# אלגוריתמים - פרופ' יובל רבני - סמסטר ב' 2021

~

# הרצאות ותרגולים

**מסכם: יחיאל מרצבך**

# תוכן העניינים

## חלק I

# הרצאות

**הרצאה מס' 1:**

**יום ראשון**

14.03.21

## 1 הקדמה

אין ספר מומלץ לקורס.

ניתן להיעזר בשני הספרים שמוצגים במודל, אך לא נלך באף אחת מהגישות שבספרים.

**קצת היסטוריה**

העיסוק באלגוריתמים התחיל בשנות ה-40 של המאה ה-20.

התיעוד הראשון שיש לנו בהיסטוריה על אלגוריתמים הוא מלוחות אבן בבבל, שם מתוארת דרך לפיתרון מערכת משוואות פשוטה - ניתן לומר כי זהו סוג של אלגוריתם.

בתקופה מעט מאוחרת יותר, מתמטיקאים יוונים הציעו אלגוריתמים אחרים: **האלגוריתם של אוקלידס** (325-270 לפנה"ס), **אלגוריתם הנפה של ארתוסטנס** למציאת מספרים ראשוניים (276-192 לפנה"ס), **ושיטת הרון** (שייתכן שמקורה עוד בבבליים), לחישוב שורש ריבועי של מספר (70-10 לפנה"ס).

בהמשך העולם העתיק, אחד ההישגים הגדולים הוא 'האריתמטיקה ההודו-ערבית' ואלו למעשה הפעולות האריתמטיות המוכרות לנו כיום. השיטה הבבלית פעלה למעשה בבסיס 60 (כיום ניתן למצוא שאריות של שיטה זו בשעון או בחישוב זוויות), אבל אנחנו משתמשים בבסיס העשרוני, שהתחילה אצל ההודים, התקדמה לפרסים ולערבים, כשמוחמד אבן מוסא אלחוארזמי הוא המוכר לנו ביותר. משמו נגזר השם 'אלגוריתם'.

בנוסף, שיטה חשובה שהתפתחה בהמשך הינה '**אלימינצית גאוס**' שנקראת על שמו של יוהאן קארל פרידריך גאוס (1777-1855). שיטה זו מוכרת כבר בעולם הסיני, ונמצאת בספר **ג'יוז'אנג סואנשו** (179 לספירה).

כל התוצאות האלו אנקדוטיאליות, במובן שאין בהם הבנה מעמיקה של הגדרת האלגוריתם. ההבנה זו הייתה רק בתחילת המאה ה-20 (כשעדיין לא היה מחשב).

למעשה, ההגדרה המתמטית היא זו שהובילה לפיתוח המחשב בסופו של דבר.

בשנת 1928 דויד הילברט ווילהלם אקרמן עסקו ב'בעיית ההכרעה' שעוסקת בשאלה האם המשפט נכון או שלילתו. בעיה זו נפתרה ב-1936, על ידי צ'רץ' וטיורינג שענו שתי תשובות (שבמהלך התבררו כזהות), לשאלות אלו. שניהם ענו כי אין תשובה ל'בעיית ההכרעה', אלא שהם נאלצו לפתח שני מודלים אבסטרקטיים שונים, צ'רץ' באמצעות תחשיב $\lambda$ וטיורינג באמצעות מכונת טיורינג. כאמור, שני מודלים אלו למעשה זהים.

אבני דרך נוספים בחקר האלגוריתמים הם התכנון ליניארי - שיטת הסימפלקס בה עסקנו קנטורוביץ היצ'קוק ודנציג (בשנים 1939-1947), ואלגוריתם אדמונדז לשידוך מקסימום (בשנת 1965, על ידי אלן קובהם וג'ק אדמונדז).

**קצת על הקורס**

במהלך הקורס נלמד אלגוריתמים לחישוב בעיות בסיסיות, שהינן אבני בעיות לבעיות מורכבות (שלא בהכרח נדבר עליהן).

בנוסף, נלמד אלגוריתמים בעלי מגוון רחב של של שימושים, שיטות לתכנון ולניתוח פתרונות אלגוריתמיים, ונדון במודלים המתמטיים שעומדים ביסוד הבעיות שהאלגוריתמים האלו פותרים.

לא נקפיד על הגדרה פורמלית של המודל החישובי - נחשוב על אלגוריתם בתור דבר שקל לתרגם אותו לשפת תכנות (למשל בשפת פייתון), במגבלות זמן ומקום נדרשות.

# 2 עקרון ההחלפה ושימושו באלגוריתמים חמדניים

**הרצאה מס' 2:**

**יום שלישי**

16.03.21

## 2.1 דוגמאות לעקרון ההחלפה

תחילה, נראה כמה דוגמאות שיעזרו לנו להבין עיקרון שנבין לאחר מכן לעומק.

**דוגמא 1 - שיבוץ משימות**

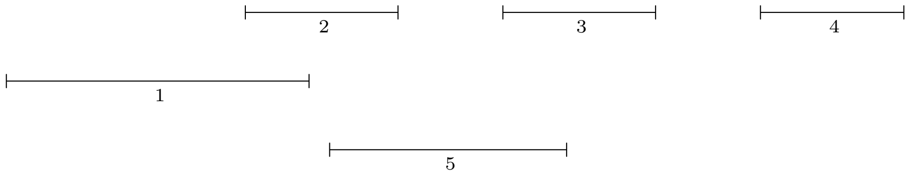
קלט:

רשימה של $n$ קטעים **סגורים** על הישר הממשי (נניח שקצוות הקטעים הם מספרים שלמים), כל קטע נתון על ידי זוג סדור של של נקודת התחלה ונקודת סיום:

$$(s_1,t_1),(s_2,t_2) \quad \cdots \quad (s_n,t_n)$$

חשוב לשים לב כי הקטעים יכולים להיות חופפים (לא מובטח שהקטעים זרים).

דוגמה לקטעים:



פלט:

רשימה באורך מירבי של קטעים לא חופפים (אינטואיטיבית, נרצה לסדר מערכת כך שהקורסים לא יהיו חופפים) - אין חשיבות לאורך הקטע.

האלגוריתם:

נמיין את הקטעים בסדר לא יורד של זמני סיום (כלומר, בה"כ $t_1 \le t_2 \le t_3 \le \ldots \le t_n$).

נעבור על הרשימה הממוינת לפי סדר ונוסיף לפי פלט כל קלט שאינו חופף לקטעים הקודמים שלקחנו.

סיבוכיות:

ראשית, נבחין כי המיון לוקח $O(n\log n)$ פעולות.

לאחר מכן, המעבר על הרשימה הממוינת אורך $O(n)$ פעולות.

על מנת לבדוק חפיפה, מספיק לנו לבדוק את הקטע האחרון שלקחנו (כי עד כה על פי הדרך שבה פעלנו, לא היו חפיפות):

למשל, אם אנו בודקים את $(s_j,t_j)$ והקטע האחרון היה $(s_i,t_i)$ עבור $i<j$ אזי בפרט אין חפיפה אס"ם $s_j > t_i$.

סך הכל, בדיקת החפיפה לוקחת $O(1)$ פעולות, וגם כאן מתבצעות סך הכל $O(n)$ פעולות, שנוספות לסך כל הפעולות.

אם כך, סך הכל התבצעו $O(n\log n)$ פעולות.

סיבוכיות המקום היא $O(1)$ (בנוסף לקלט) - המיון דורש משתנה עזר אחד בלבד ובדיקת החפיפה דורשת לזכור את נקודת הסיום של הקטע האחרון בפלט עד כה.

נבחין כי ספרנו פעולות השמה והשוואה של מספרים טבעיים ותאי זיכרון שמכילים כל אחד מספר טבעי.

נכונות האלגוריתם:

נסמן ב-$A$ את קבוצת הקטעים בפיתרון של האלגוריתם.

**משפט**

לכל קבוצה $I$ של קטעים שאינם חופפים זה לזה מתקיים כי $|A| \geq |I|$.

**הוכחה**

נניח כי $A = \{(s_{i_1}, t_{i_1}), \cdots, (S_{i_k}, t_{i_k})\}$ ו-$I = \{(s_{j_1}, t_{j_1}), \cdots, (s_{j_l}, t_{j_l})\}$, כאשר:

$$t_{i_1} \leq t_{i_2} \leq \cdots \leqslant t_{i_k}$$
$$t_{j_1} \leqslant t_{j_2} \leq \cdots \leq t_{j_l}$$

ולכן נרצה להוכיח $k \geq l$.

נוכיח באינדוקציה על $r$ כי הקטעים

$$(s_{i_1}, t_{i_1}), \cdots, (s_{i_r}, t_{i_r}), (s_{j_{r+1}}, t_{j_{r+1}}), \ldots, (s_{j_l}, t_{j_l})$$

אינם חופפים זה לזה.[1]

בסיס האינדוקציה:

עבור $r = 1$.

על פי כלל הבחירה של האלגוריתם, מתקיים כי $t_{i_1} = t_1 \leq t_{j_1}$ ולכן אם נחליף את $(s_{j_1}, t_{j_1})$ ב-$(s_{i_1}, t_{i_1})$, הקטע שהחלפנו לא חופף לשאר הקטעים בפיתרון $I$.

צעד האינדוקציה:

על פי הנחת האינדוקציה עבור $r-1$, מתקיים כי הקטעים הבאים **אינם חופפים**:

$$(s_{i_1}, t_{i_1}), (s_{i_2,} t_{i_2}), \cdots, (s_{i_{r-1}}, t_{i_{r-1}}), (s_{j_r} t_{j_r}), \ldots, (s_{j_l} t_{j_l})$$

לכן, אחרי בחירת $(s_{i_{r-1}}, t_{i_{r-1}})$ גם $(s_{i_r}, t_{i_r})$ וגם $(s_{j_r}, t_{j_r})$ זמינים לבחירה.

בעקבות כך, מתקיים כי $t_{i_r} \leq t_{j_r}$ ולכן החלפת $(s_{j_r}, t_{j_r})$ ב- $(s_{i_r}, t_{i_r})$ לא גורמת לחפיפה עם הקטעים הבאים של $I$:

$$s_{i_r} \leq t_{i_r} < s_{j_{r+1}} \leq t_{j_{r+1}} \leq s_{j_{r+2}} \leq t_{j_{r+2}} < \ldots$$

מסקנה:

---

[1] הרעיון הוא לקחת את הפתרון $I$ ובהדרגה להחליף אותו לפתרון $A$. אנחנו מוכיחים באינדוקציה שכשניקח את $r$ הקטעים הראשונים בקבוצה $A$ ולאחר מכן את $r+1$ הקטעים בקבוצה $I$, זהו פתרון חוקי לבעיה.

- אם $l \leq k$, סיימנו (החלפנו את כל הקטעים ב-$I$ בקטעים ב-$A$ ולכן הוא אופטימלי)
- אם $l > k$ אזי $(s_{i_1}, t_{i_1}), \ldots, (s_{i_k}, t_{i_k}), (s_{j_{k+1}}, t_{j_{k+1}})$ זו קבוצת קטעים לא חופפים, אבל קיבלנו סתירה לכך שהאלגוריתם סיים בלי שהביא בחשבון כי אפשר להוסיף ל-$A$ את הקטע $(s_{j_{k+1}}, t_{j_{k+1}})$ - אפשר להוסיף אותו שהרי אפשר להחליף את **כל הקטעים** בפיתרון האלגוריתם, ואז ממילא אין **חפיפות עבור הקטע האחרון**. אם כך, האלגוריתם היה אמור להוסיף קטע זה, **בסתירה** לפעולת האלגוריתם.

לסיכום, בהינתן **פיתרון אופטימלי** הראינו באמצעות **פתרון חמדני**, כי אם היה קיים פיתרון אחר, הפיתרון האופטימלי יהיה גדול או שווה לו.

האם ניתן למצוא שיטות אחרות?
מיון על פי נקודות התחלה איננו עובד.
מיון על פי אורך הקטע גם איננו עובד.

**דוגמה 2 - מיזעור האיחור**

הקלט:
קבוצה של $n$ משימות, כשכל משימה נתונה על ידי זוג מספרים טבעיים $(p_j, d_j)$.
כאשר $p_j$ הוא משך הזמן שהמשימה דורשת ו-$d_j$ הוא מועד הסיום הנדרש.

על מנת להגיש את התרגיל במועד שלו, אז על התלמיד להתחיל אותו ב-$d_j - p_j$ או מוקדם יותר.
הפלט:
שיבוץ תקין של המשימות על ציר הזמן.
כלומר, ניקח פונקציה $t : \{1, \ldots .n\} \to [0, \infty)$ שמקיימת, לכל $i, j \in [1, \ldots, n]$ כאשר $i \neq j$:

$$(t(i), t(i) + p_i) \cap (t(j), t(j) + p_j) = \emptyset$$

כלומר, לא ניתן לבצע שתי משימות במקביל ולכן החיתוך ביניהן צריך להיות ריק ($t(i)$ זהו זמן התחלת המשימה, ו-$p_i$ זהו זמן הביצוע).

המטרה:
לחשב שיבוץ תקין עבור האיחור המירבי המינימלי - ונסמן אותו ב-$L_{\max}(t)$:

$$L_{\max}(t) = \max_{j=1,\ldots,n} \{\max\{0, t(j) + p_j - d_j\}\}$$

המקסימום החיצוני נועד על מנת למצוא את **האיחור המקסימלי**, והמקסימום הפנימי נועד למנוע **ערכים שליליים**. הכוונה כאן כי אנחנו מחפשים את האיחור הכי גדול - כל איחור במשימה אחת הרי עלול לגרום איחור במשימה אחרת. המטרה שלנו היא **למזער את האיחור הכי גדול**.

האלגוריתם - EDD - Earliest Due Date:

- נמיין את המשימות בסדר לא יורד של זמני הסיום (בה"כ, $d_1 \leq d_2 \leq d_3 \leq \ldots \leq d_n$)

□ נשבץ אותן החל מזמן $0$, ללא רווחים ובסדר המיון.

כלומר, $t\left(1\right)=0$ ו-$t\left(j+1\right)=t\left(j\right)+p_j$ לכל $j\geq 1$.
בפרט, נקבל כי $t\left(j+1\right)=p_1+p_2+\ldots+p_j$.

<u>הסיבוכיות:</u>
גם כאן זמן הריצה הוא $O\left(n\log n\right)$, כי **הסיבוכיות המירבית** היא המיון.

**משפט**
$EDD$ מחזיר את $L_{\max}$.
**הוכחה**
נתבונן בשיבוץ כלשהו (אחר) $t$ שממזער את $L_{\max}$.
בה"כ אין רווחים ב-$t$. אם יש רווח במקום כלשהוא, אפשר להזיז אחורה את כל המשימות המשובצות אחרי הרווח וזה לא מגדיל את האיחור המירבי.
אם סדר המשימות אינו זהה ל-$EDD$, אזי חייבות להיות זוג משימות $i,j$ שהן עוקבות לפי $t$ (כלומר $t\left(j\right)=t\left(i\right)+p_i$) והן בסדר הפוך של $EDD$ כלומר כי $i<j$.

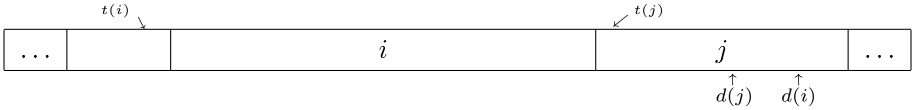
**הרצאה מס' $3$:**

**יום ראשון**

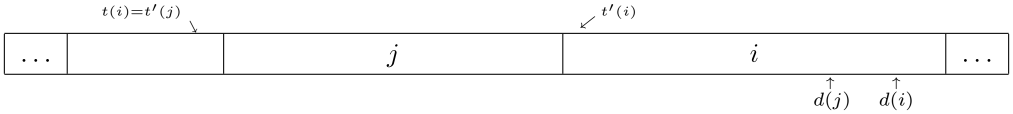
21.03.21

**טענה**
אם נחליף בין $i$ ל-$j$ לא נקבל פיתרון שערכו גדול יותר.

השיבוץ לפני ההחלפה נראה כך:



ואילו אחרי ההחלפה הוא נראה כך:



אם נחליף בין $i$ ו-$j$, נראה כי הדבר היחיד שמשתנה הוא ש-$j$ מופיע לפני $i$, וזה לא מגדיל את האיחור המירבי.
נשים לב כי $L_{\max}\geq L_j\geq L'_j$ כי הזזנו את $j$ אחורה. לעומת זאת, $L_i\geq L'_i$ כי הזזנו את $i$ קדימה.
כעת, נוכל לקבל[2]:

---

[2] אינטואיטיבית, כיוון ש-$i$ נגמר בנקודת הסיום של $j$, אבל $d_j\leq d'_i$ האיחור לא גדל.

$$L_i' = \max\left\{0, t'\left(i\right) + p_i - d_i\right\} \overset{\substack{\underbrace{\text{מסתכלים על סך הזמנים}} \\ \downarrow}}{=}$$

$$\max\left\{0, t\left(j\right) + p_j - d_i\right\} \overset{\substack{\underbrace{d_j \le d_i} \\ \downarrow}}{\le}$$

$$\max\left\{0, t\left(j\right) + p_j - d_j\right\} \overset{\substack{\underbrace{\text{הגדרה}} \\ \downarrow}}{=} L_j \le L_{\max}$$

(העיכובים של שאר המשימות לא משתנים).

סך הכל קיבלנו כי $L_i' \le L_{\max}$.

אם נחזור להוכחת המשפט, נוכל להחליף ב-$t$ (הפתרון האופטימלי שאנחנו מסתכלים על זה). כלומר, נחליף זוג משימות סמוכות בסדר הפוך ל-$EDD$, כל עוד יש זוג כזה (יש $\binom{n}{2}$ החלפות). כל צעד לא החליף את $L_{\max}$ ולכן $L_{\max}$ של $EDD$ הוא לכל היותר $L_{\max}$ **שהתחלנו איתו**, שלפי ההנחה הוא **אופטימלי**.

**דוגמה 3 - בעיית הדפדוף**

במערכת מחשב הזיכרון הוירטואלי מחולק לדפים (קטעי זיכרון בגודל קבוע). פיסית, יש זיכרון מהיר עם קיבולת של $k$ דפים (פרמטר כלשהוא), כששאר הדפים מוחזקים בזיכרון איטי. המעבד יכול לגשת רק לזיכרון המהיר. לכן, צריך לדפדף דפים בין הזיכרון המהיר לאיטי לפי הצורך.

הקלט:

סדרת הדפים שרוצים לגשת אליהם, לפי סדר הגישה. למשל, נוכל לסמן $\sigma_1, \sigma_2, \ldots, \sigma_n$ לפי סדר הגישה. לכן, יש סך הכל $n > k$ דפים ולכל $t$ מתקיים כי $\sigma_t \in \{1, \ldots, n\}$ (הדפים יכולים לחזור על עצמם).

הפלט:

לכל זמן $t = k+1, \ldots, m$ שבו מבקשים דף **שאינו** בזיכרון המהיר, יש לציין איזה דף יש "לזרוק" מהזיכרון המהיר (ה-$t$-ים הרלוונטים והדפים הנזרקים תלויים כמובן בהחלטות הקודמות)

האלגוריתם של Belady:

נסמן ב-$C_t$ את קבוצת הדפים בזיכרון המהיר בזמן $t$ (כאשר מתקבל $\sigma_t$). כלומר $C_{k+1} = \{\sigma_1, \sigma_2, \ldots, \sigma_k\}$.

אם $\sigma_t \notin C_t$, נחליף אותו בדף $\sigma \in C_t$ עבורו הפעם הבאה שמבקשים את $\sigma$ היא המאוחרת ביותר.

**משפט**

האלגוריתם של בלאדי מחשב פיתרון שמספר הדפדופים שלו מינימלי.

**הוכחה**[3]

נתבונן בפיתרון $S$ של הבעיה ויהי $t$ הצעד הראשון בו הפיתרון שונה מהאלגוריתם של בלאדי. כלומר, תוכן **הזיכרון המהיר** $C_t$ **זהה** בשני הפתרונות כאשר מקבלים את הבקשה ל-$\sigma_t$, אבל האלגוריתם של בלאדי 'זורק' את $\sigma \in C_t$ ואילו $S$ זורק את $\sigma' \in C_t$, כאשר $\sigma \neq \sigma'$.

כעת, על מנת להראות שהפיתרון של בלאדי הוא אופטימלי (כלומר מספר הדפדופים הוא מינימלי), נשתמש בדרך הבאה.

---

[3] לא הוכחנו בכיתה, ההוכחה מתוך סיכום ההרצאה.

נגדיר פיתרון חדש $S'$ כך: לפני צעד $t$ שתיארנו קודם לכן, $S'$ זהה ל-$S$ (וממילא זהה לבלאדי), אך בצעד $t$ מתקיים כי $S'$ זורק את $\sigma$. בנוסף, אחרי צעד $t$, יתקיים כי $S'$ זהה ל-$S$, חוץ מהשינויים הבאים:

$(i)$ אם בצעד $t' > t$ מתקיים כי $\sigma_{t'} \neq \sigma, \sigma'$ (כלומר הדף שנזרק בזמן זה) ו-$S$ איננו מחזיק את $\sigma_{t'}$ והוא זורק את $\sigma$, אזי בהכרח $S'$ איננו מחזיק את $\sigma_{t'}$. **במקרה זה** $S'$ זורק את $\sigma'$ ומכאן ואילך $S$ ו-$S'$ זהים.

$(ii)$ אם $\sigma_{t'} = \sigma'$ ו-$S$ זורק דף $\sigma''$: אם $\sigma'' = \sigma$ אזי $S$ לא צריך לעשות דבר (כי בשלב זה $\sigma'$ מוחזק על ידי $S'$), ואחרי צעד זה $S$ ו-$S'$ זהים.

לאחר הטרנספורמציה, נקבל כי $S'$ זהה לבלאדי עד צעד $t+1$, ולא משלם יותר מ-$S$.

כעת, אפשר להמשיך באינדוקציה עד שמקבלים פיתרון **שזהה** לבלאדי ואינו משלם יותר פיתרון המקורי, איתו התחלנו.

כלומר, ניתן לראות כיצד גם כאן השתמשנו בעיקרון ההחלפה, איתו פתחנו את שלושת הדוגמאות.

## 2.2 עץ פורש מינימום

**הרצאה מס' 4:**

**יום ראשון**

05.04.21

<u>בעיה:</u>

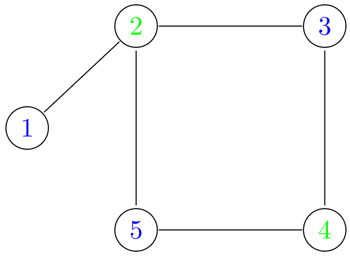
יהי $G = (V, E)$ גרף לא מכוון וקשיר. ותהי $w : E \to \mathbb{N}$ פונקציית משקל חיובית על הקשתות.

נרצה למצוא **תת גרף קשיר** שפורש את כל הצמתים, ושמשקלו **מינימלי**. תת גרף כזה, בהכרח יהיה עץ פורש.

**הגדרה**

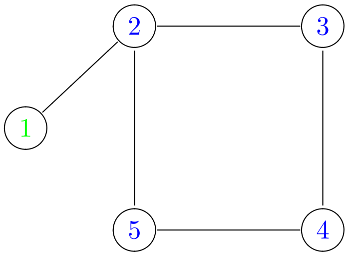
חתך בגרף הוא קבוצת הקשתות שמחברות בין הקודקודים בשני הצדדים של חלוקה של צמתי הגרף לשתי קבוצות לא ריקות.

אם נרצה מספר דוגמאות פשוטות לחתך, נוכל לראות כאן:



זהו החתך $(\{1, 5, 3\}, \{2, 4\})$.

ולעומת זאת, עוד דוגמה לחתך:



זהו החתך $(\{2,3,4,5\},\{1\})$.
הצלעות בחתך הן בשתי הדוגמאות הצלעות שמחברות קודקוד כחול לקודקוד ירוק.

**טענה**

גרף קשיר חסר מעגלים עם $n$ קודקודים, מכיל בדיוק $n-1$ קשתות.

**טענה**

אם נוסיף לעץ $T$, קשת $u$ המחברת בין שני קודקודים שלו, בגרף שנוצר קיים מעגל העובר דרך הקשר שהוספנו.

**מסקנה**

יהי $T \subseteq E$ עץ ותהי $e \notin T$ ו-$f \in T$ קשת כלשהיא על המעגל הפשוט הנוצר כתוצאה מהוספת $e$ ל-$T$. אזי $T \cup \{e\} \setminus \{f\}$ הוא עץ על $V(T)$.

**משפט**

יהי $F \subseteq E$ חתך ב-$G$ ותהי $e \in F$ קשת בעלת משקל $w(e)$ מינימלי ב-$F$. אזי בהכרח קיים עץ פורש מינימלי שמכיל את $e$.

**הוכחה**

נניח בשלילה שלא. כלומר, שלא קיים עץ פורש מינימלי שמכיל את $e$.
יהי $T$ עץ פורש מינימלי, בהכרח מהנחת השלילה הוא איננו מכיל את $e$. כעת, אם נוסיף את $e$ ל-$T$ נסגור מעגל $C$ (מהטענה הקודמת). בהכרח, מעגל זה חייב להכיל לפחות שתי קשתות מ-$F$ (על מנת שיווצר מעגל). בשלב זה ניקח $e' \neq e$ קשת נוספת של $C$. בהכרח מתקיים כי $e' \in T$ וגם $w(e') \geq w(e)$ ממינימליות $e$. מכך מתקיים כי $T \cup \{e\} \setminus \{e'\}$ הוא עץ פורש בעצמו, ומשקלו איננו גדול יותר ממשקל העץ $T$, לכן הוא בהכרח עץ פורש מינימלי, כנדרש.

**אבחנה**

יהי $T$ עץ פורש. תהי $e \in T$. הסרת $e$ מחלקת את צמתי $T$ לשני רכיבים קשירים לא ריקים ולכן מגדירה חתך $F_e$ בגרף.
כמו כן, בהכרח $e \in F_e$. אם $T$ עץ פורש מינימלי, אזי בהכרח $e$ קשת בעלת **משקל מינימלי** בחתך זה, אחרת ניתן להחליפה בקשת קלה יותר ב-$F_e$, בסתירה לכך ש-$T$ עץ פורש מינימלי.
אפשר לשים לב כי השתמשנו כאן בעיקרון ההחלפה (שוב).

**משפט**

יהי $C$ מעגל פשוט ב-$G$ ותהי $e \in C$ קשת שמשקלה **מירבי** בין קשתות המעגל. אזי קיים עץ פורש מינימלי שאינו מכיל את $e$.

**הרצאה מס' 5:**

**יום שלישי**

07.04.21

**הוכחה**

יהי $T$ עץ פורש מינימום. אם $e \notin E(T)$ אזי **סיימנו** והמשפט מתקיים.

אחרת, כלומר אם $e \in E(T)$, נבחין כי הסרת $e$ מ-$T$ מחלקת את $T$ לשני רכיבים קשירים לא ריקים, שנסמנם $T_1, T_2$.

כמו כן, נשים לב כי $V = V(T_1) \cup T(V_2)$.

נתבונן בחתך $F_{T_1 e}$, שהינן כל הקשתות ב-$G$ שמחברות בין צומת ב-$V(T_1)$ ובין צומת ב-$V(T_2)$. בפרט, מתקיים כי $e \in F_{T_1 e} \cap C$.

לכן, קיימת קשת $f \neq e$ עבורה $f \in F_{T_1 e} \cap C$. לפי ההנחה, מתקיים כי $w(f) \leq w(e)$, אך $T_1 \cup T_2 \cup \{f\}$ הוא עץ (כל אחת מה-$T$ הוא תת עץ, ו-$f$ מחברת בין צומת כלשהיא ב-$T_1$ לצומת כלשהיא ב-$T_2$), כשמשקלו הוא לכל היותר משקל $T$. לכן, גם הוא עץ פורש מינימום.

לאחר שהוכחנו את שתי התכונות הקשורות לעץ פורש מינימום, נמצא שיטה למציאת עץ פורש כזה.

**שיטה למציאת עץ פורש מינימום**

נקרא לדבר זה שיטה ולא אלגוריתם, כיוון שלא נדגים את כל פרטי המימוש. למעשה, משיטה זו אפשר לגזור מספר אלגוריתמים - בפרט האלגוריתמים של פרים ושל קרוסקל הם מקרים פרטיים של שיטה זו.

נניח כי קשתות הגרף יכולות להיות צבועות בכחול, באדום, או לא צבועות כלל.
בתחילה כל הקשתות לא צבועות כלל.

הכלל הכחול:

נבחר חתך $F \subseteq E$ שאין בו **אף קשת כחולה**, ונצבע בכחול קשת $e \in F$ בעלת משקל **מינימלי** מבין קשתות $F$ .

הכלל האדום:

נבחר מעגל $C$ שאין בו **אף קשת אדומה** ונצבע $e \in C$ בעלת משקל **מירבי** מבין קשתות $C$ באדום.

**משפט**

נפעיל את הכללים הללו בסדר כלשהוא. כל עוד יש כלל שאפשר להפעיל[4].אזי כאשר נעצור, מתקיים[5]:

1. כל הקשתות צבועות.
2. הקשתות הצבועות בכחול הן עץ פורש מינימום.

**הוכחה**

תחילה, נוכיח באינדוקציה על מספר הצעדים כי תמיד יש עץ פורש מינימום שמכיל את הקשתות הכחולות ואף קשת אדומה.

בסיס האינדוקציה:

---

[4] אפשר לראות כיצד קרוסקל ופרים מבצעים את שיטה זו.

[5] על מנת שדבר זה יהיה אלגוריתם, עלינו להחליט איזה מהכללים להפעיל. נצטרך למצוא שיטה אפקטיבית לזהות על מה אפשר להפעיל.

בתחילה, אין אף קשתות צבועות (מאיך שהגדרנו את השיטה), ולכן הטענה נכונה באופן ריק.

צעד האינדוקציה:

יהי $T$ עץ פורש מינימום שמקיים את הנחת האינדוקציה.
כעת, בצעד הבא ייתכנו שתי אפשרויות (יוצאים מנקודת הנחה שיש עוד צעד):

1. נפעיל את הכלל הכחול על חתך $F$. תהי $e \in F$ הקשת שנבחרה להיצבע **בכחול**. אם מתברר כי $e \in E(T)$ סיימנו (הטענה ממשיכה להתקיים ביחס ל-$T$). אחרת, נניח כי הוספת $e$ ל-$T$ סוגרת מעגל פשוט $C$, שהרי הוספתה ל-$T$ מיותרת. נבחין כי $e \in F \cap C$ ולכן ישנה עוד קשת אחת ב-$C$, שנסמנה $e \neq f \in F \cap C$. לפי הכלל הכחול, $f$ לא צבועה בכחול (משום שאנו יכולים להפעיל את הכלל הכחול רק על חתך בהן אין קשתות כחולות). בנוסף, $w(f) \geq w(e)$, ולכן המשקל של העץ $T \setminus \{f\} \cup \{e\}$ בהכרח לא גדול יותר יותר ממשקל $T$ - כלומר, זהו עץ פורש מינימלי, שמקיים את הטענה.

2. מפעילים את הכלל האדום על מעגל פשוט $C$. תהי $e \in C$ הקשת שנבחרה להיצבע **באדום**. אם מתברר כי $e \notin T$ סיימנו. אחרת, נתבונן בחתך $F_{T,e}$ שנוצר על ידי הסרת הצלע $e$. נבחין כי $e \in F_{T,e} \cap C$ ולכן יש עוד קשת $f \neq e$ כך ש-$f \in F_{T,e} \cap C$. נשים לב כי $f$ **איננה** אדומה (כי אפשר להפעיל את הכלל רק אם אין במעגל קשתות אדומות). כמו כן, לפי בחירת $e$ (מקסימלית) מתקיים כי $w(f) \leq w(e)$. לכן בפרט $T \setminus \{e\} \cup \{f\}$ הוא עץ שמשקלו לא גדול ממשקל $T$ ולכן הוא עץ פורש מינימלי.

כעת, עלינו להראות כי לא ייתכן שהצביעה תיפסק לפני שכל הקשתות צבועות. נניח בשלילה שזה לא המצב, כלומר שחלק מהקשתות לא צבועות, ולא ניתן להמשיך. יהי $T$ עץ פורש מינימום שמכיל את כל הקשתות הכחולות ואף קשת אדומה (קיים כזה לפי הוכחת האינדוקציה).
תהי $e$ קשת כלשהיא שאינה צבועה, נראה שאפשר להפעיל את אחד הכללים, ולכן סימן שלא עצרנו.
אם $e \in E(T)$ אזי החתך $F_{T,e}$ לא מכיל אף קשת כחולה, כי כל הקשתות הכחולות מוכלות ב-$E(T)$ ו-$F_{T,e} \cap E(T) = \{e\}$. על החתך הזה אפשר להפעיל את הכלל הכחול.
אם $e \notin E(T)$. אם נוסיף את $e$ ל-$T$, נסגור מעגל $C$. ב-$C$ אין אף קשת אדומה, כי $C \setminus E(T) = \{e\}$ ו-$e$ לא צבועה. על $C$ אפשר להפעיל את הכלל האדום ולכן הגענו לסתירה. כלומר, בסיום הריצה כל הקשתות צבועות.

אם כל הקשתות צבועות ויש עץ פורש מינימום $T$ שמכיל את כל הכחולות ואף קשת אדומה, בהכרח $T$ הוא אוסף הקשתות הכחולות.

**מסקנה**

האלגוריתמים של פרים ושל קרוסקל[6] (ווריצאציות נוספות) מחשבים עץ פורש מינימום.

## 2.3 מערכות של קבוצות

**הרצאה מס' 6:**

**יום שלישי**

11.04.21

תהי קבוצת בסיס. לדוגמא - קבוצת הקשתות של גרף לא מכוון.
בנוסף, נתונה רשימה של תת קבוצות של קבוצות הבסיס. לדוגמא - כל המעגלים הפשוטים.

---

[6] נפעיל את הכלל הכחול עד שאי אפשר. בצורה דומה אפשר להפעיל גם את הכלל האדום, אך זהו אלגוריתם יעיל פחות. זהו סדר גודל $|E|^2$ והקשתות שקיימות כעת הן העפ"מ.

**הגדרה**

מטראויד הוא זוג סדור $(E,I)$ ובו $E$ קבוצת בסיס סופית ו-$I$ אוסף של תתי קבוצות של $E$ שמקיים שתי תכונות:

1. (ירושה) - אם $A \in I$ ו-$C \subseteq A$ אזי $C \in I$.
2. (הרחבה) אם $A, C \in I$, ו-$|C| > |A|$ אזי קיים $e \in C \setminus A$ עבורו $A \cup \{e\} \in I$.

הקבוצות שנמצאות ב-$I$ נקראות בלתי תלויות. קבוצה בלתי תלוייה מקסימלית ביחס להכלה נקראת בסיס.

**טענה**

כל הבסיסים במטרואיד שווי גודל. (אחרת, אפשר להגדיל את הקטן ביותר והוא לא מקסימלי).

את תכונת ההרחבה אפשר להחליף במספר תכונות אחרות, ולהסיק מספר טענות.

**טענה - תכונת ההחלפה**

אם $A, B \in I$ שני בסיסים (מקסימליות ולכן שוות גודל) אזי לכל $a \in A \setminus B$ קיים $b \in B \setminus A$ עבורו $A \setminus \{a\} \cup \{b\}$ גם הוא בסיס של המטרואיד.

בעקבות טענה זו אפשר גם להוכיח את הטענה הבאה.

**טענה (החלפה סימטרית)**

באותם תנאים של הטענה הקודמת, לכל $a \in A \setminus B$ קיים $b \in B \setminus A$ עבורם גם $A \setminus \{a\} \cup \{b\}$ הוא בסיס וגם $B \setminus \{b\} \cup \{a\}$ הוא בסיס.

**טענה (החלפה חח"ע)**

באותם תנאים של הטענה הקודמת, קיימת העתקה חח"ע $f : A \setminus B \to B \setminus A$ עבורה לכל $a \in A \setminus B$ הקבוצה גם $A \setminus \{a\} \cup \{f(a)\}$ היא בסיס.

**דוגמאות**

1. דוגמה טריוויאלית - אוסף כל תת הקבוצות של $E$ הוא מטרואיד (ברור כי גם תכונת הירושה וגם תכונת ההרחבה מתקיימות). הקבוצה $E$ היא המטרואיד היחיד.
2. מטראויד המעגלים של הגרף - בהינתן גרף סופי לא מכוון $G = (V, E)$ אזי האוסף של כל קבוצות הקשתות $f \subseteq E$ עבורן $F$ **חסרת מעגלים**, מהווה מטרואיד מעל קבוצת הקשתות. מטרואיד זה ייקרא **מטרואיד המעגלים של הגרף**. אם $G$ קשיר, הבסיסים הם כל העצים הפורשים של $G$.
3. מטראויד הוקטורים - בהינתן מטריצה $A$, אוסף קבוצות העמודות של $A$ שהן בת"ל מהווה מטרואיד מעל קבוצת העמודות. הבסיסים הם העמודות שפורשות את המרחב.
4. המטראויד הביסרקולרי - בהינתן גרף סופי לא מכוון $G = (V, E)$, אוסף קבוצות הקשתות $F \subseteq E$ עבורן כל רכיב קשיר של $F$ מכיל לכל היותר מעגל אחד הוא מטרואיד מעל $E$.

### 2.3.1 האלגוריתם החמדן

נרצה לתאר סכמה שפותרת לנו כל בעיה הקשורה למטרואידים.

הבעייה:

נתון מטרואיד $(E,I)$, כאשר $I$ לא נתון באופן מפורש אלא באופן כללי. מבחינה אלגוריתמית, נתון אלגוריתם יעיל שבהינתן $F \subseteq E$ בודק האם $F \in I$. מעבר לכך, נתונה פונקציית משקל חיובית $w : E \to \mathbb{N}$.

המטרה:

למצוא בסיס שמשקלו **מקסימלי**.

אלגוריתם:

נמיין את איברי $E$ בסדר **לא עולה**, נעבור על כל האיברים בסדר המיון ונוסיף לפתרון החל מקבוצה ריקה כל איבר שניתן להוסיפו מבלי לצאת מ-$I$.

**טענה**

נקבל בוודאות בסיס מהאלגוריתם לעיל.

(כל איבר שדילגנו עליו בוודאי בלתי אפשרי לתוצאה הסופית).

**טענה**

התוצאה של האלגוריתם אופטימלית.

**הוכחה**

נוכיח באינדוקציה שבכל שלב של ריצת האלגוריתם קיים בסיס אופטימלי שמכיל את כל האיברים שלקחנו לפתרון עד כה.

בסיס האינדוקציה

בתחילה, הפתרון החלקי הוא הקבוצה הריקה, וכל בסיס אופטימלי מכיל אותה בהכרח.

צעד האינדוקציה

נניח כי קיים בסיס **אופטימלי** $B \subseteq E$ שמכיל את כל האיברים שבחרנו **עד כה** באלגוריתם.

נסמנו $|B| = k, B = \{b_1, b_2, b_3, \ldots, b_k\}$ וגם $w(b_1) \geq w(b_2) \geq \ldots \geq w(b_k)$. ברור כי $B$ לא מכיל איבר שדילגנו עליו עד כה.

בפרט, אם הגענו בשלב פעולת האלגוריתם ל-$l \leq k$ אז הפיתרון החלקי שלנו עד כה הוא $b_1, \ldots, b_l$. כעת:

1. אם בשלב של צעד האינדוקציה אנחנו מדלגים על האיבר, ברור כי טענת האינדוקציה נשארת נכונה (מהנחת האינדוקציה).
2. אם בשלב זה אנחנו בוחרים את האיבר הבא, $a$, כלומר כעת הפיתרון הוא $b_{1,}b_2, \ldots, b_l, a$, נחלק למקרים.
   (א) אם $a \in B$, ברור שהטענה ממשיכה להיות נכונה.
   (ב) לכן נניח כי $a \notin B$.

   נתבונן כעת בבסיס $B'$ שמרחיב את הפיתרון שלנו $B' = \{b_1, b_2, \ldots, b_l, a, \ldots\}$ - אנחנו מניחים שיש כזה, והסיבה לכך היא ש-$l \leq k$, לכן על מנת שתהיה קבוצה בת״ל מקסימלית, $B'$ חייב להיות בסיס.

   מתכונת ההחלפה, קיים איבר $b \in B \setminus B'$ כך ש-$B \setminus \{b\} \cup \{a\}$ הוא בסיס, אך $b \notin \{b_1, \ldots, b_l, a\}$ ולכן $w(a) \geq w(b)$ (מהגדרת האלגוריתם, שבוחר תמיד את הגדול יותר).

   בפרט $B \setminus \{b\} \cup \{a\}$ אופטימלי וגם מכיל את $a$ ואת כל האיברים הקודמים ולכן גם במקרה זה טענת האינדוקציה נשמרת.

**משפט (Edomonds)**

תהי $(E,I)$ מערכת קבוצות שמקיימת את תכונת הירושה. אם אלגוריתם החמדן מוצא פתרון אופטימלי עבור כל פונקציות משקל חיובית, אז כל מערכת הקבוצות היא מטרואיד.

**הוכחה**

ההוכחה תתבצע בצורה הזאת - נראה כי אם לא מדובר במטרואיד, אז יש **פונקציית משקל שלא עובדת**.

**הרצאה מס' 7:**

**יום שלישי**

13.04.21

תהי $S \subseteq E$ קבוצה כלשהיא של איברים, ותהיינה $A, B \subseteq S$ שתי קבוצות בלתי תלויות **מקסימליות ב-$S$ ביחס להכלה**, כלומר $A, B \in I$ ולא קיימת $A \subsetneq A' \subseteq S$ עבורה $A' \in I$ ובדומה לא קיימת $B \subsetneq B' \subseteq S$ עבורה $B' \in I$.

נניח בשלילה כי $(E,I)$ אינה מטרואיד.

אם $(E,I)$ אינה מטרואיד, אזי יש $S$ ויש $A, B$ כאלה עבורן $|A| < |B|$, אחרת מתקיימת תכונת ההרחבה[7]. כעת, ניקח את $S = A \cup B$. נסמן $S = \left\{e_1, e_2, \cdots, e_{|S|}\right\}$ כך שאיברי $A$ מופיעים לפני איברי $B$.

בשלב זה, נגדיר את פונקציית **המשקל שלא עובדת**. נגדיר עבור $e_i \in S$ $w(e_i) = 1 + \varepsilon_i$ ואילו עבור $e \notin S$ נגדיר $w(e) = 0$.

כמו כן, נדרוש כי $\varepsilon_1 > \varepsilon_2, \ldots > \varepsilon_{|S|}$ וגם $\sum\limits_{i=1}^{|S|} \varepsilon_i < 1$.

איזה אפסילונים ניקח? נוכל לקחת למשל $\varepsilon_i = 2^{-i}$ שיקיימו את הדרישות שלנו. האלגוריתם החמדן בהכרח יפיק את $A$, אולי בתוספת עוד איברים שמשקלם 0. אזי משקל הפתרון הוא:

$$\begin{aligned}
&w(e_1) + w(e_2) + \cdots + w\left(e_{|A|}\right) = \\
&= 1 + \varepsilon_1 + 1 + \varepsilon_2 + \cdots + 1 + \varepsilon_{|A|} = \\
&\qquad\quad \underbrace{\text{כך הגדרנו}} \\
&\qquad\qquad\quad \downarrow \\
&= |A| + \sum_{i=1}^{|A|} \varepsilon_i < \\
&\quad \underbrace{B \subseteq S \Rightarrow w(e_i) > 1} \\
&\qquad\qquad\quad \downarrow \\
&|A| + 1 \le |B| < \\
&\sum_{e \in B} w(e) \le \\
&\text{ערך פיתרון אופטימלי}
\end{aligned}$$

כלומר, הראינו למעשה כי בחירת $B$ טובה יותר מאשר הפתרון של האלגוריתם החמדן שהרי היא משיגה תוצאה גבוהה יותר. ממילא, מדובר בסתירה לאופטימליות האלגוריתם החמדן.

ניזכר בדוגמאות שראינו בעבר. נשים לב כי בעיית שיבוץ הקטעים הממושקלים איננה מטרואיד, כי נוכל למצוא קטעים בגדלים שונים, ולכן לא נוכל להשתמש באלגוריתם החמדן לפתרון בעייה זו.

כמו כן, כפי שכבר אמרנו, העצים הפורשים של גרף סופי לא מכוון וקשיר הם הבסיסים של מטרואיד המעגלים של הגרף. מכאן עולה כי האלגוריתם החמדן יכול לשמש עץ פורש מקסימום. באמצעות החלפה של משקל כל קשת $w(e)$ ב-$w'(e) = w_{\text{max}} - w(e)$[8], האלגוריתם החמדן יחשב עץ פורש מינימום (דבר זה נקרא היפוך סדר הקשתות).

---

[7] הסבר מפורט יותר: נניח שעבור כל $S$ וכל שתי קבוצות בלתי תלויות ביחס להכלה $A, B \subseteq S$ מתקיים $|B| > |A|$. ניקח $S = A \cup B$ , בהכרח $A$ אינה ב"ת מקסימלית ביחס להרחבה בתוך $S$. אזי יש איבר $b \in S \setminus A = B \setminus A$ שאפשר להוסיף ל-$A$ ולכן זה מטרואיד.

[8] עץ פורש מקסימום תחת $w'$ שקול לעץ פורש מינימום תחת $w$.

למעשה, האלגוריתם של קרוסקל הוא האלגוריתם החמדן על מטרואיד.

## 3 שיטת הזיכרון ותכנון דינמי

לעיתים, כפי שכבר ראינו, האלגוריתם החמדן לא יעבוד. לשם כך נצטרך אלגוריתם מורכב יותר, שנקרא תכנון דינמי.

אלגוריתם זה למעשה פועל רקורסיבית, ומתעדכן בצורה דינמית. באופן כלל, נסתמך בדרך כלל על מידע קודם, אותו נוכל לשמור בטבלה.

### 3.1 בעיית התרמיל

#### 3.1.1 הגדרת הבעייה וכשלונם של האלגוריתמים החמדניים

נתונה קבוצה $E$. לכל $e \in E$ יש נפח $v\left(e\right)$ ומשקל $w\left(e\right)$. נתון גם נפח התרמיל $V$.[9]

נתונים $n$ חפצים עם נפחים $v_1, \ldots v_n$ ומשקלים $w_1, \ldots, w_n$.

אנו רוצים למצוא $F \subseteq E$ עבורה $\sum\limits_{e\in F} v\left(e\right) \leq V$ ו-$\sum\limits_{e\in F} w\left(e\right)$ מקסימלי מבין האריזות המותרות (שנפחן לכל היותר נפח התרמיל).

אוסף האריזות החוקיות מקיים את תכונת הירושה אך אמנם תכונת ההרחבה לא בהכרח מתקיימת. למשל, נניח כי יש איבר אחד שנפחו $V$ והרבה איברים שהנפח שלהם $1$. אם כך, זה לא מטרואיד, ואלגוריתם החמדן לא יעבוד (כל הבאסה). מעבר לכך, זה אלגוריתם ממש גרוע: נניח אם יש איבר אחד שנפחו $V$ ומשקלו $1$ ויש $n-1$ איברים שהנפח שלהם $\frac{V}{n-1}$ ומשקלם $1-\varepsilon$ (עבור $\varepsilon > 0$ מאוד קטן). האלגוריתם החמדן ישיג משקל של $1$ אך הפתרון האופטימלי משיג משקל של $\left(1-\varepsilon\right)\left(n-1\right)$.[10]

הנה אלגוריתם חלופי (קצת יותר סבבה): נמיין את האיברים בסדר לא עולה של המשקל ליחידת נפח $\frac{w(e)}{v(e)}$ ואז נשתמש באריזה חמדנית ביחס לסדר הזה.

אבל האלגוריתם הזה עדיין לא מספיק טוב, כי אם למשל נתבונן בדוגמה **הרעה** הבאה. נניח שהקלט הוא בדיוק שני איברים: אחד עם נפח $V$ ומשקל $V$ והשני עם נפח $1$ ומשקל $1+\varepsilon$ - נקבל $1+\varepsilon$ במקום $V$ - שהוא הפתרון הטוב יותר.

כאן הסיבה שהאלגוריתם נכשל שונה: מצד אחד יש איבר בעל נפח גדול, אך גם המשקל גדול (ולכן היחס בעייתי). מצד שני, לשאר האיברים יש יחס רווח-נפח עדיף, אבל הם ממלאים רק חלק קטן מהתרמיל.

מה עם הרעיון הבא? נריץ את שני האלגוריתמים ונבחר את הפתרון הטוב מבין השניים.

למעשה, פיתרון זה משיג לפחות $\frac{1}{2}$ מהמשקל האופטימלי[11].

**הוכחה (בערך)**

נבחין כי הרווח של שני הפתרונות $\leq$ הרווח האופטימלי.[12]

#### 3.1.2 פתרון דינמי ראשון לבעיית התרמיל

**הרצאה מס' $8$:**

**יום ראשון**

18.04.21

הבה ונחשוב על פיתרון דינמי, כלומר פתרון שמתבסס על הפתרונות הקודמים.

---

[9] חשוב לשים לב כי זו בעיית $NP$ קשה - כלומר אין אלגוריתם יעיל שפותר אותה.

[10] האיבר הראשון אמנם הבטיח את הרווח הגדול מבין כל האיברים, אך הרווח שהאיבר הראשון מבטיח ליחידת נפח מאוד קטן - $\frac{1}{V}$. הרווח של $e_2$ הרבה יותר גדול - $\frac{(1-\varepsilon)(n-1)}{V}$ וכן הלאה.

[11] ספויילר לאלגוריתמים מקרבים ☻.

[12] יובל לא ממש האריך בזה השנה. אם תרצו בסיכומים של שנה שעברה יש הסברים מפורטים יותר.

לשם כך נתבונן בחפץ האחרון $n$. עבור חפץ זה, ישנן שתי אפשרויות: הוא נמצא בפיתרון האופטימלי או שאינו נמצא בפיתרון האופטימלי.

אם הוא נמצא בפיתרון, אזי הפיתרון הוא אריזה אופטימלית של $\{1,\dots n-1\}$ בנפח $V-v_n$ בתוספת האיבר האחרון $n$ (אנחנו בטוח צריכים להישאר בגודל המתאים).

אם החפץ לא נמצא בפיתרון האופטימלי, אזי הפיתרון הוא אריזה אופטימלית של מ-$\{1,\dots,n-1\}$ בנפח $V$.

**מימוש בעזרת רקורסיה**

בעקבות כך, נוכל להגיע **לנוסחה רקורסיבית**. נסמן ב-$P\left(i,V'\right)$ **אריזה אופטימלית** של של חפצים מ-$\{1,\dots,i\}$ בתרמיל **שהנפח** שלו $V'$.

את המשקל נסמן ב-$w\left(P\left(i,V'\right)\right)$.

אם כך, נקבל:

$$w(P(n,V))=\max\left\{w_n+w\left(P\left(n-1,V-v_n\right)\right),\left(P(n-1,V)\right)\right\}$$

מה בעצם עשינו כאן? אנחנו בכל פעם בודקים מהו המשקל הכי טוב, בהתאם לשתי האפשרויות שראינו קודם לכן. נבחר את המקסימלי מבין שתי הפתרונות.

מימוש פשוט של תנאי הרקורסיה הזו (בתנאי עצירה סביר) "עולה" בדיקה של $2^n$ אפשרויות (☹).

אבל אם נתבונן ב-$P\left(i,\cdot\right)$ אין בהכרח $2^i$ אפשרויות רלוונטיות ($\cdot$ מסמן את האפשרות לקבל ערכים שלמים בין 0 ל$\sum\limits_{j=1}^{i} v_j$).

כיוון שאנו דורשים כי $\sum\limits_{j=1}^{i} v_j \leq V$, יש **לכל היותר** $V+1$ ערכים רלוונטיים - כלומר יש לנו אילוץ שבעצם דורש כי נקבל מספר נפחים מתאים.

אם נניח וניתנו לנו כל הערכים של $P\left(i-1,\cdot\right)$ (יש $V+1$ כאלו) , נוכל לחשב את **כל הערכים** של $P\left(i,\cdot\right)$. כל ערך דורש $O\left(1\right)$ פעולות אריתמטיות, ותכל'ס סך הכל יש $V+1$ ערכים שצריך לחשב.

אם נרצה לחשוב על זה קצת לעומק, נוכל להבחין כי אנחנו מתבססים למעשה על המידע הקודם $P\left(i-1,\cdot\right)$ ובודקים מה יקרה באיבר ה-$P\left(i,\cdot\right)$.

במקרה ההתחלתי, מה שנרצה לעשות זה לבדוק האם עבור האיבר הראשון, הנפחים והמשקלים מתאימים.

כלומר, לכל $V'\in\{0,\dots,V\}$ - כל תת נפח - נרצה לבדוק מהו הערך של $P\left(1,V'\right)$ ומהו משקלו. כלומר, אם תרצו, מהו אורך המסלול (כמה אספנו) ומה משקלו.

אז נקבל, עבור אורך המסלול:

$$P\left(1,V'\right)=\begin{cases}\{1\} & v_1\leqslant V'\\ \emptyset & else\end{cases}$$

אם לא נוכל להכניס את המשקל לתוך $V'$ כלשהוא, אזי ממילא הקבוצה לא קיימת.

נתאים את המשקל:

$$w\left(1,V'^{;}\right)=\begin{cases}w_1 & v_1\leqslant V'^{;}\\ 0 & else\end{cases}$$

כעת, בהינתן כל הערכים עבור 1, נוכל לחשב את כל הערכים עבור 2 וכן הלאה.
העדכון יתבצע על פי הנוסחה הבאה:

$$P\left(i,V'\right)=\begin{cases}\{i\}\cup P\left(i-1,V'-v_i\right)\\ P\left(i-1,V'\right)\end{cases}$$

כאשר התנאי הראשון מתקיים אם $V'\geq v_i$ וגם $w\left(P\left(i-1,V'-v_i\right)\right)+w_i\geqslant w\left(P\left(i-1,V'\right)\right)$,
כלומר צריך כי גם הנפח מתאים (לא חרגנו), וגם כי הוספת המשקל של $i$ **תורמת** לנו.
אפשר לחשוב על זה כטבלה בעלת שורות מ-1 עד $n$ ועמודות מ-0 עד $V$.
נעבור שורה שורה על מנת למלא את הטבלה - בכל פעם נתבונן בשורה אחת.

| $i/V'$ | $0$ | $\cdots$ | $V'$ | $\cdots$ | $V$ |
|---|---|---|---|---|---|
| $1$ | $\emptyset$ | $\cdots$ | $\{1\}$ | $\cdots$ | $\{1\}$ |
| $\vdots$ | | | | | |
| $i$ | | | $P\left(i,V'\right),w\left(P\left(i,V'\right)\right)$ | | |
| $\vdots$ | | | | | |
| $n$ | | | | | $P(n,V),w(P(n,V))$ |

**פסאודו קוד**

האלגוריתם עצמו נראה כך בפסאודו קוד:

**הרצאה מס' 9:**

**יום שלישי**

20.04.21

**אלגוריתם 1** בעיית תרמיל הגב

```
1. לכל $j=0$ עד $V \leftarrow j$:
    (א) אם $j<V_1$ אזי $P\left(1,j\right)\leftarrow\emptyset$.
    (ב) אחרת: $P\left(1,j\right)\leftarrow\{1\}$
2. לכל $0 \leftarrow i$ עד $n \leftarrow i$:
    (א) לכל $0 \leftarrow j$ עד $v_i \leftarrow j$:
        i. אם $v_i>j$ אז $P\left(i,j\right)\leftarrow P\left(i-1,j\right)$
        ii. אם $P\left(i-1,j\right)>P\left(i-1,j-v_i\right)+w_i$ אזי $P\left(i,j\right)\leftarrow P\left(i-1,j\right)$
        iii. אחרת: $P\left(i,j\right)\leftarrow P\left(i-1,j-v_i\right)\cup\{i\}$.
3. תחזיר את $P\left(n,v\right)$
```

**טענה**

בסיום ריצת האלגוריתם $P\left(n,v\right)$ מחזיר פתרון אופטימלי (כלומר, קבוצת חפצים ניתנת לאריזה בתרמיל שמשקלה מירבי).

**הוכחה**

נדגים רק את רעיון ההוכחה.

יש להוכיח באינדוקציה על מספר התאים ב-$P$ שעדכנו כי לכל $i \in \{1,\ldots,n\}$ ולכל $V' \in \{0,\ldots V\}$ מתקיים כי $P\left(i,V'\right)$ הוא פתרון אופטימלי של אריזת חפצים מ-$\{1,\ldots,i\}$ בתרמיל בנפח $V'$.

למעשה, בכל פעם שאנחנו מנסים להוכיח בעיה דינמית יש להשתמש באינדוקציה. דבר זה נובע מכך כי בכל פעם אנחנו מתבססים על הצעד הקודם ועל הבסיס ולכן אינדוקציה היא הדרך האידיאלית להוכיח זאת.

**סיבוכיות**

זמן

נבחין כי יש לנו אתחול ועוד $n-1$ איטרציות על $i$. בכל צעד כזה יש מעבר על $V-1$ ערכים של נפח. כמו כן, לכל ערך כזה מבצעים $O\left(1\right)$ פעולות אריתמטיות. לכן סך הכל הזמן הינו $O\left(nV\right)$ - למעשה זמן הריצה הוא גודל הטבלה כפול זמן המילוי של תא, שבמקרה שלנו הוא $O\left(1\right)$.

מקום

יש לשמור ערכים עבור $i-1$ ועבור $i$. אם כך, אנו שומרים $O\left(V\right)$ תאי זיכרון מעבר לקלט. בכל תא שומרים את האריזה ואת משקלה, סך הכל $O\left(nV\right)$ תאים שמחזיקים ערך מספרי או מצביע (שוב, גודל הטבלה זה המקום הנדרש).

האם מדובר בזיכרון שהינו פולינומי בגודל הקלט? השאלה היא מהו הייצוג. אם $V$ מיוצג בייצוג אונארי (כלומר, כמו שאנחנו רגילים, ספירה באמצעות הידיים), אז אכן מדובר בזיכרון שהינו פולינומי בגודל הקלט, אך אם $V$ מיוצג בייצוג בינארי - לא דווקא. (ייתכן כי $V$ אקספוננציאלי בגודל הקלט, זה תלוי ב-$n$). את $V$ עצמו ניתן לייצג על ידי $\lceil \log V \rceil + 1$ ביטים.

אלגוריתם כזה, שהוא פולינומי אם"ם המספרים מיוצגים בייצוג אונארי, נקרא אלגוריתם **פסאודו-פולינומי**.

### 3.1.3 פתרון דינמי שני לבעיית התרמיל - ללא תלות בנפח

**הגדרת הבעייה**

כיוון שאנחנו תלויים גם ב-$V$ וגם ב-$n$, נחפש פיתרון שאיננו תלוי ב-$V$, שהרי ייתכן כי $V$ יהיה גדול מאוד ביחס ל-$n$.

פורמלית, נגדיר כעת את המשקל המקסימלי:

$$w_{\max} = \max\left\{w_i : i \text{ s.t. } v_i \leqslant V\right\}$$

המשקל המירבי של אריזה כלשהי הוא לכל היותר$nw_{\max}$ כי במקרה הכי טוב ניתן לארוז את **כל החפצים**, ולכולם יש את אות המשקל, המשקל המקסימלי. אם כך, המשקל של אריזה נמצא בקבוצה $\{0,\cdots,nw_{\max}\}$.

כמו כן, נגדיר $Q\left(i,W\right)$ שהיא למעשה **האריזה** של חפצים מ-$\{1,\ldots,i\}$ **שמשקלה בדיוק** $W$ ונפחה **מינימלי** מבין האריזות הללו. אם אין אריזה של חפצים מ-$\{1,\ldots,i\}$ שמשיגה משקל **בדיוק** $W$, נסמן זאת על ידי ערך מיוחד, $none$ - בשונה מהטבלה הקודמת, כאן אין בהכרח פיתרון.

פתרון בעיית התרמיל מצוי באיבר $Q\left(n,W\right)$ עבור $W$ **המקסימלי** שבו האיבר הזה **אינו** $none$.

**נוסחת הרקורסיה**

כעת, אפשר להגדיר נוסחת רקורסיה מתאימה, לכל $W \in \{0,\ldots,nw_{\max}\}$ מתקיים, עבור השורה הראשונה[13]

$$Q\left(1,W\right) = \begin{cases} \{1\} & W = w_1 \\ none & else \end{cases}$$

---

[13] ניזכר! אנחנו מחפשים בדיוק את המשקל $W$, ואת הנפח המינימלי.

מה שעשינו כאן, הוא למעשה לבדוק עבור האיבר הראשון, האם משקלו תואם את המשקל מ-$W \in \{0, \ldots, nw_{\max}\}$ - אם לא, נשים $none$.

נראה עכשיו כיצד ניתן לעשות **עדכון** של הטבלה.

כלומר, נניח שנתונה השורה ה-$i-1$ של הטבלה $Q$ ונרצה לחשב את $Q\left(i, W\right)$ לכל $W \in \{0, \ldots, nw_{\max}\}$.

יהי $W \in \{0, \ldots, mw_{\max}\}$.

ישנן שלוש אפשרויות:

1. אין פיתרון. דבר זה מתרחש כאשר בשורה ה-$i-1$ מתקיים כי $Q\left(i-1, W\right) = none$ (אין פתרון שעבור $i-1$ איברים מביא בדיוק את המשקל $W$) וגם שמתרחשות אחת משתי האפשרויות הבאות:

   (א) $w_i$ גדול מ-$W$ כלומר כי $W - w_i < 0$. כלומר, ברור כי הוספת $w_i$ ל-$W - w_i$ לא תיתן לנו בדיוק את $W$ כפי שאנחנו מחפשים.

   (ב) $Q\left(i-1, W - w_i\right) = none$ - אין שום פיתרון כך שאם נוסיף את $w_i$ נקבל בדיוק את $W$ שאנחנו מחפשים.

   במקרה זה, $Q\left(i, W\right) \leftarrow none$ - כלומר אין פיתרון.

2. יש פיתרון אבל הוא לא מדהים. אנחנו הרי מחפשים את הנפח המינימלי, ואם מתקיים כי הוספת הנפח $v_i$ לא תורמת, כלומר $v\left(Q\left(i-1, W\right)\right) < v\left(Q\left(i-1, W - w_i\right)\right) + v_i$, אזי $Q\left(i, W\right) \leftarrow Q\left(i-1, W\right)$.

3. אחרת: כלומר, יש אחלה פיתרון, אזי $Q\left(i, W\right) \leftarrow Q\left(i-1, W - w_i\right) \cup \{i\}$.

התוצאה נמצאת בתא ה-$Q\left(n, W'\right)$ כאשר $W'$ הוא המקסימלי שאין בתא שלו $none$.

**הטבלה**

| $i/w$ | $0$ | $\cdots$ | $w$ | $\cdots$ | $n \cdot w_{\max}$ |
|---|---|---|---|---|---|
| $1$ | none | $\cdots$ | $\{1\}$ | $\cdots$ | $\{1\}$ |
| $\vdots$ | | | | | |
| $i$ | | | $P\left(i, V'\right), w\left(P\left(i, V'\right)\right)$ | | |
| $\vdots$ | | | | | |
| $n$ | none | $\cdots$ | $Q(n, w)$ | $\cdots$ | $Q\left(n, n \cdot w_{\max}\right)$ |

**סיבוכיות**

כבר ראינו כי יש למעשה לחשב את גודל הטבלה כפול עלות מילוי כל תא.

כל אתחול או עדכון של $Q$ עולה $O\left(1\right)$ פעולות. סך הכל אנחנו מבצעים $O\left(n^2 w_{\max}\right)$ צעדים כאלו . מדובר באלגוריתם פולינומיאלי אם $w_{\max}$ הינו פולינומיאלי ב-$n$.

**אפשרות לשינוי המשקלים**

נבחין כי הרווחנו עוד משהו, שיכול לעזור לנו להשתפר עוד יותר!

נקבע $k$ פרמטר **כלשהוא** שערכו יינתן **בהמשך**. נרצה לעגל את המשקלים לכפולות של $k$ (כלפי מטה). במקרה זה נקבל לכל היותר $\frac{w_{\max}}{k} + 1$ ערכים שונים (מ-0 עד $\frac{W_{\max}}{k}$). במילים אחרות, נגדיר משקל חדש על ידי $W_i' = \left\lfloor \frac{W_i'}{k} \right\rfloor \in \left\{0, 1, 2, \cdots, \left\lfloor \frac{w_{\max}}{k} \right\rfloor\right\}$.

כעת, סיבוכיות הזמן של הרצת האלגוריתם לעיל (שמחשב את $Q$) על הקלט $\left(v, W', V\right)$ הינה $O\left(n^2 \frac{w \max}{k}\right)$ - אם $k$ ממש גדול, אזי הקטנו את גודל הטבלה ושיפרנו את זמן הריצה, אך אמנם אנחנו יכולים לאבד דיוק. כלומר, יש כאן איזשהו trade-off שאנחנו מבצעים (בהמשך נראה מהו ה-$k$ האופטימלי)

לגבי התוצאה, נסמן את הפתרון של האלגוריתם ב-$P \subseteq \{1, \ldots, n\}$ ופתרון אופטימלי ב-$P_{\text{OPT}} \subseteq \{1, \ldots n\}$.

נקבל:

$$w\left(p\right)=\sum_{i\in P}w_i\overset{\substack{\underbrace{\text{עיגול כלפי מטה}}\\ \downarrow}}{\geq}$$

$$k\sum_{i\in P}\left\lfloor\frac{w_i}{k}\right\rfloor\overset{\substack{\underbrace{\text{אופטימליות ביחס למשקלים המעוגלים}}\\ \downarrow}}{\geq}$$

$$k\sum_{i\in P_{OPT}}\left\lfloor\frac{w_i}{k}\right\rfloor>$$

$$>\sum_{i\in P_{OPT}}w_i-nk$$

המעבר האחרון נובע מכך שהעיגול כלפי הוא לפחות $nk$, כי למעשה אנחנו מפסידים $k-1$ על כל איבר, ומספר האיברים הוא לכל היותר $n$.

כפי שאמרנו, יש כאן טרייד אוף של דיוק וסיבוכיות. מהו ה-$k$ **האופטימלי** למציאת הדיוק?

**מציאת $k$**

עבור $\varepsilon>0$ נגדיר $k=\frac{\varepsilon w_{\max}}{n}$ (ברור כי $w_{\max}>0$, אחרת אפשר להשתמש באלגוריתם בלי החלוקה ב-$k$).
סיבוכיות זמן הריצה תהיה $O\left(\frac{n^3}{\varepsilon}\right)$.
האופטימליות של הפתרון:

$$\sum_{i\in P}w_i\overset{\substack{\underbrace{\text{אופטימליות והחסרה}}\\ \downarrow}}{\geq}$$

$$\sum_{i\in OPT}w_i-\varepsilon w_{\max}\overset{\substack{\underbrace{\geq w_{\max}\text{OPT}}\\ \downarrow}}{\geq}$$

$$\left(1-\varepsilon\right)\sum_{i\in OPT}w_i$$

המעבר השני המצוין יכול להשיג לכל הפחות $w_{\max}$, כי הוא פשוט איבד שמשקלו $w_{\max}$.

קיבלנו אלגוריתם שמקבל כקלט $(V,w,V,\varepsilon)$ , מחשב פתרון $P$ שערכו לפחות $1-\varepsilon$ כפול הערך האופטימלי, ורץ בזמן פולינומי ב-$n$ וב-$\frac{1}{\varepsilon}$.

אנחנו אמנם יכולים להתקרב לפיתרון האופטימלי, אבל אם נתקרב, זמן הריצה יעלה ביחס ל-$\frac{1}{\varepsilon}$.

אלגוריתם כזה נקרא סכימת קירוב **פולינומית לחלוטין**.

למעשה, איננו מכירים אלגוריתם שיותר לפתור את הבעיה בזמן פולינומיאלי לגמרי.

נרצה לראות דוגמה יותר פשוטה לתכנון הדינמי.

### 3.2 שיבוץ קטעים ממושקלים

נתונים קטעים על הישר הממשי $I_1, I_2, \ldots I_n$ כך ש- $I_j = (a_i, b_j)$ ומשקלים $w_1, \ldots, w_n$.
אנו רוצים למצוא קבוצת קטעים לא חופפים שמשקלה מקסימלי.

**הרעיון**

נניח כי הקטעים ממויינים לפי זמן הסיום שלהם, כלומר $b_1 \leq \ldots \leq b_n$.
נשים לב כי אם אנו מרשים פתרון שלא חורג מעבר ל-$b_j$, בהכרח מדובר בסידור מהקטעים $\{I_1, \ldots, I_j\}$.
נגדיר כעת את $S_j$ על ידי **שיבוץ של קטעים מ-**$\{I_1, \ldots, I_j\}$ **שמשקלה מקסימלי ללא חפיפה.**
ברור כי המטרה שלנו היא לחשב את $S_n$.
נסמן כעת $j_i = \text{argmax}\{b_j \mid b_j \leq a_i\}$ - זה נקודת הסיום הכי גדולה שאפשר לסדר לפני שמתחיל הקטע ה-$i$.
נוסחת הרקורסיה תהיה:

$$S_1 = \{I_1\}$$

$$S_i = \begin{cases} S_{i-1} & w\left(S_{j_{i-1}}\right) \geq w\left(S_{j_i}\right) + w_i \\ S_{j_i} \cup \{I_i\} & else \end{cases}$$

כלומר, אנחנו בודקים מה משתלם לנו. האם הקטע האחרון ועוד "הדרך" ל-$S_i$ כבד יותר או לא.
בטבלה, זה נראה כך:

| $S_n$ | | $\ldots$ | | $S_1$ |
|---|---|---|---|---|
| $\max\{S_{i-1}, S_{j_i} \cup \{I_i\}\}$ | | | | $I_1$ |

**דוגמת הרצה**

נחלק משקלים: הקטע 1 יקבל משקל 10 , הקטע 2 יקבל משקל 20, הקטע 3 יקבל משקל 5, הקטע 4 יקבל משקל 20 והקטע 5 יקבל משקל 15.
נניח כי הקטע 5 יכול 'ללכת' ל-3, שיכול ללכת ל-1 בלבד והקטע 4 יכול 'ללכת' ל-1 בלבד (מבחינת חפיפות).
כעת נוכל לבנות את הטבלה. התא הראשון יהיה 10 בהכרח, השני יקבל 20 (כי הוא משפר), השלישי יהיה $\max\{10 + 5, 20\} = 20$ והרביעי יהיה $\max\{20 + 10, 20\} = 30$. החמישי יהיה $\max\{20 + 15, 30\} = 35$:

| 10 | 20 | 20 | 30 | 35 |
|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 5 |

נבחין כי 'המסלול' הוא מ-5 ל-2, זה הדרך המקסימלית.

### 3.3 מרחק עריכה

**הרצאה מס' 10:**

**יום ראשון**

25.04.21

<u>הקלט:</u>
שתי מחרוזות $x$ ו-$y$ מעל אל"ף-בי"ת $\Sigma$.
<u>הפלט:</u>
המספר המזערי של פעולות עריכה (הוספת אות, מחיקת אות, החלפת אות באות אחרת) אשר דרושות על מנת להפוך את $x$ ל-$y$.

**דוגמא**

מילה שגויה: ה צ ת ע צ ו ת.

מילה מתוקנת: ה צ ט ע צ ע ו ת.

הדרך שלנו להפוך את המילה השגויה למילה המתוקנת הינו כזה:

ה צ **ת** ע צ ו ת –- ה צ ט ע צ ו ת–-<ה צ ט ע **צע** ו ת.
[14]

כלומר, אנחנו מחפשים האם מילה מסוימת קיימת **במילון** ואם לא, אנחנו מחפשים את המילה שהכי **קרובה** אליה במרחק עריכה.

אחת הדרכים לחשוב על זה היא להציג את הבעיה בגרף מכוון, בו השורות הן המחרוזת המקורית, והעמודות הן מחרוזת היעד. בעקבות כך, "ירידה בגרף" משמעה הוספה , "פנייה ימינה" משמעה מחיקה, ופנייה ימינה ולמטה משמעה החלפה.

באופן מפורש, ניתן לחשוב על זה כך:

$$|x| = m,\ \ |y| = n$$
$$G = (V, E)$$

כך ש- $V = \{0,1,2,\ldots,n\} \times \{0,1,2,\ldots,m\}$ **וגם:**

$$\begin{aligned} &\{((i,j),(i+1,j)):\ 0 \le i < n \land 0 \le j \le m\} \cup \\ E = &\{((i,j),(i,j+1)):\ 0 \le i \le n \land 0 \le j < m\} \cup \quad . \\ &\{((i,j),(i+1,j+1)):\ 0 \le i < n \land 0 \le j < m\} \end{aligned}$$

כאשר המחובר הראשון הינו **ההוספות**, השני הינו **המחיקות**, והשלישי הוא **ההחלפות**.

לקשתות האלו יש משקלים - **הוספה** או **מחיקה** עולה 1. ולקשתות האלכסוניות שמסמלות החלפה - תלוי האם התבצעה החלפה ממשית או לא, כלומר האם הצלעות זהות או לא.

כל מסלול כזה הוא מונוטוני ולכן אורכו לכל היותר $m+n$. כל מסלול מתאים לסדרת פעולות שממירות את $x$ ל-$y$.

האם האלגוריתם של דייקסטרה יעבוד כאן? **וואלה כן**, בסדר גודל של $O\left(m \cdot n \cdot \log\left(mn\right)\right)$. אבל נוכל אפילו לשפר את זה קצת.

נכתוב את האלגורתם בתור פסואודו קוד:

**הרצאה מס' 11:**

**יום שלישי**

27.04.21

---

[14] רק שתדעו מוצאם של בני האדם הוא לא משימפנזה. יש האומרים שיש להם אב משותף. שלא תחשבו שאני אומר משהו שאני לא אומר (י.ר)

**אלגוריתם 2** מרחק עריכה מינימלי

```
1. נחזיק טבלה $D$ כך ש-$D\left(i,j\right)$ הוא מסלול מינימלי מ-$\left(0,0\right)$ ל-$\left(i,j\right)$.
2. לכל $i \leftarrow 0$ עד $i \leftarrow n$ תגדיר $\left(i,0\right) \leftarrow i$.
3. לכל $j \leftarrow 0$ עד $j \leftarrow m$ תגדיר $\left(0,j\right) \leftarrow i$
4. לכל $i \leftarrow 1$ עד $n$:
    (א) לכל $j \leftarrow 1$ עד $m$:
        i. תגדיר $D\left(i,j\right) = \min\left\{D(i-1,j)+1, D(i,j-1)+1, D(i-1,j-1)+[x_i \neq y_j]\right\}$
```

**סיבוכיות**

סיבוכיות הזמן פשוטה - סדר גודל של $O\left(mn\right)$ ולסיבוכיות המקום גם נשמור $O\left(mn\right)$
(אפשר לשפר ל-$O\left(\min\left\{m,n\right\}\right)$ אבל זה מסובך).

**הוכחת נכונות**

נוכיח זאת באינדוקציה על סדר המילוי של איברי $D$, כי לכל $i,j$ מתקיים כי הערך $D\left(i,j\right)$ הוא המשקל המינימלי של מסלול מ-$\left(0,0\right)$ ל-$\left(i,j\right)$.

בסיס האינדוקציה

האתחול מבטיח נכונות עבור הזוגות $i,j$ שבהם $i=0$ או $j=0$.

צעד האינדוקציה

כל מסלול ל-$\left(i,j\right)$ חייב לעבור דרך אחד הצמתים $\left(i-1,j\right)$ או $\left(i,j-1\right)$ או $\left(i-1,j-1\right)$, ולאחר מכן לשלם דרך הקשת הנוספת.

הקטע עד הצומת הקודם חייב להיות מסלול במשקל מינימלי (מהנחת האינדוקציה). האלגוריתם בוחר במינימום מבין שלוש האפשרויות.

**חישוב סדרת פעולות העריכה**

נשמור טבלה נוספת $P$ בגודל $m+1 \times n+1$. הערך $P\left(i,j\right)$ יציין את הקשת האחרונה במסלול $D\left(i,j\right)$ מ-$\left(0,0\right)$ ל-$\left(i,j\right)$.

# 4 רשתות זרימה

## 4.1 שידוך מושלם

**הגדרות**

גרף סופי לא מכוון דו צדדי $G\left(U,V,E\right)$ הוא גרף בו קבוצת הצמתים היא $U\cup V$ (איחוד זר) - כלומר $U\cap V=\emptyset$.

**סימונים**

עבור $U \ni u$ נגדיר $N(u)$ - קבוצת שכני $u$ (ובהכרח $N(u) \subseteq V$). בדומה מתקיים עבור $v \in V$, כי $N(v) \subseteq U$ היא קבוצת שכני $v$.

**הגדרה**

**שידוך ב-$G$** הוא קבוצה $M \subseteq E$ כך שבכל צומת של $G$ נוגעת לכל היותר קשת אחת מ-$M$. שידוך $M$ הוא מושלם אם"ם $|m| = |U|$.

אפשר גם להגדיר זאת כי בכל צומת אחת נוגעת **בדיוק** קשת אחת.

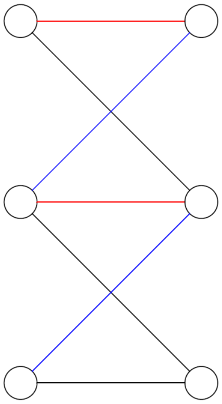
המשפט המרכזי שנתעסק בו בנושא זה הוא משפט hall.

על מנת להוכיח זאת, נצטרך הגדרה שתעזור לנו.

**הגדרה**

בהינתן שידוך כלשהוא $M$, מסלול פשוט שבו הקשתות לסירוגין מ-$E \setminus M$ ו-$M$ נקרא מסלול מתחלף.

דוגמה למסלול מתחלף:



### 4.1.1 משפט החתונה של הול

הרצאה מס' 12:

יום ראשון

02.05.21

ב-$G$ יש שידוך מושלם אם ורק אם לכל $u' \subseteq u$ מתקיים כי $|N(u')| \geq |u'|$ (מספר השכנים של כל קבוצת קודקודים גדול מגודל הקבוצה)

**הוכחה**

כיוון אחד טרוויאלי (בערך)[15]. אם קיים ב-$G$ שידוך מושלם $M$ אזי לכל צומת $u \in U$ קיים $M(u) \in V$ כך ש-$u$ משודך לו - כלומר כל צומת ב-$U$ משודך.

כל ה-$M(u)$-ם הללו שונים זה מזה. כלומר,לכל $u, u' \in U$ עבורם $u \neq u'$ מתקיים כי $M(u) \neq M(u')$.

כעת, תהי $U' \subseteq U$ . נקבל - $\{M(u) : u \in U'\} \subseteq N(U')$ ואז נקבל:

---

[15] אפשר גם להגיד כך. זיווג מושלם של קבוצה אחת לשנייה מקביל להתאמה חח"ע ועל. לא אפשרי לעשות התאמה כזאת אם קיימת קבוצה שמספר איבריה בתמונתה קטן ממספר איבריה.

$$|N(U')| \geqslant |\{M(u) : u \in U'\}| = |U'|$$

כלומר, ברור כי אם כל צמתי $U'$ משודכים, מספר השכנים הוא לפחות כמו גודל השידוך.

בכיוון השני, נוכיח בשלילה.

### ההנחה בשלילה

כלומר, נניח כי התנאי מתקיים ($|N(u')| \geq |u'|$ לכל $u' \in u$) אך עבור $M$ שהינו שידוך גדול ביותר (אין שידוך אחר, לאו דווקא הרחבה שלו, גדול יותר), $M$ איננו שידוך מושלם.
לפי ההנחה בשלילה, אנחנו מקבלים כי $|M| < |U|$(כי הוא איננו שידוך מושלם). בפרט, נניח כי יש $u \in U$ **שאיננו משודך**.

### מציאת קבוצות לשידוכים

כעת, נתבונן **באוסף המסלולים המתחלפים** ש-$u$ קצה אחד שלהם - $u$ אינו משודך ולכן ישנן שתי אפשרויות: המסלול שלו נגמר בצלע בשידוך, או שהמסלול שלו נגמר בצלע שלא בשידוך (צלע אדומה, או צלע כחולה).
כעת, נסמן ב-$U'$ את קבוצת הצמתים ב-$U$ (צד שמאל) שמשתתפים במסלולים המתחלפים הללו (בפרט $u \in U'$).
גם נסמן ב-$V'$ את קבוצת הצמתים ב-$V$ (צד ימין) שמשתתפים במסלולים הללו.[16]
לפי הנחת המקסימליות של $M$, לא ייתכן שיש **מסלול מתחלף באורך אי זוגי שלא ניתן להאריך אותו**[17].
מכאן נובע באופן ישיר כי כל מסלול מתחלף מ-$u \in U$ (שלהזכירכם לא משודך) מסתיים ב-$U'$ (צד שמאל), או מסתיים בצומת משודך ב-$V'$ (כלומר, מסתיים בצלע כחולה).
מכאן ניתן להגיע למסקנה כי **כל הצמתים ב-$V'$ משודכים לצמתים ב-$U'$**. למה? אם כל מסלול מתחלף נגמר ב-$U'$ או בצומת משודך ב-$V'$ אזי ישנן שתי אפשרויות: כל הצמתים ב-$V'$ משודכים, או שהמסלול נגמר בצד ימין ונוכל לשדך אותו חזרה לשמאל.
כמו כן, כל הצמתים ב-$U' \setminus \{u\}$ **משודכים לצמתים ב-$V'$**- דבר זה נובע מהגדרת מסלול מתחלף. $u$ לא משודך מעצם הגדרתו, אבל כל צומת אחר ב-$U'$ נמצא באוסף וחייב להיות משודך.
בעקבות כך, נקבל כי $|U' \setminus \{u\}| = |V'|$, כלומר $|U'| > |V'|$.

### הסתירה

יהי $x \in U'$. אם קיים $y \notin V'$ עבורו $(x, y) \in E$, אזי יש מסלול מתחלף שמגיע מ-$u$ ל-$y$, כי יש מסלול מתחלף שמגיע מ-$u$ ל-$x$ ($x \in U'$) ואפשר להאריך אותו עם $x, y$.

---

[16] תזכורת לשם ההבהרה:

$U$ צד שמאל, $V$ צד ימין, $U'$ אוסף הצמתים מ-$U$ שמשתתפים במסלול מתחלף מ-$u$, $V'$ אוסף הצמתים מ-$V$ שמשתתפים במסלול מתחלף מ-$u$.

$M$ שידוך שהנחנו בשלילה שלא מושלם.

מסלולים מתחלפים - מסלולים שאחד בשידוך ואחד לא, בהתאמה.

[17] למה? כי אם נניח בשלילה שיש מסלול כזה, שלא ניתן להאריך אותו, אזי מצאנו מסלול באורך אי זוגי, שקצוותיו אינם משודכים. נוכל להגדיל את $M$ על ידי זריקת הכחולות והכנסת האדומות לשידוך (הרי אם יש מסלול מתחלף כשהקצוות שלו לא משודכים, משמע שיש יותר אדומים מכחולים). כלומר למצוא מסלול אחר, בסתירה למקסימליות.

דוגמה לכך:

מסלול כזה מסתיים בקשת בשידוך. הקשת $(x,y) \notin M$ כי הנחנו ש-$y \notin V'$ אבל אז אפשר להאריך את המסלול בעזרת הקשת $(x,y)$ וזו סתירה לבנייה של $V'$, כי הרי מצאנו קודקוד ב-$V$ שנמצא במסלולים המתחלפים ולא נמצא ב-$V'$. משמע, אין כזה.

בפרט קיבלנו כי $N\left(U'\right) \subseteq V'$. כלומר, קיבלנו כי הקבוצה $U'$ מפרה את התנאי של משפט הול.

כי $\left|N\left(U'\right)\right| \leqslant |V'| < |U'|$ - זו סתירה, לכן ההנחה **שלא מדובר בשידוך מושלם - שגויה.**

המשפט עצמו לא מהווה אלגוריתם, כיוון שהוא לא עוסק באילו קשתות מדובר, אלא רק בעובדה שיש כאלו. אמנם, משפט זה כן מרמז על קיומו של אלגוריתם כזה.

האלגוריתם מקבל כקלט **גרף דו צדדי** ומחזיר שידוך מושלם או קבוצה $U' \subseteq U$ עבורה $\left|N\left(U'\right)\right| < |U'|$. הרעיון של האלגוריתם הינו כזה:

- נתחיל משידוך $M$ ריק. כלומר $M \leftarrow \emptyset$.
- בכל איטרציה נגדיל את $M$ באחד.
- אם אין $u \in U$ שאינו משודך, $M$ שידוך מושלם.
- אם יש $u \in U$ שאינו משודך, נמצא את $U'$ ו-$V'$ באמצעות פרוצדורה דמויית BFS, שהרי אנחנו צריכים למצוא מסלולים שהם לסירוגין בשידוך או לא בשידוך.

נוכל להבחין כי $G$ גרף **דו צדדי** ולכן **אין קשת** בין שני צמתים באותה שכבה.

$u$ נמצא בשכבה $0$. אם מגיעים לצומת $y$ בשכבה אי זוגית (נמצאת בצד ימין) ואין המשך לשכבה הבאה, אז מצאנו מסלול מתחלף מקסימלי מ-$u$ ל-$y$ וזה מאפשר הגדלת $M$ ב-$1$ (כלומר, נחליף בין הכחול לאדום כי יש יותר אדום).

אם אין $y$ כזה: מתקיים כי יש צומת לא משודכת $u$ ואין צומת לא מתחלפת באורך אי זוגי אזי הצמתים בשכבות האי זוגיות הם כל השכנים של הצמתים בשכבות הזוגיות. בעקבות כך, מספר הצמתים בשכבות הזוגיות גדול ממספר הצמתים בשכבות האי זוגיות. הצמתים בשכבות הזוגיות הם קבוצה $U'$ שמפרה את התנאי של משפט הול.

**סיבוכיות**

סיבוכיות הזמן

נסמן ב-$n$ את מספר הצמתים וב-$m$ את מספר הקשתות.

מבצעים לכל היותר $|U|$ איטרציות של הגדלת $M$. אנו יודעים כי $|U| \leq n$. כל איטרציה זו הרצה של פרוצדורה דמויית BFS[18] שעולה $O\left(m+n\right)$. אם כך, סיבוכיות הזמן היא $O\left(mn+n^2\right)$.

סיבוכיות המקום

בנוסף לקלט, מדובר בסדר גודל של $O\left(m+n\right)$ תאי זיכרון.

### 4.2 זרימה ברשתות

**הרצאה מס' 13:**

**יום שלישי**

04.05.21

**הגדרה**

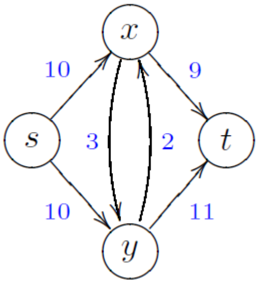
רשת זרימה היא רביעייה $(G,c,s,t)$ ובה:

$G=(V,A)$ גרף סופי מכוון, $c: E\left(G\right) \to \mathbb{N}$ פונקציית קיבול חיובית, $s,t \in V\left(G\right)$ הנקראים מקור $(s)$ ובור $(t)$.

דוגמה:

---

[18] יש פסאודו קוד במודל.



**הגדרה**

**זרימה ברשת** $(G,c,s,t)$ היא פונקציה $f:A\to\mathbb{R}$ שמקיימת:

$(I)$ שימור הזרימה - בכל קודקוד $v\in V\setminus\{s,t\}$ מתקיים כי $\sum_{a=(u,v)\in A} f(a) = \sum_{a=(v,u)\in A} f(a)$ - הזרימה הנכנסת לקודקוד שווה לזרימה היוצאת ממנו, מלבד המקור והבור.

$(I)$ שימור אילוצי הקיבול - לכל $a\in A$ מתקיים כי $0\le f(a)\le c(a)$ - הזרימה בצלע אינה גדולה מהקיבול באותו הצלע.

בהינתן זרימה $f$ ברשת $(G,c,s,t)$ , הערך של $f$ שנסמן $|f|$ הוא:

$$|f| = \sum_{a=(s,u)\in A} f(a) - \sum_{a\in(u,s)\in a} f(a)$$

כלומר, מדובר על ההפרש בין **הזרימה שנכנסת** ל-$s$ ובין **הזרימה שיוצאת** מ-$s$.

<u>מסקנה</u>

ניתן להגדיר את $|f|$ גם בצורה הבאה:

$$|f| = \sum_{a=(u,t)\in A} f(a) - \sum_{a\in(t,u)\in a} f(a)$$

כלומר, למעשה הזרימה שיוצאת **מהמקור** ומגיעה נטו **לבור**.

עבור קבוצת קשתות $A\supseteq B$ נגדיר $C(B)=\sum_{a\in B} c(a)$.

כעת, נרצה להגדיר הגדרה נוספת.

**הגדרה**

קבוצה $B\subseteq A$ תקרא חתך $s-t$[א] אם״ם קיימת קבוצת צמתים $\emptyset\neq S\subseteq V\setminus\{t\}$ כך ש-$s\in S$ עבורה $B$ היא קבוצת הקשתות שיוצאות מ-$S$.

---

[א] היזכרו בהגדרת חתך כפי שראינו אותה קודם לכן.

נסמן ב-$B'$ את קבוצת הקשתות שנכנסות ל-$S$.

**טענה**

לכל חתך $s-t$ $B,$ מתקיים כי $|f| = \sum\limits_{a\in B} f(a) - \sum\limits_{a\in B'} f(x)$ - כאשר מדובר כאן בזרימה על גבי החתך בלבד.

**טענה**

לכל חתך $s-t$ $B,$ ולכל זרימה $f$ ברשת מתקיים כי $|f| \leq c(B)$

**הוכחה**

$$|f| \overset{\underbrace{\text{טענה קודמת}}_{\downarrow}}{\leq} \sum_{a\in B} f(a) \overset{\underbrace{\text{אילוצי הקיבול}}_{\downarrow}}{\leq} \sum_{a\in B} c(a) \overset{\underbrace{\text{הגדרה}}_{\downarrow}}{=} c(B)$$

### 4.2.1 רשת שיורית

נתונה רשת זרימה $N=(G.c,s,t)$. נתונה זרימה $f$ ב-$N$.

אנו רוצים להגדיר רשת $N_f=(G_f,c_f,s,t)$ שמתארת את "המקום" שנותר ברשת להעביר עוד בזרימה.

**ניסיון** $1$:

נגדיר $G_f=G$ ו-$c_f(a)=c(a)-f(a)$ $\forall a\in A$ (מבחינה אינטואיטיבית - המקום שנשאר לאחר הזרימה).

אמנם, ניסיון זה איננו מוצלח. למה? נוכל עם הזרימה האחת - הלא מקסימלית שלנו - לחסום את כל המסלולים האפשריים בגרף, למרות ש-$f$ אינה זרימה מקסימלית.

כלומר, למעשה אנחנו מבטלים צלעות מסוימת ואין לנו יכולות לעבור בהן שוב.

**ניסיון** $2$:

עבור כל קשת שעברה בזרימה, נוסיף קשת בכיוון ההפוך שתאפשר לנו לבטל את הזרימה שעברה.

אחרי שהסברנו את האינטואיציה, ננסה לראות את ההגדרה הפורמלית.

**הגדרה**

**הרשת השיורית** $N_f = (G_f, c_f, s, t)$ מקיימת $V(G_f) = V(G)$ וגם:

$$E(G_f) = \{(u,v) : f(u,v) < c(u,v)\} U \{(v,u) : f(u,v) > 0\}$$

כלומר, **הזרימה בכל קשת קטנה ממש מהקיבול** וגם כי הזרימה בכיוון ההפוך גדולה מ-0 - זאת אחרי שהגדרנו כביכול צלעות בכייוון ההפוך.

ממילא, נקבל מההגדרות:

$$\begin{aligned} &c_f = c(u,v) - f(u,v) > 0 \\ &(f(u,v) < c(u,v)) \end{aligned}$$

וגם:

$$\begin{aligned} &c_f(v,u) = f(u,v) \\ &(f(u,v) > 0) \end{aligned}$$

**טענה**

אם $f$ זרימה חוקית ב-$N$ ו-$f'$ זרימה חוקית ב-$N_f$ אזי $f + f'$ היא זרימה חוקית ב-$N$ המוגדרת על ידי:

$$(f + f')(u,v) = f(u,v) + f'(u,v) - f'(v,u)$$

**הוכחה**

עלינו להראות כי הזרימה המוגדרת מקיימת את אילוצי הקיבול.

נבחין כי:

$$f'(v,u) \leq c_f(v,u) = f(u,v)$$

מכאן נובע כי:

$$(f + f')(u,v) > 0$$

כעת, נקבל:

$$\begin{aligned}&\left(f+f'\right)(u,v)=f(u,v)+f'(u,v)-f'(v,u)\leq\\&f(u,v)+c_f(u,v)=\\&f(u,v)+c(u,v)-f(u,v)=c(u,v)\end{aligned}$$

אם כך, הראינו כי הזרימה החדשה מקיימת את אילוצי הקיבול.
אילוצי שימור הזרימה נובעים באופן ישיר ומושארים כתרגיל.

**טענה**

$$|f+f'|=|f|+|f'|$$

כעת, אנחנו מחפשים זרימת מקסימום $f_{\max}$ ב-$N$ (כלומר זרימה שערכה $|f_{\max}|$ הוא מירבי).
הרעיון הטבעי מתוך מה שתארנו עד כה הוא כזה.

### 4.2.2 שיטת Ford-Fulkerson

אתחול:

נתחיל מזרימה $f$ שערכה 0 על כל קשת

בכל איטרציה:

- נמצא ברשת השיורית $N_f$ מסלול מכוון $P$ מ-$s$ ל-$t$ - $P$ נקרא מסלול משפר.[19]
- נגדיר: $$f'(u,v)=\begin{cases}\min_{e\in P}C_f(e) & (u,v)\in P\\ 0 & else\end{cases}$$

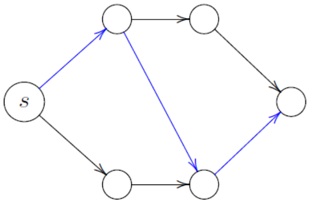
כלומר, למעשה עבור קשת במסלול, הערך של $f'$ הוא הקיבול השיורי המינימלי של איזושהי קשת במסלול $P$.

- נחליף את $f$ ב-$f+f'$.

סיום:

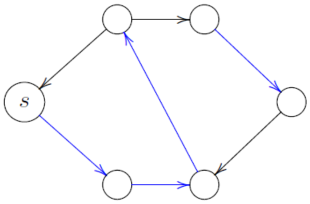
ב-$N_f$ אין מסלול מכוון מ-$s$ ל-$t$ (אם אין מסלול כזה, אין דרך להעביר מסלול כזה).

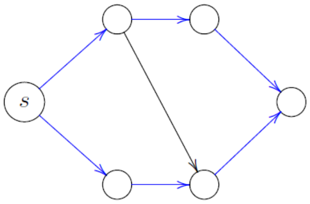
**דוגמת הרצה**



---

[19] (בהמשך נראה כיצד למצוא את המסלול, באמצעות BFS למשל)





מבחינה רעיונית, אנחנו דוחפים זרימה מ-$s$ ל-$t$ על גבי מסלול אחד. רשת זו מוגבלת על ידי צוואר הבקבוק של המסלול, שהיא הקשת עם הקיבול המינימלי $c_f$.

**משפט**

בסיום האיטרציות (שבהכרח הינן מספר סופי) $f$ היא זרימת מקסימום.

**הרצאה מס' 14:**

**יום ראשון**

09.05.21

**הוכחה**

נסמן ב-$S$ את קבוצת הצמתים שיש ב-$N_f$ (הרשת השיורית עם הזרימה בסוף) מסלול מ-$s$ אליהם. בפרט, ברור כי $s \in S$ (כי זהו המסלול הריק), וכי $t \notin S$ (זהו כלל העצירה).

אם כך, מתקיים כי $B = \{(u,v) \in A(G) : u \in S \wedge v \notin S\}$ הוא חתך $s-t$ (לפי ההגדרה). כלומר, סך הכל מדובר על אוסף הקשתות ב-$G$ **שיוצאות מ-$S$ לצומת שאיננה ב-$S$.**

נסמן ב-$B' = \{(v,u) \in A(s) : u \in S \wedge v \notin S\}$ - כלומר, את אוסף הקשתות שנכנסות ל-$S$.

מתקיים כי $B \subseteq A(G)$ (כלומר חלק מצלעות הגרף) אבל $B \cap A(G_f) = \emptyset$, כי אם נניח בשלילה שלא, אזי תהי $(u,v) \in B \cap A(G_f)$, נקבל כי $u \in S$ כלומר, מצאנו **מסלול** ב-$N_f$ מ-$s$ ל-$u$, כעת, נוסיף את הצלע $(u,v)$ ונקבל מסלול מ-$s$ ל-$v$ ב-$N_f$. מדובר **בסתירה** להנחה כי $v \notin S$, כלומר כי אין מסלול מ-$s$ ל-$v$ ב-$N_f$ .

אם כך, קיבלנו כי אין קשתות שהן גם ב-$B$ וגם ברשת השיורית. כיצד הדבר ייתכן? רק בצירוף שני הדברים הבאים:

- לכל $(u,v) \in B$ נקבל כי $f(u,v) = c(u,v)$ - (מדוע? ניזכר כי $B$ היא אוסף הקשתות שיוצאות למעשה מהרשת השיורית לצומת לא ב-$S$. כלומר $(u,v) \notin N_f$ - אם מתקיים כי $f(u,v) < c(u,v)$, אזי $(u,v)$ הייתה ברשת השיורית, בסתירה).

- לכל $(v,u) \in B'$ מתקיים כי $f(v,u) = 0$ (אחרת, ב-$N_f$ מופיעה קשת $(u,v)$ - כלומר הזרימה ההפוכה גדולה מ-0, אזי מופיעה ברשת השיורית קשת $(u,v)$ מצומת ב-$S$ לצומת לא ב-$S$, ולפי ההנחה **אין כזאת).**

לכן, נקבל סך הכל:

$$|f| = \sum_{(u,v)\in B} f(u,v) - \sum_{(v,u)\in B'} f(v,u) =$$

$$\sum_{(u,v)\in B} c(u,v) - 0 = c(B) \geqslant |g|$$

לכל זרימה חוקית $g$. בפרט, דבר זה נכון עבור זרימת מקסימום - כלומר $f$ היא זרימת מקסימום.

**מסקנות:**

1. $B$ הוא חתך בעל קיבול מינימלי.
2. זרימת מקסימום $=$ קיבול חתך $s-t$ מינימלי.

**הגדרה**

זרימה $f$ נקראת זרימה בשלמים, אם הערך שלה על כל קשת הוא מספר שלם.

**טענה**

אם הקיבולים שלמים, בכל שלב בריצה של שיטת FF הזרימה היא בשלמים.

**הוכחה**

אנחנו מתחילים עם זרימה בשלמים, ולכן גם הרשת השיורית היא מספר שלם, ואז נגדיל מספר זה במספרים שלמים.
(פורמלית - באינדוקציה על מספר האיטרציות).

**מסקנה**

בכל איטרציה הזרימה גדלה ב-1 לפחות.

**כמות האיטרציות**

כמה איטרציות עלינו לבצע? נניח כי הקיבולים ברשת הקלט $N$ הם מספרים שלמים. בעקבות כך, נקבל כי הזרימה $f$ היא זרימה בשלמים. אם כך, בעקבות המסקנה הקודמת, אם נסמן את **זרימת המקסימום** ב-$f^*$, אזי מספר האיטרציות הוא לכל היותר $|f^*|$.

**הרצאה מס' 15:**

**יום שלישי**

11.05.21

**סיבוכיות**

ניתן לראות כי השיטה לא פולינומית בגודל הקלט.
מדוע? אם נניח כי גודל הקלט הוא $O\left(k\right)$ ונבחר את הגרף הסטנדרטי כשכל הקשתות בגודל $2^k$ והקשת המחברת בגודל 1. אזי בכל פעם נגדיל את הזרימה ב-1 דרך האמצע (וההפוך שלו), וכך נמשיך $2^k$ איטרציות - זרימת המקסימום תהיה $2\cdot 2^k$ וזה גם מספר האיטרציות. נראה הסבר לכך בהמשך.
אבל מסתבר שאפשר למצוא אלגוריתם יעיל יותר, האלגוריתם של Edmonds-karp.

### 4.2.3 האלגוריתם של Edmonds-karp

מדובר למעשה במימוש של FF (בדרך מציאת המסלולים).
ברשת השיורית, נמצא **מסלול משפר שמספר הקשתות בו מינימלי.** אפשר ליישם זאת באמצעות BFS (בזמן $O\left(n+m\right)$ כאשר $m$ מספר הקשתות ו-$n$ מספר הצמתים - ברשת השיורית יש לכל היותר $2m$ קשתות).

**משפט**

נדרשות לכל היותר $O\left(mn\right)$ איטרציות.

**הוכחה**

עבור גרף $H$ נסמן ב-$d_H$ את פונקציית המרחק בין קודקודי $H$.
כלומר, נקבל כי $d_H\left(u,v\right)$ שווה למספר הקשתות **במסלול קצר ביותר** מ-$u$ ל-$v$.
נתבונן **בזרימה כלשהי** $f_0$ שחושבה במהלך ריצת האלגוריתם.

כעת, נגדיר $f_1 = f_0 + f'_0$ (כלומר, זו הזרימה לאחר השיפור בעזרת $N_{f_0}$).

יהי $v \in V$. נוכיח באינדוקציה על $d_{G_{f_1}}(s,v)$ כי $d_{G_{f_0}}(s,v) \le d_{G_{f_1}}(s,v)$ - כלומר, **המרחק של $v$ מהמקור** קטן או שווה למרחק שלו מהמקור לאחר השיפור.

בסיס האינדוקציה

$d_{G_{f_1}}(s,v) = 0$. כלומר, $v = s$.

במקרה זה, נקבל $d_{G_{f_0}}(s,s) = 0 = d_{G_{f_1}}(s,s)$ ולכן הטענה מתקיימת עבור בסיס האינדוקציה.

צעד האינדוקציה

נניח כי $d_{G_{f_0}}(s,u) \le d_{G_{f_1}}(s,u)$ .

נתבונן **במסלול קצר** ביותר $P$ מ-$s$ ל-$v$ בגרף $G_{f_1}$. מספר הקשתות בגרף זה הוא בדיוק $d_{G_{f_1}}(s,v)$.

תהי $(u,v)$ הקשת האחרונה במסלול זה.

כעת, ישנן שתי אפשרויות:

- אם $(u,v) \in E(G_{f_0})$ אזי:

$$\begin{aligned}&d_{G_{f_0}}(s,v) \le d_{G_{f_0}}(s,u) + 1 \overset{\substack{\underbrace{\text{הנחת האינדוקציה}}\\ \downarrow}}{\le}\\ &d_{G_{f_1}}(s,u) + 1 = d_{G_{f_1}}(s,v)\end{aligned}$$

כאשר המעבר הראשון נובע מכך ש-$(u,v) \in E(G_{f_0})$ - ניתן להגיע ל-$v$ באמצעות הגעה ל-$u$ והוספה של 1. מצד שני, ייתכן כי אפשר להגיע ל-$v$ ב-$G_{f_0}$ בצורה אחרת, ולכן יש א"ש.

- אם $(u,v) \notin E(G_{f_0})$ אזי בהכרח $(v,u) \in E(G_{f_0})$ (למה? הקשת $(u,v) \in G_{f_1}$ ולכן חייבים להוסיף זרימה בכיוון ההפוך ב-$G_{f_0}$. קשת מופיעה ב-$G_{f_1}$ אם ורק אם הייתה זרימה בכיוון ההפוך). כמו כן, המסלול המשפר ב-$G_{f_0}$ עובר דרך $(u,v)$ ולכן $d_{G_{f_0}}(s,v) = d_{G_{f_0}}(s,u) - 1$ זהו מסלול קצר ביותר, ולכן נקבל:

$$\begin{aligned}&d_{G_{f_0}}(s,v) =\\ &d_{G_{f_0}}(s,u) - 1 \overset{\substack{\underbrace{\text{הנחת האינדוקציה}}\\ \downarrow}}{\le}\\ &d_{G_{f_1}}(s,u) - 1 \le d_{G_{f_1}}(s,v) - 1 - 1 =\\ &d_{G_{f_1}}(s,v) - 2\\ &\le d_{G_{f_1}}(s,v)\end{aligned}$$

מעבר לכך! נוכל להסיק כי אם הקשת $(u,v)$ הייתה ברשת שיורית כלשהי במהלך ריצת האלגוריתם ואז נעלמה אז $d(s,v)$ גדל ב-2 לפחות.[20]

---

[20] כלומר, בין שתי הופעות עוקבות של הקשת $(u,v)$ שבראשונה היא נעלמה, $d(s,v)$ גדל ב-2 לפחות.

הרעיון הינו כזה: לכאורה, אם קשתות נעלמות מהרשת השיורית, אזי אפשר לחסום את מספר האיטרציות עם מספר הקשתות. אממה, קשתות יכולות לחזור ולהופיע, אך כדי שזה יקרה עבור קשת $(u, v)$, צריך לחזור ולהעביר זרימה דרך $(v, u)$ בשלב כלשהוא. **הראינו באמצעות טענת האינדוקציה כי המרחק בשלב זה גדל בלפחות 2!**

נבחין כי $1 \leq d(s, v) \leq n - 1$ (כל עוד אפשר להגיע מ-$d$ ל-$u$ והקשת $(u, v)$ קיימת. לכן מספר הפעמים שהקשת $(u, v)$ יכולה להיות צוואר בקבוק לשיפור הזרימה הוא לכל היותר $\frac{n}{2}$ - דבר זה נובע מכך שהמרחק צריך לגדול ב-2 בכל פעם כפי שראינו קודם לכן. בכל איטרצה יש לפחות קשת אחת שהיא צוואר בקבוק לשיפור הזרימה, כלומר קשת שנעלמת.

לכן מספר האיטרציות הוא $2m \cdot \frac{n}{2} = mn$.

**מסקנה**

האלגוריתם של EK מוצא זרימת מקסימום בזמן $O(m^2 n)$ פעולות אריתמטיות (בהנחה כי $m$ הוא לפחות $n - 1$ שזה נובע מההנחה שהרשת קשירה)

**סיבוכיות המקום**

$O(m + n)$ תאי זיכרון.

### 4.2.4 שידוך מקסימום בגרף דו צדדי

נרצה להראות כי בעייה זו היא מקרה פרטי של בעיית הזרימה. נבצע רדוקציה מבעיית שידוך מקסימום לבעיית הזרימה.

נמצא ברשת שנוצרה זרימת מקסימום בשלמים. הזרימה על גבי הקשתות השחורות היא בהכרח 0 או 1 (על כל קשת) ואוסף הקשתות שיש עליהן זרימה של 1 הוא בהכרח שידוך. גודל הזרימה הוא גודל השידוך.

# 5 כיסוי בצמתים ותכנון לינארי

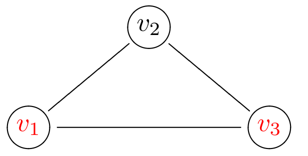
## 5.1 כיסוי בצמתים

**הגדרה**

נתון גרף לא מכוון $G = (V, E)$. קבוצה $X \subseteq V$ תיקרא כיסוי בצמתים של $G$ אם"ם לכל $e \in E$ מתקיים כי $e \cap X \neq \emptyset$.

אינטואיטיבית, מדובר בקבוצת צמתים שמכסה את כל הקשתות - כל קשת בגרף נוגעת פעם אחת לפחות בצומת.[21]

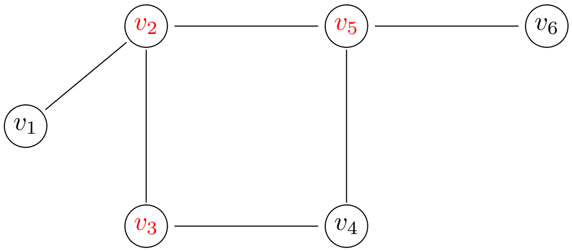
דוגמאות לכיסוי בצמתים:



או למשל:

---

[21] למה זה מעניין אותנו? כמובן, אפשר להשתמש בבעייה זו בכל מיני אלגוריתמים שימושיים.



**הגדרה**
קבוצת צמתים $I$ היא בלתי תלויה אם לכל $e\in E$ נקבל כי $|e\cap I|\le 1$.

אינטואיטיבית, הכוונה היא שאין קשת שמחברת שני צמתים בקבוצה.

**מסקנה**
$V\setminus X$ היא קבוצה בלתי תלויה ב-$G$.

מכאן נובע כי הבעיה של מציאת כיסוי בצמתים בעלת גודל מינימלי שקולה לבעיה של מציאת קבוצה בלתי תלויה בעלת גודל מקסימלי (הבעייה המשלימה)
יש להבחין כי מדובר בבעיית NP קשה, כמובן - וגם הבעייה המשלימה היא כזאת.

נניח בתחילה כי $G$ הוא גרף דו צדדי, כך שמתקיים $G=(V_L,V_R,E)$. ו-$e\in E$ כך ש- $|e\cap V_L|=|e\cap V_R|=1$.

**הרצאה מס' 16:**
**יום שלישי**
18.05.21

## 5.2 האלגוריתם של konig

יהי $M\subseteq E$ שידוך מקסימום ב-$G$.
תהי $Z$ קבוצת הצמתים הבאה:

- ☐ $Z$ מכילה את כל הצמתים ב-$V_L$ שאינם משודכים (כל הצמתים שאינם משודכים בצד שמאל).
- ☐ $Z$ מכילה את כל הצמתים שניתן להגיע אליהם מצומת לא משודך ב-$V_L$ (תכל'ס - הקבוצה הראשונה) דרך מסלול מתחלף[22]. ניזכר כי במקרה זה מספר הקשתות חייב להיות זוגי (אחרת ניתן להגדיל את השידוך[23]), וכמו כן המסלול חייב להסתיים בקשת בשידוך.

כמו כן, נגדיר קבוצה $S$ בצורה הבאה: $S=(V_L\setminus Z)\cup(V_R\cap Z)$ - כל מי שבצד שמאל ולא ב-$Z$ וכל מי שבצד ימין וכן ב-$Z$.

**טענה**
הקבוצה $S$ היא כיסוי בצמתים שגודלו מינימלי.
**הוכחה**

---

[22] מסלול מתחלף למי שלא זוכר - מסלול של קשתות כחולות ואדומות מתחלפות כאשר קשת כחולה הינה בשידוך ואדומה לא בשידוך.
[23] לשם ההרחבה, ראו בהוכחה של משפט הול.

#### הוכחת כיסוי בצמתים

תחילה, נוכיח שמדובר בכיסוי בצמתים - כלומר, כי לכל $e \in E$ מתקיים כי $e \cap X \neq \emptyset$.

תהי $e = \{x, y\} \in E$ קשת כלשהיא. נניח בה"כ כי $x \in V_L$ ו-$y \in V_R$. כאן, תיתכנה מספר אפשרויות:

1. $x \notin Z$ ולכן $x \in S$, כי $x \in V_L$ ו-$V_L \setminus Z \subseteq S$ ולכן הקשת מכוסה כי היא נוגעת בקודקוד מ-$S$.

2. $x \in Z$, נרצה להראות כי $y \in Z$ ואז $y \in S$ - כלומר הקשת תהיה מכוסה בצד ימין. יש שתי אפשרויות:

   (א) $x$ לא משודך, כלומר $e \notin M$ וכיוון ש-$x \in Z$ ו-$x \in V_L$ אפשר להגיע ל-$y$ דרך מסלול מתחלף. כלומר $y \in Z$ ולכן $y \in S$ כי $V_R \cap Z \subseteq S$.

   (ב) $x$ משודך, כלומר $e \in M$. כיוון שהנחנו כי $x \in Z$ ו-$x$ משודך, אזי בפרט הגענו ל-$x$ דרך מסלול מתחלף מצומת לא משודך ב-$V_L$. אמנם, כפי שראינו גם קודם וגם במשפט הול, המעבר לקשת בצד שמאל הינו דרך קשת בשידוך, ולכן בפרט $\{y, x\} \in M$ בעקבות כך נקבל כי $y \in Z$ (מסתיים בקשת בשידוך) ולכן בפרט $y \in S$ כי $V_R \cap Z \subseteq S$.

אם כך, $S$ הוא בוודאי כיסוי בצמתים.

#### הוכחת מינימליות

נבחין כי כל כיסוי בצמתים, גודלו לפחות $|M|$ - מדוע? עבור כל קשת $e \in M$, לפחות קצה אחד של $e$ בכיסוי, וכל הקצוות הללו שונים זה מזה.
בפרט, נקבל כי $|S| \geq |M|$.
כעת, נוכיח כי $|S| \leq |M|$ ובזה נראה כי $S$ כיסוי בצמתים מינימלי בגודלו ובפרט שווה ל-$|M|$.
ראשית, כל צומת ב-$V_L \setminus Z$ חייב להיות משודך! מדוע? כיוון ש-$Z$ מכילה את **כל הצמתים הלא משודכים** ב-$V_L$.
ניזכר כי $S = (V_L \setminus Z) \cup (V_R \cap Z)$.
ניקח $x \in V_L \setminus Z$. כעת, תהי $y \in V_R$ הצומת המשודכת ל-$x$ ב-$M$. נניח בשלילה כי $y \in Z$ (אנחנו רוצים להראות כי לכל צומת בשידוך יש בדיוק את מספר השידוכים).
אמנם, לפי ההגדרה, כיוון ש-$y \in Z$ ו-$Z$ מכיל את כל המסלולים המתחלפים מ-$V_L$, יש מסלול מתחלף מצומת לא משודך ב-$V_L$ ל-$y$. בתוספת הקשת $\{x, y\}$ נוכל לקבל מסלול מתחלף, ולכן נקבל כי $x \in Z$ , בניגוד להנחה - לכן בפרט $y \notin Z$.
קיצור- מי שבצד שמאל וב-$S$ משודך, אך לא לצומת ב-$S$.
בנוסף, כל $y \in V_R \cap Z$ נמצא בשידוך $M$. אם נניח בשלילה שלא, נקבל כי המסלול המתחלף מצומת לא משודך ב-$V_L$ ל-$y$ (שקיים בהכרח) הוא מסלול משפר שידוך (כלומר, יש מסלול מתחלף מקסימלי שמתחיל ומסתיים בצומת לא משודך, ואפשר להגדיל את $M$, בדומה למשפט הול), בסתירה להנחה כי $M$ הוא שידוך מקסימום.
למעשה, כל צומת ב-$S$ נתן לקשר לשידוך ולכן $|S| \leq |M|$ כנדרש.

אם כך, קיבלנו כי גודל כיסוי מינימום בצמתים שווה לגודל של שידוך מקסימום (בגרף דו צדדי).

### 5.2.1 האלגוריתם וסיבוכיות

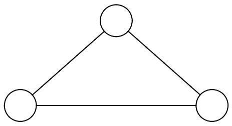
1. נחשב שידוך מקסימום $M$. אנחנו יודעים אלגוריתם שיודע לעשות זאת בזמן $O(mn)$, כאשר $m$ הוא מספר הקשתות ו-$n$ הוא מספר הצמתים (ידועים אלגוריתמים יעילים יותר).

2. נמצא את $Z$ (פרוצדורה דמויית BFS) - פעולה שלוקחת $O(m+n)$

3. חישוב $S$ - דבר שלוקח $O(n)$ פעולות.

## 5.3 גרף כללי

מה קורה במקרה של גרף כללי שאיננו גרף דו צדדי? הבסיס לאלגוריתם הקודם לא יהיה קיים כאן. אם ניקח למשל דוגמה של משולש:



גודל שידוך המקסימום (שבכל קודקוד נוגעת צלע אחת) הינו $1$, וגודל כיסוי המינימום הוא $2$. כלומר, ישנו פער בין כיסוי המקסימום ובין כיסוי המינימום.
אנחנו מכירים אלגוריתמים פולינומיים למצוא שידוך מקסימום בגרף כללי, אך הם אינם פשוטים. קל למצוא שידוך שלא ניתן להגדלה:

- נבחר קשת.
- נסיר את כל הקשתות שיש להן קודקוד משותף איתן, עד שלא נותרות קשתות בגרף.

למעשה, אנחנו לא מכירים אלגוריתם פולינומי למציאת כיסוי מינימום.

**טענה**
יהי $M$ שידוך שלא ניתן להרחבה. אזי $S = \{x \in V \mid \exists e \in M, x \in e\}$ היא כיסוי בצמתים.
כמו כן, נקבל כי $|S| = 2\,|M| \leq 2\,|M_{\max}|$
($S$ הוא כיסוי בצמתים שגודלו לכל היותר פי $2$ מהאופטימום. הסיבה לכך היא שבכל שידוך יש שני קודקודים, כלומר פי $2$ מהצלעות).

למעשה, הצלחנו לקרב את האופטימום בפקטור של $2$ לכל היותר, ואיננו יודעים לעשות דבר טוב יותר בזמן פולינומי, כי זו כאמור בעיית NP קשה.

**סינוכיות האלגוריתם**
$O\left(m+n\right)$ פעולות וגם מקום.

## 5.4 מציאת כיסוי בצמתים במקרה הממושקל

**הרצאה מס' 17:**

**יום ראשון**

23.05.21

נתון גרף סופי לא מכוון $G = (V, E)$. נתונה פונקציית משקל חיובית על הצמתים $w : V \to \mathbb{N}$. אנו רוצים למצוא כיסוי בצמתים קל ביותר.

**אלגוריתם לחישוב כיסוי בצמתים קל**
נחזיק עבור כל קשת $e = \{u, v\} \in E$ **משתנה עזר** שנסמנו $y_{uv}$ - **"הכסף"** שמשויך לקשתות ושהן מנסות לקנות איתו צמתים שמחוברות אליהן. אם כמות הכסף שחילקנו לקשתות מספיקה לקנות אותו - נוסיף אותו לפיתרון.

אתחול:

לכל קשת $\{u,v\} \in E$ נקבע $y_{uv} \leftarrow 0$.

איטרציות:

נעבור על כל הקשתות בסדר כלשהוא.

- עבור קשת $\{u,v\} \in E$ נעדכן:

$$y_{uv} \leftarrow \min\left\{\left(w\left(u\right) - \sum_{s:\{u,s\}\in E} y_{us}\right), w\left(v\right) - \sum_{s:\{s,v\}\in E} y_{sv}\right\}$$

אינטואיציה - עבור קשת $\{u,v\}$ אנחנו קובעים את הכסף של הצלע להיות כך שלעולם, לעולם, סכום כספי הצלעות לא יהיה גדול מהמשקל.

- בסיום "חלוקת הכסף", נגדיר:

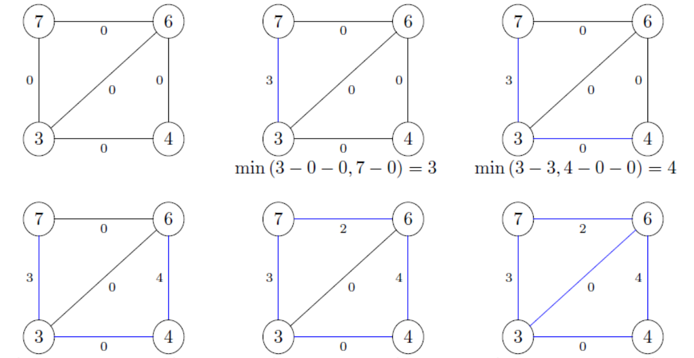
$$S = \left\{u \mid w\left(u\right) = \sum_{v:\{u,v\}\in E} y_{uv}\right\}$$

אינטואיציה - בוחרים את הקודקדים כך שסכום משקלי הצלעות שנוגעים בהם, שווה לסכום הכסף. כלומר, במקרה בו "חילקנו את כל הכסף".

פלט האלגוריתם הוא $S$.

סיבוכיות המקום והזמן הינה $O\left(m+n\right)$.

**דוגמת הרצה**



$\min\left(3-0-0, 7-0\right) = 3$

$\min\left(3-3, 4-0-0\right) = 4$

$\min\left(4-0-0, 6-0-0\right) = 4$

$\min\left(6-4-0, 7-3\right) = 2$

$\min\left(6-4-2, 3-3-0\right) = 0$

שימו לב שרק הקודקודים $3,4,6$ הם הפתרון לכיסוי הממושקל.

**טענה**

אם $S$ היא התוצאה של האלגוריתם, אזי $S$ הוא כיסוי בצמתים של $G$.

**הוכחה**

בכל מהלך ריצת האלגוריתם מתקיים כי לכל צומת $u$[24]:

$$\sum_{s:\{u,s\}\in E} y_{us} \leq w_u$$

כלומר סכום הכספים של הקשתות שיוצאות מהקודקוד קטן ממשקל הקודקוד.

נתבונן בצעד שבו עוברים על קשת $\{u,v\}$. העדכון של $y_{uv}$ מבטיח שאחריו יתקיים כי $\sum_{s:\{u,s\}\in E} y_{us} = w_u$ או $\sum_{s:\{s,v\}\in E} y_{sv} = w_v$ לכל $|S\cap\{u,v\}| \geq 1$ (הסכומים האלו מונוטוניים ולכן אם הגענו ל-$w_u$ למשל, נשאר שם).

לכן $\{u,v\}$ מכוסה על ידי $S$. זה נכון לכל קשת כי האלגוריתם עובר על כל הקשתות, כאמור.

**טענה**

בסיום ריצת האלגוריתם, עבור הפתרון שהוא מחשב $S$, מתקיים כי:

$$\sum_{u\in S} w_u \leq 2 \sum_{\{u,v\}\in E} y_{uv}$$

(אינטואיציה - משקל סכום משקלי צלעות $S$ קטן מפעמיים סכום משקלי הצלעות).[25]

**הוכחה**

מתקיים:

---

[24] ניתן להוכיח באמצעות שמורת לולאה

[25] עוד אלגוריתם מקרב.

$$\sum_{u\in S} w_u \overset{\underset{\downarrow}{\underbrace{\text{הגדרת האלגו}}}}{=}$$

$$\sum_{u\in S}\ \sum_{v:\{u,v\}\in E} y_{uv} \overset{\underset{\downarrow}{\underbrace{\text{שינוי הגדרת הסכימה}}}}{=}$$

$$\sum_{\{u,v\}\in E} y_{uv}\cdot|S\cap\{u,v\}| \overset{\underset{\downarrow}{\underbrace{\text{גודל החיתוך הוא לכל היותר 2}}}}{\le}$$

$$2\sum_{\{u,v\}\in E} y_{uv}$$

**טענה**

בסיום ריצת האלגוריתם מתקיים כי $\sum\limits_{\{u,v\}\in E} y_{uv}$ קטן או שווה ממשקל כיסוי בצמתים קל ביותר.

(אינטואיציה - סכום הכספים של כל הצלעות קטן ממשקל כיסוי בצמתים קל ביותר).

**מסקנה**

האלגוריתם שלנו מחשב 2 - קירוב לבעיית כיסוי בצמתים ממושקל[26].

**הוכחת הטענה**

נסמן ב-$z_{\text{opt}}$ את המשקל של פיתרון אופטימלי לבעיית הכיסוי הממושקלת.

נתבונן בהצבה של ערכים $x_u \in \{0,1\}$ כאשר $S=\{u\in V \mid x_u=1\}$[27] הוא פיתרון אופטימלי לבעייה:

$$z_{\text{opt}}=\min\left\{\sum_{u\in V} w_u x_u \mid \forall u\in V,\ x_u\in\{0,1\} \quad \wedge \quad \forall\{u,v\}\in E,\ \ x_u+x_v\ge 1\right\}$$

למה מדובר בפיתרון אופטימלי? נפרק את ההגדרה. בחרנו את הקודקודים שהם אופטימליים (זה מה שקורה בשורה למעלה).

אנחנו בוחרים כעת את המשקל המינימלי כך שלמעשה $x_u=1\Rightarrow u\in S$ וכך $x_u=0\Rightarrow u\notin S$, כלומר את סכום המשקלים המינימליים כך שנקבל פיתרון חוקי ואופטימלי. נבחין כי $x_u+x_v\ge 1$ חייב להתקיים על מנת שיהיה כיסוי בצמתים.

נגדיר:

$$z^*=\min\left\{\sum_{u\in V} w_u x_u \mid \forall u\in V\ \ x_u\ge 0\wedge \quad \forall\{u,v\}\in E, x_u+x_v\ge 1\right\}$$

---

[26] בתרגול נדבר על אלגוריתם מקרבים ונראה שם מה ההגדרה.

[27] נזכיר כי האינטואיציה כאן היא כביכול יש לנו וקטור $x$ עם כניסות בקודקודים המתאימים, כמו שהיה בתרגול ובתרגיל.

ברור כי $z^* \leq z_{\text{opt}}$, כי כדי לחשב את $z^*$ אנחנו מגדירים אילוצים פחות קשוחים. כלומר, למעשה הפיתרון האופטימלי מוכל בפיתרון זה.

**טענת עזר**

נניח כי $y: E \to \mathbb{R}$ מקיימת כי:

1. לכל $\{u,v\} \in E$ מתקיים כי $y_{uv} \geq 0$ (הכסף אי שלילי).
2. לכל $u \in V$ מתקיים כי $\sum\limits_{v:\{u,v\}\in E} y_{uv} \leq w_u$ (כמו שראינו, סכום הכספים היוצאים לא גדול מהמשקל).

אזי מתקיים כי $\sum\limits_{\{u,v\}\in E} y_{uv} \leq z^*$ - סכום הכספים קטן מפתרון $z^*$.

**הוכחת טענת העזר**

תהי $X: V \to \mathbb{R}$ פונקציה שמקיימת:

1. לכל $u \in V$ מתקיים כי $x_u \geq 0$
2. לכל $\{u,v\} \in E$ מתקיים כי $x_u + x_v \geq 1$

(על מנת שיתקיים הכיסוי בצמתים)

אזי מתקיים כי:

$$\sum_{\{u,v\}\in E} y_{uv} \overset{\substack{\underbrace{x_u + x_v \geq 1} \\ \downarrow}}{\leq}$$

$$\sum_{\{u,v\}\in E} y_{uv}\left(x_u + x_v\right) \overset{\substack{\underbrace{\text{שינוי הסכימה}} \\ \downarrow}}{=}$$

$$\sum_{u\in V} x_u \sum_{v:\{u,v\}\in E} y_{uv} \overset{\substack{\underbrace{\text{נתון}} \\ \downarrow}}{=}$$

$$\sum_{u\in V} w_u x_u$$

בפרט מתקיים כי $\sum\limits_{v:\{u,v\}\in E} y_{uv} \leq w_u$ . כלומר זה נכון עבור כל ה-$x$-ים שקובעים את $z^*$.

מטענת העזר נובע כי:

**הרצאה מס' 18:**

$$\sum_{\{u,v\}\in E} y_{uv} \leq z^* \leq z_{\text{opt}}$$

**יום שלישי**

כעת, נוסיף הגדרה חדשה. נגדיר:

25.05.21

$$u^* = \max \left\{ \sum_{\{u,v\}\in E} y_{uv} \mid \ \forall v \in V \quad \sum_{\{u,v\}\in E} y_{uv} \le w\left(v\right) \ \ \wedge y_{uv} \ge 0 \right\}$$

למעשה, אנחנו מחפשים את המשקל הכסף המקסימלי שהינו חוקי!
אם נפרק את זה קצת, נוכל לגלות כי בעצם הראינו כי $u^* \le z^*$.[28] כמו כן, נוכל להבחין כי שני הפתרונות הללו הם למעשה אותו סוג של בעייה: בשניהם אנו מוצאים אופטימום (מינימום או מקסימום) של פונקציה ליניארית על המשתנים בתחום שמוגדר על ידי מערכת אי שיוונים ליניאריים.

## 5.5 תכנון ליניארי

למעשה, מצאנו כאן שתי דוגמאות קונקרטיות לתכנון ליניארי. המשוואת שמגדירות נקראות ביחד **תוכנית ליניארית**.
התוכניות לחישוב $Z^*$ ו-$u^*$ הן דואליות ומתקיים כי $u^* = z^*$ (לא נוכיח את זה כאן).
קיים אלגוריתם יעיל (פולינומי) לחישוב היתרון של כל תוכניות ליניאריות (עם מקדמים רציונליים).
בעזרת תוכניות ליניאריות אפשר לבטא הרבה בעיות שימושיות.

**דוגמה**
זרימה ברשת אפשר לבטא כך:

$$\begin{aligned} G &= \left(V, A\right) \\ c &: A \to \mathbb{N} \\ s, t &\in V \end{aligned}$$

ומה עלינו למקסם:

$$\max \sum_{v:(s,v)\in A} f\left(s,v\right) - \sum_{v:v:\{v,s\}\in A} f\left(v,s\right)$$

$$s.t$$

$$(i) \quad f\left(u,v\right) \le c\left(u,v\right) \quad \forall \left(u,v\right) \in A$$

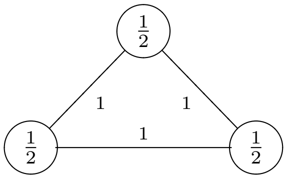
$$\forall u \in V \setminus \{s,t\} \qquad \sum_{v\in V:\{v,u\}\in A} f\left(v,u\right) = \sum_{v\in V:\{u,v\}\in A} f\left(u,v\right)$$

$$f\left(u,v\right) \ge 0 \ \ \forall \left(u,v\right) \in A$$

האם יש תמיד פתרון אופטימלי בשלמים? וואלה לא. לדוגמה התוכנית לחישוב $z^*$.

---

[28] בתכל'ס לא לגמרי הראינו איך זה 2-מקרב. אבל זה מספיק מסובך ואפשר להגיד תודה שזה נגמר כאן.



נקבל כי $z^* = \frac{3}{2}$ . ההצבה של $\frac{1}{2}$ לכל המשתנים מראה כי $z^* \le \frac{3}{2}$. הפיתרון הדואלי שמציב $\frac{1}{2}$ לכל המשתנים מראה כי $z^* \ge \frac{3}{2}$ (כי $u^* \ge \frac{3}{2}$).

נתבונן בדוגמה נוספת.

## 5.6 בעיות סיווג

### 5.6.1 אלגוריתמי המומחים

נניח כי אנו מקבלים תחזיות של $n$ מומחים שמפרסמים כל יום מה יקרה למחרת (למשל - יירד גשם או לא יירד גשם). התחזית הינה בינארית (0 או 1). בכל יום נחליט החלטה בהתאם לתחזית ונרצה **להיכשל לא יותר מהמומחה הטוב ביותר**.

**אלגוריתם פשוט**

נחזיק קבוצה $X$ של מומחים. נחליט על פי דעת הרוב בקבוצה $X$.
אם טעינו, נוריד את המומחים שטעו מ-$X$.[29]
אם $X$ ריקה (הורדנו את כל המומחים), נאתחל שוב עם כל המומחים.

**משפט**

בהינתן $n$ מומחים, אם המומחה הטוב ביותר שוגה $k$ פעמים ($k \ge 0$) אזי האלגוריתם שוגה לכל היותר $(k+1)\log_2(n)$ פעמים.

**הוכחה**

באינדוקציה על $k$, מספר הפעמים שהמומחה הטוב ביותר שוגה.

בסיס האינדוקציה:

$k = 0$.

בכל פעם שהאלגוריתם שוגה, מסירים מ-$X$ לפחות חצי מהמומחים שם. יש מומחה שאינו שוגה (כי $k = 0$), ולכן $X$ אף פעם לא ריקה.

לכן מספר השגיאות שלנו הוא לכל היותר $\log_2(n)$ (כי $|X| = n$ בהתחלה).

צעד האינדוקציה:

נניח כי הטענה נכונה, אם המומחה הטוב ביותר שוגה פחות מ-$k$ פעמים (באינדוקציה מלאה).
נסמן את המומחה הזה בתור $i$ ונניח כי הוא שוגה סך הכל $k$ פעמים.

---

[29] ”אבל מה קורה אם גילינו שכל המומחים מטומטמים? למשל, אם גיליתם שכל השיעור הזה אני אומר שטויות, לא הייתם מגיעים להרצאה. למזלי אין מרצים אחרים שמלמדים אלגוריתמים במקביל ואין לכם ברירה אלא לבוא לפה“. י״ר.

נתבונן בצעד האלגוריתם עד ש-$X$ מתרוקנת בפעם הראשונה (זה חייב לקרות, כי גם הטוב ביותר שגה $k$ פעמים). אנו יודעים כי עד לנקודה זו האלגוריתם שגה לכל היותר $\log_2 n$ פעמים - כי הוא התרוקן. כמו כן, $i$ שגה לפחות פעם אחת שם, כי $X$ התרוקנה.

החל מנקודה זו, $i$ שוגה לכל היותר $k-1$ פעמים, ולכן לפי הנחת האינדוקציה, האלגוריתם לא ישגה יותר מ-$k\log_2 n$ פעמים. סך הכל נקבל $(k+1)\log_2(n)$ שגיאות של האלגוריתם לכל היותר.

כעת נראה אלגוריתם מעט מורכב יותר, עם משקול.

**אלגוריתם הרוב הממושקל**

נניח כי התחזיות הן $\pm 1$ . נסמן ב-$c_i^t$ את התחזית של המומחה ה-$i$ בצעד ה-$t$.

לכל מומחה נחזיק משקל $w_i^t$, שתכף נבין מה הפואנטה שלו.

- נאתחל את כל המשקלים של המומחים להיות בתור 1 (לכל $i$ $w_i^1 = 1$).
- כעת, ההחלטה שלנו בצעד ה-$t$ הינה הסימן של $\sum_i w_i^t c_i^t$.
- נעדכן את המשקלים:
  1. לכל מומחה שטעה נעדכן כי $w_i^t \leftarrow \frac{1}{2}w_i^t$.[30]
  2. למומחה $i$ שצדק לא נעדכן כלום, כלומר $w_i^{t+1} \leftarrow w_i^t$.

נסמן ב-$m_i^{(t)}$ את מספר הטעויות של המומחה.

כמו כן, נסמן ב-$M(t)$ את מספר הטעויות של האלגוריתם ב-$t$ הצעדים הראשונים.

**משפט**

לכל מומחה $i$ ולכל $t$ מתקיים:

$$M(t) \le \frac{1}{\log_2\left(\frac{4}{3}\right)}\left(m_i^{(t)} + \log_2 n\right)$$

(אינטואיציה - מספר השגיאות של האלגוריתם קטן מחסם כלשהוא על כלל טעויות המומחים).

**הוכחה**

נסמן $\phi(t) = \sum_i w_i^{(t)}$ - סכום המשקלות על כל המומחים.

האתחול אם כך יהיה $\phi(1) = n$, כי בהתחלה כל המשקלים הם 1ץ

כמו כן, לכל $i$ ולכל $t$ מתקיים כי $\phi(t+1) \ge w_i^{(t+1)} = \left(\frac{1}{2}\right)^{m_i^{(t)}}$ - סכום המשקלים גדול ממשקל ספציפי, שתלוי במספר הטעויות של מומחה כלשהו.

כאשר האלגוריתם שוגה, משקל המומחים ששגו הוא לפחות חצי מהמשקל הכולל - כלומר, אם שוגים בצעד $t$, אזי $\sum_{i\text{ששגו}} w_i^{(t)} \ge \frac{1}{2}\sum_i w_i^{(t)}$ כי מחליטים על פי הרוב ולכן:

$$\phi(t+1) = \sum_i w_i^{(t+1)} = \sum_{i\text{ ששגו}} \frac{1}{2}w_i^{(t)} + \sum_{i\text{ שלא שגו}} w_i^{(t)} \le \frac{3}{4}\sum_i w_i^{(t)}$$

---

[30]

כלומר, נקבל כי:

$$\begin{gathered}\underbrace{\text{הפעלת הא״ש מלא פעמים}}\\ \downarrow \\ \phi\left(t+1\right)\le \\ \left(\frac{3}{4}\right)^{M(t)}\cdot\phi\left(1\right)=\left(\frac{3}{4}\right)^{M(t)}\cdot n\end{gathered}$$

נקבל בשילוב שני הא״שים:

$$\left(\frac{1}{2}\right)^{m_i^{(t)}}\le\phi\left(t+1\right)\le\left(\frac{3}{4}\right)^{M(t)}\cdot n$$

וגם נקבל כי (נוציא log ל-$\left(\frac{1}{2^{m_i^{(t)}}}\right)=2^{-m_i^{(t)}}$):

$$-m_i^{(t)}\le\log_2\left(\frac{3}{4}\right)M\left(t\right)+\log_2\left(n\right)$$

ולכן סך הכל נקבל:

$$M\left(t\right)\le\frac{1}{\log_2\left(\frac{4}{3}\right)}\left(m_i^{(t)}+\log_2 n\right)$$

**עדכון מתון של השגיאות**

נבחין כי על כל שגיאה של המומחים התנהגנו אליהם באכזריות מה. נוכל לעדכן זאת בצורה מתונה יותר:

$$w_i^{(t+1)}\leftarrow\frac{1}{1+\varepsilon}w_i^t$$

ונקבל (לא קשה לפתח אבל לא נעשה זאת כאן):

$$\left(\frac{1}{1+\varepsilon}\right)^{m_i^{(t)}}\le\phi(t+1)\le\left(1-\frac{\varepsilon}{2(1+\varepsilon)}\right)^{M(t)}\cdot n$$

גם נקבל כי:

$$-\ln\left(1+\varepsilon\right)m_i^{(t)}\le\left(-\ln\left(\frac{2+2\varepsilon}{2+\varepsilon}\right)\right)M\left(t\right)+\ln\left(n\right)$$

וגם:

$$M\left(t\right) \leq \frac{\ln\left(1+\varepsilon\right)}{\ln\left(\frac{2+2\varepsilon}{2+\varepsilon}\right)} m_i^{(t)} + \frac{1}{\ln\left(\frac{2+2\varepsilon}{2+2\varepsilon}\right)} \ln\left(2\right) \leq 1+\varepsilon+\frac{1}{\varepsilon}$$

אמנם $1+\varepsilon$ משתפר ככל ש-$\varepsilon$ קטן יותר, אבל $\frac{1}{\varepsilon}$ לא ממש. אזי יש לנו סוג של טרייד-אוף כאן.

**אלגוריתם המשקול הכפלי**

**הרצאה מס' 19:**

**יום ראשון**

30.05.21

נבחין כי במקרה הקודם, הנחנו מודל ספציפי של 'קנסות', ששייכים לקבוצה $\{0,1\}$[31].
כעת נניח מודל כללי יותר, כי בכל יום מוטלים על המומחים 'קנסות' בקטע $[-1,1]$.
נסמן ב-$c_i^t \in [-1,1]$ את הקנס שמוטל על המומחה ה-$i$ ביום ה-$t$. בהתחלה, מתקיים לכל מומחה $i$ כי $w_i = 1$ - כלומר, משקל כל אחד מהמומחים שווה.
האלגוריתם בוחר בכל יום $t$ התפלגות $p^t$ על המומחים (לפני שחושפים את הקנסות) - למעשה, ישנה התפלגות לגבי **איזה מומחה נבחר בסיבוב ה-**$t$.
כלומר: לכל $i$, מתקיים כי $p_i^t \geq 0$ וכמו כן $\sum\limits_{i=1}^{n} p_i^t = 1$ (סכום ההסתברויות). ביום ה-$t$ נקבל כי האלגוריתם משלם $\sum\limits_{i=1}^{n} p_i^t c_i^t = \mathbb{E}\left[c^t\right]$ - כלומר, זהו **בתוחלת** הסכום אותו המומחה קיבל ביום ה-$t$.
אם כך, על מנת לקבל את הסכום הכללי, נקבל את סכום התוחלת בכל הימים, כלומר סך כל הקנסות שמשלם האלגוריתם הינו $\sum\limits_{t=1}^{T}\sum\limits_{i=1}^{n} p_i^t c_i^t = \sum\limits_{t=1}^{T} \mathbb{E}\left[c^t\right]$.
את תשלום זה נרצה להשוות לתשלום של המומחה הטוב ביותר $\min\limits_{1\leq i\leq n} \sum\limits_{t} c_i^t$.
נחזיק משקלים לכל המומחים. נסמן ב-$w_i^t$ את המשקל של המומחה ה-$i$ בתחילת האיטרציה ה-$t$.
ההתפלגות שהאלגוריתם "משחק" הינה $p_i^t = \frac{w_i^t}{\sum\limits_{j=1}^{n} w_j^t}$.
המשקלים מתעדכנים אחרי חשיפת הקנסות באופן הבא:

$$w_i^{t+1} = \left(1-\varepsilon c_i^t\right) w_i^t$$

עבור $\varepsilon > 0$.

**משפט**

בהנחה ש-$0 < \varepsilon \leq \frac{1}{2}$ מתקיים אחרי $T$ צעדים, לכל מומחה $i$:

---

[31] מדוע המודל הספציפי הוא קנסות ששייכים לקבוצה זו? ראו בתרגול.

$$\underbrace{\sum_{i=1}^{T}\sum_{i=1}^{n} c_i^t p_i^t}_{\text{התשלום של האלגו}} \leq$$

$$\underbrace{\sum_{t=1}^{T} c_i^t}_{i\text{-התשלום של המומחה ה}} \qquad +$$

$$\underbrace{\varepsilon \sum_{t=1}^{T} \left|c_i^t\right| + \frac{1}{\varepsilon}\ln\left(n\right)}_{\text{התשלום הנוסף של האלגו במקרה הגרוע}}$$

**הוכחה**

נגדיר את $\phi\left(t\right)=\sum\limits_{t=1}^{n} w_i^t$ . על פי כלל העדכון מתקיים כי:

$$\phi\left(t+1\right) \overset{\overset{\underbrace{\text{הגדרה}}}{\downarrow}}{=}$$

$$\sum_{i=1}^{n} w_i^{t+1} \overset{\overset{\underbrace{\text{הגדרה}}}{\downarrow}}{=}$$

$$\sum_{i=1}^{n} w_i^t\left(1-\varepsilon c_i^t\right) =$$

$$\phi\left(t\right) - \varepsilon \sum_{i=1}^{n} w_i^t c_i^t$$

נבחין כי $p_i^t = \frac{w_i^t}{\phi(t)}$ (ובכן, זו ההגדרה) ולכן $w_i^t = p_i^t \cdot \phi\left(t\right)$. אם כך, נקבל כי:

$$\phi\left(t\right) - \varepsilon \sum_{i=1}^{n} w_i^t c_i^t = \phi\left(t\right) - \varepsilon\phi\left(t\right)\sum_{i=1}^{n} p_i^t c_i^t =$$

$$\phi\left(t\right)\left(1-\varepsilon\sum_{i=1}^{n} p_i^t c_i^t\right)$$

נשתמש בא"ש הידוע $1-q \leq e^{-q}$ ולכן בפרט:

$$\phi\left(t\right)\left(1-\varepsilon\sum_{i=1}^{n} p_i^t c_i^t\right) \leq \phi\left(t\right)\cdot e^{-\varepsilon\sum\limits_{i=1}^{n} p_i^t c_i^t}$$

ולכן נקבל, אם נציב $t=T$, עבור $T+1$:

$$\phi\left(T+1\right) \le \phi\left(1\right)\cdot e^{-\varepsilon\sum\limits_{t=1}^{T}\sum\limits_{i=1}^{n}p_i^tc_i^t} = ne^{\underbrace{-\varepsilon\sum\limits_{t=1}^{T}\sum\limits_{i=1}^{n}p_i^tc_i^t}_{(*)}}$$

הביטוי $(*)$ הוא מחיר האלגוריתם ב-$T$ הצעדים הראשונים [32].

נוכל להבחין גם כי:

$$\phi\left(T+1\right) \ge w_i^{T+1} = \prod_{t=1}^{T}\left(1-\varepsilon c_i^t\right)$$

ניזכר בא"ש ברנולי כי עבור $\alpha \in [0,1]$ מתקיים כי $(1-\varepsilon)^{\alpha} \le 1-\varepsilon\alpha$ וגם כי עבור $\alpha \in[-1,0]$ מתקיים כי $(1+\varepsilon)^{-\alpha} \le 1-\varepsilon\alpha$.

כעת, יתקיים כי:

$$\phi\left(T+1\right) \overset{\substack{\overbrace{\text{הורדת איברים חיוביים}} \\ \downarrow}}{\ge} w_i^{T+1} \overset{\substack{\overbrace{\text{מספר הטעויות כביכול}} \\ \downarrow}}{=} \prod_{t=1}^{T}\left(1-\varepsilon c_i^t\right) \overset{\substack{\overbrace{\text{הצבת א"ש}} \\ \downarrow}}{\ge} (1-\varepsilon)^{\sum\limits_{t:c_i^t\ge0}c_i^t}(1+\varepsilon)^{-\sum\limits_{t:c_i<0}c_i^t}$$

נקבל סך הכל כי:

$$(1-\varepsilon)^{\sum\limits_{t:c_i^t\ge0}c_i^t}(1+\varepsilon)^{-\sum\limits_{t:c_i<0}c_i^t} \le ne^{-\varepsilon\sum\limits_{t=1}^{T}\sum\limits_{i=1}^{n}p_i^tc_i^t}$$

נעביר אגפים ונחלק ב-$\varepsilon$ ונקבל :

$$\sum_{t=1}^{T}\sum_{i=1}^{n}p_i^tc_i^t \le \tfrac{1}{\varepsilon}\ln n + \frac{\ln\frac{1}{1-\varepsilon}}{\varepsilon}\sum_{t:c_i^t\ge0}c_i^t + \frac{\ln(1+\varepsilon)}{\varepsilon}\sum_{t:c_i<0}c_i^t$$

---

[32] "וזה...ביטוי נחמד" (י"ר).

נשתמש בא"ש נוסף, עבור $\varepsilon \in \left(0, \frac{1}{2}\right)$ מתקיים כי $\ln\left(\frac{1}{1-\varepsilon}\right) \le \varepsilon + \varepsilon$ וגם מתקיים כי $\ln(1+\varepsilon) \le \varepsilon - \varepsilon^2$.
ולכן נקבל, לאחר הצבה, כי:

$$\sum_{t=1}^{T}\sum_{i=1}^{n} p_i^t c_i^t \le \sum_{t=1}^{T} c_i^t + \varepsilon \sum_{t=1}^{T} \left|c_i^t\right| + \\ + \frac{1}{\varepsilon}\ln n$$

כנדרש.

כיצד נבחר את ה-$\varepsilon$ האידיאלי? אם נבחר $\varepsilon = \sqrt{\frac{\ln(n)}{T}}$, נקבל כי השגיאה פרופרציונלית ל-$\sqrt{T \ln(n)}$. כלומר, השגיאה הממוצעת לצעד היא:

**הרצאה מס' 20:**

**יום שלישי**

01.06.21

$$\frac{1}{T} \cdot \sqrt{T \ln(n)} = \sqrt{\frac{\ln(n)}{T}}$$

ככל שמתקדמים בצעדים, השגיאה הולכת ומתקרבת לאפס. מושג זה נקרא **חרטה** (regret).

**הערה**
ישנה וריאציה של שיטת המשקול הכפלי, שנקראת Hedge ובה עדכון המשקלים הוא $w_i^{t+1} \leftarrow w_i^t e^{-\varepsilon c_i^t}$.

### 5.6.2 תכנון ליניארי לבעיית המומחים והכיסוי בצמתים

האלגוריתם הזה שימושי להמון דברים. בין היתר, מדובר על אחד הכלים החשובים בלמידה חישובית[33].
אם כל אחד מהם נותן תחזית מסוימת, ואנו יודעים שאחד מהם טוב - בדרך זו האלגוריתם למידה בוחר את החזאי הטוב ביותר.

נתבונן בתוכנית הליניארית הבאה (שכבר ראינו):

$$\begin{aligned} &\min \sum_{v \in V} w_v x_v \\ &s.t \;\; x_u + x_v \ge 1 \;\; \forall \{u,v\} \in E, x \ge 0 \end{aligned}$$

נניח שאנחנו רוצים לפתור את תוכנית זו. כדי לעשות זאת, די לפתור את בעיית ההכרעה הבאה:
האם קיים $x \ge 0$ עבורו לכל קשת $\{u,v\} \in E$ מתקיים כי $x_u + x_v \ge 1$ וגם $\sum_v w_v x_V \le \beta$ עבור קלט $\beta$,
בהינתן פיתרון לבעיית ההכרעה זו, אפשר לפתור את התוכנית הליניארית על ידי חיפוש בינארי על $\beta$ (כלומר, לבדוק בכל פעם על קלט קטן יותר סטינו - לא הוכחנו אבל זה נובע באופן ישיר מכך שהפתרון הוא מקדם רציונלי עם מקדם יחסית קטן).
לכן, ניתן להחליף את הבעיה המקורית בבעייה הבאה: נבדוק האם קיים $x \ge 0$ עבורו מערכת אי שיוונים ליניאריים $Ax \ge b$ מתקיימת (עבור מטריצת אילוצים $A$ ווקטור $b$).

---

[33] מי שלומד IML מוודאי יושב לעצמו עכשיו ומחייך מתחת לשפם.

פיתרון מקורב לבעיית ההכרעה הזו:

- נזהה בוודאות שאין $x$ כזה, או:
- נמצא $x \geq 0$ עבורו לכל אי שוויון $i$ יתקיים ל-$A_i \cdot x \geq b_i - \delta$ עבור $\delta > 0$ כלשהו.

נסמן את מספר האילוצים ב-$m$ ואת מספר המשתנים ב-$n$. נקבל כי $A \in \mathbb{R}^{m \times n}, x \in \mathbb{R}^n, b \in \mathbb{R}^m$.
באופן כללי, מפתרון מקורב אפשר למצוא פתרון קרוב לאופטימלי לתוכנית הליניארית.
למשל אם נניח כי $b_i \geq 1$ לכל $i$ ו-$1-\delta$ מאוד קטן ביחס ל-1, אזי אם $x \geq 0$ מקיים כי $Ax \geq b - \begin{pmatrix} \delta \\ \vdots \\ \delta \end{pmatrix}$ אזי $x' = \frac{1}{1-\delta}x$ מקיים כי $Ax' \geq b$ (בהנחה למשל שהמקדמים במטריצה $A$ הם חיוביים). פונקציית המטרה הוכפלה ב-$\frac{1}{1-\delta}$.

נצטרך להניח כי קיים אלגוריתם $O$ לבעייה פשוטה יותר:
בהינתן התפלגות $p$ על השורות של $A$, כלומר התפלגות מתאימה לכל אחת מהשורות, ניתן לבדוק באמצעות האלגוריתם $O$ האם קיים $x \geq 0$ עבורו $\sum_{i=1}^{m} p_i A_i x \geq \sum_{i=1}^{m} p_i b_i$.
עוד נניח **כי אם קיים פיתרון** $x$ **כזה**, נמצא גם $x$ כזה, **עבורו לכל** $i$ **מתקיים** כי $|A_i x - b_i| \leq \rho$ עבור איזשהו פרמטר $\rho$ (שנקרא **הרוחב** של הבעייה).

בקצרה, נשים לב כי אם קיים $x \geq 0$ עבורו $Ax \geq b$, אזי ה-$x$ הזה מקיים את הא"ש $\sum_{i=1}^{m} p_i A_i x \geq \sum_{i=1}^{m} p_i b_i$ .

**משפט**

בהינתן $\delta > 0$ ואלגוריתם $O$ כנ"ל עם $\rho \geq \frac{\delta}{2}$, קיים אלגוריתם שקורא ל-$O$ לכל היותר $O\left(\frac{\rho^2}{\delta^2} \log m\right)$ פעמים ומחזיר $x \geq 0$ עבורו לכל $1 \leq i \leq m$ מתקיים כי $A_i x \geq b_i - \delta$ או מזהה נכון כי אין $x \geq 0$ שעבורו $Ax \geq b$.
(אינטואיציה - אם קיים אלגוריתם $O$ שמקיים את התנאים לעיל עם $\rho \geq 0$, באמצעות אלגוריתם אחר שמשתמש באלגוריתם שלנו כ-$\frac{\rho^2}{\delta^2} \log m$ פעמים, נוכל למצוא פתרון לבעיה כי $A_i x \geq b_i - \delta$).

**הערה**

קודם כל, חשוב להבחין מבחינה רעיונית כי התהליך שלנו הוא כזה - אנחנו מחפשים פתרון לתוכנית הליניארית שפותרת את בעיית המומחים לעיל. אנחנו לא יכולים, לכן אנחנו רוצים למצוא $Ax \geq b - \delta_n$ עבור $\delta$ כלשהי וזו כפי שראינו פתרון מקורב לבעיה. באמצעות ההנחה שקיים אלגוריתם כמו קודם, נרצה להראות כי קיים פתרון מקורב לבעיה.

**הוכחה**

בהינתן $x \geq 0$ כלשהו, נגדיר קנסות לאילוצים $x$ (האילוצים הם המומחים שלנו כאן).
נסמן $c_i(x) = \frac{1}{\rho} \cdot (A_i x - b_i)$ - אלו הקנסות של המומחים. לפי הגדרת הרוחב $\rho$, מתקיים כי $c_i(x) \in [-1, 1]$.
האלגוריתם מבצע איטרציות של עדכון $x$. בתחילת האיטרציה ה-$t$, האלגוריתם מחזיק התפלגות $p^t$ על האילוצים.
באתחול זו ההתפלגות האחידה.
על מנת לעדכן את ההתפלגות לאחר מכן, אנו מוצאים $x^t \geq 0$ באמצעות ה-$O$ לעיל, שמקיים כי:

$$\sum_{i=1}^{m} p_i^t A_i x \geq \sum_{i=1}^{m} p_i^t b_i$$

אם אין כזה, אזי זיהינו כי אין פיתרון $Ax \ge 0$ שמקיים $Ax \ge b$.
אחרת, כלומר, הוא בתוך הרוחב המוגדר, נחשב את $p^{t+1}$ על פי שיטת המשקול הכפלי עם המחירים $c_i^t = c_i\left(x^t\right)$ (ופרמטר עדכון $\varepsilon > 0$) - כלומר, הכפל בוקטור מבצע את העדכון.
נשים לב כי בכל צעד $t$ מתקיים כי:

$$\sum_{i=1}^{m} c_i^t p_i^t \overset{\overbrace{\text{הגדרת העדכון}}{\downarrow}}{=}$$

$$\frac{1}{\rho}\sum_{i=1}^{m}\left(A_i x^t - b_i\right) p_i^t > 0$$

מנגד, לפי מה שהוכחנו בנוגע לשיטת המשקול הכפלי, לכל אילוץ (מומחה) $i$ נקבל:

$$0 \le \sum_{t=1}^{T}\sum_{i=1}^{m} c_i^t p_i^t = \underbrace{\sum_{t=1}^{T}\frac{1}{\rho}\left(A_i x^t - b_i\right)}_{i\text{ הקנס ששילם המומחה ה-}} + \underbrace{\varepsilon\sum_{t=1}^{T}\frac{1}{\rho}\left|A_i x^t - b_i\right| + \frac{1}{\varepsilon}\ln\left(n\right)}_{\text{הסטייה}}$$

מתקיים כי $\frac{1}{\rho}\left|A_i x^t - b_i\right| \le 1$ בכל פעם, ולכן נקבל בפרט כי $\varepsilon\sum_{t=1}^{T}\frac{1}{\rho}\left|A_i x^t - b_i\right| + \frac{1}{\varepsilon}\ln\left(n\right) \le \varepsilon \cdot T$.
היינו רוצים[34] את החלק הראשון כי אז:

$$0 \le \frac{1}{T}\sum_{t=1}^{T}\frac{1}{\rho}\left(A_i x^t - b_i\right) = A_i\left(\frac{1}{T}\sum_{t=1}^{T} x^t\right) - b_i$$

נסמן $\frac{1}{T}\sum\limits_{t=1}^{T} x^t = \overline{x}$. נכפיל את הביטוי כולו ב-$\frac{\rho}{T}$ ונקבל:

$$0 \le A_i\overline{x} - b_i + \varepsilon\rho + \frac{\rho\ln\left(n\right)}{\varepsilon T}$$

נבחר $T = \left\lceil \frac{4\rho^2}{\delta^2}\ln\left(m\right)\right\rceil$ ו-$\varepsilon = \frac{\delta}{2\rho}$ נקבל כי:

$$0 \le A_i\overline{x} - b_i + \delta \Rightarrow A_i\overline{x} \ge b_i - \delta$$

---

[34] ״אם היה לנו את החלק הראשון, זה היה נפלא, לא?״. י״ר.

**הרצאה מס' 21:**

**יום ראשון**

06.06.21

למה קיים האלגוריתם $O$ עבור התוכנית הליניארית שהתחלנו איתה[35]? נתונים לנו ערכים $p_{uv} \geq 0$ עבור $\{u,v\} \in E$ שמקיימים כי $p = \sum_{\{u,v\}\in E} p_{uv} \leq 1$.
אנחנו מחפשים $x : V \to [0,1]$, עבורו:

$$\sum_{\{u,v\}\in E} p_{uv}\left(x_u + x_v\right) - \left(1-p\right)\sum_{v\in V} w_v x_v \geq p - \left(1-p\right)\beta$$

או לזהות כי אין $x$ כזה.

האלגוריתם $O$ שאנחנו צריכים כדי לפתור את התוכנית לעיל באופן מקורב, צריך לחשב את התוכנית הבאה:

$$\begin{aligned}
&\min \sum_{v\in V} w_v x_v \\
&s.t\ \ x_u + x_v \geq 1\ \ \forall \{u,v\} \in E, \\
&x \geq 0 \\
&-\sum_{v\in V} w_v x_v \geq -\beta
\end{aligned}$$

הקלט לאלגוריתם $O$ הוא פונקציית התפלגות על אילוצים אלו. כלומר, לכל $u,v$ נסמן הסתברות $p_{uv}$ והאילוץ האחרון יהיה $1-p$. כמו כן, מתקיים כי $1 - p + \sum_{u,v\in E} p_{uv} = 1$. בהינתן ההתפלגות הזו, נרצה לבדוק האם קיים $x$ שמקיים את האילוץ הממושקל הזה. כלומר, סך הכל נרצה לבדוק האם מתקיימת התוכנית, דהיינו:

$$\sum_{\{u,v\}\in E} p_{uv}\left(x_u + x_v\right) - \left(1-p\right)\sum_{v\in V} w_v x_v \geq \sum_{\{u,v\}\in E} p_{uv} - \left(1-p\right)\beta$$

האלגוריתם כאמור אמור להחזיר $x$ מסוים או לומר כי אי אפשר לפתור את הבעיה הזאת (אין $x$ כזה).
נניח כי $1-p \neq 0$ (אם $1-p=0$ הבעיה טרוויאלית כי אפשר לקחת את כל ה-$x$-ים להיות 1) ואז נסמן $q_{uv} = \frac{p_{uv}}{1-p}$. נקבל כי:

$$\sum_{\{u,v\}\in V} q_{uv}\left(x_u + x_v\right) - \sum_{v\in V} w_v x_v \geq \sum_{\{u,v\}\in E} q_{uv} - \beta$$

נבחין כי המקדם של $x_u$ הוא $\sum_{u:\{u,v\}\in E} q_{uv}$ (כי זהו סכום כל הקשתות שנוגעות ב-$v$). אם כך, נוכל להחליף את הביטוי מקודם ולקבל:

---

[35] שהינה $\min \sum_{v\in V} w_v x_v$ כך שלכל $(u,v) \in E$ מתקיים כי $x_u + x_v \geq 1$ וגם $x \geq 0$.

$$\sum_{v\in V}\left(\sum_{\{u,v\}\in E} q_{uv} - w_v\right) \geq \sum_{\{u,v\}\in E} q_{uv} - \beta$$

האלגוריתם $O$ צריך למצוא $x$ שמקיים את הדרישות הללו, או להחליט שאין $x$ כזה.
ההשמה הבאה אמורה לעבוד, אם קיים פתרון $x$ עבורו $x_v \in [0,1]$.:

$$x_v = \begin{cases} 0 & \sum_{u:\{u,v\}\in E} q_{uv} - w_v \leq 0 \\ 1 & else \end{cases}$$

אם ההצבה מקיימת את אי השוויון הנדרש, סיימנו. אחרת, אין הצבה של ערכים $x$ עבורם $x_v \in [0,1]$ שמקיימת את אי השוויון.
הראינו כיצד אפשר לפתור את התוכנית הליניארית באופן מקורב, אבל למעשה כל תוכנית ליניארית אפשר לקרב כך עם מעט שינויים.

## 6 אלגוריתמים הסתברותיים

**הרצאה מס' 22:**

**יום שלישי**

08.06.21

ישנן בעיות רבות שאיננו יודעים לפתור באופן דטרמיניסטי. מתוך בעיות אלו, ישנו מספר קטן של בעיות שאנו יודעים לפתור עם אלגוריתם הסתברותי.
באופן בסיסי, הרעיון הוא כי האלגוריתם 'יגריל' את ההסתברות שלו. כלומר, האלגוריתם יטיל מטבע כביכול ולפיו יעבוד. מדוע זה תורם לנו? אכן, אם בודקים שתי אפשרויות, עשינו זאת בעבר וזה לא נורא. האלגוריתם יכול לעזור לנו אם נגריל כביכול $n$ מטבעות - במצב זה, לעבור על $2^n$ האפשרויות זה כבר לא יעיל ולכן האלגוריתם ההסתברותי מאפשר לנו להתבונן במרחב האפשרויות שגודלו $2^n$.
נניח כי מכל $2^n$ האפשרויות ישנה אפשרות אחת שפותרת את הבעיה - עדיין לא מדובר על אלגוריתם יעיל. אלגוריתם כזה יעבוד בצורה יעילה אם באחוז מספיק טוב, נקבל פיתרון לבעיה. כלומר, נגדיר מרחב אפשרויות ובתוכו - הסיכוי לפתור את הבעיה הוא טוב (קבוע, או לפחות 1 חלקי פולינום בגודל הקלט) - אך איננו יודעים לפתור את הבעיות בצורה דטרמיניסטית - אנו יודעים שחצי מהבעיות יצליחו, אבל לא יודעים איזה חצי.
לפעמים, לבעיות מסוימות ייתכנו גם פתרונות דטרמינסטיים - נראה את הפתרון ההסתברותי גם כי קל לעבוד על דוגמאות אלו, וגם כי לפעמים ההסתברותי יעיל יותר.

### 6.1 בעיית חתך גדול ביותר - Max-cut

נתון גרף לא מכוון $G=(V,E)$ - רוצים למצוא חתך $F\subseteq E$ עם מספר מירבי של קשתות.
$F$ היא קבוצת הקשתות בין שני הצדדים של חלוקה לא טרוויאלית של $V$ לשתי קבוצות $S, V\setminus S\neq\emptyset$.
מדובר בבעיית NP קשה ולכן נחפש קירוב לבעייה זו.

#### 6.1.1 אלגוריתם הסתברותי פשוט לקירוב הבעייה

1. נבחר קשת כלשהי $\{u,v\}\in E$ ונשים את $u$ ו-$v$ בצדדים שונים של החתך.

2. לכל $w \in V \setminus \{u,v\}$ נגריל צד באופן בלתי תלוי בהגרלות האחרות ובהתפלגות אחידה.

כיוון שאנחנו מגרילים על הצמתים באופן בלתי תלוי, מספר הטלות המטבע של האלגוריתם הוא $|V|-2$. אם כך, מרחב המדגם הוא $2^{|V|-2}$ - מדובר בגודל שהינו אקספוננציאלי בגודל הקלט (אם הקלט אינו 2).

**משפט**

יהי $F$ החתך שנוצר על ידי האלגוריתם. אזי $\mathbb{E}\left[|F|\right] > \frac{|E|}{2}$.

**הוכחה**

עבור קשת $\{x,y\} \in E$, נסמן ב$C_{xy}$ את המשתנה המקרי $\mathbb{1}_{[\{x,y\}\in F]}$.

נבחין כי $\sum\limits_{\{x,y\}\in E} C_{xy} = |F|$ ואז נקבל :

$$\mathbb{E}\left[|F|\right] = \overbrace{\mathbb{E}\left[\sum_{\{x,y\}\in E} C_{xy}\right] \overset{\downarrow}{=}}^{\text{ליניאריות התוחלת}} \sum_{\{x,y\}\in E} \mathbb{P}\left[C_{xy}=1\right]$$

נבחין כי $\mathbb{P}\left[C_{xy}=1\right]$ שקולה להסתברות כי $x$ בצד של $v$ ו-$y$ בצד של $u$, או כי $x$ בצד של $u$ ו-$y$ בצד של $v$. ההגרלות של $x$ ו-$y$ הינן בלתי תלויות, הסיכוי שכל אחד מהתרחישים יקרה הוא $\frac{1}{4}$ ולכן ביחד הסיכוי הוא $\frac{1}{2}$ (אלא אם כן $\{x,y\}=\{u,v\}$ ואז ההסתברות 1). ולכן נקבל, לפי החלוקה למקרים:

$$\mathbb{E}\left[|F|\right] = \frac{|E|-1}{2} + 1 > \frac{|E|}{2}$$

**מסקנה**

$\mathbb{E}\left[|F|\right]$ הוא קירוב בפקטור 2 לגודל של חתך מקסימום.

**הוכחה**

חתך מקסימום מכיל לכל היותר את כל הקשתות.

נבחין כי לאורך ההוכחה השתמשנו בליניאריות התוחלת - ואכן זה נכון גם לגבי מקרים בהם המאורעות תלויים.
זה משמעותי, שכן לפעמים המאורעות שראינו הם תלויים ,למשל במקרה של משולש - שם המאורעות תלויים.
ננסה כעת לחשב את ההסתברות בצורה מדויקת יותר.
ראינו כי $\mathbb{E}\left[|F|\right] = \frac{|E|-1}{2} + 1 = \frac{|E|}{2} + \frac{1}{2}$. מכאן עולה כי $\mathbb{E}\left[|E| \setminus F\right] = \frac{|E|}{2} - \frac{1}{2}$. אם כך, מה ההסתברות למאורע הבא?

$$\mathbb{P}\left[|F| < \frac{|E|}{2}\right] \overset{\overbrace{\downarrow}^{\text{מעל הטבעיים}}}{=}$$

$$\mathbb{P}\left[|F| \leq \frac{|E|}{2} - 1\right] \overset{\overbrace{\downarrow}^{\text{משלים}}}{=}$$

$$\mathbb{P}\left[|E| \setminus F \geq \frac{|E|}{2} + 1\right]$$

נסמן $\alpha = \frac{\frac{|E|+2}{2}}{\frac{|E|-1}{2}} = \frac{|E|+2}{|E|-1} = 1 + \frac{3}{|E|-1}$ ואז נקבל מא"ש מרקוב :

$$\mathbb{P}\left[|E| \setminus F \geq \frac{|E|}{2} + 1\right] = \mathbb{P}\left[|E| \setminus F \geq \mathbb{E}\left[|E| \setminus F\right] \cdot \alpha\right] \leq \frac{1}{\alpha} =$$
$$\frac{|E|-1}{|E|+2} = 1 - \frac{3}{|E|+2}$$

אם כך, בהסתברות של לפחות $\frac{3}{|E|+2}$, האלגוריתם מחשב חתך $F$ שגודלו לפחות $\frac{|E|}{2}$.
אם נחזור על האלגוריתם באופן בלתי תלוי $\frac{|E|+2}{3}$ פעמים, וניקח את התוצאה הטובה ביותר $F_{\max}$, אזי:

$$\mathbb{P}\left[F_{\max} < \frac{|E|}{2}\right] \leq \left(1 - \frac{3}{|E|+2}\right)^{\frac{|E|+2}{3}} \leq \frac{1}{e} < \frac{1}{2}$$

כיוון שהנסיונות הם בלתי תלויים.
אם כך, הסתברות ההצלחה היא כעת לפחות $1 - e^{-1}$. אם נחזור על האלגוריתם $k \cdot \frac{|E|+2}{3}$ פעמים, נקבל כי ההסתברות לכישלון הינה לכל היותר $e^{-k}$.

באמצעות האלגוריתם היחסית פשוט הזה, הדגמנו כיצד מנתחים אלגוריתמים הסתברותיים ומכאן נוכל להמשיך לנתח גם אלגוריתמים הסתברותיים אחרים.
קיימים אלגוריתמים דטרמינסטיים פשוטים למצוא 2-קירוב ל-Max-Cut.[36]

**הערה**
נבחין כי אותו רעיון שראינו כעת פותר בעיות דומות.
למשל, הבעייה הבאה:
נתון אוסף משוואות ליניאריות מעל $\mathbb{Z}_2$ (דוגמה למשוואה כזו - $x_2 \oplus x_3 \oplus x_4 \oplus x_{30} = 0$).
רוצים למצוא הצבה של ערכי $0, 1$ למשתנים עבורה מספר המשוואות שמתקיימות מקסימלי.
זו הכללה של Max-Cut:
נגדיר משתנים $x_v$ לכל צומת $v \in V$. לכל קשת $\{u, v\} \in E$ נוסיף משוואה ליניארית $x_u \oplus x_v = 1$. המשוואה מתקיימת אס"ם $x_u \neq x_v$ .

---

[36] ראינו בתרגול.

לכן, הפתרון הוא חתך שבו $S = \{u \in V \mid x_u = 0\}$ ומספר המשוואות שמתקיימות הוא בדיוק מספר הצלעות שנחתכות.
במקרה הכללי, אם נגריל את המשתנים בהתפלגות אחידה ובאופן ב"ת, מה ההסתברות שמשוואה מתקיימות?
בכל הצבה בשלושת המשתנים הראשונים ולכל הצבה ב-$x_{30}$ יש בדיוק בחירה אחת לקיום המשוואה:

$$x_1 \oplus x_3 \oplus x_3 \oplus x_{30} = 0$$

אם כך, ההסתברות לקיום המשוואה שווה ל-$\frac{1}{2}$ והמסקנה מכך היא כי תוחלת מספר המשוואת המתקיימות הוא חצי מכל המשוואות.

## 6.2 בעיית חתך מינימום - Min-Cut

נתון גרף סופי לא מכוון $G = (V, E)$. נרצה למצוא את החתך $F \subseteq E$ שגודלו מינימלי.
אנו יודעים למצוא באמצעות זרימת מקסימום חתך מינימלי, אבל הבעיה היא שלא מדובר בחתך גלובלי, כלומר לאחר הבחירה של המקור והבור, צמצמנו חלק מהאפשרויות.
נוכל לפתור זאת לכאורה באמצעות $n - 1$ הפעלות של זרימת מקסימום.
אבל אפשר למצוא אלגוריתם פשוט יותר. לשם כך נגדיר את ההגדרה הבאה.

**הגדרה**
יהי גרף $G = (V, e)$ נגדיר כיווץ קשת $e = \{u, v\} \in E$ על ידי $G \setminus e = (V', E')$ כאשר:

$$V' = \{V \backslash \{u, v\}\} \cup \{uv\}$$
$$E' = \{E \backslash \{\{x, y\} : x = u \vee y = v\}\} \cup \{\{x, uv\} : x \notin \{u, v\}, \{x, u\} \in E \vee \{x, v\} \in E\}$$

(לוקחים את שני הקודקודים, מוחקים אותם, יוצרים חדש ומחברים את כל הקשתות שהיו בקודקודים המקוריים).

**האלגוריתם**

1. כל עוד $|V(G)| > 2$:
   (א) תבחר $e \in E(G)$ בהסתברות אחידה.
   (ב) $G \leftarrow G \setminus e$ .
2. תחזיר את קבוצת הקשתות שמחברת בין שני הצמתים שנשארו.

אנחנו בכל פעם מדביקים שני קודקודים, עד שנגיע לשני קודקודים ונקבל 2 קבוצות של קודוקדים מהגרף המקורי (עלולות להיווצר קשתות מקבילות אבל לא אכפת לנו).

**למה**
אם $F \subseteq E$ חתך מינימום ב-$G$, אזי $|E(G)| \geq \frac{|V| \cdot |F|}{2}$.
**הוכחה**

נסמן את $|F| = n$.

הגודל של דרגה מינימלית בגרף הוא לפחות $|F|$. כלומר $\min\limits_{v} d\left(v\right) \geq n$.[37]

אנו יודעים כי סכום הדרגות בגרף הינו $2\left|E\left(G\right)\right|$ ולכן בפרט נקבל כי :

$$2\left|E\left(G\right)\right| \geq |F|\,|V| \Rightarrow \left|E\left(G\right)\right| \geq \frac{|V|\,|F|}{2}$$

כנדרש.

**מסקנה**

ההסתברות כי $F \subseteq E\left(G \setminus e\right)$ קטנה או שווה ל-$1 - \frac{2}{|V|}$.

**הוכחה**

$$\mathbb{P}\left[F \subseteq E\left(G \setminus e\right)\right] = \mathbb{P}\left[e \notin F\right] \overset{\underbrace{\text{הסבתרות אחידה}}_{\downarrow}}{=}$$

$$\frac{\left|E\left(G\right) \setminus F\right|}{\left|E\left(G\right)\right|} \geq$$

$$\frac{\left|E\left(G\right)\right| - |F|}{\left|E\left(G\right)\right|} =$$

$$1 - \frac{|F|}{E\left(G\right)} \overset{\underbrace{\text{א"ש קודם}}_{\downarrow}}{\geq}$$

$$1 - \frac{|F|}{|V| \cdot \frac{|F|}{2}} = 1 - \frac{2}{|V|}$$

**משפט**

תהי $F \subseteq E\left(G\right)$ התשובה של האלגוריתם. אזי:

$$\mathbb{P}\left[F = F_{\min}\right] \geq \prod_{k=3}^{|V|}\left(1 - \frac{2}{k}\right)$$

**הרצאה מס' 23:**

**יום ראשון**

13.06.21

**הוכחה**

נסמן את $\left|V\left(G\right)\right| = n$ ונוכיח באינדוקציה על $n$.

<u>בסיס האינדוקציה עבור $n = 2$:</u>

---

[37] דרך לחשוב על זה: אפשר לומר שחתך מינימום הוא בעצם מספר הצלעות המינימלי שנדרש על מנת להפוך את הגרף ללא קשיר. אם נניח בשלילה כי $\min\limits_{v} d\left(v\right) < n$, כלומר יש קודקוד אחד לפחות שמספר שכניו קטן ממספר הצלעות בחתך, נוכל פשוט להעיף את כל השכנים שלו וקיבלנו שני רכיבי קשירות, כלומר חתך קטן יותר.

במקרה זה הטענה טריוויאלית (כי צד ימין הוא $1$ ו-$F$ זה כל הקשתות בין שני הקודקודים, כלומר בהכרח החתך מינימום היחיד שיש בגרף זה).

צעד האינדוקציה

נניח כי הטענה נכונה לכל גרף בעל $n-1$ קודקודים, ונוכיח עבור $G$ עם $n$ קודקודים.

$$\begin{aligned}
&\mathbb{P}\left[F=F_{\min}\right] \overset{\underbrace{\text{התפלגות מותנית}}_{\downarrow}}{=} \\
&\mathbb{P}\left[F_{\min}\subseteq E\left(G\setminus e\right)\right]\cdot\mathbb{P}_{G\setminus e}\left[F=F_{\min}\mid F_{\min}\subseteq E\left(G\setminus e\right)\right]\geq \\
&\underbrace{\left(1-\frac{2}{n}\right)}_{\text{מהלמה}}\underbrace{\prod_{k=3}^{n-1}\left(1-\frac{2}{k}\right)}_{\text{הנחת האינדוקציה}}=\prod_{k=3}^{|V|}\left(1-\frac{2}{k}\right)
\end{aligned}$$

הרעיון של המעבר הראשון הוא כי אנחנו כופלים את הסיכוי ש"שרד את הכיווץ הראשון עם הקשת $e$", ואז בהינתן העובדה שהוא שרד את הכיווץ הראשון, "מה הסיכוי שנשרוד עד הסוף", שאז נגיע לחתך מינימום.

כנדרש.

**מסקנה**

$$\mathbb{P}\left[F=F_{\min}\right]\geq\frac{1}{\binom{n}{2}}$$

**הוכחה**

$$\begin{aligned}
&\prod_{k=3}^{n}\left(1-\frac{2}{k}\right)=\prod_{k=3}^{n}\frac{k-2}{k}= \\
&\frac{1\cdot2\cdot3\cdot\ldots\cdot n-2}{3\cdot4\cdot5\cdot\ldots\cdot n}=\frac{2}{\left(n-1\right)n}=\frac{1}{\binom{n}{2}}
\end{aligned}$$

**מסקנה**

בכל גרף $G$ על $n$ קודקודים, מספר חתכי המינימום הוא לכל היותר $\binom{n}{2}$.

**מסקנה**

$$\frac{2}{\left(n-1\right)n}\geq\frac{1}{n^{2}}$$

כלומר, הסיכוי שהאלגוריתם ייכשל הוא לכל היותר $1-\frac{1}{n^2}$. אם נריץ את האלגוריתם באופן בלתי תלוי $N$ פעמים, וניקח את החתך הקטן ביותר מבין $N$ החתכים שחושבו, סיכויי הכישלון הם לכל היותר:

$$\left(1-\frac{1}{n^2}\right)^N \le e^{-\frac{N}{n^2}}$$

כלומר, אחרי $n^2$ נסיונות, מצליחים בהסתברות של לפחות $1-e^{-1}$.

**סיבוכיות**

עלינו לבצע $n^2$ איטרציות. בכל איטרציה אנו מבצעים $n-2$ כיווצים, אך עלינו להגריל קשת בכל פעם, דבר שתלוי במספר הקשתות ב-$G$.

לכן, הסיבוכיות הינה $O\left(mn^2\right)$, כאשר $m$ הוא מספר הקשתות ב-$G$ .

## 6.3 פולינומים מרובי משתנים

**הרצאה מס' 24:**

**יום שלישי**

15.06.21

נתון שדה $\mathbb{F}$ ופולינום $P$ מעל לשדה.

תחילה, נתבונן בפולינום $P$ במשתנה אחד. כלומר $P \in \mathbb{F}[x]$, כאשר $P = \sum\limits_{i=0}^{n} a_i x_i$.

**הגדרה**

$P = 0$ אם ורק אם בהצגה של $P$ כסכום מונומים, כל המקדמים $a_{\alpha_1,\ldots,\alpha_n}$ הם 0.

**בדיקת זהות פולינומים**

המטרה שלנו היא לבדוק אם $P$ הוא זהותית 0? כלומר, אם לכל הצבה של איבר $a \in \mathbb{F}$ ב-$x$ מקבלים ערך 0. אם הפולינום מיוצג באופן המפורש לעיל, התשובה טריוואלית - נבדוק אם כל המקדמים הם 0.[38]

אם כך, אם $P$ הוא פולינום עם משתנה אחד, הבעייה קלה. אמנם, מה יקרה אם $P \in \mathbb{F}[x_1, x_2, \ldots, x_n]$ אבל $P$ לא רשום מפורשות כסכום של מונומים. מה הכוונה? למשל, אם מופיעה התצוגה הבאה:

$$\sum_{\alpha_1,\ldots,\alpha_n} a_{\alpha_1,\ldots,\alpha_n} x_1^{\alpha_1} x_2^{\alpha_2} \cdot x_n^{\alpha_n}$$

אזי כמו במקרה הקודם, התשובה הינה טרוויאלית. לכן נניח כי הפולינום נתון באופן לא מפורש כזה.

לדוגמה, מכפלה של פונקציות ליניאריות:

$$\prod_i \left(c_i^{(i)} x_1 + c_2^{(i)} x_2 + \ldots + c_n^{(i)} x_n + c_0^{(i)}\right)$$

זה לא כל כך פשוט, וזה עלול לנפח את הייצוג לגודל אקספוננציאלי.

נניח כי לכל הצבה של ערכים מ-$\mathbb{F}$ למשתנים $x_1, \ldots, x_n$ אפשר לחשב את ערך הפולינום בנקודה הזאת.

---

[38] מסתבר שהבעיה הזאת עוזרת לפתרון כל מיני בעיות, נראה חלק בהמשך.

### 6.3.1 משפט Schwartz-Zippel

יהי $P \in \mathbb{F}\left[x_1,\dots,x_n\right]$ שאיננו זהותית 0.

תהי $A$, קבוצה סופית של איברי $\mathbb{F}$. נגריל הצבה $x_1=a_1,\dots,x_n=a_n$ לכל $1\le i\le n$. כאשר ה-$a_i$ כולם בלתי תלויים הדדית וכל אחד מתפלג אחיד ב-$A$, אזי:

$$\mathbb{P}\left[P\left(a_1,\dots,a_n\right)=0\right]\le\frac{\deg\left(P\right)}{|A|}$$

כאשר $\deg\left(P\right)$ היא המעלה של $P$.

(הטענה הזאת מאפשרת למעשה למצוא שיטה לבדוק האם $P$ הוא זהותית 0, בתנאי שהקבוצה $A$ גדולה מספיק. זה יכול לעבוד גם בשדות סופיים, כל עוד $\frac{\deg(P)}{|A|}$ מספיק קטן, וממילא ככל ש-$|A|>\deg\left(P\right)$)

**הוכחה**

באינדוקציה על $n$ (מספר המשתנים).

בסיס האינדוקציה, עבור $n=1$:

במקרה זה הטענה נכונה כי מספר השורשים של פולינום במשתנה אחד ממעלה $d$ שאיננו זהותית 0 הוא לכל היותר $d$[39]. לכן עלינו לבחור אחד מ-$d$ הערכים ולכן נקבל $\frac{d}{|A|}$, כי מדובר בהסתברות אחידה.

צעד האינדוקציה

נניח כי הטענה נכונה עבור $n-1$ ונוכיח עבור $n$.

נכתוב את הפולינום בצורה שונה[40] (גם אם אנחנו מקבלים אותו בצורה אחרת):

$$P\left(x_1,\dots,x_n\right)=\sum_{i=0}^{d}x_1^iP_i\left(x_2,\dots,x_n\right)$$

$P$ איננה זהותית 0 לפי ההנחה, ולכן קיים $i$ עבורו $P_i$ איננו זהותית 0. נבחר $i$ מקסימלי כזה. המעלה של $P_i$ היא לכל היותר $d-i$.

לפי האינדוקציה נקבל כי:

$$\mathbb{P}\left[P_i\left(a_2,\dots,a_n\right)=0\right]\le\frac{d-i}{|A|}$$

אם $P_i\left(a_2,\dots,a_n\right)\neq 0$, נקבל כי:

---

[39] "המשפט היסודי של האלגברה. את זה לומדים בשיעורי אלגברה לינארית, לא כאן". י"ר.

[40] דוגמה:

$$\begin{aligned}P&=\left(x_1+2x_2+x_3\right)\left(2x_1+x_2-x_3\right)\\&=2x_1^2+x_1x_2-x_1x_3+4x_1x_2+2x_2^2-2x_2x_3+2x_1x_3+x_2x_3-x_3^2\\&=2x_1^2+2x_2^2-x_3^2+5x_1x_2+x_1x_3-x_2x_3\\\Rightarrow P&=x_1^0\overbrace{\left(2x_2^2-x_3^2-x_2x_3\right)}^{P_0}+x_1^1\overbrace{\left(5x_2+x_3\right)}^{P_1}+x_1^2\overbrace{\left(2\right)}^{P_2}\end{aligned}$$

$$\sum_{j=0}^{d} x_1^j P_j\left(a_2,\ldots,a_n\right)$$

הוא פולינום במשתנה אחד שאיננו זהותית 0 (כי המקדם של $x_1^i$ איננו 0). המעלה של הפולינום הזה היא $i$- זה האינדקס המקסימלי שבחרנו להיות שונה מ-0. ולכן:

$$\mathbb{P}\left[\sum_{j=0}^{d} a_1^j P_j\left(a_2,\ldots,a_n\right)=0 \mid P_i\left(a_2,\ldots,a_n\right)\neq 0\right]\leq \frac{i}{|A|}$$

נסמן ב-$E_1$ את המאורע $P\left(a_1,\ldots,a_n\right)$ ונסמן ב-$E_2$ את המאורע $P_i\left(a_2,\ldots,a_n\right)=0$.
מנוסחת ההסתברות השלימה, עולה:

$$\begin{aligned}
&\mathbb{P}\left[E_1\right] \overset{\overbrace{\text{נה"ש}}^{}\downarrow}{=}\\
&\mathbb{P}\left[E_1\cap E_2\right]+\mathbb{P}\left[E_1\cap E_2^c\right] \overset{\overbrace{\text{בייס}}^{}\downarrow}{=}\\
&\mathbb{P}\left[E_1\mid E_2\right]\mathbb{P}\left[E_2\right]+\mathbb{P}\left[E_1\mid E_2^c\right]\overset{\overbrace{\text{הסתברות קטנה מ-1}}^{}\downarrow}{\mathbb{P}\left[E_2^c\right]\leq}\\
&\mathbb{P}\left[E_2\right]+\mathbb{P}\left[E_1\mid E_2^c\right]\leq\\
&\frac{d-i}{|A|}+\frac{i}{|A|}=\frac{d}{|A|}
\end{aligned}$$

**מסקנה**

אם $\mathbb{F}$ שדה מספיק גדול, אזי קיים אלגוריתם הסתברותי יעיל עבור הבעייה הבאה:

קלט

פולינום רב משתנים $P$ ממעלה מירבית $d$, בייצוג שמאפשר חישוב יעיל של ערכו בהינתן הצבה למשתנים.

הפלט:

זיהוי אם $P$ הוא זהותית 0.

**הוכחה**

נבחר מ-$\mathbb{F}$ קבוצה $A$ שגודלה $|A|=2d$. נגריל הצבה מקרית מ-$A$ ונציב ב-$P$.

נחזור על פעולה זאת $M$ פעמים. אם באחת ההצבות קיבלנו ערך שונה מ-0, נחזיר ש-$P$ איננו זהותית 0.

אחרת, נחזיר ש-$P$ זהותית 0.

מההוכחה הקודמת נקבל כי הסיכוי שכל $M$ הנסיונות יחזירו ערך 0 הוא לכל היותר:

$$\left(\frac{d}{2d}\right)^M=2^{-M}$$

כלומר, זה קטן באופן אקספוצינניאלי בגודל הקלט.

זמן הריצה

עלינו להגריל $M$ פעמים ערכים $a_1,\dots a_n$ ולהציב את הערכים הללו ב-$P$. לכן זמן הריצה הוא

$$(\text{סיבוכיות ההגרלה}+\text{סיבוכיות ההצבה})\cdot M$$

### 6.3.2 שימוש מעניין

נתון גרף דו צדדי $G=(V_L,V_R,E)$ שמקיים כי $|V_2|=|V_R|$. נרצה לבדוק האם קיים ב-$G$ שידוך מושלם.
נגדיר מטריצה ריבועית סימבולית $M$. השורות ממוספרות באיברי $V_L$ והעמודות ממוספרות באיברי $V_R$:

$$M_{i,j}=\begin{cases}x_{ij} & \{i,j\}\in E\\ 0 & else\end{cases}=\text{EdmondMatrix}$$

הדטרמינטה $\det(M)$ היא פולינום במשתנים $\{x_{ij}:\{i,j\}\in E\}$[41]:

$$\sum_{\sigma:V_L\underset{\text{חח"ע ועל}}{\rightarrow}V_R}(-1)^{|\sigma|}\sum_{i\in V_L}M_{i\sigma(i)}$$

אם קיים שידוך מושלם $\sigma:V_L\to V_R$ אזי:

$$\prod_{i\in V_L}M_i\sigma(i)=\prod_{i\in V_L}x_i\sigma(i)$$

והמקדם של המונום הוא $+1$ או $-1$.
אם אין שידוך מושלם, אזי הפולינום $\det(M)$ הוא זהותית 0.
אם נציב ערכים למשתנים $x_{ij}=a_{ij}$ לכל $\{i,j\}\in E$. אזי $M(\{a_j\})$ היא קלה לחישוב והיא מטריצה נומרית ואז $\det(M\{a_j\})$ ניתנת לחישוב בזמן פולינומי. [42]

מה עשינו כאן, בתכל'ס? קודם לכן ראינו אלגוריתם הסתברותי לבדיקה האם פולינום מסוים הוא זהותית 0. עכשיו אנחנו מנסים לקחת את הבעיה הזאת ולהנדס אותה במונח שיאפשר לנו למצוא זיווג מושלם בגרף. יש בעצם התאמה חח"ע ועל בין המושג זיווג מושלם והמושג "התאמה חח"ע ועל".

---

[41] זה סתם כתיב מסובך. תזכרו בפולינום האופייני מליניארית 2 ובכלל בכך שבכל פעם אנחנו מכפילים מוצאים דטרמינטה קטנה יותר.

[42] "אולי אתם לא יודעים למה, אבל לא משנה". י"ר.

אינטואיטיבית, קצת כמו שראינו בליניארית כי מטריצה הפיכה היא מטריצה שהדרמיננטה שלה שונה מאפס, ומטריצה הפיכה היא מטריצה חח"ע ועל, כך גם במקרה שלנו. מה אנחנו עושים? אנחנו מקבלים גרף, ומייצרים **מטריצה** שמייצרת בתור משתנים את הצלעות של הגרף. קיבלנו מטריצה עם מלא מלא משתנים, ובאמצעות הדרמיננטה נוכל לייצר אחלה פולינום מרובה משתנים. כעת, באמצעות האלגוריתם ההסתברותי שלנו לבירור האם פולינום מסוים הוא זהותית אפס, נמצא האם הפולינום שלנו הוא זהותית 0, וממילא נדע האם קיימת התאמה חח"ע ועל - זיווג מושלם, כפי שחיפשנו.

## 6.4 זיהוי תבניות

**הרצאה מס' 25:**

**יום ראשון**

20.06.21

הטכניקה נקראת "טביעת אצבע".
נתון טקסט $T$ - רצף של אותיות מתוך א"ב $\Sigma$ כך ש-$d=|\Sigma|$ ו-$n=|T|$.
נתונה תבנית $P$ - רצף של אותיות מתוך $\Sigma$ וגם $m=|P|$ .
נניח כי $m<<n$.
נרצה למצוא את כל המופעים של $P$ בתוך $T$.

**האלגוריתם הנאיבי לפתרון בעיה זו**

נעבור על המקומות ב-$T$ שבהם $P$ יכולה להתחיל. כלומר לכל $1\leq i\leq n-m+1$[43] נשווה בין $P$ ובין $T\left[i\ldots i+m-1\right]$.
ניתן להניח כי $\Sigma=\left\{0,\ldots,d-1\right\}$.

**סיבוכיות הזמן**

סיבוכיות הזמן תהיה $O\left(n\cdot m\right)$ פעולות של השוואה של אותיות מ-$\Sigma$ או גישה לאיברים של $T$ או $P$ על פי אינדקס או קידום של אינדקס. אם נרצה להתבונן בסיבוכיות ע"פ ביטים, אזי אות ניתנת לייצוג על ידי $O\left(\log_2 d\right)$ ביטים וגם אינדקס ניתן לייצוג על ידי $O\left(\log_2\right)n$ ביטים.

**סיבוכיות המקום**

מלבד הקלט $T,P$ והפלט, נצטרך להחזיק $O\left(1\right)$ תאי זיכרון שמחזיקים כל אחד אינדקס.

סיבוכיות הזמן לא משהו עבורנו, ביחס לגודל הקלט שהוא אחלה. האם נוכל לקצר זאת?

### 6.4.1 האלגוריתם של Karp ו-Rabin

אפשר להתייחס ל-$P$ וגם ל-$T\left[i,\ldots,i+m-1\right]$ כאל מספר שרשום בבסיס $d$.
כלומר, נקבל כי $P=d^{m-1}P\left[1\right]+d^{m-2}P\left[2\right]+\ldots+d^{0}P\left[m\right]$.
וגם נקבל כי :

$$t_i=d^{m-1}T[i]+d^{m-2}T[i+1]+\cdots+d^{0}T[i+m-1]$$

נוכל להבחין כי $T\left[i,\ldots i+m-1\right]=P$ אם ורק אם $t_i=P$.
כמו כן, נשים לב כי את $t_i$ קל לעדכן במספר קבוע של פעולות. כלומר נעדכן כך:

$$t_{i+1}=dt_i-d^{m}T[i]+T[i+m]$$

בינתיים לא שיפרנו דבר כי אנחנו עובדים עם מספרים גדולים. נקבל כי $0\leq P,t_i<d^{m}$ - כלומר עלולים לדרוש $m\log_2 d$ ביטים כדי לייצג אותם. דהיינו, לא חסכנו את התלות ב-$m$ בצורה זאת.

---

[43] כאן ברור מדוע $m<<n$. אם $m>n$ ברור כי התבנית לא נמצאת. ואם $m$ קרוב ל-$n$, יש מספר מקומות קטן לבדוק.

סך הכל, נצטרך $O\left(nm\log_2 d\right)$ פעולות על ביטים על מנת לבצע את החיפוש.

הרעיון של האלגוריתם שנציע דומה לטבלת גיבוב. בטבלת גיבוב, נרצה להחזיק טבלה שהמפתחות בה הם ממרחב גדול, אך אנו נחזיק טבלה קטנה יותר. במקרה של גיבוב, אנחנו יכולים להשתמש בפונקציות גיבוב מופרעות, אבל במקרה שלנו מדובר על 'הזזה של חלון על פני הטקסט'. לכן, נצטרך למצוא איזשהו גיבוב שקל 'להזיז' את החלון.

נבחר קבוצה גדולה $Q$ של מספרים ראשוניים, וגם נבחר $q \in Q$ באופן מקרי בהתפלגות אחידה, ונבצע את כל החישובים מודולו $q$.

נסמן $q_{\max} = \max q \in Q$. אם $\log_2 q_{\max}$ קטן יחסית לעומת $m$ (כלומר $q_{\max} << 2^m$), החישובים האריתמטיים יעילים.

כמובן, אם $t_i \not\equiv p \mod q$ אזי $T\left[i, \dots i+m-1\right] \neq P$. מה לגבי $t_i \equiv p \mod q$? לא מובטח לנו כי התבנית $P$ תואמת לטקסט $T\left[i, \dots i+m-1\right]$. עלינו לבדוק - בפרט אם $P$ מופיעה ב-$T$ $s$ פעמים, נשלם לכל הפחות $O\left(sm\right)$ פעולות.

מה לגבי מקומות שבהם התבנית שונה מהטקסט?

אם $t_i \neq P$, עבור אילו ערכים של $q$ נקבל כי $t_i \equiv p \mod{}_q$? זה קורה אם ורק אם $q$ מחלק את $|t_i - P|$. כלומר, $q$ הוא גורם ראשוני של $|t_i - P|$. כמה גורמים ראשוניים כאלו יכולים להיות? ברור כי $0 \leq |t_i - P| \leq d^m$ .

הביטוי $|t_i - P|$ הוא מכפלה של כל הגורמים הראשוניים שלו, שהם בעצמם גדולים או שווים ל-2.

לכן מספר הגורמים הראשוניים השונים זה מזה הוא קטן ממש מ-$\log_2\left(d^m\right) = m\log_2 d$.

כלומר, נקבל כי:

$$\mathbb{P}\left[t_i \equiv p \mod q \mid t_i \neq P\right] < \frac{m\log_2 d}{|Q|}$$

סך הכל, תוחלת מספר המקומות $i$ שנבדוק לריק היא קטנה מ-$\frac{nm\log_2 d}{|Q|}$. אם כך, תוחלת מספר הפעולות שנבצע הינה:

$$O\left(\underbrace{m+n}_{t_i,P\ \text{חישובים של}} + m\left(\underbrace{s}_{\text{מחרוזת נמצאת}} + \underbrace{\frac{nm\log_2}{|Q|}}_{\text{מחרוזת לא נמצאת}}\right)\right)$$

אם כך, אם $s$ נמצא מספר רב פעמים של מקומות, לא עשינו יותר מדי. אבל אם המחרוזת יחסית נדירה ו-$Q$ גדול באופן יחסי, כלומר $|Q| >> nm\log_2 d$, ניצחנו.

כדי לבחור את $Q$ אפשר להשתמש בצפיפויות של המספרים הראשוניים:

מספר המספר הראשוניים שקטנים מ-$x$ הוא $\frac{x}{\ln(x)}\left(1 \pm o\left(1\right)\right)$.

# חלק II

# תרגולים

תרגול מס' 1:

יום שלישי

16.03.21

## 1 חזרות על נושאים מתמטיים

### 1.1 טענות לוגיות

חשוב להבחין מהי משמעות 'טענה טרוויאלית'. נאמר כי היא טרוויאלית, אם היא נובעת ישירות מההגדרה.
לדוגמה, הטענה כי בעץ עם $n$ קודקודים, יש $n-1$ צלעות, **איננה טרוויאלית**. לעומת זאת, הטענה כי גרף $G$ עם 2 רכיבי קשירות אינו עץ היא טרוויאלית.

נאמר שטענה מתקיימת באופן ריק, אם היא מהצורה "לכל האיברים ב-$S$" וגם $S = \emptyset$.
למשל, הטענה כי "כל הפינגווינים שצופים בתרגול הם בעלי מקור ירוק" - **מתקיימת באופן ריק**.
הטענה כי "כל הסטודנטים שצופים בשיעור, הם פינגווינים - **לא מתקיימת באופן ריק**.

### 1.1.1 אינדוקציה

שיטה להוכחת קבוצה גדולה (בדרך כלל אינסופית, בת מנייה) של טענות.
בשלב הראשון, נצטרך למספר את הטענות.

בסיס:

נוכיח בידיים קבוצה קטנה של טענות מ-$A$, בדרך כלל $a_1$.

הנחה:

נניח נכונות עבור קבוצה גדולה של טענות, בדרך כלל $a_k$.

צעד:

נוכיח את נכונות $a_{k+1}$ בהתבסס על הנחה שהנחתי.

**דוגמה**

לכל מספר טבעי מתקיים: $\sum_{n=1}^{n} i = \frac{n(n+1)}{2}$.

בסיס:

עבור $n=1$ מתקיים כי $\sum_{i=1}^{1} i = 1 = \frac{1(1+1)}{2}$.

הנחה:

נניח נכונות עבור $k \in \mathbb{N}$.

צעד:

עבור $k+1$ מתקיים:

$$\sum_{i=1}^{k+1} i = \sum_{i=1}^{k} i + (k+1) \overset{\text{הנחה}}{=} \frac{k(k+1)}{2} + (k+1) =$$
$$\frac{k(k+1)+2(k+1)}{2} = \frac{(k+1)(k+2)}{2}$$

**טעויות נפוצות**

הקבוצה $\mathbb{N}$ היא סופית (שקרי).

<u>בסיס:</u>

עבור $n=1$, הקבוצה $\{1\}$ היא מגודל סופי.

<u>הנחה:</u>

נניח נכונות עבור כל $1 \leq k \leq n$.

<u>צעד:</u>

עבור $n+1$, נשים לב כי $\{1,\ldots,n+1\} = \{1,\ldots,n\} \cup \{n+1\}$, כיוון שאיחוד קבוצות סופית הוא בהכרח סופי, הקבוצה $\{1,\ldots,n+1\}$ סופית.

בנוסף, פעמים רבות אנו מוכיחים באינדוקציה מבלי לוודא שהצעד מאפשר מעבר בין הבסיס לטענה הבאה (דוגמת הסוסים באותו הצבע). חשוב לוודא זאת כשמשתמשים באינדוקציה.

## 1.2 חסמים אסימפטוטיים

**הגדרה**

יהיו $f,g:\mathbb{N}\to\mathbb{N}$.

נאמר ש-$g$ היא **חסם תחתון** של $f$ ונסמן $f\left(n\right)=\Omega\left(g\left(n\right)\right)$ אם קיימים $n_0\in\mathbb{N}$ ו-$c\in\mathbb{R}^+$ המקיימים:

$$\forall n>n_0 : c\cdot g(n) \leqslant f(n)$$

נאמר ש-$g$ היא **חסם עליון** של $f$ ונסמן $f\left(n\right)=O\left(g\left(n\right)\right)$ אם קיימים $n_0\in\mathbb{N}$ ו-$c\in\mathbb{R}^+$ המקיימים:

$$\forall n>n_0 : c\cdot g(n) \geq f(n)$$

אם $g$ היא גם **חסם עליון וגם חסם תחתון** של $f$, נאמר ש-$g$ חסם הדוק של $f$ ונסמן $f\left(n\right)=\Theta\left(g\left(n\right)\right)$.

**דוגמאות**

- $500\cdot\sqrt{n}=O\left(n\right)$. נבחר $n_0=1$ ו-$C=500$ ונקבל כי לכל $n>1$ מתקיים כי $500\cdot\sqrt{n}\leq 500\cdot n = C\cdot n$.
- $10\cdot n\cdot\log(\log(n))=\Omega(n)$. נבחר $n_0=4$ ו-$C=10$ ונקבל, לכל $n_0>4$:

$$10 \cdot n \cdot \log(\log(n)) \geqslant$$
$$10 \cdot n \cdot \log(\log(4)) = 10 \cdot n = C \cdot n$$

### 1.2.1 זמני ריצה ויעילות אלגוריתמים

**הגדרה**

נאמר על אלגוריתם שהוא רץ בזמן פולינומיאלי באורך הקלט (או בקיצור, אלגוריתם פולינומי) אם קיים $m \in \mathbb{N}$ כך שמספר הפעולות המקסימלי שמבצע האלגוריתם על קלט באורך $n$ הוא $O\left(n^m\right)$.

בקורס, נאמר שאלגוריתם הוא יעיל, אם הוא פולינומי.

### 1.2.2 זמני ריצה פסאודו פולינומיאלים

**דוגמא**

קלט - מספר טבעי $n \in \mathbb{N}$.

פלט - $1$ אם המספר ראשוני, $0$ אחרת.

<u>אלגוריתם:</u>

עבור $i = 2, \ldots, \sqrt{n}$ נבדוק האם $n$ מתחלק ב-$i$ ללא שארית. אם כן, נחזיר $0$, אם סיימנו לעבור על $2, \ldots, \sqrt{n}$ נחזיר $1$.

<u>זמן הריצה:</u>

עוברים על בערך $\sqrt{n}$ מספרים, לכל אחד מבצעים מספר קבוע של פעולות, לכן זמן הריצה הוא:

$$O\left(\sqrt{n}\right) = O\left(n^{\frac{1}{2}}\right) = O\left(2^{\log\left(n^{\frac{1}{2}}\right)}\right) = O\left(2^{\frac{1}{2}\log n}\right)$$

קיבלנו סיבוכיות אקספונציאלית לגודל הקלט. מתוך כך, נגיע להגדרה הבאה:

**הגדרה**

אלגוריתמים שרצים בסיבוכיות זמן ריצה פולינומית בגודל האבסולוטי של הקלט ולא באורכו, נקראים **אלגוריתמים פסאודו פולינומיאלים**.

**דוגמא לבעיה בקורס**

בעיה - מציאת האיבר הכפול.

<u>קלט:</u>

רשימה $A$ באורך $n+1$ המכילה את כל המספרים השלמים בין $1$ ל-$n$ עם חזרה בודדת.

למשל, $A = [1, 3, 4, 2, 4]$.

<u>פלט:</u>

מספר שלם $k \in \mathbb{N}$ המופיע פעמיים ב-$A$.

**אלגוריתם 3** ניסיון 1

```
1. לכל $i = 1, \ldots, n$:
    (א) לכל $j = i + 1, \ldots, n + 1$:
        i. אם $A[i] = A[j]$:
            א'. החזר את $A[i]$
```

הוכחת נכונות של ניסיון 1:

צריך להראות כי האלגוריתם מספר $k$ אם"ם $k$ מופיע פעמיים ב-$A$.

⇐: נניח שהאלגוריתם החזיר את $k$, אזי קיימים 2 אינדקסים $i \neq j$ כך שהתנאי $A[i] = A[j]$.
מתקיים מכאן ש-$k$ מופיע פעמיים ב-$A$.

⇒ נניח ש-$k$ מופיע פעמיים ב-$A$.
כיוון שהאלגוריתם עובר על כלל זוגות האינדקסים, הוא יעבור על 2 האינדקסים בהם מופיע $k$ ובאיטרציה זו יחזיר את $A[i] = k$.

ניתוח זמן ריצה של ניסיון 1:

המקרה הגרוע ביותר עבור אלגוריתם זה, הוא אם $A[n] = A[n-1]$. במקרה זה, אלגוריתם יבצע את כלל האיטרציות וזמן הריצה יהיה $n^2$.

**אלגוריתם 4** ניסיון 2

```
1. הגדר מערך $B$ בגודל $n$ ועדכן לאפס.
2. לכל $i = 0, \ldots, n$:
    (א) אם $B[A[i] - 1] = 0$
        i. עדכן את $B[A[i] - 1] = 1$.
    (ב) אחרת:
        i. החזר את $A[i]$.
```

זמן הריצה של ניסיון 2:

המקרה הגרוע ביותר הוא שהאיבר הכפול מופיע ב-$A[n]$. במקרה זה, האלגוריתם יבצע $O(n)$ פעולות. נשים לב כי באלגוריתם זה סיבוכיות הזיכרון גם היא $O(n)$.

**תרגיל**

נסו למצוא אלגוריתם שעובד בסיבוכיות מקום $O(1)$ וזמן ריצה $O(n)$.

# 2 אלגוריתמים חמדניים

**תרגול מס׳ 2:**

**יום שלישי**

06.04.21

**הגדרה**

אלגוריתם ייקרא חמדן אם בכל שלב הוא בוחר באפשרות הזמינה ביותר בלי להתחשב בהשלכות לטווח הרחוק.

סכמה כללית להוכחת **אופטימליות** אלגוריתם חמדן:

1. מוכיחים **חוקיות** של הפתרון החמדני $B$.
2. טענה באינדוקציה, שלכל $0 \leq k \leq m$, ישנו פתרון אופטימלי $C$ שמסכים עם $B$ על $k$ הצעדים הראשונים:

בסיס:

$k = 0$, אין מה להוכיח.

צעד:

מהנחת האינדוקציה עבור $k-1$ יש פתרון $C = \{b_1, b_2, \ldots, b_{k-1}, c_k, \ldots, c_n\}$ - כלומר פיתרון שמסכים עם $B$ על $k-1$ הצעדים הראשונים.

נבנה בעזרת $C$ פתרון $C'$ שמכיל גם את $b_k$ - האיבר ה-$k$ של $B$. צריך להראות חוקיות של $C'$, ואופטימליות בהתבסס על דרך הפעולה החמדנית של $B$.

מסקנה

גם עבור $k = m$ קיים פיתרון אופטימלי $C$ שמכיל את $m$ האיברים של $B$ ולכן $B$ אופטימלי.

## 2.1 בעיית תא הדלק הקטן

קלט:

$N \in N$, מספר הקילומטרים שניתן לנסוע עם **מיכל מלא**. כמו כן, נתונים $a_1, a_2, \ldots a_n$, מיקומים של **תחנות הדלק** במסלול. אנחנו מעוניינים למזער את מספר העצירות שהמכונית מבצעת.

הנחה: $a_{i-1} - a_i \leq N$ לכל $1 \leq i \leq n$.[44]

פלט:

תת סדרה $b_1, \ldots, b_n$ באורך מינימלי שמקיימת[45]:

1. $b_1 = a_1$, $b_m = a_n$
2. $b_{i+1} - b_i \leq N$ לכל $1 \leq i \leq m$.

פתרון:

---

[44] המרחק מכל תחנה קטן ממספר הק״מ שניתן לנסוע במרחק מלא. כלומר לא ייתכן ש׳ניתקע׳ באמצע הדרך.

[45] אנחנו מחפשים את מספר העצירות המינימלי.

---

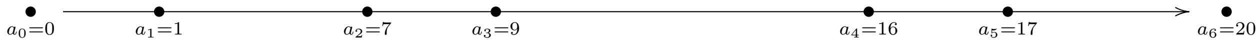
**אלגוריתם 5** בעיית תא הדלק הקטן $((a_1,\dots,a_n),N)$

---

```
1. $b_1 \leftarrow a_1$
2. $\text{prev} \to 1$.
3. לכל $i \to 2$ עד $(n-1)$:
    (א) אם $a_{i+1} - b_{\text{prev}} > N$:
        i. $b_{\text{prev+1}} \to a_i$.
        ii. $\text{prev++}$.
4. $b_{\text{prev+1}} \to a_n$
5. תחזיר את $(b_1,\dots,b_{\text{prev+1}})$.
```

---

הרעיון של האלגוריתם בפשטות הינו כזה: נאתחל את $b_1$ להיות נקודת ההתחלה, ובכל פעם נמשיך עד **המקום הרחוק ביותר** אליו נוכל להגיע, עד שנגיע אל היעד (ייתכן כי הפיתרון איננו יחיד).
קלט לדוגמה -$N=10$ - $(0,1,7,9,16,17,20)$.



**הוכחת נכונות**

חוקיות:

מדרך פעולת האלגוריתם, מתקיים כי $b_1 = a_1, b_m = a_n$.
עלינו להראות כי לכל $1 \le i \le m$ מתקיים כי $b_{i+1} - b_i \le N$ (כלומר, שיש מספר ק"מ קטן מספיק בין כל עצירה ועצירה).
נניח בשלילה כי קיים $1 \le j \le m$ שעבורו מתקיים כי $b_{j+1} - b_j > N$. מההנחה על הקלט, קיימת לפחות תחנה אחת בין $b_j$ ל-$b_{j+1}$ [46]. נסמן ב-$a_k$ את התחנה המקסימלית בקטע $[b_j, b_{j+N}]$ (הכי רחוק שאפשר להגיע).
כאשר $i = j$ נקבל כי $b_{\text{ prev}} = b_j$ (מהגדרת האלגוריתם). ממקסימליות $a_k$ בקטע, האלגוריתם היה צריך לבחור $b_{j-1} = a_k$ (התחנה הכי רחוקה שאפשר להגיע אליה) בסתירה לכך שלא הכניס את $a_k$ לסדרה ולכך ש-$b_{j+1} - b_j > N$.

אופטימליות (למת החלפה)

נסמן ב-$B = \{b_1,\dots,b_m\}$ את הפיתרון החמדן. על מנת להוכיח את האופטימליות, ניעזר בטענה הבאה:

**טענה**

לכל $1 \le k \le m$ , קיים פיתרון אופטימלי מהצורה $C = (b_1,\dots,b_k,c_{k-1},\dots,c_m)$.
הוכחה באינדוקציה על $k$.

בסיס:

$k = 1$. יהי $C$ פתרון אופטימלי. מחוקיות $C$ נובע כי $c_1 = a_1$ כנדרש (כולם מתחילים באותה נקודה).

הנחה:

נניח כי הטענה נכונה עבור $1 \le k < m$. כלומר, קיים $C = (b_1,\dots,b_k,c_{k+1},\dots,c_m)$ אופטימלי.

---

[46] ניזכר כי $a_{i-1} - a_i \le N$ לכל $1 \le i \le n$.

צעד:

נתבונן בפיתרון $C' = (b_1, \ldots, b_{k+1}, c_{k+2}, \ldots, c_m)$. עלינו להראות כי פיתרון זה אופטימלי (זה למעשה פיתרון שמתחיל ב-$k+1$ איברים של $b$). נשים לב כי $|C| = |C'|$ ולכן מספיק להראות **חוקיות** של $C'$ ואין צורך להראות אופטימילות.

**מחוקיות** $C$, מתקיים כי $c_{i+1} - c_i < N$ לכל $k+2 \leq i \leq m$.

**מחוקיות** $B$, מתקיים כי $b_{i+1} - b_i \leq N$ לכל $1 \leq i \leq k+1$.

נותר להראות כי $c_{k+2} - b_{k+1} \leq N$.

מהנחת האינדוקציה, אנו יודעים כי $c_{k+1} - b_k \leq N$ ולכן מדרך פעולת האלגוריתם, נקבל כי $c_{k+1} \leq b_{k+1}$ (כי אנחנו הולכים הכי רחוק).

ניזכר כי $c_{k+1} - b_{k+1} \leq c_{k+2} - c_{k+1} \leq N$ (כאשר המהלך האחרון נובע מחוקיות $c$ - הנחת האינדוקציה).

כלומר, הראינו כי $C'$ חוקי, ובפרט אופטימלי, כנדרש.

## 2.2 עצים פורשים מינימלים

**הגדרה**

עץ הוא גרף קשיר חסר מעגלים.

**מסקנות**

בין כל שני קודקודים בעץ $T$ יש בדיוק מסלול אחד.

הוספת צלע ל-$T$ סוגרת מעגל אחד .

**הגדרה**

יהיה $G = (V, E)$ גרף קשיר לא מכוון. אזי עץ פורש $T$ של $G$ הוא תת גרף של $G$ שהוא עץ שמכיל את כל קודקודי $G$.

**הגדרה**

תהי $w : E \to \mathbb{R}$ פונקציית משקל על $G$. המשקל של עץ פורש $T = (V, E_T)$ של $G$ הוא $w(T) = \sum\limits_{e \in E_T} w(e)$.

**הגדרה**

עץ פורש מינימלי Mst הוא עץ פורש של $G$ בעל משקל מינימלי,

כלומר לכל $T'$ פורש של $G$ מתקיים כי $w(T) \leq w(T')$.

**הגדרה**

יהי $G=(V,E)$ גרף לא מכוון ויהיו $A,B$ שתי קבוצות קודקודים המקיימות $A\cap B=\emptyset$ ו-$A\cup B=V$.
נגדיר את החתך $(A,B)$ להיות קבוצת הצלעות המקיימת:

$$(A,B)=\{(u,v)\in e \mid u\in A, v\in B\}$$

**בעיית עץ פורש מינימלי**

קלט:

גרף קשיר לא מכוון $G=(V,E)$ , פונקציית משקל $w:E\to\mathbb{R}$.

פלט:

עץ פורש מינימלי עבור $G$.

**כלל כחול וכלל אדום**

בהינתן גרף $G=(V,E)$ עם פונקציית משקל $w:E\to\mathbb{R}$, נגדיר את כללי הצביעה הבאים.

כלל כחול:

נבחר חתך שלא מכיל קשתות כחולות. נצבע את הצלע **הזולה** ביותר בכחול.

כלל אדום:

נבחר מעגל שלא מכיל קשתות אדומות, ונצבע את הקשת **היקרה** ביותר באדום.

**משפט**

יהי $G=(V,E)$ גרף קשיר לא מכוון שחלק מצלעותיו צבועות אדום וחלק בכחול באופן שרירותי. אם לא כל הצלעות צבועות, קיימת קשת **לא צבועה** שניתן לצבוע בעזרת הכלל הכחול או הכלל האדום.

**הוכחה**

נסתכל על הגרף $G'=(V,E')$ שמתקבל מהורדת **כל הצלעות האדומות** מ-$G$ ונתבונן ב-$e=(u,v)$ צלע לא צבועה ב-$G'$.

אם $e$ חלק ממעגל ב-$G'$ , כיוון שאין בו צלעות אדומות והוא גם חלק מ-$G$ ניתן להפעיל את **הכלל האדום**.

אם $e$ לא חלק ממעגל, אזי כל המסלולים בין $u$ ל-$v$ ב-$G'$ עוברים דרך $e$ (אחרת היה מעגל). הסרת $e$ מ-$G'$ משאירה את $v,u$ בשני רכיבי קשירות שונים, שנסמנם $U,S$.

נגדיר חתך $A,B$ על $G$ באופן הבא: $A=S$ ו-$B=V\setminus S$ (כל השאר). $e$ היא הצלע היחידה שאינה צבועה בחתך. אם קיימות צלעות נוספות בחתך, הן לא ב-$G'$ ולכן הן **אדומות**[47]. אם כך, אין צלעות כחולות בחתך ולכן ניתן להפעיל את הכלל הכחול על צלע זאת, וסיימנו.

**מתכון לאלגוריתם חמדן לבעיית העץ הפורש המינימלי:**

1. נאתחל $E_B=\emptyset, E_R=\emptyset$.
2. כל עוד יש צלעות לא צבועות בגרף, נבחר בין:
    (א) להפעיל את הכלל הכחול ולהוסיף את הצלע הנבחרת ל-$E_B$.

---

[47] כלומר, $e$ היא הצלע היחידה שמחברת בין שני רכיבי הקשירות לפי הדרך בה הלכנו. היא לא אדומה, ולא ייתכן שיש עוד כחולות, כי אחרת היה מעגל.

(ב) להפעיל את הכלל האדום ולהוסיף את הצלע הנבחרת ל-$E_R$.

3. נחזיר את $G = (V, E_B)$

**טענה (הוכחה בהרצאה)**

כל אלגוריתם שמתקבל מהמתכון לעיל מחזיר עץ פורש מינימלי ל-$G$.

**דוגמה 1 - האלגוריתם של פרים**

1. נתחיל עם $E_B = \emptyset$ ו-$V_T = \emptyset$.
2. נבחר קודקוד אקראי $V$ ונוסיף ל-$V_T$.
3. נפעיל את הכלל הכחול על החתך $(V, V \setminus V_T)$ ונצבע את הקשת הנבחרת $(u, v) \in E$ בכחול.
4. נוסיף את $v, u$ ל-$V_T$ ואת $(v, u)$ ל-$E_T$.
5. נחזור לשלב 3 עד ש-$V_T = V$.

**זמן ריצה**

פתרון נאיבי - לכל איטרציה, נעבור על כל הצלעות ונבחר את המינימלית שנמצאת בחתך ונקבל זמן ריצה של $O\left(V \cdot E\right)$.

תזכורת:

תור קדימות הוא מבנה נתונים אבסטרקטי המאפשר את הפעולות הבאות עבור מערך בגודל $n$:

□ אתחול $O\left(n\right)$.

□ שליפת מינימום $O\left(\log n\right)$.

□ הורדת ערך לאיבר $O\left(\log n\right)$.

פתרון טוב יותר:

נשמור בתור קדימויות - לכל קודקוד $V$ מחיר $C\left(v\right)$ המייצג את המשקל של הוספת $v$ ל-$V_T$. באתחול, $C\left(v\right) = \infty$ לכל $v \in V$.

בכל איטרציה, נוסיף את הקודקוד $u$ עם **משקל החיבור המינימלי**. לאחר הוספת קודקוד, נעדכן לכל $v$ בשכני $u$ את מחיר $C\left(v\right)$ להיות $\min\left(C\left(v\right), C\left(u\right) + w\left(u, v\right)\right)$ - כלומר המינימום מבין המחיר הנוכחי ובין מחיר $u$ והוספת הצלע המחברת בינו ובין השכן.

זמן הריצה החדש

$O\left(E + E\log V\right) = O\left(E \log V\right)$ כאשר זמן הריצה הראשון הוא על המיון ההתחלתי, והשני על הרצת האלגוריתם.

נכונות:

בכל איטרציה של האלגוריתם, הפעלנו תמיד את הכלל הכחול על החתך (יש בו רק צלעות לא צבועות, ובפרט לא כחולות). לכן מנכונות המתכון, גם האלגוריתם הזה מחזיר פיתרון אופטימלי.

**דוגמה 2 - האלגוריתם של קרוסקל.**

1. נתחיל עם $E_T=\emptyset$ ו-$V_T=\emptyset$.
2. נמיין את הצלעות בסדר עולה $w\left(e_1\right)\leq w\left(e_2\right)\leq\ldots\leq w\left(e_{|E|}\right)$.
3. נעבור על הצלעות לפי הסדר:
   - (א) אם הקשת $e=(u,v)$ סוגרת מעגל עם הצלעות ב-$E_T$ נפעיל על מעגל זה את **הכלל האדום**, ונצבע את $e$ באדום.
   - (ב) אחרת, $e$ הצלע הקלה ביותר בחתך $\left(V_T\cup\{u\}, V\setminus V_T\cup\{u\}\right)$. לכן נצבע אותה **בכחול** ונוסיף את $e$ ל-$E_T$ ואת $v\in V_T$.

זמן הריצה

בשלב הראשון, נצטרך למיין את הצלעות, כלומר זמן ריצה של $O\left(|E|\log\left(|E|\right)\right)$. על מנת למצוא את המעגלים - אפשר לעשות זאת עם Union-Find בזמן ריצה של $O\left(E\cdot\alpha\left(n\right)\right)$, כאשר $\alpha$ היא הופכית לפונקציית אקרמן. לכן, זמן הריצה הוא $O\left(|E|\log\left(|E|\right)\right)$.

הוכחת נכונות

בתרגיל 2.

## 2.3 מטרואידים

**תרגול מס' 3:**

**יום שלישי**

13.04.21

**הגדרה**

מטרואיד הוא זוג סדור $(E,I)$ כאשר $E$ הינה קבוצה סופית כלשהי ו-$I\subseteq 2^{|E|}$ קבוצה לא ריקה המקיימת:

1. ירושה – אם $A\in I$ וגם $B\subseteq A$ אז $B\in I$.
2. הרחבה – אם $A,B\in I$ וגם $|A|>|B|$ אז קיים $a\in A\setminus B$ כך ש- $B\cup\{a\}\in I$

**דוגמאות**

1. $M_1=\left(E_1,I_1\right)$ כאשר $E=\{1,2\}$ ו-$I=\{\{1\},\{2\},\{1,2\}\}$ - לא מטרואיד. נשים לב כי בכל מטרואיד $\emptyset\in I$ משום ש- $I$ לא ריקה ו-$\emptyset$ היא תת קבוצה של כל קבוצה.

2. $M_2=\left(E_2,I_2\right)$ כאשר $E_2=\{1,2,3,4\}$ ו-$I_2=\{\emptyset,\{1\},\{2\},\{3\},\{4\},\{1,2\},\{3,4\}\}$. הירושה מתקיימת, אבל ההרחבה לא, כי אם למשל נתבונן ב-$A=\{1,2\},B=\{3\}$ לא מתקיימת תכונת ההרחבה. אם נגדיר $I_2'=I_2\cup\{1,4\}\cup\{2,3\}$ אזי $M_2'=\left(E_2,I_2'\right)$ מטרואיד.

3. המטרואיד הוקטורי - נסמן $M_v=\left(E_v,I_v\right)$ כאשר $E_v$ קבוצת וקטורים במרחב וקטורי כלשהוא, ו-$I_v$ קבוצות של וקטורים בת"ל מתוך $E_v$ (ההוכחה נשארת כתרגיל).

4. המטרואיד הגרפי - יהי $E=(V,E)$ גרף.
נסמן $M_G=\left(E_G,I_G\right)$ כאשר $E_G=E$ קבוצת הצלעות ו-$I_G=\{A\subseteq E\mid G_A=\langle V,A\rangle \text{ is a forest }\}$.
ראשית, נבחין כי $\emptyset\in I_G$ משום שגרף ריק הוא חסר מעגלים.
ירושה

אם $A \in I_G$ הוא יער $G_A$ יער. אם $B \subseteq A$ תת קבוצת צלעות מ-$A$ אזי $B$ גם יער, שכן $G_B$ לא מכיל מעגלים.

הרחבה

אם $A, B \in I_G$ וגם $|B| < |A|$ נראה שיש צלע $a \in A \setminus B$ כך ש- $B \cup \{a\} \in I_G$.

נגדיר $G_A = (V, A)$ וגם $G_B = (V, B)$ ונבחין כי שניהם חסרי מעגלים.

**טענת עזר**

הוספת צלע לגרף מקטינה את מספר רכיבי הקשירות ב-1 או סוגרת מעגל.

מכיוון שבגרף $G_A$ אין מעגלים, **מספר רכיבי הקשירות בו קטן ממש** מאשר ב-$G_B$. נטען שקיים רכיב קשירות ב-$G_A$ שנחתך עם שני רכיבי הקשירות ב-$G_B$. דבר זה נכון, שכן אחרת מספר רכיבי הקשירות ב-$G_A$ היה גדול או שווה ל-מספר רכיבי הקשירות של $G_B$. לכן אפשר לבחור את הצלע שמחברת בין שני רכיבי הקשירות ב-$G_B$ וזו לא תסגור מעגל ב-$G_{B'} = (V, B \cup \{a\})$ כאשר $a$ היא הצלע המחברת.

### 2.3.1 האלגוריתם החמדן של המטרואיד

קלט:

מטרואיד $M = (E, I)$ ופונקציית משקל חיובית ממש $w : E \to \mathbb{R}^+$.

פלט:

תת קבוצה $A \in I$ שגודלה מקסימלי, כלומר לכל $B \in I$ מתקיים כי $|A| \geq |B|$ וגם שמשקלה מקסימלי, כלומר לכל $B \in I$ מתקיים:

$$w(A) = \sum_{a \in A} w(a) \geq \sum_{b \in B} w(b) = w(B)$$

**הערה**

במונחים של חוקיות ואופטימליות, פתרון **חוקי** יהיה קבוצה כלשהיא מ-$I$ ופתרון **אופטימלי** יהיה להחזיר מבין כל הקבוצות $A \in I$ את $A$ עם המשקל המקסימלי. אם כך, ניתן להסתכל על $I$ כקבוצת הפתרונות החוקיים לבעייה.

**האלגוריתם הגנרי**

1. נמיין את איברי $E$ בסדר יורד $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_n)$.
2. נאתחל $A = \emptyset$. נעבור על איברי $E$ לפי הסדר ולכל $i \in [n]$: אם $A \cup \{e_i\} \in I$ נוסיף את $e_i$ ל-$A$.
3. לאחר שעברנו על כל האיברים נחזיר את $A$.

זמן ריצה:

1. מיון לוקח $O(n \log n)$.
2. לכל $i$ צריך לבדוק האם $A \cup \{e_i\} \in I$. זמן זה תלוי בבעייה הספציפית. נסמן זמן ריצה זמן ב-$O(f(n))$.

לכן זמן הריצה הכולל הינו: $O(n \log n + n f(n))$.

**הערות**

1. במידה והצלחנו להראות שבעיית אופטימיזציה מסויימת מהווה **מטרואיד**, ניתן פשוט להשתמש באלגוריתם החמדן הגנרי בלי צורך להוכיח נכונות. ניתן יהיה להסתמך על הוכחת הנכונות שנראה בהרצאה.[48]
2. נשים לב כי האלגוריתם החמדן ייתן תמיד פתרון מגודל **מקסימלי**.

---

[48] דבר זה לא מבטיח יעילות בהכרח.

3. נשים לב **שמציאת מינימום** היא בעיה שקולה. כל שעלינו לעשות הוא להפוך את המיון לסדר עולה \ להחליף את הסימן של $w$. במצב זה, כבר לא נכון להגדיר את הבעיה כבעיית משקל מינימלי, אלא מציאת קבוצה מגודל מקסימלי בעלת משקל מינימלי.
4. לא תמיד חייבים לרוץ על כל $i \in [n]$. לפעמים ניתן לעצור אחרי מספר צעדים (תלוי בבעייה).

אם מפעילים את האלגוריתם הגנרי (עם מיון בסדר עולה) על המטרואיד הגרפי, מקבלים יער מגודל מקסימלי (עץ פורש) וממשקל מינימלי. למעשה, מדובר באלגוריתם קרוסקל.

### 2.3.2 מטרואיד השידוכים

תזכורות:

גרף דו צדדי $G = (L, R, E_G)$ הוא גרף המוגדר על 2 קבוצות זרות של קודקודים וכל הצלעות שלו מחברות בין קודקודים מקבוצות שונות.

**התאמה** בגרף דו צדדי היא תת קבוצה של $E$ עבורה אין 2 שנוגעות באותו קודקוד.

**התאמה מושלמת** היא התאמה שנוגעת **בכל הקודקודים**.

יהי $G = (L, R, L)$ גרף דו צדדי.

נגדיר $E_M = L$ וכמו כן:

$$I_M = \{L' \subseteq L \mid \exists R' \subseteq R, \exists \pi : L' \to R' | \pi \text{ is a perfect matching } \}$$

**טענה**

$M = (E_M, I_M)$ מטרואיד.

**הוכחה**

1. $I_M$ קבוצה לא ריקה כיוון ש- $\emptyset \in I_M$.
2. ירושה:

תהי $A \in I_M$ ותהי $B \subseteq A$. אנחנו צריכים להוכיח כי $B \in I_M$.

מכך ש-$A \in I$ נובע כי קיימת התאמה מושלמת $M_A : A \to R' \subseteq R$. נסתכל על $M_B : B \to R'' \subseteq R$ שמוגדרת על ידי $M_B(b) = M_A(a)$ לכל $b \in B$ . זו התאמה מושלמת מ-$B$ ל-$R$ ולכן $B \in I_M$.

3. הרחבה:

יהיו $A, B \in I_M$ כך ש-$|A| > |B|$.

אנחנו צריכים להוכיח כי קיים $a \in A \setminus B$ כך שקיימת התאמה מושלמת מ-$B \cup \{a\}$ ל-$R' \subseteq R$.

כיוון ש-$A \in I_M$ נובע כי קיימת התאמה מושלמת $M_A$ מ-$A$ ל-$R_A \subseteq R$. בצורה מקבילה קיימת גם התאמה $M_B$ מ-$B$ ל-$R_B \subseteq R$.

כעת, נגדיר את הגרף הבא $G' = (L, R, M_A \cup M_B)$.ננתח את רכיבי הקשירות של $G'$.

(**הערה:** על מנת שלא להתבלבל, חשוב להבחין. אנחנו מאחדים את שתי ההתאמות המושלמות. לכן תיתכנה שתי האפשרויות הבאות בלבד)[49]

הדרגה של של כל הקודקודים חסומה על ידי 2 (כל קודקוד נמצא בשידוך של $M_A$ ובשידוך של $M_B$) ובעקבות כך בגרף $G'$ ישנם שני סוגים של רכיבי קשירות:

1. מעגל פשוט ובו דרגת כל קודקוד היא 2 - כל קודקוד שייך הן ל-$A$ והן ל-$B$.
2. מסלול פשוט, ובו דרגת כל קודקוד פנימי היא 2 וכל קודקוד חיצוני הוא מדרגה 1 (קצוות המסלול).

---

[49] במונחים של יובל, זה המסלולים המתחלפים רק של צלעות בשני שידוכים שונים.

בשלב זה נבחין כי בכל רכיב קשירות של $G'$, הצלעות הן חלק מ-$M_A$ ו-$M_B$ לסירוגין.

כמו כן, נזכיר כי $|A| > |B|$ ולכן משיקולי ספירה קיים מסלול באורך אי זוגי באיחוד, שנסמנו $C$, שיש בו יותר צלעות מ-$M_A$ מאשר צלעות מ-$M_B$ (כיוון שגודל הקבוצה של $B$ קטן יותר, יש פחות צלעות בשידוך). זהו מסלול פשוט באורך אי זוגי ונסמנו ב-$C$[50]. נסמן את צלעות $C$ ב-$E_C$ ואת הצלעות ב-$E_C$ מ-$M_A, M_B$ נסמן $E_{C_A}$ ו-$E_{C_A}$ בהתאמה.

כיוון ש-$|E_{C_A}| = |E_{C_B}| + 1$ (אפשר להבין זאת מכך שמדובר בהתאמה מושלמת ושיש רק צלע אחת 'עודפת' ), נובע שהמסלול **מתחיל ומסתיים** בצלע מ-$M_A$ וכיוון שהמסלול באורך אי זוגי, אחד הקצוות נמצא ב-$L$. נסמן את הקודקוד בקצה זה ב-$a$ , ונשים לב כי $a \in A \setminus B$.

נגדיר את השידוך עבור $B \cup \{a\}$ באופן הבא. נשאיר את כל צלעות $M_B \setminus E_{C_B}$ כמו שהן. את הצלעות ב-$E_{C_B}$ נחליף ב-$E_{C_A}$. פורמלית, נקבל כי $M_{B'} = M_B \cup E_{C_A} \setminus E_{C_B}$.

נשים לב כי לכל קודקוד $b \notin C$ לא שינינו את ההתאמה. עבור $b \in C$, מהיותה של $M_A$ התאמה, מצאנו התאמה שכוללת את כל הקודקודים בו מ-$B$ יחד עם $a \notin B$. אם כך, $M_{B'} = M_B \cup E_{C_A} \setminus E_{C_B}$ התאמה, כנדרש.

על מנת להשתמש באלגוריתם הגנרי למטרואיד למציאת שידוך מקסימלי, יהיה עלינו לבדוק בכל שלב האם $A \cup \{e_i\} \in I_M$, עבור $A$ קבוצת הקודקודים הנוכחית ו-$e_i$ שהקודקוד ה-$i$ במיון, כלומר האם קיים שידוך מושלם עבור $A \cup \{e_i\}$. באופן נאיבי, בדיקה זו יקרה, למשל מעבר על כל תתי הקבוצות $S \subseteq A \cup \{e_i\}$ ובדיקה האם $|S| < |\text{Ne}(S)|$ (משפט הול מדיסקרטית). בהמשך הקורס נראה דרך יעילה למציאת שידוך מקסימום.

**תרגול מס' 4:**

**יום שלישי**

20.04.21

## 3 תכנון דינמי

תכנון דינמי זו גישה לפתרון בעיות רקורסיביות שמשתמשת בזכרון פולינומיאלי כדי להפחית את זמן הריצה, הרבה פעמים מאקספוננציאלי לפולינומיאלי. התובנה המרכזית היא שבהרבה בעיות רקורסיביות אנחנו עושים את אותם החישובים שוב ושוב באופן שמנפח את זמן הריצה, כשלמעשה אפשר לשמור את כל חישובי הביניים ולמחזר אותם. באופן טיפוסי, מדובר על בעיות שאפשר לפרק לתתי־בעיות, כך שהפתרון של תת־הבעיה שימושי כדי למצוא פתרון של הבעיה.

נוכל להבין זאת דרך דוגמאות.

### 3.0.1 דוגמה 1: סדרת פיבונאצ'י

נמצא את האיבר ה-$n$ בסדרת פיבונאצ'י באמצעות אלגוריתם נאיבי:

**אלגוריתם 6** סדרת פיבונאצ'י

```
1. אם $n \le 1$ תחזיר את $n$.
2. אחרת: תחזיר את $\text{Fib}(n-1) + \text{Fib}(n-1)$.
```

אפשר להראות באינדוקציה כי זמן הריצה הינו $O(2^n)$. אפשר להראות כי זמן הריצה הוא למעשה $\Theta(\varphi^n)$, כאשר $\varphi$ הוא יחס הזהב.

דבר זה נובע מכך שנוסחת הנסיגה היא $T(n) = T(n-1) + T(n-2)$.

זהו זמן ריצה מוגזם מאוד, שנובע מכך שאנו חוזרים על אותם איברים פעמיים.

---

[50] זה חייב להיות מסלול פשוט, אחרת מספר הצלעות באותו רכיב קשירות ב-$A$ וב-$B$ שווה.

אפשר לייעל זאת באמצעות שמירת חישובי הביניים בטבלה, ובכל שלב להשתמש בערכים שהשתמשנו בהם:

**אלגוריתם 7** אלגוריתם למציאת פיבונאצ'י - פחות נאיבי

```
1. $history \leftarrow [0,1]$
2. לכל $i \leftarrow 2$ עד $i \leftarrow n$:
   (א) $history\,[i] \leftarrow history\,[i-1] + history\,[i-2]$
```

אפשר לראות כי זמן הריצה הוא $O\,(n)$ שכן מתבצעות $n$ פעולות, וזהו גם זמן הזיכרון.
שימו לב שזמן הריצה הזה עדיין אקספוננציאלי בגודל הקלט. עם זאת הצלחנו לשפר משמעותית את זמן הריצה על ידי חישוב יחיד של כל תת־בעיה. שימו לב שהאלגוריתם הנ"ל הוא לא אופטימלי. ניתן להוריד את הזיכרון ל-$O\,(1)$ על ידי שימוש בהיסטוריה קצרה יותר, ולמעשה בזמן לוג-לינארי:

$$\text{fib}\ (n) = \frac{1}{\sqrt{5}}\left(\left(\frac{1+\sqrt{5}}{2}\right)^n - \left(\frac{1-\sqrt{5}}{2}\right)^n\right)$$

## 3.1 שלבים לפתרון בעייה באמצעות תכנון דינמי

נתאר את המתכון לפתרון הבעיות:

1. הגדרת תתי הבעיות: נתאר איזה תתי בעיות נרצה לשמור במערך ההיסטוריה שיצרנו, בצורה מתמטית.
2. כתיבת נוסחת רקורסיה: נרשום מתמטית כיצד לחשב תת בעייה מתתי בעיות קודמות.
3. הגדרת טבלה + סדר מילויה: נגדיר את טבלת ההיסטוריה בה נשמור את תתי הבעיות ונסביר באיזה סדר אפשר למלא אותה.
4. אופן חילוץ הפתרון האופטימלי: בהינתן טבלת ההיסטוריה המלאה, כיצד נחלץ פתרון.
5. ניתוח זמן ריצה: בדרך כלל, זמן מילוי כל תא כפול גודל הטבלה.
6. הוכחת נכונות: נוכיח את נכונות האלגוריתם שכתבנו. נוכיח כי כל התאים בטבלת ההיסטוריה מלאים בתתי הבעיות שהגדרנו ב-1.לאחר מכן, נוכיח כי ניתן להשתמש בטבלה המלאה על מנת לקבל את הפתרון כפי שהסברנו ב-4.

כדי להוכיח שהתאים בטבלה אכן מכילים את תתי הבעיות, נעשה בדרך כלל אינדוקציה על סדר מילוי הטבלה ב-3, בה נראה כי נוסחאת הרקורסיה שהגדרנו ב-2 אכן פותרת נכונה את תתי הבעיות.

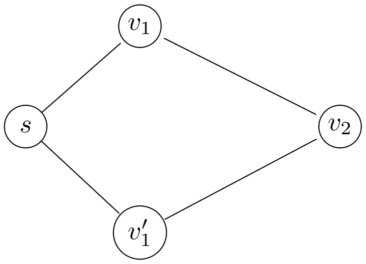
### 3.1.1 דוגמה 2: בלמן-שכבות

נתון גרף מכוון $G=(V,E)$ המכיל $K+2$ שכבות $V_0,V_1,\ldots,V_{K+1}$ ופונקציית משקל $w:E\to\mathbb{R}$ כך שמתקיים:

$$\begin{aligned}
&V_0=\{s\},\ \ V_{K+1}=\{t\}\\
&|V_1|=|V_2|=\ldots=|V_K|=M\\
&\bigcup_{k=0}^{K+1}V_k=V,\quad \forall i\neq j:V_i\cap V_j=\emptyset\\
&\forall(u,v)=e\in E,\quad \exists 0\leq k\leq K\quad u\in V_k,v\in V_{k+1}
\end{aligned}$$

למעשה, מדובר על גרף שמתחיל ב-$s$ ונגמר ב-$t$, כשבין קודקודים אלו יש $K$ שכבות בגודל $M$, כך שיכולות להיות רק צלעות שמחברות בין שכבות עקובות.

דוגמה לגרף כזה:



כאשר $s$ הוא כביכול בשכבה $V_0$ ו-$v_1,v_1'\in V_1$ וכמו כן $v_2\in V_0$.

**אוסף תתי הבעיות**

לכל קודקוד $v\in V$, נמצא את המסלול הקצר ביותר מ-$s$ אל $v$. נשים לב כי עבור $t\in V$, זה מה שאנחנו מחפשים.

**נוסחת הרקורסיה**

נוכל לסמן ב-$D\left[v\right]$ את משקל המסלול הקל ביותר מקודקוד $s$ לקודקוד $v$. כמו כן, נסמן ב-$V_k$ את השכבה בה $v\in V_k$.

המסלול הקצר ביותר מקודקוד לעצמו יהיה 0 - $D\left[s\right]=0$.

יתר על כן, נשים לב כי משקל המסלול הקל ביותר מקודקוד $s$ **לכל קודקוד** $v$ **אחר** הוא סכום משקל המסלול הקל ביותר מ-$s$ לקודקוד לפני $v$, ומשקל הצלע שמחברת ביניהם.

כיוון שמדובר בגרף שכבות, אזי הקודקוד שלפני $v$, נמצא **בשכבה שלפניו**, ואם כך משקל המסלול הקל ביותר יתקבל בקודקוד שנמצא בשכבה שלפני $v$.

לפי זה, מספיק לנו להסתכל על השכנים של $v$ (שכולם נמצאים בשכבה שלפני $v$). בעקבות כך, הנוסחה הרקורסיבית הינה:

$$D[v]=\min_{u\in d_{ni}(v)}(D[u]+w(u,v))$$

אם נחבר את המקרים, נקבל:

$$D\left[v\right] = \begin{cases} 0 & k=0 \\ \min_{u\in d_{ni}(v)}(D[u]+w(u,v)) & else \end{cases}$$

**הגדרת הטבלה**

נבנה טבלה בגודל $M \times K$. כעת, נמלא את הטבלה לפי נוסחת הרקורסיה, עמודה כל פעם, משמאל לימין.
תחילה נחשב עבור $k=1$ ולאחר מכן את השכבות הבאות.

**חילוץ הפיתרון**

בזמן מילוי הטבלה נשמור גם את הקודקוד ממנו הגענו (במרחק הכי קצר. ניתן לשמור באמצעות חץ).
כעת, נלך מסוף הטבלה להתחלה: נבחר את התא בעמודה שהצלע ממנו אל $t$ פלוס המשקל מינימלי. נוכל להשתמש בקודקודים ששמרנו בכל תא, כדי לעבור על הטבלה בסדר הפוך ולחלץ את הפתרון.

**זמן הריצה**

גודל הטבלה כפול כל זמן למלא תא.
במקרה שלנו, גודל הטבלה הוא $O\left(V\right)$ , וזמן למלא כל תא הוא במקרה הגרוע $O\left(n\right)$, אך זמן הריצה איננו $O\left(nV\right)$
כיוון שניתן לייעל את זמן הריצה קצת: נוכל לשים לב כי בכל פעם אנו בודקים כל צלע פעם אחת (עבור כל קודקוד בודקים את שכניו), ולכן זמן המילוי ייקח $O\left(|E|\right)$. לכן סך הכל מדובר בזמן ריצה של $O\left(|V|+|E|\right)$.

הוכחת נכונות

**טענה**

נוכיח באינדוקציה על סדר מילוי הטבלה כי הטבלה מולאה נכון.

בסיס

בתא המקביל לקודקוד $s$ יש $0$ וזה בדיוק המרחק.

הנחה

כל התאים עד כה מולאו נכון.

צעד

תחילה, נוכיח כי קיים מסלול כזה.
נתבונן בקודקוד $v$ שנמצא בשכבה $k$, כך שמתקיים: $V^k = \left\{v_1^k, \dots v_m^k\right\}$.
בתא שאנו מתבוננים בו, מתקבל:

$$\min_{u\in V_k}\left(D[u]+w\left(u,v_m^k\right)\right)$$

האיבר ה-$u'$ מקיים כי $u' = \min\left(D\left[u\right]+w\left(n,v_m^k\right)\right)$.
מהנחת האינדוקציה קיים מסלול ל-$u'$ באורך $D\left[u'\right]$. אם נמשיך בו בצלע $\left(u', v_m^k\right)$, נקבל מסלול באורך הנדרש.

אופטימליות

נניח בשלילה שהמסלול לא הכי קצר. אז בשכבה ה-$k-1$ המסלול עבר ב-$u'' \neq u'$.

מה האורך של המסלול?

$$D\left[u''\right]+w\left(u'',v_m^k\right)$$

ומכאן נובע כי המינימום עבר על $u''$ ולא בחר בה, בסתירת לפעולת המינימום.

**הערות**

(1) דבר זה פועל על כל גרף מכוון חסר מעגלים, על ידי מיון טופולוגי $O\left(|E|+|V|\right)$. עלינו לעבור על הקודקודים על פי הסדר שמתקבל מהמיון הטופולוגי , כשעבור כל קודקוד $v$ נבדוק מיהו הקודקוד שממזער את סכום משקל המסלול הקל ביותר ומשקל הקשת המחברת אותו אל $v$.
(2) ניתן לחשוב על כל אלגוריתם דינאמי בתור גרף שכבות. כל תת בעייה תהווה קודקוד בגרף, ונוסיף צלע $(u,v)$ אם כדי לפתור את תת הבעייה $v$ צריך להשתמש בקודקוד $u$.

## 3.2 תת־מחרוזת משותפת מקסימלית

קלט:
שתי מחרוזות.
פלט:
תת מחרוזת משותפת מקסימלית.

אלגוריתם נאיבי:
נעבור על כל תתי הקבוצות ונשווה. זהו זמן ריצה של $2^n$.

**פתרון דינמי**

תתי בעיות
נסמן $X=x_1,\ldots,x_i$ רישא של $X$ ו-$Y=y_1,\ldots,y_j$ רישא של $Y$.
נניח כי $|X|=n$ ו-$|Y|=m$.
לכל $1\le i\le n$ ולכל $1\le j\le m$ נמצא תת מחרוזת של $X_i$ ו-$Y_j$.

**נוסחת הרקורסיה**

נסמן את האורך של תת מחרוזת $X^i$ ו-$Y^j$ ב-$f\left(i,j\right)$, ואז נקבל את הנוסחה הבאה:

$$f(i,j)=\begin{cases}0 & i=0\vee j=0\\ f(i-1,j-1)+1 & x_i=y_j\\ \max\{f(i-1,j),f(i,j-1)\} & x_i\neq y_j\end{cases}$$

**חילוץ הפיתרון**

בזמן מילוי הטבלה נשמור מצביעים אל התא ממנו חישבנו את הפתרון.
כעת, נעבור על הטבלה מלמעלה למטה:

נתחיל מהתא $(n,m)$ ונלך לפי החיצים. בכל פעם שתא $(i,j)$ מצביע לתא $(i-1,j-1)$, נוסיף את האות הנוכחית למחרוזת משמאל ולבסוף נחזיר את המחרוזת.

**מילוי טבלה**

נבנה טבלה $M$ בגודל $n \times m$. נמלא את הטבלה למעשה לפי נוסחת הרקורסיה, קודם כל את מקרי הבסיס ואז שורה אחרי, מלמטה למעלה, לשמאל לימין. החל מהפינה השמאלית התחתונה. את השורה התחתונה והעמודה השמאלית ניתן למלא לפי בסיס הרקורסיה, ובהמשך לכל $i,j$ נצטרך להשתמש בתאים $(i-1,j-1)$ וגם $(i,j-1)$ ו-$(i-1,j)$ שחישבנו.

**דוגמת הרצה:**

ניקח $X = ABCD$ ו-$Y = BDC$, נקבל את הטבלה:

| | | | | | |
|---|---|---|---|---|---|
| $\leftarrow 2$ | $\swarrow$ **2** | $\downarrow 1$ | 0 | 4 | D |
| $\swarrow 2$ | $\leftarrow 1$ | $\downarrow 1$ | 0 | 3 | C |
| $\leftarrow 1$ | $\leftarrow 1$ | $\swarrow 1$ | 0 | 2 | B |
| $\leftarrow 0$ | $\leftarrow 0$ | $\leftarrow 0$ | 0 | 1 | A |
| 0 | 0 | 0 | 0 | 0 | $\phi$ |
| 3 | 2 | 1 | 0 | | |
| C | D | B | $\phi$ | | |

במקרה זה, אנחנו מוסיפים בתא $(4,2)$ את $D$ ובתא $(2,1)$ נוסיף את $B$. המחרוזת $BD$ אכן תקינה.

**זמן הריצה**

גודל הטבלה הוא למעשה $(n+1)\cdot(m+1)$. מילוי כל תא לוקח $O\left(1\right)$ - כי בודקים 3 תאים סמוכים, ולכן קיבלנו $O\left(n \cdot m\right)$.

**הוכחת נכונות**

נציג את הנכונות רק לגבי אורך תת המחרוזת, ולא לגבי המחרוזת עצמה (לא בעייה לשלב).

**טענה**

לכל $0 \leq i \leq n$ ולכל $0 \leq j \leq m$ התא $M\left[i,j\right]$ מכיל את האורך של תת מחרוזת $X^i$ ו-$Y^j$.

**הוכחה**

נוכיח באינדוקציה על סדר מילוי הטבלה. נניח שתאי הטבלה מולאו ושמילאנו אותם טוב, ונוכיח לתא הבא.

בסיס האינדוקציה

אם $j=0$ או $i=0$, במקרה זה תמ"א ריקה ואורכה 0, ואכן לפי נוסחת הרקורסיה אנו ממלאים 0 בתאים אלו, כנדרש.

צעד האינדוקציה

יהיו $0 < i \leq n$ ו-$0 < j \leq m$. נניח כי התאים לפני $M\left[i,j\right]$ מכילים ערכים נכונים, ונוכיח כי התא $M\left[i,j\right]$ מלא בתא הנכון.

תהי $S = s_1,\ldots,s_r$ תמ"א כלשהיא של $X^i$ ו-$Y^j$.

נחלק למקרים:

- אם $x_i = y_j$: במקרה זה בהכרח $s_r = x_i = y_j$, אחרת אפשר להוסיף את $x_i = y_j$ למחרוזת ולקבל מחרוזת ארוכה יותר. לכן גם $S' = s_1, \ldots, s_{r-1}$ היא תמ"א של $X^{i-1}$ ו-$Y^{j-1}$. מהנחת האינדוקציה, בהכרח אורכה נמצא בתא $M\left[i-1, j-1\right] = r-1$. מכלל העדכון נצטרך להוסיף 1 ואכן נקבל את $r$.

- אם $x_i \neq y_j$: בהכרח $s_r \neq x_i$, או $s_r \neq y_j$:

א) אם $s_r \neq x_i$, אזי $S$ תמ"א של $X^{i-1}, Y^j$ ולפי הנחת האינדוקציה, אורכה נמצא בתא $M\left[i-1, j\right]$.

ב) אם $s_r \neq y_j$, אזי $S$ תמ"א של $X^i, Y^{j-1}$ ולפי הנחת האינדוקציה, אורכה נמצא בתא $M\left[i, j-1\right]$.

$S$ מקסימלית, לכן אורכה בהכרח $\max\left\{M\left[i-1, j\right], M\left[i, j-1\right]\right\}$ - שזהו הערך שמילאנו בתא ה-$(i, j)$, כנדרש.

**תרגול מס' 5:**

**יום שלישי**

20.04.21

## 3.3 בעיית מסילת הרכבת

כדאי קודם כל להבין את הבעייה בפשטות, לפני שנתעסק בפירוק המתמטי.

נתונים $N$ חלקים עם $K$ סוגי חיבורים. לכל חלק באורך $\ell_i$ יש חיבור סיום $e_i \in K$, חיבור התחלה $s_i \in K$ ומחיר $p_i$.

נרצה לבנות מסילה באורך $L$ עם מחיר מינימלי, כאשר נתון לנו אילוץ כי כל חיבור ימני חייב להתחבר לאחד שמאלי מאותו הסוג.

ניתן לחזור על חלקים פעמיים.

למשל $L = 3$, $K = 4$, ו-$N = 5$ - $\{(\ni, \circ, 1, 30), (>, \in, 1, 10), (\sqsupset, \sqsubset, 1, 30), (\circ, \circ, 2, 40), (\sqsupset, \in, 3, 100)\}$.

### אבחנה

גישה נפוצה לעיצוב אלגוריתם דינאמי היא להסתכל על הצעד האחרון בפתרון אופטימלי כלשהו, ולנסות להבין האם כשמורידים צעד זה נשארים עם פתרון אופטימלי עבור משימה כלשהי. אם כן, המשימות האלו מגדירות לנו את תתי הבעיות. הדרך בה מחליטים מה הוא הצעד האחרון על סמך אוסף פתרונות תתי הבעיות תגדיר עבורנו את נוסחת הרקורסיה.

אם מורידים חלק אחד מהפתרון האופטימלי $i$, נקבל פתרון אופטימלי מאורך $L - \ell_i$ שנגמר בחיבור מסוג $s_i$ (כי זה ההתחלה של החיבור שאנחנו מסירים).

### הגדרת תתי הבעיות

עבור $0 \leq \ell \leq L$ ועבור $k \in K$ נמצא את מחיר המסילה הכי זול למסילה באורך $\ell$ שנגמרת בחלק $k$.

### הגדרת הרקורסיה

נסמן $f\left(l, k\right)$ - המחיר המינימלי למסילה באורך $\ell$ שנגמרת בחיבור ימני מסוג $k$ ונקבל:

$$f(\ell, k) = \begin{cases} 0 & \ell = 0 \\ \min\limits_{1 \leq i \leq N:\ e_i = k \wedge\ \ell - \ell_i \geq 0} \left\{p_i + f\left(\ell - \ell_i, s_i\right)\right\} & \ell > 0 \end{cases}$$

כאשר התנאי השני נובע כדי למנוע מצב בו $\ell < \ell_i$. כמו כן, על מנת לכסות מקרה בו אין חלק שמסתיים בסוג חיבור מסוים, נגדיר $\min$ על קבוצה ריקה בתור $\infty$.

### בניית הטבלה

נבנה טבלה בגודל $(L+1)\times K$ ונמלא לפי הגדרת הרקורסיה. נתחיל בשורה התחתונה (כיוון שאפשר למלא אותה באמצעות בסיס האינדוקציה) ונמשיך משם כלפי מעלה, שורה אחרי שורה (קודם כל נעבור עמודה עמודה).
לבסוף, נחזיר את המינימום על השורה העליונה, כלומר $\min\limits_{1\leq k\leq K}\left\{M\left[L,k\right]\right\}$.

בדוגמה שראינו קודם לכן, למשל:

| | | | | |
|---|---|---|---|---|
| 90 | $\infty$ | 70 | 100 | 3 |
| 60 | $\infty$ | 40 | $\infty$ | 2 |
| 30 | $\infty$ | 30 | 10 | 1 |
| 0 | 0 | 0 | 0 | 0 |
| $\sqsubset$ | $<$ | $\circ$ | $\ni$ | |

### זמן הריצה

יש לנו $L\left(K+1\right)$ תאים כשזמן מילוי התא הוא $O\left(N\right)$, לכן זמן הריצה הוא $O\left(NLK\right)$ - אפשר לשים לב כי הוא לא פולינומיאלי, כיוון שאורך המסילה נתון על ידי $\log L$.

### הוכחת נכונות

נוכיח באינדוקציה על סדר מילוי הטבלה כי בתא ה-$\ell,k$ יש את מחיר הטבלה החוקית המינימלית.

בסיס

אם $\ell=0$ אז אפשר לבנות רק את המסילה הריקה שמחזירה 0, כנדרש.

הנחה

נניח כי כל התאים עד כה מולאו נכון.

צעד

נתבונן בתא $\ell,k$ ונוכיח כי יש מסילה באורך זה.
אם רשום בתא זה $\infty$, אז המינימום הוא קבוצה ריקה ואין מסלול כזה.
אחרת, נבחין כי מסילה חוקית שמסתיימת בחיבור $k$ חייבת להסתיים בחיבור ימני מסוג $k$. לכן החלק האחרון במסילה חייב להיות חלק עם חיבור ימני מסוג $k$. נניח כי מדובר באיבר ה-$i$. תת המסילה שלא מכילה את האיבר הזה, היא מסילה חוקית באורך $\ell-\ell_i$ והיא מסתיימת בחלק מסוג $s_i$.
כעת, מחיר המסילה בלי האיבר ה-$i$ הוא $f\left(\ell-\ell_i,s_i\right)$ - נוסיף אליה את משקל האיבר ה-$i$, כלומר את $p_i$ ונקבל את $f\left(\ell-\ell_i,s_i\right)+p_i$ - בדיוק כפי שמופיע בנוסחת הרקורסיה.
מהנחת האינדוקציה, התא ה-$\ell-\ell_i,s_i$ $\left(\ell-\ell_i<\ell\right)$ מכיל את מחיר המסילה החוקית באורך $\ell-\ell_i$ שמסתיימת בחלק מסוג $s_i$ ולכן בפרט התא $\ell,k$ מכיל את מחיר המסילה החוקית בעלת המחיר המינימלי באורך $\ell$ המסתיימת בחיבור מסוג $k$, כנדרש.

### אופטימליות

נניח בשלילה כי יש מסילה קצרה יותר. נסמן את החלק האחרון בה, ב-$i''$. נחשב את אורכה.
המסילה עד אליה נגמרה ב-$s_i''$. עד החלק האחרון המסילה הכי קצרה הינה $f\left(\ell-\ell_i,e''\right)$ ונוסיף אליה את $p_{i''}$.
המסילה הזאת מתאימה לתנאים של המינימום, אך האלגוריתם לא לקח אותה, בסתירה לפעולת האלגוריתם.

## 3.4 פלוייד ורשל - כל הדרכים הקצרות

קלט

גרף מכוון $G=(V,E)$ ופונקציית משקל $w:E\to\mathbb{R}$ לאו דווקא חיובית.

פלט

מטריצה בגודל $|V|\times|V|$ כך שבתא ה-$i,j$ נמצא משקל המסלול הקל ביותר מ-$i$ ל-$j$.

הנחה

בגרף אין מעגלים שליליים (מעגל שסכום הערכים שלו שלילי).

**ניסיון נאיבי** לכל $1\le i,j\le n$ נמצא את משקל המסלול הקל ביותר מ-$v_i$ ל-$v_j$. יש בעייתיות כי זאת הבעייה הכללית ולא תתי הבעיות.

כמו כן, לא ברור כיצד להגדיר את נוסחת הרקורסיה במקרה זה.

גם לא ברור כיצד אפשר לייצא את הפתרון.

### 3.4.1 אלגוריתם נוסף

**הגדרת תתי בעיות**

לכל $1\le i\le n$ ולכל $1\le j\le n$ ולכל $0\le m\le n-1$ נמצא את משקל המסלול הקל ביותר בין $v_i$ ל-$v_j$ שמשתמש בלכל היותר $m$ צלעות.

**נוסחת הרקורסיה**

תחילה נבדוק עבור מקרה הבסיס, בגודל 0:

$$f(i,j,0)=\begin{cases}0 & i=j\\ \infty & i\neq j\end{cases}$$

כלומר, אם יש צלע, המסלול הוא 0, אם $i\neq j$ אין דבר כזה ולכן גודל המסלול הוא $\infty$.

במקרה הכללי, נגדיר:

$$f(i,j,m)=\min_{v_x\in V}\left\{f(i,x,m-1)+w\left(v_x,v_j\right)\right\}$$

**הגדרת הטבלה**

נבנה טבלאות עבור כל גודל.

בתחילה, עבור $m=0$:

$$L^{(0)} = \begin{pmatrix} 0 & \infty & \infty & \cdots & \infty \\ \infty & 0 & \infty & \cdots & \infty \\ \infty & \infty & 0 & \cdots & \infty \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \infty & \infty & \infty & \cdots & 0 \end{pmatrix}$$

ולאחר מכן עבור טבלאות בגדלים הבאים, על סמך הטבלאות הקודמות.

**זמן הריצה**

גודל הטבלה הינו $|V|^4$ וזמן מילוי כל תא הוא $|V|$ ולכן נקבל סך הכל $O\left(|V|^4\right)$.
נתן לשפר את זמן הריצה אם לוקחים את המינימום רק על השכנים, ונקבל $O\left(|V|^2|E|\right)$ - כיוון שלכל בחירה של $i$ ולכל בחירה של $j$ צריך לעבור על כל הצלעות פעם אחת. למעשה זהו זמן הריצה אם נריץ בלמן פורד על כל נקודה בגרף. אפשר לשפר את זמן הריצה הזה.

### 3.4.2 האלגוריתם של פלוייד וורשל

פלוייד וורשל הציעו רעיון אחר שיאפשר לנו לקצר את זמן הריצה.

**הגדרת תתי הבעיות**

נמספר את הקודקודים $v_1, \ldots, v_n$ וכעת לכל $1 \le i \le n$ ולכל $1 \le j \le n$ ולכל $1 \le k \le n$ נמצא את המסלול הכי קל שמשתמש לכל לכל היותר בקודקודים $v_1, \ldots, v_k$.

**הגדרת הרקורסיה**

נחלק למקרה הבסיס ושאר המקרים.
עבור מקרה הבסיס, נקבל, עבור קודקודים מתוך $k = 0$.

$$f(i,j,0) = \begin{cases} 0 & i = j \\ w(v_i, v_j) & (v_i, v_j) \in E \\ \infty & (v_i, v_j) \notin E \end{cases}$$

אם $k > 0$. נגדיר כעת את $P$ בתור המסלול הקצר ביותר בין $v_i$ ל-$v_j$ שמשתמש רק בקודקודים מתוך $\{v_1, \ldots, v_k\}$.
ישנם שתי אפשרויות:
אם $P$ לא משתמש ב-$v_k$ אזי $P$ הוא מסלול קצר שמשתמש בקודקודים מתוך $v_1, \ldots, v_{k-1}$.
אם $P$ משתמש ב-$v_k$ אזי יש מסלול קצר בין $v_i$ ל-$v_k$ שמשתמש בקודקוד מתוך $\{v_1, \ldots, v_{k-1}\}$ ומסלול קצר בין $v_k$ ל-$v_j$ שמשתמש בקודקודים מתוך $\{v_1, \ldots, v_{k-1}\}$.
נבחר את המינימלי מביניהם ונקבל:

$$f(i,j,k) = \min\{\underbrace{f(i,j,k-1)}_{v_k \text{ is unused}}, \underbrace{f(i,k,k-1) + f(k,j,k-1)}_{v_k \text{ is used}}\}$$

**בניית הטבלה**

כמו המקרה הקודם, רק שכעת עבור $k$ ולא $m$.

למשל:

$$D^{(0)}=\begin{pmatrix} 0 & 3 & 8 & \infty & -4 \\ \infty & 0 & \infty & 1 & 7 \\ \infty & 4 & 0 & \infty & \infty \\ 2 & \infty & -5 & 0 & \infty \\ \infty & \infty & \infty & 6 & 0 \end{pmatrix}$$

**זמן הריצה**

גודל הטבלה -$O\left(n^{3}\right)$ ומילוי תא הוא בזמן של $O\left(1\right)$. ולכן נקבל $O\left(\left|V\right|^{3}\right)$, כנדרש

**הוכחת נכונות**

נוכיח באינדוקציה על סדר מילוי הטבלה.

<u>בסיס</u>

עבור $k=0$ טריוואלי, כפי שראינו קודם לכן.

<u>הנחה</u>

נניח כי כל התאים עד כה מולאו כפי שצריך.

<u>צעד</u>

נחלק למקרים כפי נוסחת הרקורסיה.

נתבונן בתא ה-$(i,j)$. נסמן את המסלול הקצר מ-$v_i$ ל-$v_j$ שעובר לכל היותר בקודקדים מתוך $\{v_1,\ldots,v_{k-1}\}$.

מקרה א':

$v_k\in p\left(v_i,v_j\right)$ - במקרה זה אפשר לחלק את המסלול לשניים על ידי $p\left(v_k,v_j\right)$ ו-$p\left(v_i,v_k\right)$. מכאן $f\left(i,j,j\right)=f\left(i,k,k-1\right)+f\left(k,j,k-1\right)$.

מקרה ב':

$v_k\notin p\left(v_i,v_j\right)$ ולכן בהכרח לא משתמשים בו, ולכן ניקח את $f\left(i,j,k\right)=f\left(i,j,k-1\right)$ כנדרש.

# 4 רשתות זרימה

**תרגול מס' 6:**

## 4.1 הגדרות

**יום שלישי**

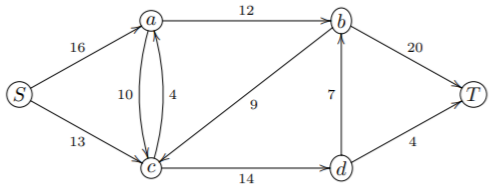
04.05.21

**הקדמה**

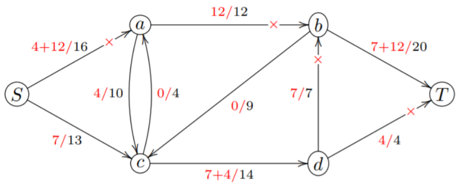
נניח כי נתונה לנו רשת שבה כל צלע יש קיבולת, קודקוד $s$ (מקור) וקודקוד $t$ בור.

**דוגמא**

נתבונן ברשת הזרימה הבאה:



נוכל להגדיר עליה את הצביעה הבאה:



בדוגמא זו אנחנו מעבירים $23$ יחידות חומר מ-$s$ אל $t$. כל צינור שמגיע לרוויה אנחנו מסמנים ב-$\times$. איננו יכולים להעביר עוד חומר בדוגמה זו כיוון שכל מסילה עוברת דרך אחת הצלעות המסומנות ב-$\times$ וכיוון שאנחנו מעבירים את המקסימום דרכן, הן מהוות צוואר בקבוק.

**הגדרה**

רשת זרימה היא רביעייה $(G, c, s, t)$ כאשר:

1. $G = (V, E)$ גרף מכוון.
2. $c : E \to \mathbb{N}$ פונקציית קיבול.
3. $s \in V$ קודקוד מקור.
4. $t \in V$ קודקוד בור.

**הנחה**

אין קשתות שנכנסות ל-$s$ ואין קשתות שיוצאות מ-$t$.[51]

**הגדרה**

זרימה חוקית ברשת זרימה היא פונקצייה $f : E \to \mathbb{R}^{+}$ המקיימת את האילוצים הבאים:

1. אילוץ הקיבול - $\forall e \in E,\ \ f(e) \leq c(e)$ .
2. חוק שימור הזרימה - $\sum_{u' : (v, u') \in E} f(v, u') = \sum_{u : (u, v) \in E} f(u, v), \qquad \forall v \in V \backslash \{s, t\}$.

---

[51] אפשר להתעלם מהקשתות שנכנסות למקור ושיוצאות מהבור, כי הן חסרות משמעות.

**הגדרה**

השטף של זרימה $f$, המסומן על ידי $|f|$ הוא סך כל הזרימה היוצאת מהבור $|f| := \sum_{u:(s,u)\in E} f(s,u)$.

**הערה**

בכיתה הגדרנו אחרת. חשוב להבחין שדבר זה נובע מהנחה שמובאת קודם לכן.

**טענה**

השטף שווה לסך כל הזרימה הנכנסת אל הבור:

$$|f| = \sum_{u:(u,t)\in E} f(u,t)$$

ניתן להוכיח טענה זאת על ידי שימוש בחוק שימור הזרימה.

### 4.1.1 בעיית הזרימה

קלט

רשת זרימה $N = (G, c, s, t)$

פלט

זרימה חוקית $f$ עבור רשת זרימה $N$, כך שהערך $|f|$- השטף - מקסימלי.

**הערות**

1. מדובר בבעיית אופטימיזציה. פתרון חוקי הוא זרימה $f$ שחוקית לרשת $N$.
2. הפתרון האופטימלי לא בהכרח יחיד.
3. בכל רשת יש לפחות זרימה חוקית אחת (זרימת האפס?).
4. השטף חסום מלמעלה על ידי **הקיבול בחתך** שיוצא מ-$s$ (צוואר בקבוק).
5. אם נבנה רשת צנורות ונזרים בה מים, נקבל פתרון אופטימלי לבעייה (זהו אלגוריתם אפשרי, אבל לא פשוט ובטח לא ניתן להרחבה).

בכיתה ראינו שני פתרונות - FF שעובד בזמן $O(|E| \cdot |f^*|)$, ואחד EK שעובד בזמן $O\left(|V| \cdot |E|^2\right)$. במהלך התרגול, נשתמש באלגוריתמים אלו כ'קופסה שחורה'. בנוסף, נעדיף את האלגוריתם אדמונד קארפ, כיוון שיש לו זמן ריצה טוב יותר, וגם כי הוא מחזיר פתרון בטבעיים אם קיים כזה.

אמנם, נעיר כי אם ידוע לנו שהזרימה האופטימלית קטנה ממספר הצלעות באופן יחסי, נעדיף להשתמש ב-FF שיהיה בעל זמן ריצה טוב יותר.[52]

---

[52] בדרך כלל, אם אנחנו בונים רשת בעצמנו היא תהיה פשוטה ולכן נעדיף להשתמש ב-FF. מצד שני, אם הביאו לנו רשת והיא מורכבת יותר או שלא ידועים עליה הרבה פרטים, נעדיף להשתמש ב-EK.

## 4.2 שימושים

**מציאת התאמה מקסימלית בגרף דו-צדדי**

קלט

גרף דו צדדי לא מכוון, $G=(L,R,E)$.

פלט

התאמה מקסימלית ב-$G$.

**פתרון**

1. נגדיר רשת זרימה $(G'=(V',E'),c,s,t)$ באופן הבא:
   (א) $V'=L\cup R\cup\{s,t\}$.
   (ב) נסמן ב-$\overrightarrow{E}$ את קשתות הגרף המקורי מכוונות מ-$L$ ל-$R$ - $\vec{E}=\{(u,v)\in E\mid u\in L,v\in R\}$.
   (ג) נגדיר קשתות יוצאות מקודקוד המקור ונכנסות לקודקוד המטרה ונסמן $E_L=\{(s,u)\mid u\in L\}$ וגם $E_R=\{(v,t)\mid v\in R\}$.
   (ד) נקבע את $E'$ כאיחוד כל הקשתות הנ"ל $E'=\vec{E}\cup E_L\cup E_R$.
   (ה) נקבע את הקיבול ל-1 עבור כל הקשתות $\forall e\in E',c(e)=1$.
2. נריץ על הרשת הנ"ל את אלגוריתם FF למציאת זרימה מקסימלית. נסמן ב-$f$ את הזרימה שחזרה.
3. נשים לב ש-$f$ זרימה בשלמים ולכן $f(e)\in\{0,1\}$. נחזיר את ההתאמה $M=\{e\in\vec{E}\mid f(e)=1\}$.

**הוכחת נכונות**

נניח בשלילה ש-$M$ **לא התאמה**. אם כך, קיים קודקוד $x\in L\cup R$ שיש זוג צלעות ב-$M$ שנוגע בו.
- אם $x\in L$ אזי יכולה לצאת ממנו זרימה בגודל 1 לכל היותר, בסתירה לכך שיש 2 צלעות ב-$\overrightarrow{E}$ עם $f(e)=1$ וגם שתיהן נוגעות ב-$x$, כלומר בסתירה לכך שנכנסת אליו זרימה בגודל של לפחות 2.
למעשה, באמצעות ההנחה בשלילה אנחנו מניחים שהזרימה שיוצאת מקודקוד מסוים היא 2, אבל הזרימה הנכנסת אליו היא מקסימום 1.
- אם $x\in R$, אותו דבר עבור $t$.

**הוכחת אופטימליות**

נניח בשלילה ש-$M$ לא מקסימלית. אזי קיימת התאמה $M'$ עם $|M'|>|M|$. כעת, עלינו להוכיח 2 דברים:

1. קיימת $|f'|=|M'|$.
2. $|M|=|f|$.

תחילה, נוכיח את 2:

$$|f| = \sum_{v\in L} f(s,v) \overset{\substack{\underbrace{\text{שימור החומר}} \\ \downarrow}}{=}$$

$$\sum_{(u,v)\in\overrightarrow{E}} f(u,v) \overset{\substack{\underbrace{\text{סוכמים רק צלעות שונות מאפס}} \\ \downarrow}}{=}$$

$$\sum_{(u,v)\in\overrightarrow{E}} 1 \overset{\substack{\underbrace{\text{הגדרה}} \\ \downarrow}}{=} |M|$$

עבור $1$, נקבל:

תהי $M'$ התאמה. נסמן $L' \subseteq L$ הקודקודים מ-$L$ ש-$M'$ נוגעת בהם וגם $R' \subseteq R$ הקודקודים מ-$R$ ש-$M'$ נוגעת בהם.

נגדיר כעת זרימה באופן הבא:

$$f'(e) = \begin{cases} 1 & e \in \{(s,u) \mid u \in L'\} \cup M' \cup \{(u,t) \mid u \in R'\} \\ 0 & \text{otherwise} \end{cases}$$

$f'$ בהכרח חוקית, שכן $c(e) = 1$ וגם שימור החומר מתקיים - $M'$ התאמה.

כל שנותר לנו להראות הוא כי $|f'| = |M'|$:

$$|f'| = \sum_{v\in L} f(s,v) \overset{\substack{\underbrace{\text{שימור החומר}} \\ \downarrow}}{=}$$

$$\sum_{(u,v)\in\overrightarrow{E}} f(u,v) \overset{\substack{\underbrace{\text{סוכמים רק צלעות שונות מאפס}} \\ \downarrow}}{=}$$

$$\sum_{(u,v)\in\overrightarrow{E}} 1 \overset{\substack{\underbrace{\text{הגדרה}} \\ \downarrow}}{=} |L'| = |M'|$$

הוכחנו כי קיימת $|f'| = |M'|$ והוכחנו כי $|f| = |M|$.

דבר זה גורר סתירה, שכן אנו יודעים כי $|f'| = |M'| > |M| = |f|$, בסתירה לאופטימליות EK.

### בעיית הסוכנים

$n$ סוכני מודיעין נמצאים ב-$n$ ערים באמריקה $(d_1,\dots,d_n)$. רוצים להעביר אותם ל-$n$ ערים באסיה $(c_1\dots,c_n)$. אין טיסות ישירות אלא רק דרך הערים באירופה $(e_1,\dots,e_n)$. בכל עיר באירופה יכולים לשהות בו-זמנית לא יותר מ-$k$ סוכנים. בהינתן קווי הטיסות מאמריקה לאירופה:

$$A=\{(d_i,e_j)\mid\forall 1\leq i,j\leq n\ \ \text{there is a flight from } d_i\text{to } e_j\}$$

וקווי הטיסות:

$$B=\{(e_j,c_k)\mid\forall 1\leq j,k\leq n \text{ there is a flight from } e_j\text{to } c_k\}$$

האם ניתן להעביר את כל הסוכנים לאסיה כך שבכל $c_i$ יהיה סוכן אחד. (כל הטיסות ממריאות ונוחתות באותו הזמן).

**פתרון**

1. נגדיר רשת זרימה $N=(G,c,s,t)$ כך:
   (א) נשכפל את הקודקודים $e_1,\dots,e_n$ כך שנקבל שתי קבוצות $\left\{e_1^1,\dots,e_n^1\right\},\left\{e_1^2,\dots,e_n^2\right\}$.[53]
   (ב) נגדיר $V=\{d_1,\dots,d_n\}\cup\left\{e_1^i,\dots,e_n^i\right\}\cup\{c_1,\dots,c_n\}\cup\{s,t\}$.
   (ג) נגדיר $E_d=\{(s,d_i)\mid 1\leq i\leq n\}$ וגם $E_e=\left\{\left(e_i^1,e_i^2\right)\mid 1\leq i\leq n\right\}$ וגם $E_c=\{(c_i,t)\mid 1\leq i\leq n\}$.
   (ד) נגדיר $E^1=\left\{\left(d_i,e_j^1\right)\mid (d_i,e_j)\in A\right\}$ וגם $E^2=\left\{\left(e_i^2,c_j\mid (e_i,c_j)\in B\right)\right\}$.
   (ה) לבסוף נגדיר $E=E_d\cup E^1\cup E_e\cup E^2\cup E_c$.
   (ו) נגדיר את הקיבול להיות $1$, למעט בין הצלעות $e_l^1,e_l^2$, עליהם נרשום $k$ (כלומר, זו סך הכל הזרימה שיכולה לעבור).
2. נריץ אלגוריתם למציאת זרימה ברשת המדוברת, ונסמן אותה ב-$f$.
3. אם $|f|=n$, נחזיר כי אפשר, אחרת נחזיר שאי אפשר.

**הוכחת נכונות**

צריך להוכיח כי השטף המקסימלי ב-$n$ הוא $|f|=n$ אם ורק אם ניתן להעביר את הסוכנים לפי האילוץ בבעייה. יש צורך להוכיח את שני הכיוונים כי האלגוריתם משתמש בשניהם: אנחנו מחזירים כן אס"ם מצאנו כי $|f|=n$.

בכיוון הראשון, נניח כי ניתן להעביר את הסוכנים. ניקח את תכנית ההעברה, נבנה בעזרתה זרימה $f$ ב-$N$ ונוכיח כי:

1. $f$ חוקית.
2. $|f|$ מקסימלי.
3. $|f|=n$.

נגדיר את $f$ כך:

- לכל צלע מהצורה $(s,d_i)\cup(c_k,t)$ נזרים $1$.
- לכל צלע מ-$\left(e_i^1,e_i^2\right)$ נזרים כמספר הסוכנים שעוברים בה (לכל היותר $k$).

---

[53] שימו לב שמדובר בטריק ידוע, כשאנו רוצים להגביל כמות בתוך קודקוד מסוים.

- לכל צלע מהצורה $\left(d_i, e_j^1\right), \left(e_j^2, c_k\right)$ נזרים $1$ אם"ם ב-$d_i$ מועבר ל-$c_k$ דרך $e_j$. כלומר, הזרמנו יחידת זרימה בדיוק לאורך כל מסלול בו עובר סוכן.
- בכל הצלעות האחרות נזרים $0$.

נוכיח כי $f$ מקיימת את כל התנאים לעיל:

1. קל לוודא, כי לכל $d_i$ יכולה להכנס זרימה אחת בלבד, מכל $c_i$ יוצאת יחידת זרימה אחת, ובכל $e = \left(e_i^1, e_i^2\right)$ עוברת זרימה כמספר הסוכנים שעוברים ב-$e$.
2. $f$ מקסימלית, כל כל קשת שיוצאת מ-$s$ רוויה אז לא ניתן להזרים יותר חומר.
3. לכל $1 \leq i \leq n$ מתקיים $f\left(s, d_i\right) = 1$ ולכן $|f| = \sum_{i=1}^{n} f\left(s, d_i\right) = \sum_{i=1}^{n} 1 = n$.

בכיוון השני, נניח כי הערך המקסימלי הוא , בגלל שכל הקיבולים **שלמים**, קיימת זרימה בשלמים עם **שטף** $n$, שנסמנה $f$.

בהכרח מובטח כי לכל $e \in A \cup B$ מתקיים כי $f\left(e\right) \in \{0, 1\}$. נתבונן באוסף הצלעות $\{e \in A \cup B \mid f(e) = 1\}$ ונגדיר העברה של הסוכנים.

לכל $i, j$, אם $f\left(d_i, e_j^1\right) = 1$ אזי סוכן שנמצא ב-$d_i$ עובר דרך $e_j$. לכל $j, k$ אם $f\left(e_j^2, c_k\right) = 1$ אזי הסוכן שהועבר דרך $e_j$ הועבר לעיר $c_k$ באסיה.

עלינו להראות כי מתקיימים התנאים:

1. $|f| = n$ ולכן $f\left(s, d_i\right) = 1$ לכל $i$. משימור החומר עבור קודקודי $d_i$, נקבל כי כל סוכן הועבר לעיר אחת באירופה.
2. משימור החומר עבור $e_j^i$, נקבל כי לכל $j$ מתקיים כי $\sum_{i=1}^{n} f\left(d_i, e_j^1\right) = f\left(e_j^1, e_j^2\right) \leq k$. כלומר יש לכל היותר $k$ סוכנים שעוברים דרך העיר $e_j$.
3. מאחר והערך הוא $n$, לכל $k$ מתקיים כי $f\left(c_k, t\right) = 1$. משימור הזרימה עבור קודקודי $k$ חייבת להיות קשת $\left(e_j^2, c_k\right)$ שעוברת דרכה יחידת זרם אחת. כלומר, הועבר סוכן אחד לכל אחת מ-$n$ הערים באסיה.

**תרגול מס' 7:**

**יום שלישי**

11.05.21

## 4.3 חתכים

**הגדרות**

**הגדרות**

נניח רשת זרימה $N$, זרימה חוקית $f$. אזי **הערך** (או השטף) של הזרימה הוא $|f| = \sum\limits_{u:s,u\in E} f(s,u)$.

נחלק את הרשת לשתי קבוצות קודקודים $S,T$ זרות, כאשר $t \in T$ ו-$s \in S$. כלומר $S \cap T = \emptyset,\ \ S \cup T = V$.
נגדיר את החתך להיות קבוצת הצלעות הבאה:

$$\{(u,v) \in E \mid u \in S, v \in T\}$$

קיבול של חתך $S,T$ יהיה $c(S,T) = \sum\limits_{(u,v)\in S\times T} c(u,v)$, כאשר אם $(u,v) \notin E$ אזי $c(u,v) = 0$ .

**טענה**

לכל זרימה חוקית $f$ ולכל חתך $S,T$ מתקיים כי $c(S,T) \geq |f|$.

אינטואיציה: מחוק שימור החומר, ומכאן שקיבולי כל החתכים הם אלו שיוצרים את צוואר הבקבוק של המסלול כולו.

**מסקנה**

אם מצאנו זרימה $f$ וחתך $(S,T)$ כך ש-$|f| = c(s,t)$, אזי מתקיים כי $f$ זרימת מקסימום (בעלת שטף מקסימלי וגם מתקיים כי $S,T$ חתך מינימום.

**הוכחה**

נניח כי $f$ לא מקסימלית. כלומר קיימת $f'$ כך ש-$|f'| > |f|$. אזי נקבל כי:

$$|f'| > |f| = c(s,t)$$

וקיבלנו סתירה לטענה הקודמת.
כמו כן, נניח כי $S,T$ לא בעל קיבול מינימלי, אזי קיימים $S',T'$ כך ש-$c(S',T') < c(S,T)$.
אזי מתקיים כי:

$$c(S',T') < c(S,T) = |f|$$

בסתירה.

**משפט השטף והחתך**

קיימת תמיד זרימה חוקית $f$ וחתך $S,T$ כך ש-$|f| = c(S,T)$.

### 4.3.1 דרגת קשירות של גרפים

יהי $G=(V,E)$ גרף קשיר לא מכוון. נרצה לבדוק את **מספר הצלעות המינימלי** שהופך אותו לגרף לא קשיר - נסמן אותו ב-$C(G)$.

קלט

גרף $G=(V,E)$.

פלט

$C(G)$.

**אלגוריתם**

- נבנה $|V|-1$ רשתות זרימה, כשכל קודקוד $t$ מגדיר רשת זרימה, שנסמנה על ידי $N_t=(E_t,V_t,c,s,t)$
- נבחר $s\in V$ שרירותית.
- לכל $v\in V\setminus\{s\}$:
  - $V_t=V$
  - $E_t$ יהיה שכפול של כל הצלעות ב-$E$.
  - $c=1$
- נחשב את הקיבול של החתך המינימלי $c(t)$, שנסמנו בתור $\min\limits_{\{t\neq s\}}(c(t))$ ונחזיר אותו.

**הוכחת נכונות**

צריך להוכיח כי $\min\limits_{\{t\neq s\}}(c(t))=C(G)$ - כלומר, קיבול החתך המינימלי שווה למספר הצלעות שהופכות אותו ללא קשיר.

נוכיח כי $\min\limits_{\{t\neq s\}}(c(t))\geq C(G)$

לכל חתך $S,T$, (מהגדרת חתך) הורדת כל הצלעות החתך מפרידה את הגרף לשני רכיבי קשירות (שבמקרה שלנו באחד יש את $t$ ובאחד יש את $s$), כיוון שכל הקיבולים הם $1$, אזי מספר הצלעות בחתך הוא בדיוק הקיבול (זה המקסימום זרימה שיכולה לעבור בחתך כולו). בעקבות כך, לכל $t\neq s$ מתקיים כי $c(t)\geq C(G)$ ולכן $\min\limits_{t\neq s}(c(t))\geq C(t)$.

נוכיח כי $C(G)\geq\min\limits_{\{t\neq s\}}(c(t))$.

נבחין כי הורדת הצלעות לפי $C(G)$ מייצרת שני רכיבי קשירות, שנסמם $S,T$ כאשר $s\in S$.

כיוון ש-$T\neq\emptyset$, יש בה לפחות קודקוד אחד, וכיוון שעברנו על הקודקודים בלולאה, מתישהו נקבל כי $t\in T$ ולכן:

$$\min_{t\neq s}c(t)\leq c(t)$$

$$\underbrace{\text{רק חלק מהצלעות}}_{\downarrow}$$

$$\leq C(T,S)\leq .$$

$$C(G)$$

מכאן נובע כי $C\left(G\right) = \min\limits_{t \neq s} c\left(t\right)$.

**זמן ריצה**

בכל איטרציה נבצע EK שלוקח $O\left(VE^2\right)$ וכיון שעשינו זאת $O\left(V\right)$ פעמים נקבל $O\left(V^2E^2\right)$.

### 4.3.2 בעיית המשקיעים והשחקנים

<u>קלט</u>

- $A = \{a_1, \ldots, a_n\}$ קבוצת שחקנים. לכל שחקן $a_i$ יש משכורת $s_i$ מתאימה.
- $B = \{b_1, \ldots, b_k\}$ קבוצת משקיעים. לכל משקיע $b_i$ יש תרומה של $d_i$.

לכל משקיע $b_j$ יש קבוצת שחקנים $A_j \subseteq A$, כך שאם כל השחקנים ב-$A_j$ בסרט, המשקיע ייתן לנו כסף $d_j$.

<u>פלט</u>

$P\left(A', B'\right) = \sum\limits_{b_i \in B} d_i - \sum\limits_{a \in A_i} s_i$ מקסימלי.

**האלגוריתם**

1. נבנה רשת זרימה כך:
   - $V = \{s, t\} \cup A \cup B$
   - $E = \{(s, b_i) \mid b_i \in B\} \cup \{(b_i, a_j) \mid a_j \in A_i\} \cup \{(a_i, t) \mid a_i \in A\}$
   -

$$c(v,u) = \begin{cases} d_i & u = s, v \in B \\ \infty & u = b_i \in B, v \in A_i \\ s_i & u \in A, v = t \end{cases}$$

2. נמצא חתך מינימלי $(S,T)$ ברשת.
3. נגדיר $A' = S \cap A$ ו-$B' = S \cap B$ ונחזיר את $A', B'$.

**הוכחת נכונות**

נגדיר פונקציה חח"ע ועל בין קבוצות $A', B'$ בין משקיעים לחתכים:

- לכל חתך $(S,T)$ נגדיר $B' = S \cap B$ – ו-$A' = S \cap A$
- לכל $A', B'$ נגדיר את החתך $S = \{s\} \cup A' \cup B'$ -$T = V \setminus S$.

**טענה**

$A', B'$ חוקיות אם ורק אם $c\left(S,T\right)$ סופיים.

**הוכחה**

$c(S,T)$ סופי אם ורק אם אין לו צלעות מקיבול $\infty$, אם ורק אם אין צלעות מ-$b_i$ ל-$a_j$, אם ורק אם לכל $a_i \in A_i$ קיים $S$, משמע הפתרון חוקי.

**טענה**

יהי $(S,T)$ חתך מקיבול סופי, אזי $c(A',B') = D - p(S,T)$ כאשר $D$ קבוע ולא תלוי ב-$A',B'$.

מכאן ניתן להוכיח את חוקיות האלגוריתם (על סמך זה שאין צלעות מקיבול $\infty$) ואת האופטימיליות (בהסתמך על הטענה הקודמת ומינימליות החתך).

**תרגול מס' 8:**

**יום שלישי**

18.05.21

# 5 תכנון ליניארי

## 5.1 הגדרות

בדומה לרשת זרימה, תכנון ליניארי היא דרך להכליל בעייה מוכרת לנו לבעיות אחרות.

**הגדרה**

בעיית אופטימיזציה תקרא בעיית תכנון ליניארי ($LP$) אם ניתן לכתוב אותה בצורה הבאה:

$$\begin{aligned} &\max_{x\in\mathbb{R}^m} c^\top x \\ &s.t\ \ Ax \le b \\ &x \ge 0 \end{aligned}$$

כאשר $x \in \mathbb{R}^m$, ו-$c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$ וגם $A \in \mathbb{R}^{m\times n}$

**הערות**

1. הצורה הנ"ל תיקרא הצורה הסטנדרטית לבעיית $LP$. ישנן צורות שונות ששקולות לבעייה הסטנדרטית. (נראה בתרגיל).
2. אי השוויונים $Ax \le b$ ו-$x \ge 0$ מוגדרים כא"ש לכל קואורדינטה בנפרד.
3. זוהי בעיית אופטימיזציה קלאסית. כלומר, יש סט פתרונות חוקיים (כל וקטור $x$ המקיים את $m$ א"ש $Ax \le b$) ואנו מחפשים בתוך קבוצה זו את $x$ האופטימלי (שממקסם את $c^\top x$).
4. ניתן להגדיר את בעיית התכנון הליניארי גם כבעיית מינימיזציה:

$$\begin{aligned} \text{min} \quad & c^\top \cdot x \\ \text{s.t} \quad & Ax \ge b \\ & x \ge 0 \end{aligned}$$

על ידי החלפת הסימנים של $A, b, c$.

**הגדרה**

יהי וקטור $a \in \mathbb{R}^m$ וסקלר $b \in \mathbb{R}$:

- ☐ הקבוצה $\left\{x \in \mathbb{R}^n \mid a^\top x = b\right\}$ נקראת **על מישור** (Hyperplane).
- ☐ הקבוצה $\left\{x \in \mathbb{R}^n \mid a^\top x \leq b\right\}$ נקראת **חצי מרחב** (Halfspace).

**הגדרה**

קבוצת נקודות ב-$\mathbb{R}^n$ שניתן לתאר בצורה $\left\{x \in \mathbb{R}^n \mid A^\top x \leq b\right\}$ כאשר $A \in \mathbb{R}^{m \times n}$ ו-$b \in \mathbb{R}^m$ נקראת פוליהדרון.

נשים לב כי קבוצת הפתרונות לבעיית התכנון הליניארי היא פוליהדרון, הבנוי מחיתוך של $m+n$ חצאי מרחבים.

**דוגמה**

נניח כי $x \in \mathbb{R}^2$. נגדיר את $A, b, c$ להיות:

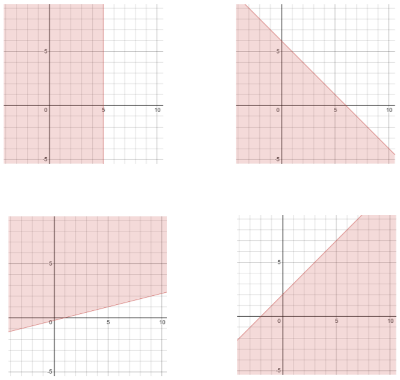
$$A = \begin{bmatrix} 1 & 1 \\ 1 & 0 \\ -2 & 2 \\ 1 & -4 \end{bmatrix}, b = \begin{bmatrix} 6 \\ 5 \\ 4 \\ 1 \end{bmatrix}, c = \begin{bmatrix} 3 \\ -1 \end{bmatrix}$$

נפרק את המטריצה והווקטור ל-4 א"ש:

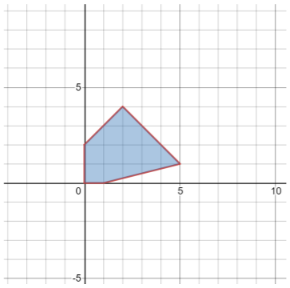
$$Ax \leq b \Leftrightarrow \quad \begin{matrix} x_1 + x_2 \leq 6 \\ x_1 \leq 5 \\ -2x_1 + 2x_2 \leq 4 \\ x_1 - 4x_2 \leq 1 \end{matrix}$$

למעשה, בתוך קבוצה זו אנחנו מחפשים את $x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ הממקסם את $c^\top x = 3x_1 - x_2$. אפשר לראות כי הוקטור הממקסם הוא $\begin{bmatrix} 5 \\ 1 \end{bmatrix}$.

מבחינה אלגברית, נחפש את חצאי כל המרחבים הבאים:



שיוצרים את הקבוצה הבאה:



הוקטור הממקסם הוא כמובן $\begin{bmatrix} 5 \\ 1 \end{bmatrix}$.

## 5.2 פתרון בעיית $LP$

ידועים כמה אלגוריתמים שפותרים את בעיית התכנון הליניארי בזמן פולינומי בגודל הקלט $O(m+n)$. עם זאת, בקורס לא נראה אלגוריתם כזה. נניח כי נתונה קופסה שחורה שפותרת את בעיית $LP$ בזמן פולינומי ב-$m+n$.

### 5.2.1 התרמיל השברי כבעיית $LP$

**תזכורת:** [54]

נתונים $n$ פריטים שלכל אחד משקל $w_i$ וערך $v_i$. מחפשים וקטור $x \in \mathbb{R}^n$ כך ש-$x_i \leq 1$ וגם $\sum\limits_{i=1}^{n} w_i x_i \leq W$ הממקסם את $\sum\limits_{i=1}^{n} v_i x_i$.

נרצה להציג זאת כבעיית תכנון ליניארי, כלומר להגדיר את $A, b, c$.

נוכל לעשות זאת כך:

---

[54] למעשה, למדנו רק את התרמיל השלם ולא את התרמיל השברי.

$$A=\begin{pmatrix} w_1 & w_2 & \cdots & w_n \\ 1 & 0 & & 0 \\ 0 & 1 & & 0 \\ & & \ddots & \\ 0 & 0 & & 1 \end{pmatrix}, b=\begin{pmatrix} W \\ 1 \\ 1 \\ \vdots \\ 1 \end{pmatrix}, c=\begin{pmatrix} v_1 \\ \\ \vdots \\ v_n \end{pmatrix}$$

נשים לב כי כדי לקבל פתרון לתרמיל השלם, היינו צריכים להוסיף אילוצים $x_i \in Z$. לבעייה כזו קוראים תכנון ליניארי בשלמים ($ILP$).

**הגדרה**

בעייה תיקרא $ILP$ אם ניתן לכתוב אותה כך:

$$\begin{aligned} \max \quad & c^\top \cdot x \\ \text{s.t} \quad & Ax \leq b \\ & x \geq 0 \\ & x \in \mathbb{Z}^n \end{aligned}$$

חלק גדול מהבעיות שמעניינות אותנו ניתנת להצגה כ-ILP. לצערנו, מסתבר ש-ILP גם היא בעיית NP קשה.

### 5.2.2 שטף מקסימלי כבעיית $LP$

בהינתן רשת זרימה $N=(G,c,s,t)$, נסמן לכל $(i,j)\in E$ את $c_{i,j}$ - הקיבול של $(i,j)$. באופן דומה, $f_{ij}$ זו הזרימה על $(i,j)$.

נשים לב כי אנו רוצים למצוא $\max_{f\in\mathbb{R}^{|E|}} \sum_{(s,i)\in E} 1\cdot f_{si}$.

נחפש $x\in\mathbb{R}^{|E|}$ ונגדיר את פונקציית המטרה להיות $d^\top x$ כאשר :

$$x=\begin{pmatrix} 1 \\ f_{ij} \\ | \end{pmatrix}\in\mathbb{R}^{|E|},\quad d_{ij}=\mathbb{1}_{[i=s]}$$

נרצה כי הזרימה תהיה חיובית וזה מגיע מהדרישה $x\geq 0$. כמו כן, נרשום את אילוץ הקיבול כך:

$$\forall(i,j)\in E,\quad f_{i,j}\leq c_{ij}$$

את אילוץ שימור החומר נכתוב כך:

$$\forall i \in V\backslash\{s,t\}, \qquad \sum_{(i,x)\in E} f_{ix} - \sum_{(x,i)\in E} f_{xi} = 0$$

הבעייה כי LP דורש א"ש ובמקרה כאן יש לנו שויון. נוכל לבצע את הפתרון כך:
נשים לב כי $a=0$ אם ורק אם $(a \le 0) \wedge (a \ge 0)$ ששקול למעשה לכך ש-$(a \le 0) \wedge (-a \le 0)$.
אם כך, במקרה שלנו שימור החומר יתורגם לאילוצים כי:

$$\forall i \in V\backslash\{s,t\}: \ \begin{array}{l} \sum_{(i,x)\in E} f_{ix} - \sum_{(x,i)\in E} f_{xi} \le 0 \\ \sum_{(x,i)\in E} f_{xi} - \sum_{(i,x)\in E} f_{ix} \le 0 \end{array}$$

**בניית הצורה הסטנדרטית**

אנו מחפשים את $\max\limits_{x\in\mathbb{R}^{|E|}} d^\top x$ כך ש- $Ax \le b$ וגם $x \ge 0$ כאשר $x = \begin{pmatrix} 1 \\ f_{ij} \\ | \end{pmatrix} \in \mathbb{R}^{|E|}, d_{ij} = \mathbb{1}_{[i=s]}$.

כמו כן, נוכל לסמן:

$$A = \left(\begin{array}{ccccc} 1 & 0 & 0 & \cdots & 0 \\ 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & 1 \\ \hline \\ & & M & & \end{array}\right) \in \mathbb{R}^{(|E|+|V|-2)\times|E|}, \quad b = \begin{pmatrix} | \\ c_{ij} \\ \frac{|}{0} \\ \vdots \\ 0 \end{pmatrix} \in \mathbb{R}^{|E|+|V|-2}, c_e = \mathbb{1}_{[e=(s,i)]}$$

כאשר $M$ מקיימת:[55]

$$M_{lk} = \begin{cases} +1 & \exists u \in V, k=(u,l) \in E \\ -1 & \exists u \in V, k=(l,u) \in E \\ 0 & \text{otherwise} \end{cases}$$

**זמן הריצה**

נסמן ב-$f(m+n)$ את זמן הריצה של פתרון בעיית $LP$ עם $(m+n)$אילוצים.

- הגדרת $d$ עולה $O(|E|)$.
- הגדרת $b$ עולה $O(|V|+|E|)$.

---

[55] $M$ מייצגת את האילוצים של שימור החומר. יש שתי שורות לכל קודקוד ועמודה לכל צלע. עבור קודקוד $l$ נסמן ב-$l^1$ את האילוץ הראשון על $l$ וב-$l^2$ את האילוץ השני. נסמן ב-$k$ את הצלע ה- $k$ . אם $k$ נכנסת לקודקוד ה- $l$ אז צריך לשים מקדם חיובי לזרימה של הצלע הזאת כדי שאי השוויון יתקיים. באותו אופן עבור צלע שיוצאת מ- $l$ צריך מקדם שלילי.

- הגדרת $A$ עולה $O\left(|E|\cdot|V|\right)$.
- פתרון $LP$ עולה $O\left(f\left(|E|+|V|\right)\right)$.

לכן סך הכל קיבלנו זמן ריצה של:

$$O\left(|E|+|V|+f\left(|E|+|V|\right)\right)$$

## 5.3 בעיית הסרת משולשים

קלט:
גרף לא מכוון $G=(V,E)$.

פלט:
תת קבוצה של צלעות $S\subseteq E$ שהסרתה מ-$G$ תשאיר גרף $G'=(V,E\setminus S)$ ללא משולשים (משולש הוא קליקה מגודל 3).

**בניית הפתרון**
נגדיר וקטור משתנים $x\in\{0,1\}^{|E|}$ המציין לכל צלע $e\in E$ האם היא ב-$S$ $(x\left[e\right]=1)$ או לא $(x\left[e\right]=0)$. נשים לב כי $|S|=\sum\limits_{e\in E}x\left[e\right]$.
לכן , נגדיר את פונקציית המטרה להיות:

$$\min_{x\in\mathbb{R}^{|E|},e\in E}x\left[e\right]$$

בנוסף, נדרוש שלכל משולש ב-$G$, לפחות אחת מהצלעות במשולש מקיימת $x\left[e\right]=1$ (כאשר נסיר צלע זו, המשולש יישבר). סך הכל, נקבל את התוכנית הלינארית בשלמים הבאה:

$$\min_{x\in\mathbb{R}|E|}\textstyle\sum_{e\in E}x$$

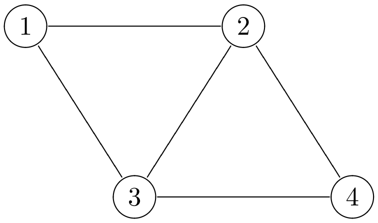
$$\forall u,v,w\in V\quad \text{s.t. }(u,v),(v,w),(w,u)\in E:$$

$$x[(u,v)]+x[(v,w)]+x[(w,u)]\geq 1$$

$$\forall e\in E\quad 0\leq x[e]\leq 1$$

$$\forall e\in Ex[e]\in\mathbb{Z}$$

למשל, עבור הדוגמה הבאה:



נקבל כי $X\in\mathbb{R}^5$ ו-$C=\mathbb{1}^5$.

כמו כן, נקבל:

$$A=\left[\begin{array}{ccccc} {\scriptstyle (1,2)} & {\scriptstyle (1,3)} & {\scriptstyle (3,4)} & {\scriptstyle (3,2)} & {\scriptstyle (2,4)} \\ 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ & & -I & & \end{array}\right]$$

כמות משולשים:

$$b=\begin{pmatrix} \text{כמות משולשים}\left\{\begin{matrix}1\\|\\1\end{matrix}\right. \\ |E|\left\{\begin{matrix}-1\\|\\-1\end{matrix}\right. \end{pmatrix}$$

**תרגול מס' 9:**

## 6 אלגוריתמי קירוב

**יום שלישי**

25.05.21

**הגדרה**

יהי $X$ מרחב הפתרונות החוקיים לבעיית אופטימיזציה. עבור פונקציה $f:X\to\mathbb{R}^+$ וקלט מסוים לבעיה, נסמן ב-$x^*$ את הפתרון האופטימלי. כלומר, לכל $x\in X$ יתקיים כי $f(x^*)\le f(x)$.

**הגדרה**

יהי $c\ge 1$. נאמר שאלגוריתם הוא $c$-מקרב עבור הבעיה $\min\limits_{x\in X} f(x)$ אם לכל קלט, האלגוריתם מחזיר פתרון חוקי $x\in X$ בזמן יעיל המקיים $f(x)\le f(x^*)\cdot c$.

עבור מקסימיזציה מתקיים כי $f(x)\ge\frac{1}{c}f(x^*)$.

**הערות**

1. לרוב נקצר ונסמן ב-opt את ערך הפתרון האופטימלי.
2. באלגוריתמי קירוב, הפתרון שלנו תמיד יהיה גדול (במינימיזציה\קטן במקסימיזציה) מהפתרון האופטימלי.

## 6.1 בעיית ה-3-SAT

באופן כללי, לא מדובר בבעיית אופטימיזציה אלא בבעיית הכרעה (כן או לא). האם קיימת לנוסחת 3-CNF השמה מספקת.

הגדרת הבעייה

- משתנים בוליאנים $x_i \in \{\mathbb{T}, \mathbb{F}\}$
- ליטרלים - משתנה $x_i$ או שלילתו $\neg x_i$.
- פסוקית בנוסחת 3-CNF : אוסף של שלושה ליטרלים ודיסיונקציה[56] (או אימום) ביניהם. למשל, $(x_1 \vee \neg x_n \vee x_4)$.
- קוניונקציה[57] (או גימום) של פסוקיות. למשל: $(x_1 \vee \neg x_n \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$.

קלט

נוסחת $3\text{-}CNF$ בעלות פסוקיות $C_1, \ldots, C_n$ מעל $n$ המשתנים $x_1, \ldots, x_n$. כמו כן, נניח כי $n \leq 3m$.

**הנחה**

המשתנים בכל פסוקית שונים זה מזה.

פלט

האם קיימת השמה ל-$x_1, \ldots, x_n$ כך שערך הנוסחה הוא $\mathbb{T}$ .

מדובר בבעיה (סופר מעניינת!) $NP$ קשה. לכן סביר שלא נמצא פתרון יעיל עבור. נתפשר על פתרון מקורב לבעית אופטימיזצייה דומה.

פלט אלטרנטיבי

השמה ל-$X_1, \ldots, X_n$ שממקסמת את מספר הפסוקיות שמקבלות $T$ .
למשל:

$$\underbrace{(\overbrace{x_1}^{\text{literal}} \vee \overbrace{\neg x_3}^{\text{literal}} \vee \neg x_4)}_{C_1} \wedge \underbrace{(\neg x_1 \vee \neg x_3 \vee x_4)}_{C_2}$$

ההשמה $(x_1, x_2, x_3, x_4) = (\mathbb{T}, \mathbb{F}, \mathbb{T}, \mathbb{F})$ מספקת פסוקית 1. לעומת זאת, ההשמה $(x_1, x_2, x_3, x_4) = (\mathbb{F}, \mathbb{F}, \mathbb{F}, \mathbb{T})$ מספקת תשובה לבעיה.

---

[56] דיסְיוּנְקְצְיָה: 1. אִחוּד לוֹגִי – פעולת "אוֹ" המסומנת בסימן "$\vee$" ולעיתים גם בסימן "$+$". טענה מהצורה "$\alpha$ או $\beta$" (למשל "היום יום ראשון או עכשיו יורד גשם") נקראת דיסיונקציה או טענה דיסיונקטיבית ואפשר לכתוב אותה: $\alpha \vee \beta$.

[57] קוֹנְיוּנְקְצִיָּה: 1. חִתּוּךְ לוֹגִי – פעולת "גַּם" המסומנת בסימן "$\wedge$" ולעיתים גם בסימן "$\cdot$". טענה מהצורה "$\alpha$ וגם $\beta$" (למשל "היום יום ראשון וגם עכשיו יורד גשם") נקראת קוניונקציה או טענה קוניונקטיבית ואפשר לכתוב אותה: $\alpha \wedge \beta$.

### 6.1.1 אלגוריתם $2$-מקרב נאיבי

1. נבדוק כמה פסוקיות מסופקות על ידי ההשמה $\overrightarrow{x_{\mathbb{F}}} = (\overbrace{\mathbb{F},\mathbb{F},\ldots,\mathbb{F}}^{n\text{ times}})$ ונסמן מספר זה ב-$f$.
2. נבדוק כמה פסוקיות מסופקות על ידי ההשמה $\overrightarrow{x_{\mathbb{T}}} = (\overbrace{\mathbb{T},\mathbb{T},\ldots,\mathbb{T}}^{n\text{ times}})$ ונסמן מספר זה ב-$t$.
3. אם $f < t$ נחזיר את ההשמה $\overrightarrow{x_{\mathbb{T}}}$, אחרת נחזיר את $\overrightarrow{x_{\mathbb{F}}}$.

**הוכחת נכונות $2$- קירוב**

לכל $c_i$ מתקיים כי $\overrightarrow{x_{\mathbb{F}}}$ מספקת אותה או $\overrightarrow{x_{\mathbb{T}}}$ אותה, או שתיהן.
מכאן נובע כי $t + f \geq m$ ולכן $\max\{f,t\} \geq \frac{m}{2}$ ולכן מכאן אנחנו מחזירים השמה שמספקת לפחות חצי מ-$C_1,\ldots,C_m$. כאן, כיוון ש-$\text{opt} \leq m$, קיבלנו $2$-קירוב.

**הערות**

1. האלגוריתם היה עובד עבור כל השמה $\vec{x}$ ושלילתה.
2. האלגוריתם לא משתמש בכך שיש $3$ ליטרלים בכל פסוקית. לכן הוא עובד לכל נוסחת CNF.

## 6.2 בעיית התרמיל השלם

תזכורת

נתונים $n$ פריטים עם נפחים $V_1,\ldots,V_n$ משקלים $w_1,\ldots,w_n$, והגבלת נפח $V$.
(מניחים $\sum_i v_i \geq V$ ו-$V_i \leq V$).
המטרה - למצוא תת קבוצה של פריטים עם סכום משקלים מקסימלי וסכום נפחים שאינו עולה על $V$.

### 6.2.1 שאלה ממבחן $2016$ מועד א'

1. הראו כי האלגוריתם החמדן שבוחר פריטים לפי ערכם הסגולי $p_i = \frac{w_i}{V_i}$ אינו משיג $2$-קירוב.
2. הציעו תיקון לאלגוריתם החמדן, כך שישיג $2$-קירוב.

**פתרון לשאלה $1$**

בהינתן $c$, נצטרך להראות קלט עם פתרון אופטימלי $x^*$ כך שהאלגוריתם החמדן משיג $x$ עם $f(x) < \frac{1}{c} f(x^*)$.
בהינתן $c$, נבחר $V$ המקיים כי $c < V - 1$. נסתכל על הקלט:

| מספר פריט | $w_i$ | $v_i$ | $p_i = \frac{w_i}{v_i}$ |
|---|---|---|---|
| 1 | 1 | 1 | 1 |
| 2 | $V-1$ | $V$ | $1-\frac{1}{V}$ |

האלגוריתם החמדן יבחר את $(0,1)$ למרות שהפתרון הוא $(0,2)$.
לכן $f(x^*) = V - 1$. כלומר מתקיים כי $f(x) = \frac{1}{V-1} f(x^*) < \frac{1}{c} f(x^*)$.
בפרט, לא מדובר ב-$2$ קירוב.

**פתרון לשאלה $2$**

□ מיין את הפריטים לפי הערך הסגולי, כלומר $p_i = \frac{w_i}{V_i}$.

□ מצא את $k$ המינימלי המקיים $\sum_{i=1}^{k} V_i > V$.

□ מבין $x_1 = (\overbrace{1, \ldots, 1}^{k-1}, 0, \ldots, 0)$ ו-$x_2 = (\overbrace{0, \ldots, 0}^{k-1}, 1, 0, \ldots, 0)$ נחזיר את $x \in \{x_1, x_2\}$ שממקסם את הפתרון, כלומר $f(x) = \max\{f(x_1), f(x_2)\}$

**חוקיות הפתרון**

הפתרון הוא בחירה בין שני פתרונות חוקיים ($x_1$ חוקי ממזעריות $k$, השני חוקי כי לכל $k$ מתקיים כי $v_k \leq V$).
לכן הפתרון כולו חוקי.
יהי $x^*$ פתרון אופטימלי לבעיה.

**טענה**

הפתרון $x$ שמוחזר על ידי האלגוריתם, מקיים $f(x) \geq \frac{1}{2} f(x^*)$.

**הוכחה**

יהי $z^*$ הפתרון האופטימלי לבעיית התרמיל **השברי** באותו קלט.

מכיוון שכל פתרון לבעיית התרמיל השלם הוא גם פתרון לשברי, נקבל כי:

$$f(x^*) \leq f(z^*)$$

ניזכר כי:

$$z^* = (\overbrace{1, \ldots, 1}^{k-1}, \alpha, \ldots, 0)$$

עבור $\alpha \in [0, 1]$.

מכאן נובע כי:

$$\begin{aligned}
&f(x^*) \leq f(z^*) = \overbrace{\sum_{i=1}^{k-1} v_i}^{f(x_1)} + \alpha \overbrace{v_k}^{f(x_2)} \\
&= f(x_1) + \alpha f(x_2) \\
&\overset{\alpha \leq 1}{\leq} f(x_1) + f(x_2) \\
&\leq 2 \max\{f(x_1), f(x_2)\} = 2f(x)
\end{aligned}$$

ולכן נקבל כי $\frac{1}{2} f(x^*) \leq f(x)$, כנדרש.

**זמן הריצה**

נדרוש מיון ב-$O(n \log n)$ וזהו יהיה הזמן המרכזי.

## 6.3 בעיית החתך המקסימלי Max-Cut

יהי $G=(V,E)$ גרף לא מכוון.
חתך $A,B$ בגרף $G$ הוא אוסף הצלעות:

$$\{(u,v)\in E \mid u\in A, v\in B\}$$

כאשר $A,B$ חלוקה של $V$ ל-2 קבוצות זרות.

<u>קלט</u>
גרף לא מכוון $G=(V,E)$
<u>פלט</u>
חתך בו מספר הצלעות הוא מקסימלי.
מדובר בבעיית NP קשה ולכן נמצא אלגוריתם 2 -מקרב.

**אלגוריתם**
נמספר את הקודקודים $V=(V_1,\dots,V_n)$.

1. נתחיל בחלוקה טריוויאלית $A=V, B=\emptyset$.
2. נעבור על הקודקודים לפי סדר המספור, ולכל קודקוד נבדוק, אם מספר השכנים שלו בקבוצה **שלו גדול ממספר השכנים בקבוצה השנייה - נעביר אותו**. לא תמיד מדובר על מעבר מ-$A$ ל-$B$, אלא לפעמים גם מעבר בכיוון השני (תכף נראה).
3. נחזור על שלב 2 עד שלא יישארו קודקודים להעביר.

**הוכחת נכונות**
מספר הצלעות בחתך חסום מלמעלה על ידי $|E|$. בכל העברה של קודקוד, מספר הצלעות בחתך עולה ב-1 לפחות.
לכן האלגוריתם בהכרח עוצר.
עלינו להראות חוקיות ונכונות הקירוב.

**חוקיות**
אנחנו מתחילים עם חתך חוקי, ובכל איטרציה אנחנו מעבירים קודקוד בודד מקבוצה אחת לאחרת. לכן בכל איטרציה אנחנו נשארים עם חתך חוקי.

**נכונות הקירוב**
נסמן ב-$C\left(A,B\right)$ את החתך שמחזיר האלגוריתם.
לכל $v\in V$ נסמן בתור $d\left(v\right)$ את הדרגה של $v$ (מספר השכנים). כמו כן, נגדיר כי $v_C$ בתור הצלעות שנוגעות ב-$V$ וחוצות את החתך $C$.

**טענה**
$|C|\geq\frac{1}{2}\text{opt}$
**הוכחה**
נבחין כי מתקיים:

$$
\begin{aligned}
&|C| \overset{\substack{\underbrace{\text{כל צלע נספרת פעמיים}} \\ \downarrow}}{=} \\
&\frac{1}{2}\sum_{v\in V}|v_C| \overset{\substack{\underbrace{\text{אחרת היינו מעבירים לקבוצה השנייה}} \\ \downarrow}}{\geq} \\
&\frac{1}{2}\sum_{v\in V}\frac{1}{2}d\left(v\right) = \\
&\frac{1}{4}\sum_{v\in V}d\left(v\right) \overset{\substack{\underbrace{\text{סכום דרגות קודקדים}} \\ \downarrow}}{=} \\
&\frac{1}{4}\cdot 2\left|E\right| = \frac{1}{2}\left|E\right| \geq \\
&\frac{1}{2}\text{opt}
\end{aligned}
$$

**זמן הריצה**

בשלב 2 - האלגוריתם עושה לכל היותר $|E|$ איטרציות.

בכל איטרציה עוברים על כל הקודקודים וכל הצלעות.

לכן זמן הריצה הוא:

$$O\left(\left|E\right|\cdot\left(\left|E\right|+\left|V\right|\right)\right)$$

## 7 בעיות סיווג - Online learning

**תרגול מס' 10:**

**יום שלישי**

01.06.21

בבעיות סיווג (clasfification) המטרה היא לחילוק קבוצה של פריטים לתת קבוצות. לדוגמא - לסווג תפוחים לקבוצה של תפוחים בשלים וקבוצה של פתוחים לא בשלים.

אנחנו נתמקד בשיעור בסיווג בינארי - נצטרך לחלק לשתי קבוצות בלבד. בתרגול זה, נתמקד בסוג מסוים של בעיות סיווג - למידת אונליין בעזרת מומחים.

**סמנטיקה**
ישנם $T$ סיבובים, כך ש-$1 \leq t \leq T$.
בכל סיבוב מקבלים פריט $x_t \in X$. לכל פריט יש סיווג $y_t \in \{-1,1\}$ שאותו נרצה למצוא.
בנוסף, יש לנו קבוצה של $N$ מומחים, $f_1,\ldots,f_n$ כך שכל מומחה נותן את דעתו על הסיווג של $x_t$.
באופן פורמלי, נגדיר כי $f: X \to \{-1,1\}$.
איננו יודעים מראש כמה ה"מומחים" טובים.

נשחק "משחק":

1. נאתחל את מספר הטעויות שלנו. $mistakes = 0$.
2. עבור כל אחד מהסיבובים $1 \leq t \leq T$:
   - (א) מקבלים אובייקט מהסביבה $x_t$ ורשימה של $N$ סיווגים מכל המומחים $f_1^t,\ldots,f_n^t$.
   - (ב) יוצרים סיווג משלנו $\hat{y}_t$ כפונקציה של סיווג המומחים.
   - (ג) מקבלים מהסביבה את הסיווג האמיתי $y_t$.
   - (ד) אם $\hat{y}_t \neq y_t$ מעלים באחד את הערך של $mistakes$.

**המטרה**
לייצר סיווגים $\hat{y}_1,\ldots,\hat{y}_t$ כך שמספר הטעויות הכולל יהיה נמוך כמה שניתן.

## 7.1 אלגוריתם halving

בכל סיבוב נשתמש רק במומחים שצדקו בכל הסיבובים הקודמים. מבין הסיווגים האלו, נבחר את זה שהרוב מסכימים עליו. פורמלית:
בכל סיבוב נגדיר את $\text{experts}_t = \left\{ f_i \mid f_i^d = y_d \ \ \forall 1 \leq d \leq t-1 \right\}$.
נחזיר את הסיווג:

$$\hat{y}_t = \begin{cases} 1 & \mid \{ f_i \in \text{ experts }_t : f_i^t = 1 \} \mid \geq \mid \{ f_i \in \text{ experts }_t : f_i^t = -1 \} \mid \\ -1 & \text{otherwise} \end{cases}$$

**טענה**
אם יש מומחה מושלם $f_i$ שצודק תמיד, אזי מתקיים כי $mistakes \leq \log(n)$
**הוכחה**
בכל סיבוב $t$ בו עשינו טעות, רוב המומחים ב-$\text{experts}_t$ טעו ולכן:

$$\left|\text{experts}_{t+1}\right| \leq \frac{1}{2}\left|\text{experts}_t\right|$$

כיוון ש-$\left|\text{experts}_t\right| = N$, לאחר לכל היותר $\log(N)$ טעויות, נישאר עם קבוצה בגודל 1 מכיוון שהמומחה שנותר הוא מושלם, לא נבצע יותר טעויות.

נשים לב כי קיבלנו חסם עליון על כמות הטעויות שהאלגוריתם יכול לעשות. כמו כן, חסם זה לא תלוי במספר הסיבובים. נוכל להבחין כי חסם זה הינו הדוק.
אם נבחר $N = 2^n$ (חזקה כלשהי של 2), ונחליט שרק אחד הוא מושלם. נסדר את הדוגמאות ככה שבכל סיבוב בדיוק חצי יטעו וחצי יצדקו. בהרצה זו, האלגוריתם יעשה בדיוק $\log(n)$ טעויות לפני שנישאר עם קבוצה בגודל $1$.
ניתן להוכיח שזו השגיאה המינימלית לכל אלגוריתם סיווג (תחת ההנחה של מומחה מושלם). לכן, אלגוריתם halving מסווג את התוצאה הטובה ביותר האפשרית.
אמנם - הנחנו הנחה חזקה מאוד והוא שקיים מומחה מושלם. ברוב המקרים לא יהיה כזה, ולכן נצטרך לפנות לאלגוריתמים אחרים.

## 7.2 אלגוריתם רוב ממושקל

לכל מומחה $f_i$ ניצור משקולת $w_i$ שתייצג כמה "משקל" אנחנו מייחסים לדעתו של המומחה $f_i$.
נאתחל $w_i = 1$ לכל $i \in [n]$. בכל סיבוב נחזיר את הניחוש

$$\hat{y}_t = \text{sign}\left(\sum_i w_i^t f_i^t\right)$$

בסוף כל סיבוב, נעדכן כל מומחה שטעה באופן הבא:

$$w_i^{t+1} = \frac{1}{2} w_i^t$$

נסמן ב-$mistakes$ את מספר הטעויות של המומחה $f_i$. הוכחנו בכיתה כי:

$$\forall i \;\; mistakes \leq \frac{1}{\log\left(\frac{4}{3}\right)}\left(mistakes_i + \log(N)\right)$$

נשים לב כי אם קיים מומחה מושלם, אזי מספר הטעויות חסום על ידי:

$$mistakes \leq \frac{1}{\log\left(\frac{4}{3}\right)}\left(0 + \log(N)\right) = 2.41 \log(N)$$

כלומר, אנחנו במצב פחות טוב מאלגוריתם halving, אבל יש לנו אלגוריתם שעובד טוב גם בלי מומחה מושלם.

**הערה**

נשים לב שמכיוון ש-$\hat{y}_t \in \{-1, 1\}$, צריך לשנות את הפונקציה sign שתחזיר 1 או $-1$ במקרה של $\text{sign}(0)$, לכן נבחר שרירותית במקרה זה.

## 7.3 משקולות כפליים

נכליל מעט את הבעייה - במקום לספור לכל מומחה כמה טעויות הוא ביצע, נגדיר פונקציה על הסיווג של כל מומחה בכל סיבוב - $(\text{punishment})_i^t \in [-1,1]$. נשים לב שניתן לתת "קנס" שלילי (בעצם זהו פרס) על תשובה נכונה. נוכל להבחין כי מדובר בהכללה על הבעייה הקודמת, אם נגדיר:

$$(\text{punishment})_i^t = \begin{cases} 1 & f_i^t \neq y_t \\ 0 & \text{otherwise} \end{cases}$$

כלומר על תשובות שגויות תמיד "קנס" של $1$ ,ועל תשובות נכונות קנס של $0$, נקבל כי במהלך כל הסיבובים כל מומחה יקבל תמיד "קנס" בגודל מספר הטעויות שעשה.

במקרה הקודם, מדדנו את עצמנו למול המומחה שטעה הכי מעט. כעת, נמדוד את עצמנו למול המומחה שקיבל את הקנס הכי נמוך:

$$\min_i \sum_{t=1}^{T} (\text{punishment})_i^t \doteq \sum_{t=1}^{T} (\text{punishment})_{\text{OPT}}^t$$

נרצה להגדיר כמה קנס נשלם עבור הסיווגים שלנו. כדי לעשות זאת, במקום סיווג משלנו, נבחר מומחה להקשיב לו ונסווג לפי המומחה בסיבוב זה.

נקבל בסופו של דבר את הקנס שקיבל המומחה.

**האלגוריתם**

1. לכל מומחה נחזיק משקולת $w_1, \ldots, w_n$.
2. נאתחל $w_i = 1$ לכל $i \in [n]$.
3. בבואנו לבחור מומחה להקשיב לו בסיבוב $t$, נדגום מומחה מקבוצת המומחים לפי ההתפלגות:

$$p_i^t = \frac{w_i^t}{\sum\limits_j w_j^t}$$

בעקבות העובדה כי בכל סיבוב אנו מגרילים מומחה לו נקשיב, נוכל לקבל קנסות שונים כתלות בסוכן שאיתו באמת דגמנו. לכן, במקום להסתכל על ה"קנס" הספציפי שקיבלנו בהרצה מסוימת, נתבונן בתוחלת הקנס על פני $T$ הסיבובים:

$$\text{punishment OP=our } = \sum_{t=1}^{T} \mathbb{E}_p \left[ (\text{punishment})_i^t \right] = \sum_{t=1}^{T} \sum_{i=1}^{N} p_i^t (\text{punishment})_i^t$$

4. נסמן את הקנס בסיבוב ה-$t$ של המומחה הטוב ביותר ב-$(\text{punishment})^{t}_{\text{OPT}}$. נרצה כי $\dot{\text{O}}\text{P}$ יהיה קרוב ככל הניתן ל-$\sum_{t=1}^{T}(\text{punishment})^{t}_{\text{OPT}}$.
5. נגדיר את $w_i^{t+1}=(1-\varepsilon(\text{ punishment })^{t}_{i})\,w_i^{t}$, כאשר $\frac{1}{2}\geq\varepsilon>0$.

הראינו בכיתה כי:

$$\text{OP} \leq \sum_{t=1}^{T}(\text{punishment})^{t}_{\text{OPT}}+\varepsilon\sum_{t=1}^{T}\mid(\text{punishment})^{t}_{\text{OPT}}\mid+\frac{1}{\varepsilon}\log N$$

אם נחזור לבעייה הקודמת, בה:

$$(\text{punishment})^{t}_{i}=\begin{cases}1 & f_i^t\neq y_t\\ 0 & \text{otherwise}\end{cases}$$

נוכל לחסום את מספר השגיאות שהאלגוריתם עושה על ידי:

$$\text{OP}\leq(1+\varepsilon)\,mistakes_{\text{OPT}}+\frac{1}{\varepsilon}\log N$$

כלומר, אם קיים מומחה מושלם ו-$\varepsilon=\frac{1}{2}$ נקבל כי:

$$\text{OP}\leq 2\log N$$

כלומר, קיבלנו אלגוריתם טוב יותר מהרוב הממושקל. אם כך, הוספת הבחירה ההסתברותית גרמה לנו לפעול באופן טוב יותר בתוחלת.

**תרגול מס' 11:**

# 8 אלגוריתמים הסתברותיים

**יום שלישי**

08.06.21

**הגדרה**
אלגוריתם הסתברותי הוא אלגוריתם אשר במהלך ריצתו משתמש ב"הטלות מטבע". כלומר, מבצע הגרלות בזמן הריצה.

### 8.0.1 בעיית צעצוע: חיפוש מספרים במערך

קלט - מערך $A=[a_1,\ldots,a_n]$ אשר חצי מהערכים בו הם $1$ וחצי $0$.

פלט - אינדקס $i \in [n]$ עם $a_i = 1$.

**פתרון נאיבי**

נרוץ על המערך ונחזיר את האינדקס הראשון בו $a_i = 1$.

**זמן ריצה**

במקרה הכי גרוע, האיברים יהיו מסודרים חצי בצד אחד וחצי בצד השני, ולכן נקבל סך הכל $O\left(n\right)$.

נראה כיצד אפשר לשפר זאת באמצעות אלגוריתמים הסתברותיים.

## 8.1 סוגי אלגוריתמים הסתברותיים

### 8.1.1 סוג ראשון - לאס וגאס

אלגוריתמים שבהם הטלות המטבע אינן משפיעות על נכונות הפלט. לדוגמא, מיון מהיר (Quick-Sort).

**פתרון לאס וגאס לבעיית הצעצוע**

נגריל אינדקס $i$ בלי החזרה, וברגע שיוצא $a_i$ עם $1$, נחזיר אותו.
נחזור על התהליך עד שנצליח (לא נדגום אותו אינדקס פעמיים).

**זמן ריצה**

במקרה הגרוע ביותר, עדיין נקבל $O\left(n\right)$.
עם זאת, נשים לב כי שמאחר וההגרלות אקראיות לחלוטין, לא משנה מהו הקלט, בכל הגרלה, הסיכוי להצליח גדול או שווה מ-$\frac{1}{2}$. לכן בתוחלת על כל קלט נצטרך שתי הגרלות.
זמן ריצה כזה נקרא שיעורי (amortized). לכן נקבל זמן ריצה שיעורי של $O\left(1\right)$ (זה סוג חדש של זמן ריצה).

### 8.1.2 סוג שני - מונטה קרלו

אלגוריתמים שבהם הטלת המטבע משפיעה על נכונות הפלט. בפרט, ריצות שונות על אותו קלט יכולות להסתיים בתוצאות שונות, והאלגוריתם יכול להיכשל.

**פתרון מונטה קרלו לבעיית הצעצוע**

הגרל אינדקס $i$. אם $a_i = 1$ החזר אותו, אחרת החזר $fail$.

**זמן ריצה**

מבצעים הגרלה אחת, לכן זמן הריצה הוא $O\left(1\right)$. שיפור משמעותי מהאלגוריתם הנאיבי, אבל ישנו מחיר - האלגוריתם עשוי להיכשל.

כדי להתמודד עם הסיכוי לכשלון, נדרוש את הדבר הבא:
לכל קלט, האלגוריתמים (מ"ק) שנציג, יציגו פתרון חוקי ואופטימלי, בהסתברות "גדולה כרצוננו".
דרך מקובלת להגדיר "גדולה כרצוננו" היא שהאלגוריתם מחזיר תשובה נכונה, בהסתברות של לפחות $1 - \frac{1}{e^k}$ עבור $k \in \mathbb{N}$.

**הערה**

בקורס נתעסק בעיקר באלגוריתמי מ"ק.

### 8.1.3 ניפוח באלגוריתמים הסתברותיים

אחד החוזקות של אלגוריתמים הסתברותיים היא שניתן בקלות "לנפח" את ההסתברות לקבלת תשובה נכונה ע"י הרצה חוזרת. כדי להמחיש זאת, נסתכל שוב על דוגמת הצעצוע שלנו.

**פתרון הסתברותי עם הסתברות "גדולה כרצוננו" לבעיית הצעצוע**

בהינתן $k \in \mathbb{N}$ ומערך $A$, נריץ את האלגוריתם הקודם $\log_2(e) \cdot k$ פעמים.
אם קיבלנו באחת ההרצות אינדקס $i$ עם $a_i = 1$, נחזיר אותו. אחרת, נחזיר $fail$.

**טענה**
לכל פלט האלגוריתם טועה בהסתברות של לכל היותר $\frac{1}{e^k}$.
**הוכחה**
מכיוון שבמערך יש $\frac{n}{2}$ אפסים, ההסתברות כי האלגוריתם הבסיסי נכשל היא $\frac{1}{2}$.
בעקבות העובדה כי כל ההגרלות בלתי תלויות, ההסתברות שניכשל ב-$\log_2(e) \cdot k$ היא:

$$\mathbb{P}\left(\text{האלגוריתם הכללי נכשל}\right) = \mathbb{P}\left(\log_2(e) \cdot k\ \text{האלגוריתם הבסיסי נכשל}\right) =$$

$$\mathbb{P}\left(\text{האלגוריתם הבסיסי נכשל}\right)^{\log_2(e) \cdot k} = \left(\left(\frac{1}{2}\right)^{\log_2(e)}\right)^k = \frac{1}{e^k}$$

מריצים $O(k)$ פעמים אלגוריתם שרץ ב-$O(1)$, זמן הריצה הוא $O(k)$.

## 8.2 אלגוריתמי קירוב הסתברותיים

נזכיר כי אלגוריתם קירוב הוא אלגוריתם שמחזיר פתרון חוקי עם קירוב מסוים לבעייה. אלגוריתם הסתברותי הוא כפי שראינו בקיצור.

**הגדרה**
אלגוריתם קירוב הסתברותי הוא אלגוריתם שלכל קלט, בזמן פולינומיאלי, מחזיר פלט חוקי ו-$c$-מקרב לבעיה בהסתברות גדולה כרצוננו, ובהסתברות נמוכה, נכשל.

נזכיר את הבעיה הבאה:

### 8.2.1 בעיית ה-3-SAT

באופן כללי, לא מדובר בבעיית אופטימיזציה אלא בבעיית הכרעה (כן או לא). האם קיימת לנוסחת 3-CNF השמה מספקת.

הגדרת הבעייה

- משתנים בוליאנים $x_i \in \{\mathbb{T}, \mathbb{F}\}$
- ליטרלים - משתנה $x_i$ או שלילתו $\neg x_i$.

- ☐ פסוקית בנוסחת 3-CNF : אוסף של שלושה ליטרלים ודיסיונקציה (או אימום) ביניהם. למשל, $(x_1 \vee \neg x_n \vee x_4)$.
- ☐ קוניונקציה (או גימום) של פסוקיות. למשל: $(x_1 \vee \neg x_n \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_3)$.

קלט

נוסחת $CNF$-3 בעלות פסוקיות $C_1, \dots, C_n$ מעל $n$ המשתנים $x_1, \dots, x_n$. כמו כן, נניח כי $n \leq 3m$.

### אלגוריתם $\frac{7}{8}$-מקרב לבעיית Max-3-SAT

תחילה נציג אלגוריתם בסיסי:

לכל משתנה $x_i$ נטיל מטבע הוגן:

- אם יצא 1, נגדיר $x_i = \mathbb{T}$.
- אם יצא 0 נגדיר $x_i = \mathbb{F}$ .

אם ההשמה מספקת $\frac{7}{8}m$ פסוקיות, נחזיר אותה, אחרת נחזיר $fail$.

כעת אלגוריתם כללי:

נחזור על האלגוריתם הבסיסי $k \cdot (m+1)$ פעמים באופן ב"ת.

אם באחת ההרצות קיבלנו השמה שמספקת $\frac{7}{8}m$ נחזיר אותה, אחרת נחזיר $fail$.

### זמן ריצה

האלגוריתם הבסיסי עולה $O(n+m) = O(m)$.

חוזרים על האלגוריתם $k \cdot (m+1)$ פעמים ולכן זמן הריצה הכולל הוא $O\left(k \cdot m^2\right)$.

### הוכחת נכונות

צריך להוכיח שהאלגוריתם מצליח בהסתברות מספיק גבוהה ושאם הוא מצליח, אז הוא $\frac{8}{7}$-מקרב.

אם האלגוריתם הצליח, אזי הוא וודאי $\frac{8}{7}$- מקרב, כי $\frac{7}{8}m \geq \frac{7}{8}\text{opt}$.

נותר להוכיח כי ההסתברות להצלחה גבוהה מספיק.

לשם כך, נוכיח את הטענה הבאה:

$$\mathbb{P}\left(\text{אלגו בסיסי מצליח}\right) \geq \frac{1}{m+1}$$

נראה שתוחלת מספר הפסוקיות המסתפקות היא $\frac{7}{8}m$.

נגדיר $X_i$ מ"מ אינדיקטור על האם הפסוקית $C_i$ סופקה על ידי ההשמה לכל $1 \leq i \leq m$.

נקבל:

$$\begin{aligned}&\mathbb{E}\left[X_i\right] = 0 \cdot \mathbb{P}\left(X_i = 0\right) + 1 \cdot \mathbb{P}\left(X_i = 1\right) = \\ &\mathbb{P}\left(X_i = 1\right) = 1 - \mathbb{P}\left(X_i = 0\right)\end{aligned}$$

המאורע $\{X_i = 0\}$ שקול לכך שנקבל $\mathbb{F}$ בכל אחת מהפעמים. ולכן סך הכל:

$$1 - \mathbb{P}\left(X_i = 0\right) = 1 - \left(\frac{1}{2}\right)^3 = \frac{7}{8}$$

נגדיר $X = \sum\limits_{i=1}^{m} X_i$. נשים לב ש-$X$ סופר את מספר הפסוקיות המקבלות ערך אמת.
אם כך, נקבל:

$$\begin{aligned}
&\mathbb{E}\left[X\right] = \mathbb{E}\left[\sum_{i=1}^{m} X_i\right] \overbrace{\overset{\downarrow}{=}}^{\text{ליניאריות התוחלת}} \\
&\sum_{i=1}^{m} \mathbb{E}\left[X_i\right] = \sum_{i=1}^{m} \frac{7}{8} = \frac{7}{8}m
\end{aligned}$$

על מנת שנעזר במרקוב ידידנו, נגדיר מ"מ $Y$ שסופר את כמות הפסוקיות שלא הסתפקו. נשים לב כי $Y = m - X$.
נקבל:

$$\mathbb{E}\left[Y\right] = \mathbb{E}\left[m - X\right] = \frac{1}{8}m$$

**טענה**

עבור $Y, m \in \mathbb{N}$ ו-$c > 0$ נקבל:

$$Y > \frac{m}{c} \Leftrightarrow Y \geq \frac{m}{c} + \frac{1}{c}$$

**הוכחה**

$$Y > \frac{1}{c}m \Leftrightarrow c \cdot Y > m \overset{(*)}{\Leftrightarrow} c \cdot Y \geq m + 1 \Leftrightarrow Y \geq \frac{1}{c} \cdot m + \frac{1}{c}$$

הביטוי $(*)$ מתקיים כיוון ש-$cY \in \mathbb{N}$ ולכן לא ייתכן כי $m < cY < m + 1$.
ואז נקבל כי:

$$\begin{aligned}
&\mathbb{P}\left(\text{אלגו בסיסי נכשל}\right) = \mathbb{P}\left(Y \geq \frac{1}{8}m\right) = \mathbb{P}\left(Y \geq \frac{1}{8}m + \frac{1}{8}\right) = \\
&\mathbb{P}\left(Y \geq \frac{1}{8}m\left(1 + \frac{1}{m}\right)\right) \leq \frac{\mathbb{E}\left[Y\right]}{\frac{1}{8}m\left(1 + \frac{1}{m}\right)} = \\
&\frac{1}{\frac{m+1}{m}} = \frac{m}{m+1}
\end{aligned}$$

אם כך, כאשר האלגוריתם מצליח, מתקיים:

$$\mathbb{P}\left(\text{אלגו בסיסי מצליח}\right) \geq 1 - \frac{m}{m+1} = \frac{m+1}{m+1} - \frac{m}{m+1} = \frac{1}{m+1}$$

**טענה**

האלגוריתם הכללי מצליח בהסתברות של $1 - \frac{1}{e^k}$ לפחות.

**הוכחה**

$$\begin{aligned}
&\mathbb{P}\left(\text{האלגוריתם הכללי נכשל}\right) = \mathbb{P}\left(\text{נכשל הבסיסיאלגוריתם}k\left(m+1\right)\right) = \\
&\mathbb{P}\left(\text{האלגוריתם הבסיסי נכשל}\right)^{k(m+1)} \leq \\
&\left(\frac{m}{m+1}\right)^{(m+1)k} = \left(1 - \left(\frac{1}{m+1}\right)^{m+1}\right)^{k} < \\
&\left(\frac{1}{e}\right)^{k} = \frac{1}{e^k}
\end{aligned}$$

אי השוויון האחרון לפני האחרון נכון כוון שישנה שאיפה ל-$\frac{1}{e}$ במצב זה.

## 8.3 בעיית פתרון מערכת משוואות מודולו $2$ (Max Lin 2)

**תרגול מס' $12$:**

**יום שלישי**

15.06.21

<u>קלט :</u>

מטריצה $A$ מסדר $m \times n$ מעל השדה $\mathbb{F}_2$. וקטור $b$ באורך $n$ מעל השדה $\mathbb{F}_2$ המייצגים מערכת משוואות ליניארית $Ax = b$ עם $x = \begin{pmatrix} x_1 \\ \vdots \\ x_n \end{pmatrix}$.

<u>פלט:</u>

השמה למשתנים $x_1, \ldots, x_n$ המספקת מספר מקסימלי של משוואות.

**דוגמה**

$m = 4, n = 2$, והמטריצה נתונה על ידי:

$$A = \begin{bmatrix} 1 & 0 \\ 1 & 1 \\ 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix} b$$

נבחין כי למערכת אין פתרון כיוון שאנחנו דורשים כי:

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

אבל מתקיים כי:

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

נבחין כי ההשמה $x_1 = x_2 = 0$ מספקת 3 משוואות וזהו המספר המקסימלי.
כמו כן נשים לב כי כל השמה מספקת את המשוואה $\left\langle \begin{pmatrix}0\\0\end{pmatrix}, \begin{pmatrix}x_1\\x_2\end{pmatrix}\right\rangle = 0$ ולכן נניח כי לא קיימות משוואת כנ"ל בקלט.
בעייה זו היא NP קשה ונציג אלגוריתם 2- קירוב הסתברותי.

### 8.3.1 אלגוריתם 2-מקרב לבעייה

**אלגוריתם בסיסי**

נגריל השמה מקרית $S \in \mathbb{F}_2^n$ (לכל משתנה $x_i$ נגריל $x_i = 1$ בהסתברות $\frac{1}{2}$ ו-$x_i = 0$ בהסתברות $\frac{1}{2}$).
אם $S$ מספקת $\frac{1}{2}$ מהמשוואות - החזר אותה.
אחרת, החזר $fail$.

נראה שהאלגוריתם נכשל בהסתברות קטנה או שווה מ-$\frac{m}{m+1}$.
נחשב את תוחלת המשוואות שמסופקות על ידי השמה מקרית.

1. נגדיר סימונים לטובת ההמשך:
   (א) נסמן ב-$r_i$ את השורה ה-$i$ של המטריצה $A$.
   (ב) נסמן ב-$b_i$ את הקואורדינטה ה-$i$ של הוקטור $b$.
   (ג) נאמר כי משוואה $i$ מסופקת על ידי השמה $S$, אם"ם $\langle r_i, S\rangle = b_i$.
   (ד) נסמן ב-$X$ את המשתנה המקרי שסופר את מספר המשוואות שמסופקות על ידי $S$.
   (ה) נגדיר $X_i$ משתנה מקרי אינדיקטור על המאורע "משוואה $i$ סופקה על ידי $S$".
   (ו) נחשב את תוחלת $X_i$. נניח בה"כ כי $b_i = 1$ עבור השורה ה-$i$. נסמן ב-$d$ הקואורדינטה הגדולה ביותר ששונה מ-0. נקבל כי $\mathbb{E}\left[X_i\right] = \mathbb{P}\left[X_i = 1\right] = \mathbb{P}\left[\langle r_i, S\rangle = b_i\right]$.
   מנוסחת ההסתברות השלימה עולה כי

$$\mathbb{P}\left[\langle r_i, S\rangle = b_i\right] = \mathbb{P}\left[\sum_{j=1}^{d-1} r_{ij}s_j = 0 \cap r_{id}S_d = 1\right] + \mathbb{P}\left[\sum_{j=1}^{d-1} r_{ij}s_j = 1 \cap r_{id}S_d = 0\right]$$
$$= \mathbb{P}\left(\sum_{j=1}^{d-1} r_{i,j}s_j = 1\right) \cdot \mathbb{P}\left(s_d = 0\right) + \mathbb{P}\left(\sum_{j=1}^{d-1} r_{i,j}s_j = 0\right) \cdot \mathbb{P}\left(s_d = 1\right)$$
$$= \mathbb{P}\left(\sum_{j=1}^{d-1} r_{i,j}s_j = 1\right) \cdot \frac{1}{2} + \mathbb{P}\left(\sum_{j=1}^{d-1} r_{i,j}s_j = 0\right) \cdot \frac{1}{2}$$
$$= \frac{1}{2} \cdot \underbrace{(\mathbb{P}\left(\sum_{j=1}^{d-1} r_{i,j}s_j = 1\right) + \mathbb{P}\left(\sum_{j=1}^{d-1} r_{i,j}s_j = 0\right))}_{=1} = \frac{1}{2}$$

(ז) נחשב את תוחלת $X$ שהיא $\mathbb{E}\left[X\right] = \mathbb{E}\left[\sum_{i=1}^{d} X_i\right] = \frac{1}{2}m$.

2. נגדיר $Y = m - X$ ונקבל כי $\mathbb{E}\left[Y\right] = \frac{1}{2}m$ ומהלמה שראינו בשבוע שעבר בצירוף של א"ש מרקוב נקבל כי:

$$P(\text{algorithm fails}) = P\left(Y > \frac{1}{2}m\right) = P\left(Y \geq \frac{1}{2}m + \frac{1}{2}\right)$$
$$= P(Y \geq \frac{1}{2}m \cdot \left(1 + \frac{1}{m}\right)) \leq \frac{1}{(1 + \frac{1}{m})} = \frac{m}{m+1}$$

**אלגוריתם כללי**

נריץ את האלגוריתם הבסיסי $D$ פעמים. אם באחת הפעמים קיבלנו השמה $S$ שמספקת $\frac{1}{2}$ מהמשוואות, נחזיר אותה. אחרת, נחזיר $false$.

נחשב את סיכויי הכישלון של האלגוריתם:

$$P \text{ (general alg' fails) } = P \text{ (basic alg' fails } D \text{ times)}$$
$$= P(\text{ basic alg' fails })^D$$
$$\leq \left(\frac{m}{m+1}\right)^D$$
$$= \left(\left(1 - \frac{1}{m+1}\right)^{m+1}\right)^{\frac{D}{m+1}} < \frac{1}{e^{\frac{D}{m+1}}}$$

כעת נקבל כי $\frac{D}{m+1} = k \Leftrightarrow D = k\left(m+1\right)$, כנדרש.

זמן ריצה

האלגוריתם הבסיסי רץ ב-$O\left(mn\right)$. אנו מריצים אותו $O\left(km\right)$ פעמים ולכן זמן הריצה הכללי הוא $O\left(km^2n\right)$.

## 8.4 צביעה מקסימלית של גרף ב-3 צבעים

**בעיית הצביעה**

קלט: גרף לא מכוון $G = \langle V, E\rangle$

פלט: מספר הצבעים המינימלי בו ניתן לצבוע את קודקודי $G$ כך שאין זוג קודקודים שכנים שצבועים באותו הצבע.

**בעייה אחרת**

קלט: גרף לא מכוון $G = \langle V, E\rangle$

פלט: האם ניתן לצבוע את קודקודי $G$ ב-$3$ צבעים כך שאין זוג קודקודים שכנים שצבועים באותו הצבע.

זו גם בעיית NP קשה, ולכן ננסה לקרב אותה לבעייה נוספת. אך קודם לכן, נהפוך אותה לבעיית אופטימיזציה.

**בעייה שלישית**

קלט: גרף לא מכוון $G = \langle V, E\rangle$

פלט: צביעה של הקודקודים ב-$3$ צבעים כך שמספר הצלעות שקודקודיהן אינם באותו צבע הוא מקסימלי.

פורמלית, נרצה למצוא פונקציית צביעה $c : E \to \{1, 2, 3\}$

כך שנמקסם את גודל הקבוצה $A = \{(i, j) \in E \mid c(i) \neq c(j)\}$.

נשים לב כי אם $|A| = |E|$ אז יש פתרון לבעייה האחרת - יש צביעה של הגרף בשלושה צבעים, ונוכל להשתמש בבעייה הנ"ל כדי לפתור את הבעיה המקורית. זו בעייה קשה יותר מהקודמת, ולכן זו גם בעיית NP קשה.

### 8.4.1 אלגוריתם הסתברותי $\frac{3}{2}$-מקרב לבעיית צביעה מקסימלית של גרף ב-$3$ צבעים

**אלגוריתם בסיסי**

1. נעבור על קודקודי הגרף:
   (א) לכל קודקוד נגריל צבע בהסתברות אחידה.
2. נחשב את $A$ על ידי מעבר על כל הצלעות בגרף.
3. אם $|A| \geq \frac{2}{3}|E|$ נחזיר את הצביעה $c$.
4. אחרת, נחזיר $fail$.

**טענה**

האלגוריתם הבסיסי מצליח בהסתברות של לפחות $\frac{1}{|E|+1}$

**הוכחה**

נסמן ב-$X$ את כמות הצלעות ב-$A$. נבחין כי מתקיים:

$$X = \sum_{(i,j)\in E} \mathbb{1}_{[c(i)\neq c(j)]}$$

נשים לב כי $\mathbb{1}_{[c(i)\neq c(j)]}$ היא פונקציית אינדיקטור המקבלת $1$ אם"ם $(i, j) \in A$.

ההסתברות שדבר זה יתקיים הינה, אם נניח בה"כ כי $c(i) = 1$:

$$\begin{aligned}\mathbb{P}[(i, j) \in A] = \mathbb{P}(c(j) = 2 \cup c(j) = 3) &= \\ \mathbb{P}(c(j) = 2) + \mathbb{P}(c(j) = 3) &= \\ \frac{1}{3} + \frac{1}{3} = \frac{2}{3}\end{aligned}$$

ולכן נקבל כי $\mathbb{E}\left[X\right]=\frac{2}{3}\left|E\right|$. אם נגדיר $Y=\left|E\right|-X$ נקבל כי $\mathbb{E}\left[Y\right]=\frac{1}{3}\left|E\right|$.
בהתבסס על הלמה שראינו בשבוע שעבר, לפיה אם $a,b,c\in\mathbb{N}$ מתקיים כי $a>\frac{b}{c}\Leftrightarrow a\geq\frac{b}{c}+\frac{1}{c}$ ובהתבסס על העובדה כי האלגוריתם הבסיסי נכשל אס"ם $Y>\frac{1}{3}\left|E\right|$ נקבל:

$$\begin{aligned}P[\text{ basic algorithm fails }] & =P\left[Y>\frac{1}{3}|E|\right]\\
 & =P\left[Y\geq\frac{1}{3}|E|+\frac{1}{3}\right]\\
 & =P\left[Y\geq\frac{1}{3}|E|\left(1+\frac{1}{|E|}\right)\right]\\
 & \leq\frac{\mathbb{E}[Y]}{\frac{1}{3}|E|\left(1+\frac{1}{|E|}\right)}\\
 & =\frac{1}{1+\frac{1}{|E|}}=\frac{|E|}{|E|+1}
\end{aligned}$$

ולכן מתקיים:

$$\begin{aligned} & P[\text{basic algorithm succeeds}]=1-P[\text{basic algorithm fails}]\\
 & =1-\frac{|E|}{|E|+1}\\
 & =\frac{1}{|E|+1}
\end{aligned}$$

**אלגוריתם כללי**

1. נריץ את האלגוריתם $k\cdot\left(\left|E\right|+1\right)$ פעמים.
2. אם באחת הריצות קיבלנו צביעה עבורה $\left|A\right|\geq\frac{2}{3}\left|E\right|$ נחזיר אותה.
3. אחרת, נחזיר $fail$.

**טענה**

האלגוריתם מצליח בהסתברות של לפחות $1-\frac{1}{e^{k}}$.

**הוכחה**

$$\begin{aligned}p(\text{algorithm succeed}) & =1-p(\text{algorithm failed})\\
 & =1-p(\text{base algorithm failed})^{k(|E|+1)}\\
 & =1-(1-p(\text{base succeed}))^{k(|E|+1)}\\
 & \geq1-\left(1-\frac{1}{|E|+1}\right)^{k(|E|+1)}\\
 & =1-\left(\left(1-\frac{1}{|E|+1}\right)^{|E|+1}\right)^{k}\\
 & \geq1-\left(\frac{1}{e}\right)^{k}\\
 & =1-\frac{1}{e^{k}}
\end{aligned}$$

כנדרש.

זמן ריצה

כל צביעה לוקחת $O\left(V\right)$. חישוב $A$ לוקח $O\left(E\right)$. אם כך, האלגוריתם הבסיסי אורך $O\left(|E|+|V|\right)$.
אנו חוזרים עליו $O\left(k\cdot|E|\right)$ פעמים ולכן זמן הריצה הינו $O\left(k\cdot|E|^{2}+k\cdot|E|\cdot|V|\right)$.

**תרגול מס' $13$:**

**יום שלישי**

22.06.21

## 8.5 פולינומים מרובי משתנים

פולינום מדרגה $d$ מעל שדה $\mathbb{F}$ במשתנה $x$ הוא ביטוי מהצורה $P\left(x\right)=\sum\limits_{i=0}^{d}a_{i}x^{i}$.
כאשר $d\in\mathbb{N}$ ו-$a\in\mathbb{F}$. ביטוי מהצורה $a_{i}x^{i}$ נקרא מונום.
אפשר להכליל את הגדרת הפולינום לפונקצייה מעל $n$ משתנים.:

$$P\left(x_{1},\ldots,x_{n}\right)=\sum_{i_{1},\ldots,i_{n}\in\left(d_{1},\ldots d_{n}\right)}a_{i_{1},\ldots,i_{n}}\cdot x_{1}^{i_{1}}\cdot\ldots\cdot x_{n}^{i_{n}}$$

לדוגמא:

$$P\left(x_{1},x_{2}\right)=3x_{1}+5x_{1}^{2}x_{2}^{2}-x_{1}x_{2}^{2}$$

נקבל במקרה זה כי $a_{1,0}=3,a_{2,2}=5,a_{1,2}=-1$.

**הגדרה**

דרגה של פולינום מרובה משתנים היא סכום החזקות במונום המקסימלי.
כלומר:

$$\deg\left(P\left(x_{1},\ldots x_{n}\right)\right)=\max_{i_{1},\ldots i_{n}\in\left(d_{1},\ldots d_{n}\right)}\left\{ \sum_{j=1}^{n}i_{j}\mid a_{i_{1}\ldots i_{n}}\neq0\right\}$$

בדוגמה לעיל, דרגת הפולינום היא $4$.

**הגדרה**

שורש של פולינום $P$ הוא איבר $x\in\mathbb{F}^{n}$ שעבורו מתקיים כי $P\left(x\right)=0$.
נאמר כי פולינום הוא פולינום ה-0 אס"ם לכל $x\in\mathbb{F}^{n}$ מתקיים כי $P\left(x\right)=0$.

**טענה**

יהי שדה $\mathbb{F}$ בעל $|\mathbb{F}|$ איברים. יהי פולינום $P$ המקיים כי $\deg(P) \leq |\mathbb{F}|$ . הפולינום $P$ הוא פולינום האפס אם"ם כל המקדמים בו הם 0.בהצגה הסטנדרטית קל לראות אם פולינום הוא זהותית 0, אם כל המקדמים הינם אפסים. אבל בהצגה לא סטנדרטית זה קשה, למשל הפולינום הבא, בשני משתנים, הוא זהותית 0:

$$p(x_1, x_2) = (x_1 + 3)(x_2 - 5) - (x_1 + 2)(x_2 - 4) + x_1 - x_2 + 7$$

לפתוח את כל הסוגריים ולחשב זאת עלול להיות ארוך מאוד חישובית.

**טביעת אצבע**

נסתכל על הבעייה מנקודת מבט הסתברותית. לפולינומים שאינם זהותית אפס, יש כמות קטנה של ערכים עבורם הם מחזירים 0 ביחס לכמות הערכים שאינם מחזירים 0 - לערכים אלו קוראים שורשים.
למעשה, ככל שהשדה $\mathbb{F}$ מכיל יותר איברים, כמות השורשים של הפולינום הופכת להיות זניחה ביחס לגודלו של $\mathbb{F}$. על כן, אם פשוט נגריל ערך מתוך השדה, **הסיכוי** שנתקל בשורש של הפולינום **נמוך**. אבחנה זאת יכולה לשמש לנו כבסיס לאלגוריתם הסתברותי - פשוט נגריל ערך מהשדה, ונבדוק את הערך של הפולינום עבורו (הרבה יותר מהיר מלפשט את הפולינום) אם נקבל 0, נגיד כי הפולינום הוא זהותית 0 ,ואחרת, נגיד כי לא. זהו כמובן אלוגריתם שאינו פועל תמיד - לעיתים יש לנו פולינומים שאינם זהותית אפס, אבל הגרלנו עבורם את אחד מהשורשים שלהם, ועל כן האלגוריתם שלנו יחזיר בטעות שהם כן זהותית 0. עם זאת, בהרצאה הראינו כי בעזרת מספר קטן של חזרות ניתן להבטיח **בהסתברות גדולה כרצוננו** כי אלגוריתמים מסוג זה מחזירים תשובה נכונה ־ וזה יאפשר לנו לשפר באופן דרסטי את זמן הריצה של האלגוריתם הדטרמיניסטי.

**אלגוריתם בסיס:**

- נבחר קבוצה סופית $A \subseteq \mathbb{F}$.
- נגריל בהתפלגות אחידה וקטור $x \in \mathbb{F}^n$ (לכל קואורדינטה של $x$ מגרילים איבר מ-$A$ בהתפלגות אחידה).
- נציב $x$ ב-$P$. אם $P(x) = 0$ נחזיר שכן ואחרת נחזיר שלא.

**זמן ריצה**

נניח שדגימה בהסתברות אחידה לוקחת $O(\log|A|)$, בהנחה שבפולינום $O(g)$ ביטויים ושבכל ביטוי יש $O(m)$ סוגריים ובכל סוגריים $(n)$ איברים, זמן הריצה הוא $O(g \cdot m \cdot n + n \cdot \log|A|)$.

**משפט שוורץ- זיפל**

האלגוריתם הבסיסי טועה בהסתברות של:

$$\mathbb{P}(\text{fail}) \leq \frac{\deg(P)}{|A|}$$

אם נבחר $|A| > \deg(P)$ נקבל כי האלגוריתם צודק בהסתברות גדולה מ-0 נוכל כעת להשתמש בניפוח כדי להשיג הסתברות גדולה כרצוננו.

### ניפוח

נשים לב שאפשר לבחור קבוצה $A$ עם $|A| > \deg(P) \cdot e^k$ ונקבל:

$$\mathbb{P}(\text{fail}) \leq \frac{1}{e^k}$$

במקרה זה, זמן הריצה יהיה:

$$\begin{aligned}
&O\left(g \cdot m \cdot n + n \cdot \log\left|\deg(P) \cdot e^k\right|\right) = \\
&O\left(g \cdot m \cdot n + n \log\left(\deg(P) + nk\right)\right)
\end{aligned}$$

### ניפוח בעזרת חזרות - אלגוריתם כללי

1. נחזור על האלגוריתם הבסיסים $\frac{k}{\ln\left(\frac{|A|}{\deg(P)}\right)}$ פעמים.
2. אם באחת הפעמים קיבלנו כי הפולינום אינו $0$, נחזיר שלא ($P$ אינו זהותית $0$). אחרת, נחזיר כי $P$ זהותית $0$.

**טענה**

האלגוריתם נכשל בהסתברות קטנה או שווה ל-$\frac{1}{e^k}$:
נשים לב כי מתקיים ש:

$$\begin{aligned}
\frac{k}{\ln\left(\frac{|A|}{\deg(p)}\right)} &= k \cdot -\frac{1}{\ln\left(\frac{\deg(p)}{|A|}\right)} = \\
\frac{\ln\left(\frac{1}{e^k}\right)}{\ln\left(\frac{\deg(p)}{|A|}\right)} &= \log\frac{\deg(p)}{|A|}\left(\frac{1}{e^k}\right)
\end{aligned}$$

ולכן נקבל:

$$\begin{aligned}
\mathbb{P}(\text{general algorithm fails}) &= \mathbb{P}(\text{basic algorithm fails})^{\frac{k}{\ln\left(\frac{kA|}{\deg(p)}\right)}} \\
&\leq \left(\frac{\deg(p)}{|A|}\right)^{\log_{\frac{deg(p)}{|A|}}\left(\frac{1}{e^k}\right)} \\
&= \frac{1}{e^k}
\end{aligned}$$

## 8.5.1 שימושים

### כפל מטריצות

קלט:

מטריצות $A, B \in \mathbb{R}^{n \times n}$.

פלט

1 אם $AB = C$ ו-0 אם $AB \neq C$ .

**רעיון**

נחשב את $AB$ ונשווה איבר איבר ל-$C$.

**זמן ריצה**

כפל מטריצות עולה $O\left(n^3\right)$, השוואת איברים $O\left(n^2\right)$ ולכן סך הכל נקבל $O\left(n^3\right)$.

**פתרון הסתברותי**

- נגדיר וקטור $\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$.
- נחשב את $A \cdot (B \cdot x), C \cdot x$.
- נגדיר קבוצה סופית $S \in \mathbb{R}$ ונדגום ערכים מ-$S$ באופן אחיד לווקטור $x$.
- נציב ב-$A \cdot (B \cdot x)$ ו-$Cx$. אם קיבלנו שכל הערכים שווים, נחזיר 1 ואחרת נחזיר 0.

**טענה**

לכל $k \in \mathbb{N}$ ניתן לבחור $|S| \geq e^k$ כך שיתקיים כי:

$$\mathbb{P}\left(\text{אלגו נכשל}\right) \leq \frac{1}{e^k}$$

**הוכחה**

נשים לב כי $AB = C \Leftrightarrow ABx = Cx$ לכל $x \in \mathbb{R}^n$.

אך $ABx = Cx \Leftrightarrow (AB - C)\,x = 0$ , כלומר אם ורק אם $(AB - C)\,x$ הוא פולינום האפס.

נשים לב כי $(AB - C)\,x$ הוא פולינום מדרגה 1, שהוא זהותית 0 אם"ם $AB = C$.

נסמן $P(x_1, \ldots x_n) = (AB - C)\,x$ .

ממשפט שוורץ זיפל נקבל כי:

$$\mathbb{P}(\text{fails algorithm}) = \mathbb{P}\left(p\left(x_1, \ldots, x_n\right) = 0\right) \leq \frac{\deg(p)}{|S|} = \frac{1}{e^k}$$

**זמן ריצה**

חישוב $Bx$ אורך $O\left(n^2\right)$ , וכך גם חישוב $A \cdot (Bx)$ וחישוב $Cx$.

נבחין כי דגימת וקטור מ-$S$ לוקח $O(n \cdot k)$ .

בדיקה האם הוקטורים $ABx = Cx$ לוקחת $O(n)$. ולכן סך הכל נקבל כי $O\left(n^2 + nk\right)$.

**המטריצה של Edmonds**

**תזכורת**

תהי $A$ מטריצה $n \times n$. נסמן ב-$S_n$ את קבוצת הפרמוטציות על $\{1, \dots, n\}$.
הדטרמינטה של $A$ מוגדרת להיות:

$$|A| = \det(A) = \sum_{\sigma \in S_n} \text{sign}(\sigma) A_{1,\sigma(1)} \cdot A_{2,\sigma(2)} \dots \cdot A_{n,\sigma(n)}$$

נשים לב כי הדטרמיננטה היא פולינום מעל $n^2$ משתנים שהם $(a_{11} \dots, a_{nn})$.

**הערה**

ניתן לחשב את הדטרמיננטה בעזרת דירוג ב-$O\left(n^3\right)$. הפולינום לא יהיה בהצגה הסטנדרטית אלא ייצוג ככפל של $n$ פולינום.
פיתוח בהצגה הסטנדרטית ידרוש מספר אקספוננציאלי של מונומים.

**טענה**

יהי $G = \langle L, R, E \rangle$ גרף דו צדדי עבורו מתקיים כי $n = |R| \cdot |L|$ , המטריצה של אדמונדס מוגדרת באופן הבא:

$$M_{ij} = \begin{cases} x_{ij} & \{i,j\} \in E \\ 0 & \text{otherwise} \end{cases}$$

מקיימת כי $|M| = 0$ אם"ם אין ב-$G$ שידוד מושלם.

**אלגוריתם הסתברותי לשידוד מושלם**

- בהינתן $G = \langle R, L, E \rangle$ , נחשב את מטריצת אדמונדס של $G$.
- נבחר קבוצה $S \subseteq R$ ונגריל ממנה ערכים ל-$M$ בהתפלגות אחידה.
- נחשב את $|M|$. אם יצא $0$ נחזיר כי אין זיווג מושלם. אחרת נחזיר כי יש.

**טענה**

לכל $k \in \mathbb{N}$ ניתן לבחור כי $|S| = n \cdot e^k$ כדי שיתקיים:

$$\mathbb{P}\left(\text{fail}\right) \leq \frac{1}{e^k}$$

**הוכחה**

מהטענה הקודמת, $|M| = 0$ אם"ם אין ב-$G$ שידוד מושלם.
נסמן $|M| = P\left(x_{11}, \dots, x_{nn}\right)$.
ממשפט שוורץ זיפל עולה כי:

$$\mathbb{P}(\text{algorithm fails}) = \mathbb{P}\left(p\left(x_{11}, \dots, x_{nn}\right) = 0\right) \leq \frac{\deg(p)}{|S|} \leq \frac{n}{n \cdot e^k} = \frac{1}{e^k}$$

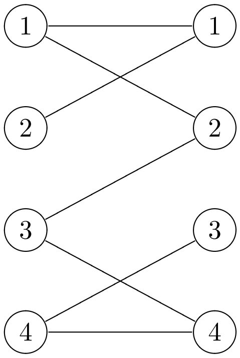
**זמן ריצה**

חישוב $M$ לוקח $O\left(n^2\right)$ פעולות. להציב במטריצה לוקח:

$$O\left(|V|^2\log\left(|V|\cdot e^k\right)\right)=O\left(|V|^2\log(|V|)+|V|^2k\right)$$

חישוב הדטרמינטה לוקח $O\left(|V|^3\right)$ פעולות. כיוון שלרוב $n >> k$, זמן הריצה הוא $O\left(|V|^3\right)$.

**דוגמת הרצה**



ואם נהפוך את הגרף למטריצה:

$$\begin{bmatrix} x_{11} & x_{12} & 0 & 0 \\ x_{21} & 0 & 0 & 0 \\ 0 & x_{32} & 0 & x_{34} \\ 0 & 0 & x_{43} & x_{44} \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 2 & 0 & 0 \\ 4 & 0 & 0 & 0 \\ 0 & 4 & 0 & 3 \\ 0 & 0 & 1 & 2 \end{bmatrix} \longrightarrow \det(M)=24$$