מבני נתונים – תרגיל 5

תאריך פרסום:

תאריך הגשה:

מרצה ומתרגל אחראים: צחי רוזן, תומר כהן

נהלי הגשת עבודה:

- את העבודה יש להגיש בזוגות.
- את הפתרון לעבודה זו עליכם לכתוב בקובץ word (או כל כתבן אחר לפי טעמכם האישי), עבודות הכתובות בכתב יד לא יתקבלו.
- את הקובץ יש להגיש למערכת ההגשה (Submission System).

שאלות לגבי העבודה יש להעלות בפורום של הקורס או בשעות הקבלה של המרצה או המתרגל האחראיים על העבודה.

נושאים:

- Heap.
- Graphs
- Sorting
- MST
- Union Find

הערה כללית לגבי השאלות בעבודה:

- שימו לב כי בכל שאלה בה אתם מתבקשים להציע מימוש למבנה נתונים התומך בפעולות נתונות, עליכם לתת תיאור של מבנה הנתונים (האם למשל יורכב ממערך, שתי מחסניות, תור של רשימות ומשתנה וכד'...). בנוסף, עליכם לתאר את האלגוריתמים לביצוע כל אחת מהפעולות המבוקשות.

---

1. **<u>מערך כמעט ממוין עם טווח k</u>** הוא מערך שכל איבר בו נמצא במרחק של לכל היותר k מקומות מהמקום בו הוא אמור היה להיות לו המערך היה ממוין.

לדוגמא המערך הבא הוא כמעט ממוין בטווח אחד.

2	1	3	5	4	7	6	9	8	10

שכן, לו המערך היה ממוין (למטה), כל איבר בו היה במרחק של לכל היותר מקום אחד ממקומו במערך הממוין.

1	2	3	4	5	6	7	8	9	10

תארו אלגוריתם למיון מערך כמעט ממוין בגודל n עם טווח k בזמן $O(nlog(k))$.

פתרון:

נעשה זאת בעזרת ערימה.

האלגוריתם:

א. איתחול i =1.
ב. הכנס את k האיברים הראשונים לערימת מינימום.
ג. כל עוד הערימה לא ריקה:
   a. הוצא את האיבר המינימאלי מהערימה והכנס למערך במקום i.
   b. אם $i + k \le n$: הכנס את A[i+k] לערימה.
   c. $i = i + 1$.

ניתוח זמן ריצה:

בניית הערימה - O(k)

הכנסת והוצאת איבר מערימה : O(log(k))

חזרה על הלולאה n פעמים.

סה"כ: O(n*log(k)+k) = O(nlog(k))

2. נתון גרף מכוון $G = (V, E)$. תארו אלגוריתם אשר בהינתן קדקודים $s, t \in V$ מוצא ב-G מסלול מ-s ל-t שאורכו מודול 3 שווה לאפס. זמן הריצה הנדרש: $O(E + V)$. במידה ולא קיים מסלול כזה, על האלגוריתם להדפיס הודעה מתאימה.

פתרון:

האלגוריתם יהיה דומה ל-BFS המתחיל בקדקוד s עם השינויים הבאים:

- בכל קדקוד v נחזיק מערך A בגודל 3 מאותחל באפסים. האינדקסים במערך יתחילו באפס. אם באיזשהו שלב A[i] יהיה שווה ל- 1, אזי יש מסלול מ- s ל- v שאורכו מודולו 3 שווה ל- i.
האלגוריתם בודק אם באיזשהו שלב A[0] של הקדקוד t מקבל את הערך 1. אם כן יש מסלול מ- s ל- t שאורכו מודולו 3 שווה 0, אחרת – אין.
- בנוסף, נחזיק בכל קדקוד v שדה d שיציין את אורך המסלול הנוכחי מודולו 3 מ- s ל- t בסריקת ה-DFS שלנו.

האלגוריתם:

```
let Q be an empty queue
s.d = 0
s.A[s.d] = 1
Q.enqueue(s)
while Q is not empty
  v ← Q.dequeue()
  for all edges u in G.adj(v) do
    if u.A[(v.d+1)%3] = 0
      u.d = (v.d+1)%3
      u.A[u.d] = 1
      Q.enqueue(u)
```

ניתוח זמן ריצה:

כל קדקוד יכנס לתור לכל היותר 3 פעמים ובכל פעם עוברים על כל השכנים שלו לכן בדומה לBFS זמן הריצה יהיה O(V+E)

3. A הוא מערך של n מספרים שלמים. ידוע שA מורכב מ- k רצפים של מספרים עוקבים וש- k ≤ log(n).
כלומר קיימות k קבוצות מספרים שלמים מהצורה:

$S_1 = \{a_1, a_1+1, a_1+2, \dots, a_1+n_1\}$
$S_2 = \{a_2, a_2+1, a_2+2, \dots, a_2+n_2\}$
$\dots$
$S_k = \{a_k, a_k+1, a_k+2, \dots, a_k+n_k\}$

כך ש- n1+n2+…+nk=n ו- A מכיל את איחוד הקבוצות $S_1 \dots S_k$.
שימו לב. הקבוצות לא בהכרח זרות והן אינן מופיעות בהכרח כרצפים ממוינים במערך A. אם ערך מסוים מופיע במספר קבוצות, הוא יופיע במערך A מספר פעמים כמספר הקבוצות בהן הוא מופיע.
למשל המערך A=[11, 3, 2, 13, 4, 6, 12, 5, 4] מקיים את התנאים הנדרשים עבור k = 3 והקבוצות $S_1 = \{2,3,4,5\}$, $S_2 = \{11,12,13\}$ ו- $S_3 = \{4,5,6\}$
הציעו אלגוריתם למיון A בזמן ריצה קצר ככל האפשר, והוכיחו חסם הדוק לזמן הריצה של האלגוריתם שלכם.

פתרון:

כדי לפתור את השאלה נשים לב שכל מערך Arr של O(n) מספרים שלמים אשר מקיים:

$$(*)\max(Arr) - \min(Arr) = O(n)$$

ניתן למיון בעזרת מיון מניה בזמן O(n) (פשוט צריך להחסיר מכל איבר ב- Arr את $\min(Arr) - 1$).

בפתרון נשתמש בעובדה זו ונמיין בעזרת מיון מניה קבוצות של מספרים המקיימות את התנאי .

נסתכל על המערך A . נניח ללא הגבלת הכלליות כי $\max(A) \in S_k$. אברי הקבוצה S_k מקיימים את התנאי (*) . ניתן למצוא ב- A את כל אברי S_k בזמן O(n) ע"י כך שנמצא את $\max(A)$ ואת כל האיברים הגדולים שווים מ- $\max(A) - n$ בזמן O(n). את האיברים האלה אנחנו יכולים למיין במיון מניה בזמן O(n) . אחרי שעשינו זאת אנו יכולים לחזור על הפעולה שוב ושוב בזמן O(n) , כאשר בכל פעם אנחנו מפרידים וממיינים קבוצת איברים המכילה בתוכה את הקבוצה Si אשר מכילה את האיבר המקסימלי מבין האיברים שנותרו ב- A .

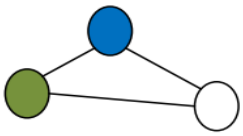
```
Sort-Them(A)
        i = 0
        While A is not empty do
                i=i+1
                Put all the numbers in A in the [max(A) − n … max(A)] range in array A_i
                A = New array containing A\A_i
                A_i = A_i sorted in O(n) time using modified counting sort.

        C = new array containing in order { A_i , A_{i−1} … , A_1 }
```

האלגוריתם בבירור ממיין את A משום שהוא בכל פעם לוקח קבוצה שמכילה את כל המספרים מעל גודל מסוים ,ממיין אותם, ובסוף מסדר את המערכים הממוינים לפי טווחי הערכים.

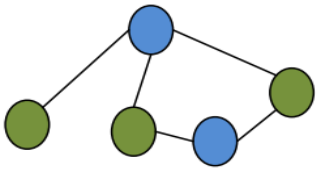
ניתוח זמן ריצה:

האלגוריתם רץ ב- O(nk) זמן משום שלולאת ה- While נפטרת בכל איטרציה לפחות מאחת מהקבוצות Siולכן רצה k פעמים לכל היותר, וגוף לולאת ה- While מורכב ממספר סופי של פעולות שזמן הריצה שלהם חסום ע"יO(n) .

4. גרף לא מכוון $G = (V, E)$ הוא 2 צביע אם ניתן לצבוע את קדקודיו בשני צבעים כך שלכל צלע $(u, v) \in E$ מתקיים $color(u) \neq color(v)$
דוגמא:



גרף זה לא 2 צביע כי לא ניתן לצבוע את הקודקוד הלבן בירוק או כחול כך שהצביעה תהיה חוקית



גרף זה 2 צביע, כי מצאנו צביעה חוקית.

א) הראו שאם גרף מכיל מעגל באורך אי זוגי אזי גרף זה לא 2 צביע.
ב) הראו שגרף לא מכוון וקשיר חסר מעגלים (עץ) הינו 2 צביע . תנו הסבר מפורט . הציעו אלגוריתם שצובע עץ בשני צבעים: לבן ושחור. מה זמן ריצתו
ג) הציעו אלגוריתם שמקבל גרף לא מכוון וקשיר וקובע האם הגרף הינו 2 צביע או לא.

פתרון:

א) נניח בשלילה כי ניתן לצבוע את הגרף ב-2 צבעים, נביט על מעגל $v_1, v_2, \dots v_{2k+1}$ אי זוגי בגרף. v_1 צבוע בצבע 1, מכיוון ש$(v_1, v_2) \in E$, v_2 חייב להיות צבוע בצבע 2, v_3 בצבע 1 וכך הלאה. מכיוון שאורך המעגל הוא אי-זוגי נקבל כי v_{2k+1} צבוע בצבע 1, אך $(v_1, v_{2k+1}) \in E$ לכן הצביעה אינה חוקית (כלומר סתירה).
ב) נראה זאת באינדוקציה:

בסיס: לכל $n \leq 2$ הטענה ברורה.

הנחה: נניח כי ניתן לצבוע עץ בעל n-1 קודקודים

צעד: נביט על T עץ בעל n קודקודים.

מכיוון ש T הוא עץ קיים לו לפחות עלה אחד (בעל דרגה 1) נסמנו v_i , נביט על $T' = T/v_i$,

T' עדיין עץ והוא בעל n-1 קודקודים, לכן לפי ההנחה ניתן לצבוע אותו ב-2 צבעים.

כעט נביט על v_i, v_i הוא עלה ולכן מחובר רק לקודקוד אחד, נצבע את v_i בצבע ההפוך לקודקוד שהוא מחובר אליו.

האלגוריתם: נפעיל BFS, לכל קודקוד שמרחקו זוגי מהשורש ניתן את הצבע 1, ולכל קודקוד שמרחקו איזוגי ניתן את הצבע 2.

נכונות האלגוריתם : נניח בשלילה שחזרה צביעה לא חוקית – נניח בה"כ כי קיימת קשת בין v_i מרמה אי זוגית לv_j מרמה אי זוגית, לכן קיים מסלול באורך זוגי מהשורש ל v_j וגם מסלול זוגי לv_j (עד לv_i ואז לv_j) לכן קיים מעגל בגרף – סתירה.

זמן ריצה: BFS = O(V+E), לעבור על קודקודים O(V).

סה"כ – O(V+E)

ג) האלגוריתם יהיה BFS עם שינוי קל: כאשר נקבע את המרחק, שנסמנו d, של קודקוד מהשורש, נבדוק את כל השכנים של הקודקוד, אם קיים שכן שגילינו כבר בסריקה שמרחקו r (r<d) כך ש d%2 = r%2 נחזיר שאי אפשר לצבוע עם 2 צבעים, אחרת נמשיך בסריקת ה BFS, אם הגענו לסוף הסריקה נחזיר שניתן לצבוע את הגרף עם 2 צבעים.

נכונות האלגוריתם: קל לראות כי אם יש מסלול זוגי ואי זוגי מהשורש לקודקוד מסוים קיים מעגל אי זוגי בגרף, ולכן לא צביע. אם סיימנו את האלגוריתם אז ניתן לכל קודקוד v_j את הצבע $distance(v_j)\%2$ ומכיוון שאין שכנים שהמרחק שלהם מהשורש מודולו 2 שווה קיבלנו צביעה חוקית.

5. קוטר של גרף הינו אורך המסלול הפשוט הארוך ביותר בגרף (מסלול פשוט – מסלול ללא חזרות על קדקודים, להוציא אולי את הקדקודים בשתי קצוות המסלול). תארו אלגוריתם שבהינתן **<u>עץ</u>** מחזיר את קוטר הגרף. זמן ריצה נדרש: $O(E+V)$

פתרון:

ראשית נשים לב לעובדה כי אם המרחק בין v_i ל v_r הוא המרחק הגדול ביותר בגרף, כאשר נפעיל את האלגוריתם BFS על v_i או v_r בתור שורש נקבל כי המרחק המקסימאלי שהאלגוריתם יתן הוא הקוטר.

האלגוריתם:

a. בחר קדקוד v_j בגרף.
b. הפעל BFS מv_j
c. מצא את הקדקוד u הרחוק ביותר מ-v_j
d. הפעל BFS מ u
e. החזר את המרחק הגדול ביותר.

נכונות האלגוריתם: נניח כי המרחק המקסימאלי הוא בין v_i ל v_r ונניח כי האלגוריתם מצא כי u הוא הרחוק ביותר מv_j נראה כי המרחק בין u ל v_r או בין v_i ל u הוא הגדול ביותר.

נביט על המסלול מהשורש לu נסמנו $r, u_1, u_2 \dots u$, ועל המסלול בין v_i ל v_r נסמנו $v_i, v_2 \dots v_r$,

נשים לב כי $\exists i,j: u_i = v_j$, (מכיוון שאחרת נוכל לקבל מסלול ארוך יותר מהמסלול בין v_i ל v_r).

בה"כ v_j יותר קרוב ל v_i, נבחין כי במקרה זה המסלול $u_i \dots u$ ארוך לפחות כמו $v_i, v_2 \dots v_j$, (כי אחרת נקבל מסלול ארוך יותר מהשורש בניגוד להנחה), לכן המסלול $u \dots u_i, v_j \dots v_r$ הוא לפחות באורך המסלול בין v_i ל v_r ולכן נקבל שהמרחק בין u ל v_r הוא הגדול ביותר בעץ.

זמן ריצה:

הפעלה פעמיים של BFS : O(E+V), חיפוש קדקוד עם מרחק מקסימאלי: O(V). סה"כ: O(E+V)

<u>העשרה:</u>

6. נאמר שגרף $G = (V, E)$ עם פונקציית משקל $w: E \to R$ על הקשתות הוא **<u>גרף מיוחד</u>** אם יש ל-G עץ פורש מינימלי (MST) יחיד.
תארו אלגוריתם המקבל כקלט גרף $G = (V, E)$ עם פונקציית משקל $w: E \to R$ על הקשתות ובודק אם G הוא גרף מיוחד.
על האלגוריתם להחזיר את העץ הפורס המינימלי היחיד שלG , אם G הוא גרף מיוחד, או אחרת על האלגוריתם להודיע שהגרף אינו מיוחד.
זמן הריצה הנדרש: $O(Elog(V))$.

פתרון:

גרף G הוא מיוחד אם לא קיים עץ פורש מינימאלי נוסף.

האלגוריתם:

```
A = ∅
curr = -∞
foreach (u,v) in G.E:
      (u,v).color = white
foreach v ∈ G.V:
   MAKE-SET(v)
foreach (u, v) ordered by weight(u, v), increasing:
    if weight(u,v) > curr:
      curr = weight(u,v)
      foreach (u',v') s.t. weight(u,v)=weight(u',v'):
         if FIND-SET(u') ≠ FIND-SET(v'):
           (u',v').color = grey
    if FIND-SET(u) ≠ FIND-SET(v):
      (u,v).color = white
        A = A ∪ {(u, v)}
        UNION(u, v)
```

```
foreach (u,v) in G.E:
      if (u,v).color = grey:
            return false
return A
```

נכונות האלגוריתם: ניזכר כי האלגוריתם של קרוסקל כל פעם בוחר את הקשת הקלה ביותר שאפשר להוסיף. נשים לב כי אם קיימות 2 קשתות באותו משקל ששתיהן בטוחות בעת הוספת אחת מהקשתות, אך לאחר הוספת אחת מהקשתות, הקשת השניה אינה בטוחה, האלגוריתם של קרוסקל היה נותן פתרון אחר אם היינו בוחרים את הצלע השניה. ולכן קיים עוד עץ פורש כלומר G איננו מיוחד.
כעט נותר להראות כי אם יש 2 עצי MST (שאחד מהם נוצר ע"י קרוסקל) אז קיימות 2 צלעות כנ"ל.
יהי T_1, T_2 שני העצים, תהי e_2 הצלע הקלה ביותר ב T_2 שאינה ב T_1 וe_1 הצלע הקלה ביותר ב T_1 שאינה בT_2.
נראה כי משקל e_1 שווה למשקל e_2. נניח בשלילה שהם לא שוות, בה"כ e_2 היא הקלה יותר. נוסיף את e_2 ל T_1 כעט יש מעגל ב T_1 נשים לב כי לא יכול להיות כי כל הצלעות במעגל קטנות ממש מe_2 כי אז קיימת צלע במעגל שלא שייכת ל T_2 ואז נקבל סתירה להנחה שe_2 היא הקלה יותר, בנוסף לא יכול להיות שיש צלע יותר כבדה מe_2 כי אז נוציא אותה ונקבל עץ ששוקל פחות. לכן קיימת צלע במעגל ששוקלת כמו e_2 ואינה ב T_2 בסתירה לכך שe_1 הצלע הקלה ביותר ב T_1 שאינה בT_2.

ניתוח זמן ריצה: האלגוריתם מבוסס על קרוסקל לכן די שננתח רק את מה שהוספנו.

הלולאה בשורה 2 עוברת על כל הקשתות – O(E)

הלולאה בשורה 9 עוברת על כל קשת לכל היותר פעם אחת וכל הפעלה לוקחת log(V) לכן סה"כ– O(Elog(V))

הלולאה בשורה 16 עוברת על כל הקשתות – O(E)

לכן סה"כ זמן ריצה : זמן של קרוסקל + O(E) + O(E) = O(Elog(V))

7. תארו אלגוריתם הממיין n נקודות שנדגמו בצורה אחידה (יוניפורמית) ממישור היחידה באורך n (המלבן שהנקודה השמאלית העליונה שלו היא (0, 1) והנקודה הימנית התחתונה שלו היא (n, -1)). על המיון למיין את הנקודות לפי מרחקן מראשית הצירים. על המיון לעבוד בזמן ממוצע (צפי) של O(n).

רמז: חישבו איך אפשר לחלק את מישור היחידה לקטעים רלוונטיים למיון, כך שהשטח של כל אחד מהם (להוציא אולי קטע אחד) הוא סדר גודל של 1/n מהשטח הכולל של מישור היחידה בגודל n (כפי שהוגדר למעלה)

פתרון:

נשתמש במיון דליים כאשר החלוקה לדליים היא לפי המרחק מהראשית. נקודה שנמצאת בתוך מעגל היחידה שייכת לדלי מס' 1. נקודה שנמצאת בתוך מעגל בעל רדיוס i אבל מחוץ למעגל בעל רדיוס i-1 שייכת לדלי מס' i . סה"כ n+1 דליים. כל שצריך להראות הוא שהשטח שכל דלי מכסה (להוציא הדלי שמכסה את השטח הרחוק ביותר מהראשית), הוא בסדר גודל של 1/n מהשטח שממנו נדגמו הנקודות. אבל. גודל השטח ממנו נדגמו הנקודות הוא בגודל של 2*n, ושטח כל דלי אינו עולה על 2*2=4 ואינו נופל מ- pi/4 ולכן בסדר גודל של 1/n.