אלגוריתמים קומבינטוריים 104291 | תרגולים

תום מובס

1 ביולי 2022

1 תרגול

1.1 סימונים אסימפטוטים

**__הגדרה:__ עבור פונקציה $f:\mathbb{N}\to\mathbb{R}^+$ (תמיד יהיו מונוטוניות) נסמן ב־$O(f)$ את אוסף הפונקציות $g:\mathbb{N}\to\mathbb{R}^+$ כך שקיימים $c>0$ ו־$N_0\in\mathbb{N}$
כך שלכל $n>N_0$ מתקיים $0\le g(n)\le c\cdot f(n)$, נכנתוב עבור g כנ"ל $g(n)=O(f)$**

__דוגמאות:__

1. אם $g(n)=c>0$ אז $g(n)=O(1)$: לוקחים $N_0=1$, ו־$c=c$.
2. $g(n)=\sum_{k=0}^{l}\alpha_k n^k \Leftarrow g(n)=O(n^l)$

__משפט:__ אם $f,g:\mathbb{N}\to\mathbb{R}^+$ ומתקיים $Lim_{n\to\infty}\frac{f(n)}{g(n)}$ קיים וסופי $\Leftarrow f(n)=O(g(n))$
__הוכחה:__ תרגיל.

**__הגדרה:__ עבור פונקציה $f:\mathbb{N}\to\mathbb{R}^+$ נסמן ב־$o(f)$ את אוסף הפונקציות $g:\mathbb{N}\to\mathbb{R}^+$ כך שקיימים $c>0$ ו־$N_0\in\mathbb{N}$
כך שלכל $n>N_0$ מתקיים $0\le g(n)<c\cdot f(n)$, נכנתוב עבור g כנ"ל $g(n)=O(f)$
הגדרה אלטרנטיבית אם $Lim_{n\to\infty}\frac{f(n)}{g(n)}=0$.**

__דוגמאות__

1. $2n=o(n^4)$
2. $2n=O(n)$ אבל $2n\neq o(n)$

__הגדרה:__ עבור f כנ"ל נגדיר $\Omega(f)$ להיות אוסף g כך שקיים $c>0$, $N_0\in\mathbb{N}$ כך שלכל $n>N_0$ $0\le c\cdot f(n)\le g(n)$.
**__הגדרה:__ עבור f כנ"ל נגדיר $\omega(f)$ להיות אוסף g כך שקיים $c>0$, $N_0\in\mathbb{N}$ כך שלכל $n>N_0$ $0\le c\cdot f(n)<g(n)$.
הגדרה אלטרנטיבית אם $Lim_{n\to\infty}\frac{g(n)}{f(n)}=\infty$**
**__הגדרה:__ נאמר ש$g(n)=\Theta(f(n))$ אם $g(n)=O(f(n))$ וגם $g(n)=\Omega(f(n))$
או במפורש אם קיימים $c_1,c_2>0$ כך שלכל $n>N_0\in\mathbb{N}$ $\leftarrow$ $0\le c_2f(n)\le g(n)\le c_1f(n)$**

1.2 הבחנות

1. אם $g(n)=O(f)$ אז $f(n)=\Omega(g)$
2. $g(n)=o(f(n))$ אז $f(n)=\omega(g(n))$
3. וכן הלאה..

1.3 תרגיל

להוכיח ש $\sum_{k=1}^{n} t^k = \Theta(n^{k+1})$
__תשובה:__ $1^k + 2^k + \cdots + n^k \leq n^k + n^k + \cdots + n^k = n^k n = n^{k+1}$
$g(x) = (\lfloor x \rfloor + 1)^k$ **ונשים לב** $\sum_{k=1}^{n} t^k = \int_0^n g(X)dx \geq \int_0^n x^k dx = \frac{x^{k+1}}{k+1}|_0^n = \frac{1}{k+1}n^{k+1}$
סה"כ קיבלנו $\frac{1}{k+1}n^{k+1} \leq \sum_{k=1}^{n} t^k \leq n^{k+1}$

1.4 תרגיל

להוכיח $log(n!) = \Theta(nlogn)$
__תשובה:__ $log(n!) = \sum_{k=1}^{n} log(k) \leq nlogn$ **צד אחד**
צד שני - מתקיים $\sum_{k=1}^{n} log(k) \geq \int_1^n log(x)dx = xlogx + 1|_1^n = nlogn - n$ **וכמובן ש-** $nlogn - n = \Theta(nlogn)$ **סה"כ**
נקבל $\sum_{k=1}^{n} log(k) = \Theta(nlogn)$

1.5 תרגיל

נגדיר פונקציה ריקורסיבית T **ע"י** $T(1) = 1$ **וגם** $T(n) = 3T(\frac{n}{2}) + n^2$**, למצוא את ההתנהגות האסימפטוטית.**
__תשובה:__ ראשית נשים לב שלכל n קיים k כך ש- $4^k \leq n \leq 4^{k+1}$.
נשים לב שT **מונוטונית ולכן** $T(4^k) \leq T(n) \leq T(4^{k+1})$ **, נסמן** $m = 4^k$

$$T(m) = 3T(\frac{m}{4}) + m^2 = m^2 + 3(3T(\frac{m}{4^2}) + \frac{m^2}{4^2}) = m^2 + \frac{3}{4^2}m^2 + 3^2T(\frac{m}{4^2}) = \sum_{k=0}^{log_4 m} \frac{3^i}{???}m^2 + 3^{log_4 m} = \Theta$$

סה"כ נקבל: $T(n) \leq T(4^{k+1}) \leq c \cdot (4^{k+1})^2 \leq c \cdot (4n^2) = 16c \cdot n^2$

2 תרגול - מיונים

2.1 תרגיל

תנו אלגוריתם שמקבל רשימה ממוינת באורך n **ורשימת ממוינת באורך** k**, ויוצר רשימה ממוינת שמכילה את איברי שני הרשימות - בסיבוכיות זמן ריצה** $\Theta(klogn)$

__פתרון:__

- **__האלגוריתם:__ בהינתן** l_1, l_2**רשימות כך ש** $|l_1| = n$ **ו-**$|l_2| = k$ **←לכל** $a \in l_2$ **נבצע חיפוש בינארי ב**l_1 **כדי למצוא את המקום של** a **ונכניס אותו לשם**

- <u>אלגוריתם עוצר:</u> האלגוריתם מבצע k חיפושים בינארים, וכל חיפוש בינארי עוצר. ולכן כל האלגוריתם עוצר.

- <u>נכונות אלגוריתם:</u> נשים לב שלפני ביצוע של כל חיפוש בינארי הרשימה l_1 ממויינת, ואחרי הוספת האיבר הרשימה עדיין ממויינת מתכונות של חיפוש בינארי. מוסיפים כל איבר בl_2, לכן בסוף האלגוריתם l_1 ממויינת ומכילה את כל האיברים כנדרש.

- <u>סיבוכיות:</u> סיבוכיות החיפוש הבינארי ה־k שמבצעים היא $\Theta(log(n+k-1))$ ולכן סה"כ סיבוכיות:

$$klog(n) \leq \sum_{i=1}^{k} log(n+i-1) \leq_{i\leq k\leq n} k \cdot log(2n) = klog(n) + klog(n) = O(klogn)$$

ולכן סה"כ סיבוכיות זמן הריצה היא $\Theta(klog(n))$

<u>הנחה של הקורס:</u> נכון שחיפוש הוא $logn$ ברשימה ממוינת, אבל בקורס הזה ניתן "להכניס" איבר לרשימה בלי לדאוג ל"הזזה" של שאר האיברים (נתייחס להכנסה שלו ב$O(1)$)

2.2 תרגיל

תנו אלגוריתם שמקבל k רשימות ממוינות וממזג אותם לרשימה ממויינת אחת (סה"כ יש n איברים).
סיבוכיות זמן ריצה של $\Theta(nlog(k))$

<u>פתרון:</u>
בצורה נאיבית היינו עושים השוואה בכל פעם לאיבר הראשונה ולוקחים את המינימום, אז זה היה עולה לנו יותר מידי.

- <u>האלגוריתם:</u> בהינתן $l_1, l_2, \ldots, l_k$, ונניח ש$k = 2^m$ עבור $m \in \mathbb{N}$ כלשהו
נמזג בעזרת $merge$ את l_1 עם l_2 וl_3 עם l_4 $\ldots$ באותה צורה נמשיך רקורסיבית עם הרשימות החדשות, עד שנקבל רשימה אחת מאוחדת.

- <u>אלגוריתם עוצר:</u> בשלב ה־i מבצעים לכל היותר $\frac{k}{2^i}$ פעמים $merge$. ובסה"כ יש $logk$ שלבים (סופי) ולכן האלגוריתם עוצר.

- <u>נכונות אלגוריתם:</u> באינדוקציה על m.
  - **בסיס:** $m = 1$, עובד כי $merge$ עובד.
  - **צעד:** בהינתן $k = 2^m$ רשימות ממוינות, אחרי הצעד של האלגוריתם נקבל $\frac{k}{2} = 2^{m-1}$ רשימות ממויינות, ולפי הנחת האינדוקציה הן ימוזגו.

- <u>סיבוכיות:</u> בהינתן $k = 2^b$ רשימות $l_1, \ldots, l_k$ האלגוריתם מבצע $\frac{k}{2}$ פעולות $merge$.. הסיבוכיות של המיזוגים האלה:

$$\Theta(|l_1| + |l_2|) + \Theta(|l_3| + |l_4|) + \cdots + \Theta(|l_{k-1}| + |l_k|) = \Theta(\sum_{i=1}^{k} |l_i|) = \Theta(n)$$

יש בדיוק $logk$ שלבים של האלגוריתם, ולכן סה"כ הסיבוכיות $\Theta(nlogk)$

__נשים לב:__ עבור המקרה שמספר הרשימות הוא לא חזקה של 2, אלא עבור $k \in \mathbb{N}$.
קיים $m \in \mathbb{N}$ כך ש-$2^{m-1} \leq k \leq 2^m$. ונחלק כל פעם את הרשימה הארוכה ביותר ל-2 עד שיהיו 2^m רשימות ואז נפעיל את האלגוריתם
הסיבוכיות במקרה הזה:

$$exercise \leq \Theta(nlog2^m) = \Theta(nlog(2^{m-1}) + nlog(2)) = O(nlog(2^{m-1})) = O(nlog(k))$$

__הערה:__ אם $k = \Theta(n)$ פשוט נמיין את כל האיברים

3 תרגול 3

__תזכורת:__ ראינו אלגוריתם שמקבל רשימות ממויינות $A_1, \ldots, A_k$ עם סה"כ n איברים, ונמזג אותם לרשימה ממויינת אחת, בסיבוכיות $O(nlogk)$

3.1 תרגיל

להראות שלא קיים אלגו' מבסוס השוואות לבעיה הזו עם זמן ריצה פחות מ-$\Omega(nlogk)$ (נניח שכל הרשימות באותו אורך) $|A| = n$, $A_1, \ldots, A_k$ כך שלכל $i \in [k]$ נתקיים $|A_i| = \frac{n}{k}$
__תשובה:__
יהא אלגוריתם השוואות שמקבל k רשימות וממזג אותם לרשימה ממוינת אחת.
$A_1 = \{a_1^i, \ldots, a_{l_1}^i\}$,
נניח שיש 2 רשימות $A = \{a_1, a_2, \ldots, a_n\}$ ו $B = \{b_1, \ldots, b_l\}$ ו-$C = \{c_1, \ldots, c_k\}$
למיזוג (לא ממוין) 3 הרשימות האלו, יש $\binom{n+l}{n}\binom{n+l+k}{k}$ אפשרויות (במערך יש $n+l$ איברים, וצריך לבחור סה"כ את מקום של n איבריםוהשאר פשוט נניח את l האיברים הנותרים) וכו'
__לדוגמה:__ $A = [1, 2, 5, 7]$ ו-$B = [a, b, c] \leftarrow 1, a, 2, b, 5, 7, c$
אז באופן כללי בהינתן k רשימות שמקיימות $|A_i| = l_i$ מספר הפלטים האפשריים של האלגוריתם הוא:

$$\binom{l_1}{l_2}\binom{l_1 + l_2}{l_2}\binom{l_1 + l_2 + l_3}{l_3} \cdots \binom{n}{l_k} = \binom{n}{l_1, l_2, \ldots, l_k} = \frac{n!}{l_1! l_2! \ldots l_k!}$$

אם נתבונן בעץ ההשוואות של האלגוריתם, יהיו בו- $\binom{n}{\frac{n}{k}, \cdots k, \frac{n}{k}}$ עלים.
אז לכן זמן הריצה הגרוע ביותר יהיה לפחות

$$log\binom{n}{\frac{n}{k}, \cdots k\ , \frac{n}{k}} = log(\frac{n!}{(\frac{n}{k!})^k}) = log(n!) - klog(\frac{n}{k}!) \underset{(\frac{k}{e})^k \leq k! \leq k^k\ sterling}{\geq} c_1 nlog(n)$$

$$= klog[(\frac{n}{k})^{\frac{n}{k}}] = (c_1 nlog(n) - nlog(n)) + nlog(k) \geq_{for\, big\, n} nlog(k)$$

כנדרש.

3.2 $Quicksort$

נתבונן באלגוריתם הבא: בהינתן רשימה A באורך n:

1. **בוחרים ($pivot$) $a_k \in A$**
2. **נגדיר רשימות $B = \{a_i \in A | a_i \leq a_k\}$ ו־$C = \{a_i \in A | a_i > a_k\}$ (אם c ריקה ־ נבחר a_kאחר)**
3. **נמיין את B, C ריקורסיבית ונחזיר (B, C)**

<u>הוכחת נכונות+אלגוריתם עוצר:</u> תרגיל

3.2.1 זמן ריצה

תלוי בבחירה בשלב 1 $\leftarrow$ מה הבחירה בטובה ביותר ?
נזמן ב־$T(n)$ את זמן ריצת האלגוריתם , אז $T(n) = \min\limits_{0 \leq q \leq n-1}[T(q) + T(n-q-1)] + \Theta(n)$ (כאשר $\Theta(n)$ זה הזמן של שלב 2)
ניחוש ־ $T(n) \leq cnlog(n)$ עבור c קבוע כלשהו, נראה עבור q הטוב ביותר:

$$T(n) = min[T(q) + T(n-q-1)] + \Theta(n) \leq T(\lceil \frac{n}{2} \rceil) + T(\lfloor \frac{n}{2} \rfloor) + \Theta(n) =_{like\, mergesort} \Theta(nlongn)$$

3.2.2 סיבוכיות ממוצעת

נניח שבשלב 1 בוחרים תמיד את a_1, על פני כל הקלטים האפשריים, מה זמן הריצה של האלגורית

$$\tilde{T(n)} = \frac{\sum_{q=0}^{n-1} \tilde{T}(q) + \tilde{T}(n-q-1)}{n} + c \cdot n =_{q\, twice} \frac{\sum_{q=0}^{n-1} 2\tilde{T}(q)}{n} + c \cdot n$$

$$n\tilde{T}(n) = \sum 2\tilde{T}(q) + c \cdot n$$

$$n\tilde{T}(n)-(n-1)\tilde{T}(n-1)=\sum^{n-1}2\tilde{T}(q)+cn^2-\sum^{n-2}2\tilde{T}(q)-c(n-1)^2=2\tilde{T}(n-1)+c(n+1)$$

$$\tilde{T}(n)=\frac{2(n+1)\tilde{T}(n-1)}{n}+c\frac{n+1}{n}=_{exercise}O(nlog(n))$$

4 תרגול 4

4.1 תרגיל

לתת אלגוריתן שמקבל רשימה לא ממוינת A באורך n ומחלק אותה ל־k קבוצות (שוות).
$A_1, A_2 \ldots, A_k$ ($|A_i|=\frac{n}{k}$) כך שלכל i כל איבר ב־A_i קטן מכל איבר ב־A_{i+1}

<u>פתרון:</u>

1. **<u>האלגוריתם:</u>**
   (א) **אם $k=1$ נחזיר את A**
   (ב) **אחרת, נמצא ע"י אלגוריתם $select$, את האיבר ה־$\frac{n}{k}\cdot\lfloor\frac{k}{n}\rfloor$ ב־A.**
   (ג) **נבנה שני רשימות חדשות $B=\{a_i:a_i\le X\}$, $|B|=\lfloor\frac{k}{2}\rfloor\cdot\frac{n}{k}$ ו־ $C=\{a_i:a_i>X\}$ $|C|=\lceil\frac{k}{2}\rceil\cdot\frac{n}{k}$**
   (ד) **נמשיך רקורסיבית. נחזיר $Part(B,\lfloor\frac{k}{2}\rfloor\frac{n}{k},\lfloor\frac{k}{2}\rfloor)$ ו־$Part(C,\lceil\frac{k}{2}\rceil\cdot\frac{n}{k},\lceil\frac{k}{2}\rceil)$.**
2. **<u>נכונות+עצירה:</u>** תרגיל
3. **<u>סיבוכיות זמן ריצה:</u>**
   (א) **שלב: $O(1)$**
   (ב) **שלב 2: $O(n)$**
   (ג) **שלב 3: $O(n)$**

← סה"כ אם נזמן ב$T(n,k)$ את הזמן הריצה, מתקיים $T(n,k)=T(\frac{n}{k}\cdot\lfloor\frac{k}{2}\rfloor,\lfloor\frac{k}{2}\rfloor)+T(\lceil\frac{k}{2}\rceil\cdot\frac{n}{k},\lceil\frac{k}{2}\rceil)+c\cdot n$
נוכיח ש־$T(n,k)\le cnlogk$, נוכיח באינדוקציה:
עבור $T(n,1)=c^\star$, אז עבור $n=1$ ברור
נניח ראשית ש־$k=2^m$
אז: $T(n,2^m)=2T(\frac{n}{2},2^{m-1})+cn\le_{induc..}2(c'\frac{n}{2}log2^{m-1})+cn=c'n(m-1)+cn$
$\le_{c<c'}c'n(m-1)+c'n=c'n\cdot m=c'n\cdot logk$
<u>תרגיל:</u> להראות עבור $n\neq 2^m$ ⇐ הסיבוכיות $O(nlogk)$

4.2 תרגיל

להוכיח שלא קיים אלגוריתם השוואות שעובד בסיבוכיות קטנה מ־$nlogk$.

<u>תשובה:</u>

נתבונן באלגוריתם שפותר את הבעיה. יש $\binom{n}{\frac{n}{k},\frac{n}{k},\dots k,\frac{n}{k}} = \frac{n!}{(\frac{n}{k}!)^k}$ (כל הדרכים לבחור k קבוצות בגדול $\frac{n}{k}$ מתוך n) עלים בעץ השוואות, לכן חסם תחתון לסיבוכיות זמן הריצה הוא $\frac{n!}{(\frac{n}{k})^k} = \Omega(nlogk)$

4.3 תרגיל

לתת אלגוריתם שמקבל רשימה באורך n ומוצא מינימום ומקסימום. בלכל היותר $\frac{3}{2} \cdot n - 2$ השוואות

<u>תשובה:</u>

אפשרות 1 ־ ריקורסיה (תרגיל)

אפשרות 2 ־

4.4 תרגיל

להוכיח שאין אלגוריתם שעושה פחות השוואות מ$\frac{3}{2} \cdot n - 2$ עבור מעבר על רשימה מגודל n ומוצא מינימום ומקסימום.

<u>תשובה:</u>

יהא אלגוריתם שפותר את הבעיה, אז בזמן הריצה, נעכוב אחרי שתי רשימות: A_M, A_m.

ב־A_M יהיו האינדקסים שבהם ייתכן שיש מקסימום, כאשר בתחילת האלגוריתם $A_m = A_M = (1, \dots, n)$

נבדוק כמה השוואות צריך כדי להבטיח ש$|A_M| = |A_m| = 1$

נתבונן בסוגי השוואות האפשריים:

1. אם שווים a_i, a_jכך ש $i, j \in A_M$ וגם $i, j \in A_m$אחרי השוואה גדול A_mוגם A_Mיקטנו ב־1 כל אחד.
2. אם משווים a_i, a_jכך שהב"כ $i \notin A_M$אז גודל A_M אולי לא יקטן.

נשים לב שאפשר לעשות לכל היותר $\frac{n}{2}$השוואות מסוג 1, כי אחרי $\frac{n}{2}$השוואות A_M וA_n זרים ו־$|A_m| = |A_M| = \frac{n}{2}$ צריך לפחות ו־$\frac{n}{2}$ השוואות כדי למצוא מקסימום ב־A_Mוכנ"ל מינימום A_mלכן סה"כ לפחות $\frac{3}{2}n - 2$ השוואות.

5 תרגול

<u>טענה:</u> קיים עץ בינארי עם עלים $V_1, \dots, V_m$ ו־$l_1 \leq l_2 \leq \dots \leq l_m$ כך שהגובה של V_iהוא l_i $\iff \sum_{i=1}^{m} \frac{1}{2^{l_i}} \leq 1$

<u>הוכחה:</u>

<u>$\Leftarrow$:</u> נניח שקיים עץ T כנתון בשאלה

נגדיר T' להיות T בתוספת צמתים כך שT' עץ שלם בגובה l_m. ונגדיר $A_i = \{V \in V(T') | V\, descendant\, of\, V_i, V\, leaf\}$

$\Leftarrow$ $|A_i| = 2^{l_m - l_i}$ ו־$A_i \cap A_j = \emptyset$ $i \neq j$ $\Leftarrow$

$$2^{l_m} \underset{\text{every object in } \cup A_i\, leaf}{\geq} |\cup_{i=1}^{m} A_i| = \sum_{i=1}^{m} |A_i| = \sum_{i=1}^{m} 2^{l_m - l_i} = 2^{l_m} \cdot \sum_{i=1}^{m} \frac{1}{2^{l_i}}$$

$\underline{\Rightarrow}$**:** נוכיח באינדוקציה על m (עלים).

- **<u>בסיס:</u>** $m=1$ - v_i בגובה l_i , וכיוון ש־$l_i \in Z^+$ אז $\frac{1}{2^{l_i}} \leq 1$ (נשים לב שאומנם יש עלה אחד אבל יכולה להיות שרשרת גדולה)
- **<u>הנחה:</u>** נניח שקיים עץ T עם עלים $V_1,\ldots,V_{m-1}$ כך שהגובה של V_i הוא l_i
- **<u>צעד:</u>** נגדיר T' להיות העץ השלם בגובה l_m שמתקבל מ־T ע"י הוספה של צמתים.
ונגדיר A_iכמו קודם ו־

$$|\cup_{i=1}^{m-1} A_i| = \sum_{i-1}^{m-1} |A_i| = \sum_{i=1}^{m-1} 2^{l_m - l_i} = 2^{l_m} \cdot \sum_{i=1}^{m-1} \frac{1}{2^{l_i}} \underset{\text{sum less than 1}}{<} 2^{l_m}$$

נשים לב ש־ $\cup_{i=1}^{m-1} A_i$ הוא אוסף כל העלים ב־T' שהם צאצאים של V_iכלשהו. כיוון ש־ $|\cup_{i=1}^{m-1} A_i| < 2^{l_m}$ קיים עלה u שאינו צאצא של אף V_i,
ולכן נוכל להגדיר עץ בינארי T'' ע"י הוספת u ואבותיו ל־T $\Leftarrow$ T'' הוא עץ כנדרש.

5.1 צפנים - קידוד

<u>תזכורת:</u> אומרים שצופן (קידוד) $\{w_1,\ldots,w_l\}$ הוא צופן רישא, אם לכל w_i, w_j $i \neq j$, w_iלא רישא של w_j

5.1.1 תרגיל

נתונים $l_1 \leq l_2 \leq \cdots \leq l_m$ כך ש־$\sum_{i=1}^{m} \frac{1}{2^{l_i}} \leq 1$, לבנות צופן רישא $\{w_1,\ldots,w_m\}$ כך ש־$|w_i| = l_i$
<u>תשובה:</u>
נגדיר את w_i להיות ה־l_iספרות הראשונות אחרי הנקודה בפינוח הבינארי של - $\sum_{k=1}^{i-1} \frac{1}{2^{l_k}}$.
זה צופן רישא $i<j$ אז מתקיים סכום $\sum_{k=1}^{j-1} \frac{1}{2^{l_k}} \geq \sum_{k=1}^{i-1} \frac{1}{2^{l_k}} + \frac{1}{2^{l_i}}$ $\Leftarrow$ $\frac{1}{2^{l_i}} \leq \sum_{k=1}^{j-1} \frac{1}{2^{l_k}} - \sum_{k=1}^{i-1} \frac{1}{2^{l_k}}$

<u>דוגמה:</u> $l_1 = 2, l_2 = 4$ $\Leftarrow$ $w_1 = 00$ $\frac{1}{2^2} = 0.0100..$ ו־$w_2 = 0100$

6 תרגול

6.1 משפט פרלס

עבור קבוצות $A_1,\ldots,A_m,B_1,\ldots,B_m \subset \{1,\ldots,n\}$ לא ריקות כך:

1. שלכל i, מתקיים $A_i \cap B_i = \emptyset$,

2. ולכל $i \neq j$ או $A_i \cap B_j \neq \emptyset$ או $A_j \cap B_i \neq \emptyset$

מתקיים:

$$\sum_{i=1}^{m} \frac{1}{2^{|A_i|+|B_i|}} \leq 1$$

<u>הוכחה:</u> נשתמש בהסתברות.
נגריל באופן אחיד איבר מ־$\{0,1\}^n$ (n־יה של אפסים ואחדים) ונסמנו ב־$(x_1,\ldots,x_n)$ ונגדיר את המאורע $C_i :=$
$\begin{cases} x_j = 1 & \forall j \in A_i \\ x_j = 0 & \forall j \in B_i \end{cases}$ (אפשרי כי $A_i \cap B_i \neq \emptyset$)

$$\Rightarrow P(C_i) = \frac{1}{2^{|A_i|+|B_i|}}$$

כעת, נטען שהמאורעות $C_1,\ldots,C_m$ זרים בזוגות. אם $i \neq j$, C_i, C_j זרים כי לפי הנתון בה"כ $A_i \cap B_j \neq \emptyset$, לכן קיים $k \in A_i \cap B_j$, ובמאורע C_i מתקיים $x_k = 1$
ובמאורע C מתקיים $X_k = 0$ לכן הם זרים.

סה"כ, נקבל

$$1 \geq P(\bigcup_{i=1}^{m} C_i) = \sum_{i=1}^{m} P(C_i) = \sum_{i=1}^{m} \frac{1}{2^{|A_i|+|B_i|}}$$

<u>מסקנה:</u> (המשפט שראינו בכיתה) אם $w_1,\ldots,w_m$ צופן רישא, אז $\sum \frac{1}{2^{|w_i|}} \leq 1$.
<u>הוכחה:</u>
בהנתן צופן רישא $w_1,\ldots,w_m$ ונכתוב $w_i = x_1^i, x_2^i \ldots, x_n^i$ ונגדיר $A_i := \{j | x_j^i = 1\}$ ו־ $B_i := \{i | x_j^i = 0\}$ ומתקיים
כמובן $A_i \bigcap B_i = \emptyset$
$\{w_i\}$ זה צופן רישא ולכן עבור $i \neq j$, w_iאינו רישא של w_j $\Leftarrow$ קיים אינדקס k , כך שב־k w_iוw_j לא מסכימות.
$\Leftarrow$ לכן מתקיים או $A_i \bigcap B_j \neq \emptyset$ או $A_j \bigcap B_i \neq \emptyset$.
$\Leftarrow$ ואז המסקנה מתקיימת מהמשפט.

6.2 קידוד חסר רעש

<u>בעיה:</u> בהינתן $P = (p_1,\ldots,p_m)$ וקטור כך ש־$p_i > 0$, $\sum p_i = 1$.
רוצים צופן רישא $w_1,\ldots,w_m$כך ש־$\sum |w_i| \cdot p_i$ קטן ככל האפשר.

באופן שקול, רוצים עץ T עם עלים $V_1,\ldots,V_m$ כך ש־$C(T,p) = \sum p_i \cdot h(V_i)$

<u>הגדרה:</u> פונקציה האנטרופיה של p היא

$$H(p_1,\ldots,p_m)=\sum_{i=1}^{m}P_i\cdot log_2(\frac{1}{p_i})$$

<u>משפט:</u> אם נסמן $f(p)=\underset{T}{min}C(T,p)$ מתקיים:

$$H(P)\le f(P)<H(P)+1$$

<u>הוכחה:</u>

<u>$H(P)\le f(P)$:</u>

יהי T עץ, נראה שמתקיים $H(p)\le C(T,p)$ ← יהיו $V_1,\ldots,V_m$העלים של T ומתקיים

$$\sum_{i=1}^{m}\frac{1}{2^{h(V_i)}}\le 1\underset{log}{\iff}0\ge log(\sum\frac{1}{2^{h(V_i)}})=log(\sum_{i=1}^{m}p_i\cdot\frac{1}{p_i}\cdot 2^{-h(V_i)})\underset{inf i\,1-Concave}{\ge}\sum[p_i log(\frac{1}{p_i})-p_i\cdot h(V_i)]$$

סה"כ

$$0\ge\sum p_i\cdot log(\frac{1}{p_i})-\sum p_i h_i(v_i)$$

$$H(p)=\sum p_i log(\frac{1}{p_i})\le\sum p_i h(v_i)=C(T,p)$$

אם לכל T $H(p)\le C(T,p)$ מתקיים גם $H(p)\le\underset{T}{min}C(T,p)$ כנדרש.

7 תרגול 7 - תכנון דינמי

7.1 תרגיל

נתון מוט בארוך n , ווקטור $P=(p_1,\ldots,p_n)$ כאשר $p_i\ge 0$ וp_i - מחיר של מוט באורך i.

<u>המטרה:</u> לחתוך את המוט לחתיכות קטנות יותר כדי למקסם את מחיר סה"כ חתיכות.

<u>פורמלית:</u> למנצוא $l_1,\ldots,l_k$כך ש- $\sum_{i=1}^{k}l_i=n$ ו- $\sum_{i=1}^{k}p_{l_i}$מקסימלי .

$$f_p(n):=\underset{l_i}{max}\sum p_{l_i}$$

לכתוב אלגוריתם שמוצא את $f_p(n)$

<u>תשובה:</u>

נסתכל על הפתרון הנאיבי:

```
cut(n,p)
    if(n=0)
        return 0
    q=-1
    for i=1 to n
        q=max{q,p[i]+cut(n-i,p))
    return q
```

נסתכל על פיתרון אחר:

נשים לב שכל תת קבוצה $A \subset \{i, \ldots, n-1\}$ מגדירה חיתוך של מוט $A = \{2,4\} \leftarrow$

נסמן ב$g_p(A)$ את ערך החלוקה המוגדרת ע"י A. **(**$A \subset \{1, \ldots .n-1\}$**)**

```
cut(n,p)
    return max g(A) (A in {1,...,n-1})
```

<u>סיבוכיות</u> : $O(2^{n-1})$.

נשים לב שאנחנו מחשבים $cut(n-k,p)$ מספר פעמים עבור k קבוע $\Leftarrow$ במקום , נחזיק רשימה עם הערכים הנ"ל (בתקווה שיחסוך לנו)

```
1  cut_aux(n,P,A)
2      if n=0
3          return 0
4      q=-1
5      for i=1 to n
6          if(A[k]<0)
7              A[i]=cut_max(n-i,P,A)
8          q=max{q,P[i]+A[i]}
9      return q
10
11 cut(n,p)
12     A=new arr size n with value -1
13     return cut_aux(n,P,A)
```

<u>האלגוריתם עוצר:</u> תרגיל

<u>נכונות אלגוריתם:</u> נוכיח באינדוקציה על n ש־cut_aux(n,P,A) מחזיר את המספר האופטימלי של חיתוך מוט באורך n.

- **<u>בסיס:</u>** $n = 0$ מחזיר 0 ע"ע שורות $2,3$
- **<u>צעד:</u>** נניח שלכל $i \geq 1$ cut_aux(n-1,p,a) מחזיר את המחיר האופטימלי. קיימת חלוקה אופטילית $l_1, \ldots, l_k$ כאשר $l_i \geq 1$ ו$\sum_{i=1}^{k} l_i = n$.
בשורה 4 , כאשר $l_i = i$, ע"פ הנחת האינדוקציה cut_aux(n-li,p,a) יחזיר חלוקה אופטימלית של מוט באורך

$n - l_i$, ובתוספת p_{l_i}זה המחיר המקסימלי.

<u>סיבוכיות:</u> נשים לב שהקריאה הרקורסיבית ב־$A[i] = \text{cut\_aux}(n-i, P, A)$ מתבצעת <u>בדיוק</u> פעם אחת עבור כל i. הסיבוכיות של $cut(n-i, P, A)$ פרט לרקורסיה היא $O(n-i)$, לכן סה"כ הסביכויות $\sum_{i=1}^{n} O(n-i) = O(n^2)$.

8 תרגיל

8.1 תזכורת

עבור גרף $G = (V, E_G)$ קשיר

- <u>עץ פורש</u> הוא גרף $T(V, E_T)$ כך ש־$E_T \subset E_G$
- אם יש משקלים על הקשתות $w : E_G \to \mathbb{R}$, נאמר ש־T הוא <u>עץ פורש מינימום</u> אם $\sum_{e \in E_T} w(e)$ הוא הקטן ביותר על פני כל העצים הפורשים האפשריים.

8.2 תרגיל

<u>הגדרה:</u> <u>חתך</u> בגרף $G = (V, E_G)$ הוא זוג (A, B) כך ש־ $A, B \subset V$ כך שמתקיים: $A, B \neq \emptyset$ ו־$A \bigcap B = \emptyset$ ו־$A \bigcup B = V$
נאמר שקשת $e \in (A, B)$ אם קצה אחד שלה שייך לA והקצה שני שייך ל־B

<u>שאלה:</u> יהי $G = (V, E_G)$ גרף קשיר עם משקלים $w : E_G \to \mathbb{R}$ ותהי $e \in E_G$.
להראות ש־e שייך לעץ פורש מינימום $\iff$ קיים חתך (A, B) ב־G כך שe היא הקשת הכי קלה בו.
<u>תשובה:</u>
<u>$\Leftarrow$:</u> יהי T עץ פורש מינימלי שמכיל את e. נסמן $e = (v_0, u_0)$ נשים לב שלכל צומת $v \in V$ יש מסלול יחיד ב־T מ־v ל־u_0.
אז נגדיר את A להיות קבוצת הצמתים שעבורן מסלול זה עובר ב־e . ו־$B = V \backslash A$.
נשים לב ש־(A, B) זה חתך (למה?)
e היא קשת קלה ביותר ב־(A, B) כי אם יש $e' \in (A, B)$ כך ש־$w(e') < w(e)$ אז מתקיים ש־$T' = (T - \{e\}) \bigcup \{e'\}$ הוא עץ פורש קל יותר מ־T.
(תרגיל למה T' עץ?) וסיימנו.
<u>$\Rightarrow$:</u> יהי (A, B) חתך ב־G כך ש־e היא הקשת הקלה ביותר בו. נקח T להיות עץ פורש מינימלי של G.
אם $e \in T$, סיימנו.
אחרת, נשים לב שיש קשת $e' \in T$ כך ש־$e' \in (A, B)$ (אחרת ב־Tאין מסלול מצמתים ב־A לצמתים ב־B)
נגדיר, $T' = (T - \{e'\}) \bigcup \{e\}$ זה גם עץ פורש, כי יש בו $|V| - 1$ קשתות, ואין בו מעגלים (תרגיל)
ומתקיים ש־

$$\sum_{e \in T'} w(e) = w(e) + \sum_{e \in E(T)} w(e) - w(e') \leq \sum_{e \in (T)} w(e) \underset{minimal}{\leftarrow} ?$$

ולכן T' עץ פורש מינימלי ו־$e \in T'$.

8.3 תרגיל - תכנות דינמי

<u>תזכורת:</u> כפל מטריצות מקיים $(A \cdot (B \cdot C)) = ((A \cdot B) \cdot C)$

נתונות מטריצות $A_1, \dots, A_{nn}$ כאשר A_i מטריצה $p_{i-1} \times p_i$, רוצים למצוא את מספר הפעולות הכפל המינימלי שהכרחי לחישוב למכפלת כל המטריצות.

<u>תשובה:</u>

האלגוריתם -

```
def MULT (P,n):
    reun MULT_AUX(1,n,P)

def MULT_AUX(i,j,p):
    if(i==j):
        return 0
    b=infinity
    for k=i to j-1:
        b=min{b,MULT_AUX(i,k,P)+MULT_AUX(k+1,j)+P[i-1]*P[k]*P[j]}
    return b

```

ואחרי שיפור:

```
#improve, B is 3d array that save results - to make to code faster
def MULT (P,n):
    B(n,n)=infinity
    reun MULT_AUX(1,n,P,B)

def MULT_AUX(i,j,p):
    if(i==j):
        return 0
    c=infinity
    for k=i to j-1:
        if (B(i,k)=infinity):
            B(i,k)=MULT_AUX(i,k,P,B)
        if(B(k+1,j)=infinity):
            B(k+1,j))=MULT_AUX(...)
        c=min{c,B(i,k)+(B(k+1)+P[i-1]*P[k]*P[j]}
    return c

```

סיבוכיות של השיפור : $O(n^3)$.

9 תרגול

9.1 אלגוריתמים בגרפים

<u>בעיה:</u> נתון גרף מכוון G עם משקולות W (נתון שאין מעגל עם משקל שלילי ב־G - כדי לא להיכנס ללופ אין סופי) כאשר $W : E(G) \to \mathbb{R}$, רוצים למצוא משקל לכל זוג צמתים $u, v \in V(G)$, משקל של מסלול קל ביותר מ־v ל־u ב־G.

<u>הנחה:</u> נניח ש־ $V(G) = \{v_1, \ldots, v_n\}$ ונניח שאם עבור $u, v \in V(G)$ אם $(v, u) \notin E(G)$ נכתוב $w(u, v) = \infty$

<u>ניסיון:</u> ננסה ברקורסיה:

```
def APSP(G,W):
    if v(G)==1:
        return A(1,1)=0
    A(n-1,n-1)=APSP(G-Vn,W)
        for i=1 to n-1:
            A(i,n)=min{A(i,j)+W(j,n)} # min on 1<j<n
            A(n,i)=min{A(j,i)+W(n,j)} # min on 1<j<n
        for i,j=1 to n-1:
            A(i,j)=min{A(i,j),A(i,n)+A(i,j)}
```

ניסיון2 נשפר - במקום למחוק כל פעם צומת, נעבוד איטרטיבית ובאינטרציה ה־k נרשה רק מסלולים שהמסלולים שבצמת הביניים שלהם בקבוצה $\{v_1, \ldots, v_k\}$

```
def APSP_Floyd_Warshall(G,W):
    def n times A(n,n) #make n+1 matrix n x n
    if (i=j):
        A[0](i,j)=0
    else:
        A[0](i,j)=w(i,j)
    for k=1 to n:
        for i,j:
            A[k](i,j)=min{A[k-1](i,j),A[k-1](i,k)+A[k-1](k,j)}
    return A[n]
```

<u>נכונות:</u> נוכיח באינדוקציה שאחרי השלב ה־k בלולאה x בטבלה $A[k]$ במקום ה־(i, j) נמצא המשקל הסמלול הקל ביותר מ־v_i ל־v_j שצומתי הביניים שלו בקבוצה $\{v_1, \ldots, v_k\}$

<u>בסיס:</u> $k = 0$ ברור

<u>צעד:</u> יהי מסלול קל ביותר מ־v_iל־v_j שכל צמתי היניים שלו ב־$\{v_1, \ldots, v_k\}$ יש שני אפשרויות:
הראשונה : המסלול לא עובר ב־v_k. במקרה זה , זה גם המסלול קל ביותר שכל צמתי הביניים שלו ב־$\{v_1, \ldots, v_k\}$ ולכן ערכו $A[k-1](i,j)$ ע"פ הנחת האינדוקציה
השניה: המסלול עובר דרך v_k אז :

$$v_1 \to u_1 \to \cdots \to v_k \to u_t \to u_{t+1} \to \cdots \to u_l \to u_j$$

נשים לב ש־$v_i \to u_1 \to \cdots \to u_{t-2} \to v_k$ מסלול קל ביותר מ־v_iל־v_k וכל $u_z \in \{v_1, \ldots, v_k\}$ לכן משקל מסלול זה שווה ל־$A[k-1](i,k)$ ע"י הנחת האינדקוציה.
בדומה למשקל המסלול $v_j \to \cdots \to v_k$ שווה ל־$A[k-1](k,j)$ לכן סה"כ משקל כל המסלול שווה ל־ $A[k-1](i,k) + A[k-1](k,j)$
$\Leftarrow$ לכן $A[n]$ מכיל את הערכים הדרושים

<u>אלגוריתם עוצר:</u> הלולאה קוראת n פעמים, $n-1$ סופי...
<u>סיבוכיות:</u> $O(|V|^3)$

10 תרגול

10.1 תרגיל

שאלה: נתון G גרף פשוט עם משקלים $w: E \to \mathbb{R}$ **, כאשר** $w(e) \geq 0$ **לכל** $e \in E$.
נתונים $s, t \in V$ <u>לתת אלגוריתם</u> שמוצא את כל הצמתים ששייכים למסלול קל ביותר מ־s ל־t

תשובה: <u>אלגוריתם</u>

0. נגדיר $A = \{\}$

1. נחשב את $d(s,v)$ לכל $v \in V$ ע"י דייסטרא

2. נחשב את $d(t,v)$ לכל $v \in V$ ע"י דייסטרא

3. לכל צומת $u \in V$ נוסיף את u ל־A אם $d(s,t) = d(s,u) + d(t,u)$

4. נחזיר את A

<u>נכונות:</u> **טענה:** בסוף האלגוריתם צומת u שייך ל־A $\iff$ u שייך למסלול קל ביותר בין s ל־t .
הוכחה טענה: $\Leftarrow$ נניח ש־$u \in A$ $\Leftarrow$ אז יש מסלול קצר ביותר מ־s ל־u $d(s,u)$ ויש מסלול קצר ביותר מ־u ל־t $d(t,u)$ ומתקיים שיש מסלול שהוא חיבור של המסלולים. והוא גם יהיה המסלול הקל ביותר מ־s ל־t ומתקיים שמשקלו $d(s,u) + d(t,u) = d(s,t)$
$\Rightarrow$ יהי $v \in V$ כך שקיים מסלול בין t ל־s שמשקלו קטן ביותר. כיוון שתת מסלול של מסלול קל ביותר הוא קל ביותר, משקל המסלול בין t ל־v שמשקלו $d(v,t)$ ובאותה צורה מסלול מv ל־s שמשקלו $d(s,v)$ ולכן מתקיים $d(t,v) + d(s,v) = d(s,t)$ ולכן $v \in A$

<u>אלגוריתם עוצר:</u> דייקסטרא עוצר, ועשינו מספר פעולות קבוע, כל צומת בנוסף.

<u>סיבוכיות:</u> שלבים $1,2$ הם דייסטרא $O(|V| + |E| log(|V|))$ ושלב 3 מתבצע בסיבוכיות $O(|V|)$ וסה"כ $O(|V| + |E| log(|V|))$.

10.2 תרגיל

שאלה: נתון גרף $G=(V,E)$ (ללא משקלים) ונתון $s\in V$. תנו אלגוריתם שנוצא לכל $v\in V$ אורך הקצר ביותר של מסלול באורך זוגי מ־s ל־v.

פתרון: <u>האלגוריתם:</u>

1. נבנה גרף G' באופן $V_{G'}=V_G\mathring{\bigcup}A$ כאשר $A:=\{v'|v\in V_G\}$. עבור $v,u\in V_{G'}$ קשת בין u ו־ v $\iff$ $v\in V_G$, $u'=u\in A$ ו־בG יש קשת בין v לצומת המתאימה ל־u
2. נריץ BFS ב־G' מ־S. לכל $v\in V_G\mathring{\bigcup}A$ נחזיר את המרחק ב־G' מ־S ל־v.

<u>נכונות:</u> יהי $v\in V_G$נטען שאורך המסלוך הזוגי הקצר ביותר מ־s ל־v ב־G שווה לאורך הסלול הקצר ביותר מ־s ל־v ב־G'

אם המסלול מs ל־v מסלול הזוגי הקצר ביותר ב־G אז המסלול בין s לv קיים ב־G'. מצד שני, אם קיים מסלול נוסף בין s ל־v , שהוא ב־G' אז המסלול הנוסף קיים גם ב־G (יש התאמה חח"ע בין המסלולים הגרפים).

<u>סיבוכיות ועוצר:</u> תרגיל

11 תרגול - זרימה

11.1 תזכורת

<u>הגדרה:</u> נתון גרף מכוון G עם משקלים $c : E \to \mathbb{R}^+$, ושני צמתים $s, t \in V_G$, אז נגיד שפונקציה $f : E \to \mathbb{R}^+$ היא <u>זרימה</u> אם:

(1) לכל $s, t \neq v \in V_G$ מתקיים

$$\sum_{(v,u)\in E_G} f((v,u)) = \sum_{(u,v)\in E_G} f((u,v))$$

(2) לכל $e \in E_G$ מתקיים $f(e) \leq c(e)$

<u>הגדרה:</u> <u>הערך</u> של זרימה f הוא:

$$\sum_{(s,u)\in E_G} f((s,u)) - \sum_{(u,s)\in E} f((u,s))$$

הערה: קיימים אלגוריתמים שמוצאים זרימה מקסימלית - כרגע נשתמש בהם בקופסא שחורה:

(1) פורד פלקרסון - סיבכויות $O(|E| \cdot f^*(G))$ (כאשר $f^*(G)$ הוא ערך הזרימה המקסימלית)

(2) אדמונדס-קארפ - סיבכויות $O(|E|^2 \cdot |V|)$

11.2 תרגיל

<u>שאלה</u> נתון גרף דו צדדי $G = (A\dot{\bigcup}B, E)$ רוצים למצוא זיווג מקסימלי ב-G, הראו האלגוריתם כזה

<u>תזכורת</u> זיווג היא קבוצת קשתות זרות

<u>תשובה</u> בהינתן $G = (A\dot{\bigcup}B, E)$ נגדיר G' באופן הבא:

$$V_{G'} = A\bigcup B\bigcup\{s,t\}$$
$$E_{G'} = E_G\bigcup(\{s\}\times A)\bigcup(B\times\{t\})$$

(נכוון כל קשת בE_Gמ-A ל-B)

נגדיר $c : E_{G'} \to \mathbb{R}^+$: אם $e \in E_G$ אז $c(e) = 1$ אחרת $c(e) = \infty$

<u>האלגוריתם-</u> בהינתן G:

(1) - נבנה G' כנ"ל

(2) - נמצא זרימה מקסימלית ב-G' ע"י פורד-פלרקסון

(3)- נחזיר את הזיווג $D := \{e \in E_G | f(e) = 1\}$

נכונות- נטען שהאלגוריתם מחזיר זיווג מקסימלי

(1) - האלגוריתם מחזיר זיווג, נניח בשלילה כי שלא, כלומר ב-D יש שני קשתות e, e' שאינן זרות
יש שני אפשרויות - $e \bigcap e' = v \in A$, נשים לב של-v נכנסת קשת אחת בלבד עם קיבול 1, לכן לא יתכן שלשני קשתות שיוצאות מ-v יש זרימה 1

(2) - $e \bigcap e' = u \in B$ (תרגיל להשלים)

$\Leftarrow$ קיבלנו שהאלגוריתם נותן זיווג.

הזיווג המקסימלי: נניח שקיים זיווג גדול יותר $D' \leftarrow$ נוכל להגדיר זרימה $\hat{f}$ב-G' באופן הבא:
לכל $e \in D'$ $\hat{f}(e) = 1$, לכל $e \in D'$ מתקיים $\hat{f}(s, e \bigcap A) = \hat{f}(e \bigcap B, t) = 1$ ולכל קשת אחרת הזרימה היא-0
זה מגדיר זרימה (נראה לפי התנאים של ההגדרה):

(1) מתקיים : כל צומת ב-A ששייך לקשת ב-D', שייך לקשת ????? להשלים....

(2) נכון מההגדרה

נשים לב לב שערך הזרימה $\hat{f}$ שווה ל- $|D'|$, וערך הזרימה f שווה ל-$|D|$ לכן f אינה זרימה מקסימלית וזו סתירה לנכונות ff.

האלגוריתם עוצר- הבנייה סופית, FF עוצר, 3 סופי ולכן האלגוריתם עוצר

סיבוכיות-

סיבוכיות הבניה - $O(|V| + |E|)$

סיבוכיות FF - $O(|E|^2)$ בגלל הזיווג

סיבוכיות שורה 3 - $O(|E|)$

סה"כ : $O(|V| + |E|^2)$

12 תרגול

12.1 תרגיל

שאלה: נתון בית חולים עם n רופאים. בשנה יש m חגים, כאשר נסמן ב־A_i את התאריכים של החג ה־i.
נסמן $A=\bigcup A_i$ רוצים לשבץ רופאים לתורנויות בימי חג תחת התנאים הבאים:
(1) ־ בכל יום ב־Aמשובץ רופא
(2) ־ לכל רופא $i\in[n]$ יש רשימה של תאריכים B_i שבהם הוא לא יכול לעבוד
(3) ־ רופא לא יעבוד יותר מיום אחד בכל חג
(4) ־ רופא יעבוד לא יותר מ־c ימי חג סה"כ (cקבוע נתון)
לתת אלגוריתם שמוצא שיבוץ או אומר שלא קיים כזה

פתרון: בנייה בהנתן m,n,A,B,c. נגדיר גרף G מכוון הבא:
<u>קודקודים:</u>

$$V_G=\{s,t\}\bigcup\{1,\ldots,n\}\bigcup\{A_i^j|1\le i\le m,1\le j\le n\}\bigcup\{A_i^k|1\le i\le m,1\le k\le|A_i|\}$$

<u>הקשתות:</u>
־בין S ל־i יש קשת עם קיבול c
־בין i ל־A_j^i יש קשת עם קיבלו **1**
־בין A_j^i ל־A_j^k יש קשת אם הרופא יכול לעבוד ביום ה־k של החג ה־j, והקיבול הוא 1
־ביןA_j^k ל־t יש קשת עם קיבול **1**

<u>האלגוריתם:</u>
־ בהינתן c,B,A,m,n נבנה את הגרף G
־נמצא זרימה מקס' בגרף ע"י FF
־אם ערך הזרימה שווה ל־$|A|$, נשבץ את הרופא ה־i ליום A_j^k אם עוברת זרימה חיובים מ־i ל־A_j^k. אם ערך הזרימה קטן ב־$|A|$ נחזיר שאין שיבוץ.

<u>נכונות:</u> נעבור על הבנייה, ונוודה שאם יש זרימה עם ערך $|A|$ כל התנאים מתקיימים.
1. מתקיים כי החתך עם ערך זרימה בגדול של $|A|$, החתך $(V_G\backslash\{t\},\{t\})$ רווי, כלומר עוברת זרימה חיובית בכל צומת A_i^j, כלומר לכל יום מושבץ רופא.
2. תרגיל
3. הקיבול של כל הקשתות מ־i ל־A_j^i הוא 1. לכן הרופא ה־i לא משובץ ליותר מיום 1 בחג ה־j.
4.תרגיל

נצטרך גם להסביר שאם יש שיבוץ אז ניתן להגדיר זרימה ברשת הנ"ל באופן הבא: תרגיל

<u>עוצר:</u> הבנייה סופית, FF עוצר

<u>סיבוכיות:</u> הסיבוכיות בניית הגרף הוא$O(n+|A|+|B|+n\cdot m+n\cdot|A|)$ וסיבוכיות FF $O(|A|\cdot(n\cdot m+n\cdot|A|))$

12.2 תרגיל

שאלה: עבור גרף G נתבונן בכיסויים של G ע"י מסלולים:

כיסוי של G ע"י מלסולים הוא חילוק צמתי G לקבוצות $P_1,\ldots,P_k$ כך ש־P_i מסלול

נתון ש־G DAG (גרף מכוון חסר מעגלים). תנו אלג' שמוצא כיסוי ע"י מסלולים כך שמספר המסלולים הוא הקטן ביותר (כמובן שאף צומת לא חוזרת פעמיים)

פתרון: בנייה בהינתן G כנ"ל נגדיר G' כאשר $V_{G'}=V_G\dot{\bigcup}\{v'|v\in V_G\}$ ומתקיים שעבור הקשתות $(v,u')\in E_{G'}\iff(v,u)\in E_G$ (גרף דו צדדי כמו שהגדרנו בעבר)

<u>אלגוריתם:</u> בהנתן G כנ"ל נבנה את G'

-נמצא זיווג מקסימלי בG' ע"י האלגוריתם מהתרגיל הקודם.

-נמצא מ־G את כל הקשתות שלא בזיווג

-ונחזיר מה שנשאר

<u>נכונות:</u> **טענה:** *יש התאמה $1:1$ בין זיווגים בגודל k ב־G' לבין כיסויים במסלולים של G עם $n-k$ מסלולים.*

הוכחה: בהינתן זיווג בגודל k בגרף G' נבנה כיסוי במסלולים ב־G: הקשתות יהיו בדיוק הקשתות בזיווג. נשים לב ששאחרי מחיקת הקשתות שלא בזיווג, נקבל גרף שבו לכל צומת v מתקיים : $d_{in}(v),d_{out}\leq 1$. גרפים כאלו הם גרפים שכל רכבי הקשירות שלהם הוא מסלול או מעגל, אבל נתון שב־G אין מעגלים, ולכן זה אכן כיסוי ע"י מסלולים.

אם הזיווג בגודל k אז יש k קשתות סה"כ בכל המסלולים. אם $P_1,\ldots,P_k$ הם המסלולים, מתקיים ש־ $\star\star=\sum|V(P_i)=n,\star=(\sum|V(P_i)|)-1=k$

$$n-k=\star-\star\star=\text{num of path}$$

תרגיל ליים

<u>עוצר:</u>

<u>סיבוכיות:</u>

13 תרגול - אלגוריתמים אריתמטים

13.1 תרגיל

שאלה: לתת אלגוריתם שבהינתן $a,b,m \le n$ כולם ב-$\mathbb{Z}$. ומחשב את $a^b modm$

תשובה: אלגוריתם:

```
Answer:
    Def c=0,d=1
    Find b=<b[k],b[k-1],...,b[1] #binary form of b
    for i=k to 0:
        c=2c
        d=d*d mod (m)
        if b[i]=1:
            d=d*a mod(n)
            c=c+1
    return d
```

לדוגמה: $3^5 mod(7)$, אז הייצוג הבינארי של 5 הוא 101 .

נכונות: נטען שבסוף האיטרציה ה-i של הלולאה בשורה 3 מתקיים : $c=<b_k b_{k-1}\dots b_1>$ ו$d=a^c modm$
נוכיח זו באינדוקציה על i:

בסיס: עבור $i=k$: $c=1=<b_k>$ ו, $d=amodm=a^cmodm$

צעד: נניח שהטענה נכונה עבור האיטרציה הi ונוכיח עבור $i-1$.

בתחילת האיטרציה מהנחת האינדוקציה $c=<b_k\dots b_i>$ ו$d=a^cmodm$, אחרי שורה 4 $c=<b_k\dots b_i0>$ ואחרי שורה 5 $d=a^{2\cdot<b_k\dots b_i>}modm$.

אם $b_{i-1}=0$ סיימנו והטענה מתקיים. אחרת, $b_{i-1}=1$ התנאי בשורה 6 מתקיים, ולכן אחרי שורה 7 מתקיים $d=a^{c+1}modm$ ואחרי שורה 8
$c=<b_k\dots b_i0>+1=<b_k\dots b_{i-1}>c+1$ והטענה מתקיימת.

בסוף הלולאה כאשר $i=0$ מתקיים $c=<b_k\dots b_0>=b$ ו$d=a^cmodm=a^bmodm$ לכן האלגוריתם מחזיר את הדרוש

עוצר: הלולאה סופית

סיבוכיות:

שורה 4 - קבוע

שורה 5,8 - $(logn)^2+(logn)^2$

כל השאר קבוע

הלולאה קורת $logn$ פעמים, ולכן בסה"כ הסיבוכיות $(logn)^3$

13.2 תרגיל

תרגיל: לפתור את המשוואה $x^2 = 1mod85$, כאשר נתון $85 = 5 \cdot 17$

פתרון: ממשפט השאריות הסיני מתקיים $x^n = 1mod85 \iff \begin{cases} x^2 = \pm 1mod17 \\ x^2 = \pm 1mod15 \end{cases} \iff \begin{cases} x = \pm 1mod17 \\ x = \pm 1mod15 \end{cases}$.

84 - $(-1,-1)$, **69** - $(1,-1)$,**16** - $(-1,1)$, **1** - $(1,1)$