אלגוריתמים בתורת הגרפים – תרגול מס׳ 1 (חורף 01׳)

הקדמה - סיבוכיות

חזרה קצרה על סימוני סיבוכיות של פונקציה f(n):

O(g(n)) – חסם עליון לסיבוכיות – קיימים מספרים חיוביים $n_0 c,$ כך שלכל $n > n_0$:

$$f(n) \le c \cdot g(n)$$

Θ (g(n)) – חסם עליון ותחתון לסיבוכיות – קיימים מספרים חיוביים c_1, c_2, n_0 כך שלכל

$$c_1 \cdot g(n) < f(n) < c_2 \cdot g(n) \ : n > n_0$$

הגדרות

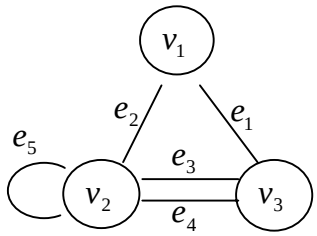
גרף $G(V,E)$ הינו מבנה המורכב מקבוצת צמתים $V = \{v_1, v_2, ..., v_n\}$ וקבוצת קשתות $E = \{e_1, e_2, ..., e_m\}$, כאשר כל קשת $e \in E$ פוגעת בזוג צמתים $e = (v_i, v_j);\ \ v_i, v_j \in V$.

הקדקדים v_i, v_j נקראים צמתי הקצה של הקשת, ושכנים זה לזה.

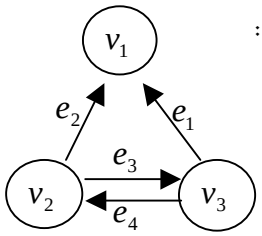
בגרף מכוון כל קשת $e = (v_i \to v_j)$ והמשמעות היא שהקשת יוצאת מצומת v_i ונכנסת ל- v_j.

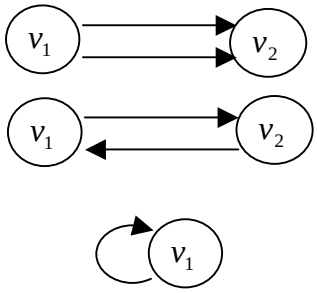
נהוג לשרטט גרף בצורה גרפית באופן הבא:

גרף לא-מכוון:



גרף מכוון:





קשתות מקבילות הן קשתות שצמתי הקצה שלהן זהים. בגרף מכוון ניתן להבחין בין קשתות מקבילות ולבין קשתות אנטי-מקבילות.

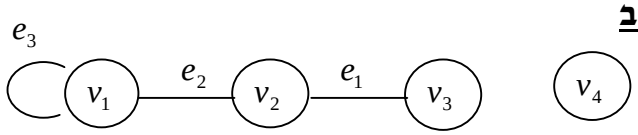
קשת עצמית (לולאה עצמית) היא קשת ששני צמתי הקצה שלה זהים.

בגרף פשוט אין קשתות מקבילות או קשתות עצמיות.

האפליקציות של גרפים רבות ומגוונות – מיישומים טריוויאלים כגון מערכת כבישים, או תכנית טיסות ועד לשימושים מורכבים כגון מיפוי גנים (כאשר כל צומת היא רצף גנטי והצלעות מייצגות את רמת הדמיון בין הרצפים). תורת הגרפים הינה מפותחת מאד ולכן קיים מאמץ לייצג בעיות רבות בצורה של גרף, דבר המאפשר שימוש במגוון הכלים הרחב שכבר קיים עבורם.

ייצוג גרפים במחשב

דוגמא:



מטריצת שכנויות

במקום ה- (i, j) נסמן:

1 – אם יש קשת בין הצומת v_i לצומת v_j

0 - אם אין קשת בין הצומת v_i לצומת v_j

	v_1	v_2	v_3	v_4
v_1	1	1	0	0
v_2	1	0	1	0
v_3	0	1	0	0
v_4	0	0	0	0

בגרף לא מכוון המטריצה סימטרית ולכן מספיקה רק חצי מטריצה – משולש תחתון או עליון.

בגרף מכוון יש צורך במטריצה השלמה.

סיבוכיות מקום: $O(|V|^2)$

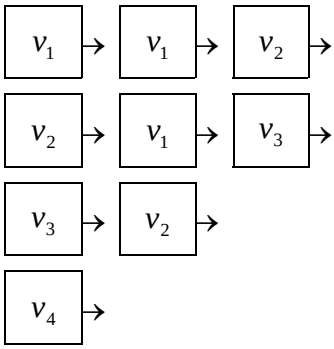
ניתן להשתמש במספרים שונים מ-0,1 במטריצה לשימושים שונים. לדוגמא, משקולות על הקשתות. בגרף לא-פשוט המספרים יכולים לייצג את מספר הקשתות בין שני הצמתים.

בייצוג מטריציוני נוח להשתמש כאשר בגרף יש הרבה קשתות. כמו-כן, ניתן להשתמש בפעולות מטריציוניות לביצוע חישובים על הגרף. לדוגמא, מה מייצגות המטריצות G^2, G^n?

רשימת שכנויות

כאשר הגרף דליל, השימוש במטריצה הינו בזבזני במקום, וגם בזמן הלוקח לסרוק אותו (נדבר על כך יותר בהמשך).

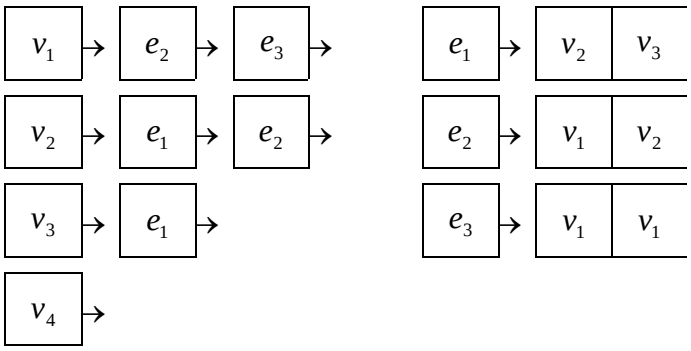
ברשימת שכנויות שומרים, עבור כל צומת v_i, רשימה של כל הצמתים השכנים לו (בגרף לא מכוון). בגרף מכוון נהוג לשמור רק את הצמתים אליהם יוצאת קשת מהצומת v_i (כלומר כל צומת v_j שקיימת בגרף קשת מכוונת $v_i \to v_j$), אך אפשריות גם גישות אחרות. מבנה הנתונים המקובל לצורך זה הוא רשימה מקושרת (linked list).



אפשרות נוספת היא לשמור במבנה הנתונים:

1. עבור כל צומת, רשימת של כל הקשתות הפוגעות בה (בגרף לא מכוון) או של הקשתות היוצאות ו/או נכנסות שלה (בגרף מכוון).
2. עבור כל קשת, את צמתי הקצה שלה.

מבנה זה מאפשר לשמור מידע נוסף עבור כל אחת מהקשתות.



<u>סיבוכיות מקום</u>: $O(|V|+|E|)$

<u>ניתוח סיבוכיות זמן (עבור שימושים נפוצים)</u>

1. בהנתן צומת v – מצא את כל שכניו

<u>במטריצת שכנויות</u>: מעבר על כל השורה במטריצה - $\Theta(|V|)$

<u>ברשימת שכנויות</u>: מציאת הצומת v ($O(|V|)$) ומעבר על כל שכניו ($O(|V|)$). סה"כ: $O(|V|)$

2. האם קיימת קשת בין צומת v_i וצומת v_j ?

<u>במטריצת שכנויות</u>: בדיקת המקום ה- (i, j) - $O(1)$

<u>ברשימת שכנויות</u>: מציאת הצומת v ($O(|V|)$) ומעבר על כל שכניו ($O(|V|)$). סה"כ: $O(|V|)$

<u>דוגמא לשימוש ברשימת שכנויות</u>

ביצוע "סריקה" מצומת v:

<u>אתחול</u>: סמן את כל הקשתות כפנויות (ניתן להוסיף flag מתאים לרשימת הקשתות).

$v \rightarrow v_{\text{current}}$

<u>לולאה</u>: עבור על רשימת הקשתות היוצאות מ- v_{current}. אם קיימת קשת פנויה $e = (v_{\text{current}}, u)$:

$u \rightarrow v_{\text{current}}$

סמן את e כתפוסה

האלגוריתם ייעצר כאשר יגיע לצומת שכל הקשתות היוצאות ממנה כבר תפוסות.

<u>המטרה</u>: מציאת מסלול מצומת v אשר לא ניתן להאריכו (לפחות לא באופן פשוט, יתכן שיש בגרף מסלולים ארוכים יותר).