<u>מבני נתונים תשפ"א – פתרון 9</u>

<u>שאלה 1</u> בהינתן ייצוג של גרף מכוון על ידי רשימות שכנויות (כלומר, רשימה מקושרת של הצמתים, כאשר לכל צומת יש מצביע אל רשימה מקושרת של כל השכנים שלו), כמה זמן דרוש לחישוב דרגת היציאה של כל צומת? כמה זמן דרוש לחישוב דרגת הכניסה?

<u>פתרון</u>

- חישוב דרגת יציאה- דרגת היציאה של צומת שווה למספר השכנים שלו. לכן, כדי לחשב את דרגת היציאה של כל צומת בגרף, עלינו לעבור על הרשימה המקושרת של הצמתים, ועבור כל צומת לעבור על הרשימה המקושרת של שכניו ולספור כמה איברים יש בה. בסך הכל עוברים פעם אחת על כל הצמתים ופעם אחת על כל הקשתות ולכן הזמן הדרוש הוא $\Theta(|V|+|E|)$.
- חישוב דרגת הכניסה – כאשר יש קשת הנכנסת מצומת x לצומת y, אז y יופיע ברשימת השכנים של x. מכאן, שדרגת הכניסה של צומת y תהיה שווה למספר הרשימות המקושרות של שכנים בהם y מופיע. גם חישוב זה ניתן לבצע בעלות של $\Theta(|V|+|E|)$ אם נשמור בצד מערך A בגודל $|V|$, וכל פעם שנראה ש- y הוא שכן של צומת מסוים נבצע A[y]++. בסוף המעבר על כל הצמתים והקשתות מערך A יכיל את דרגת הכניסה של כל צומת.

<u>שאלה 2</u> **הגרף המוחלף (transpose)** של גרף מכוון $G=(V,E)$ הוא הגרף $G^T=(V^T,E)$, כאשר $E^T=\{(v,u)\in V\times V \mid (u,v)\in E\}$. כלומר, G^T מתקבל מ- G על ידי היפוך כיווני כל הקשתות ב- G.

תארו אלגוריתמים יעילים לחישוב G^T בהינתן G, הן עבור ייצוג G על ידי מטריצת שכנויות והן על ידי ייצוג G על ידי הייצוג המעורב. נתחו את זמני הריצה של האלגוריתמים.

<u>פתרון</u> <u>אלגוריתם לחישוב G^T בייצוג של מטריצת שכנויות:</u>

מכיוון ואנחנו לא יודעים באילו מיקומים במטריצה המייצגת את גרף G יופיע 1, אין ברירה אלא לסרוק את כל המטריצה המייצגת את G.

האלגוריתם:

1. ניצור מטריצה G^T בגודל $n\times n$ ונמלא אותה באפסים.
2. באמצעות שתי לולאות for מקוננות שרצות מ- 1 ועד n, נעבור על כל אחד מהתאים במטריצה G:
   2.1. אם $G[i,j]==0$ נמשיך לתא הבא.
   2.2. אם $G[i,j]==1$, נגדיר ש- $G^T[j,i]=1$, כלומר בגרף המוחלף יש כעת קשת מ- j ל- i, במקום מ- i ל- j כמו בגרף המקורי.

נכונות: חישבנו את הגרף המוחלף, לפי הגדרה.

עלות: זמן הריצה של האלגוריתם הוא $\Theta(n^2)$ כי סרקנו את כל המטריצה המקורית.

<u>אלגוריתם לחישוב G^T בייצוג המעורב:</u>

כדי ליצור את גרף G^T בייצוג המעורב, עלינו לשנות את רשימות השכנים.

האלגוריתם:

1. ניצור מערך G^T בגודל n.
2. נעבור על כל אחד מהתאים במערך G:
   2.1. נעבור על כל אחד מהשכנים של הקודקוד הנוכחי v:

2.1.1. נסמן את השכן הנוכחי שמצאנו כ- u. כעת נכניס את v לראש רשימת השכנים של u במערך G^T:

$$v \to next = G^T[u] \to next$$
$$G^T[u] \to next = v$$

נכונות: עברנו על כל הקשתות בגרף המקורי, וחישבנו את הגרף המוחלף, לפי הגדרה.

עלות: עברנו על כל אחד מהקודקודים בגרף, $\Theta(|V|)$, ולאחר מכן עברנו על כל אחת מהקשתות ב- G ויצרנו את הקשת ההפוכה ב- G^T, $\Theta(|E|)$, לכן זמן הריצה הכולל של האלגוריתם הוא $\Theta(|V| + |E|)$.

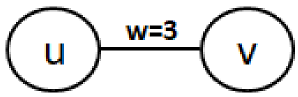
__שאלה 3__ נתון גרף קשיר לא מכוון. לכל קשת (u, v) משויך משקל שלם $w(u, v)$ שערכו בין 1 לבין קבוע k. תארו אלגוריתם שזמן הריצה שלו הוא $O(|V| + k|E|)$ אשר בהינתן קודקוד s מוצא את המסלול בעל המשקל הכולל הקטן ביותר מ- s לכל קודקוד אחר בגרף. (שימו לב לנתח את זמן הריצה ולהראות שהוא אכן כפי הנדרש.)

__פתרון__

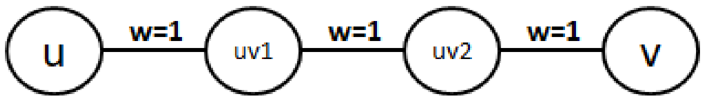
כדי לחשב את המסלול בעל המשקל הכולל הקטן ביותר מקודקוד u לכל קודקוד אחר בגרף G, נבנה גרף חדש G^w באופן הבא: עבור כל קשת (u, v) בגרף המקורי G בעלת משקל $w > 1$, נפצל את הקשת ל- w קשתות שונות, כלומר נוסיף $k = w - 1$ קודקודים חדשים בין u ל- v.

לדוגמא:

הקשת (u, v) בעלת משקל 3 בגרף המקורי:



תיראה כך בגרף G^w:



כעת. נפעיל את פונקציית $BFS(G^w, u)$ וכתוצאה מכך המרחק d שיחושב עבור כל קודקוד יהיה שווה למשקל הכולל הקטן ביותר מקודקוד u בגרף המקורי G.

__עלות__: נשים לב שבמקרה הגרוע ביותר, המשקל של כל קשת בגרף המקורי הוא k, ולכן בגרף G^w יהיו $k|E|$ קשתות, ומכאן שסיבוכיות האלגוריתם תהיה $O(|V| + k|E|)$.