אלגוריתמים קומבינטוריים
סיכומים של תרגילי כיתה מסמסטרים קודמים בנושא

DFS חיפוש לעומק
(Depth First Search)

1. נתון גרף (לא מכוון או מכוון) $G(V,E)$.

2. הרעיון של חיפוש לעומק הוא שמצומת הנוכחי מיד נתחיל לבקר בצאצאים שלו. ניסוג מצומת אחרי שניסגנו מכל צאצאיו. במשך האלגוריתם מגדירים את זמן גילוי הצומת $d[v]$ ואת זמן הנסיגה ממנו $f[v]$. נתחיל את הביקורים הצומת כלשהו, ולאחר הנסיגה ממנו, נתחיל שוב בצומת שתרם בוקר.

```
DFS(G)
1   for each vertex u ∈ V
2       do color[u] ← WHITE
3          π[u] ← NIL
4   time ← 0
    ▹ In some applications we may do the next step in a special order.
5   for each vertex u ∈ V
6       do if color[u] = WHITE
7             then DFS-VISIT(u)
```

```
DFS-VISIT(u)
1   color[u] ← GREY   ▹ White vertex u just discoverd.
2   d[u] ← time ← time + 1
3   for each v ∈ Adj[u]   ▹ Explore the edge (u, v).
4       do if color[v] = WHITE   ▹ Newly discovered.
5             then π[v] ← u
6                  DFS-VISIT(v)
7   color[u] ← BLACK   ▹ Finished exploring u's neighbors.
8   f[u] ← time ← time + 1
9a  ▹ TOPOLOGICAL-SORT(G) puts u in the front of the linked list here.
```

3. סיבוכיות: זמן הריצה של DFS הוא $O(|V|+|E|)$. נימוק: האיתחול דורש זמן $O(|V|)$ והלולאה בשורות 5 עד 7 ד- DFS דורש זמן $O(|V|)$ פרט לזמן הריצה של הקריאות של DFS-VISIT. לכל צומת v יש בדיוק קריאה אחד של DFS-VISIT(v) שרצה בזמן $O(\text{outdeg}(v)) + O(1)$. סה"כ זמן ריצה בתוך הקריאות של DFS-VISIT הוא $O(|V|+|E|)$ כי $\sum O(\text{outdeg}(v)) = O(|E|)$.

4. קשתות הגרף מחולקות לארבע סוגים:
   - (א) קשת (u,v) הוא קשת עץ (tree edge) אם דרכו התגלה הצומת v לראשונה. היער המושרה על ידי אוסף כל קשתות כאלו נקרא "עץ" ה- DFS.
   - (ב) קשת (u,v) הוא קשת אחורית (back edge) אם u הוא צאצא של v בעץ ה-DFS.
   - (ג) קשת (u,v) הוא קשת קידמית (forward edge) אם v הוא צאצא של u בעץ ה-DFS.
   - (ד) קשת שאינו מאחד הסוגים האלו נקרא קשת מוצלב (cross edge).

הערה: בחיפוס לעומק על גרף לא מכוון יש רק קשתות עץ וקשתות אחוריות.

5. תרגיל: הראה שקשת (u,v) הוא:

(א) קשת עץ אס"ם $d[u] < d[v] < f[v] < f[u]$ ו- $\pi(v) = u$. (בשלב הסריקה של הקשת (מ- u ל-v) v היתה לבנה.)

(ב) קשת קידמית אס"ם $d[u] < d[v] < f[v] < f[u]$ ו- $\pi(v) \neq u$. (בשלב הסריקה של הקשת (מ- u ל-v) v היתה שחורה ו- $d(u) < d(v)$.)

(ג) קשת אחורית אס"ם $d[v] < d[u] < f[u] < f[v]$. (בשלב הסריקה של הקשת (מ- u ל-v) v היתה אפורה.)

(ד) קשת מוצלב אס"ם $d[v] < f[v] < d[u] < f[u]$. (בשלב הסריקה של הקשת (מ- u ל-v) v היתה שחורה ו- $d(v) < d(u)$.)

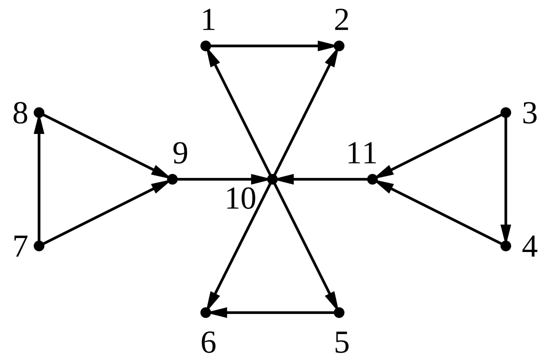
(הערה: פתרון מסתמך על משפט "הסוגריים".)

6. שימושים של DFS:

(א) מיון טופולוגי של גרף מכוון חסר מעגלים הוא סידור של הצמתים כך שכל קשת הולך משמאל (קטן) לימין (גדול). האלגוריתם הבא לחישוב מיון טופולוגי מתבסס על העובדה שבגרף מכוון חסר מעגלים אין קשתות אחוריות באף עץ DFS שלו.

```
TOPOLOGICAL-SORT(G)
  call DFS(G) to compute the finishing times f[v] for each vertex.
  as each vertex is finished, insert it into the front of a linked list.
  return the linked list of vertices.
```

דוגמה: (נניח שבשורה 5 של DFS עוברים על הצמתים בסדר עולה של מספרים, ושבשורה 3 של DFS-VISIT(u) עוברים על השכנים של u בסדר עולה של מספרים!)



v	1	2	3	4	5	6	7	8	9	10	11
$d[v]$	1	2	5	6	9	10	17	18	19	8	7
$f[v]$	4	3	16	15	12	11	22	21	20	13	14

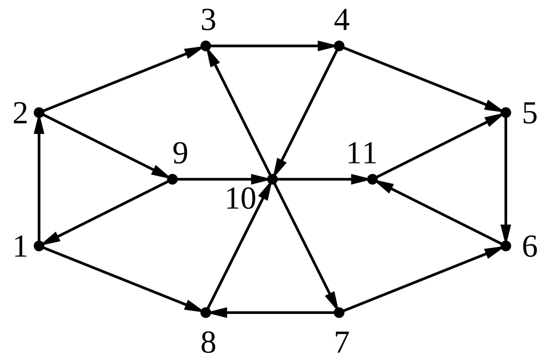
מומלץ לעבור על האלגוריתם צעד, צעד ולבדוק איך מגיעים למיספור הנ"ל.

הסדר של המיון: 7, 8, 9, 3, 4, 11, 10, 5, 6, 1, 2. בדוק שאכן כל קשת בגרף הולך משמאל לימין!

(ב) רכיבים קשירים חזק (היטב): נגדיר יחס שקילות (equivalence relation) על הצמתים בגרף מכוון לפי $v \equiv u$ אם"ם יש ב- G מסלול מכוון מ- u ל- v וגם מסלול מכוון מ- v ל- u. (בדוק שזה אכן יחס שקילות.) הגרף המושרה על ידי מחלקת שקילות נקראת רכיב קשיר חזק (היטב) של G. (במילים אחרות זהו תת-גרף מושרית על ידי קבוצת צמתים מקסימלית שקשירה היטב.) ניתן לחשב את הרכיבים (מחלקות השקילות) על ידי שימוש כפול ב- DFS. נסמן ב-G^{T} את הגרף המתקבל מ- G על ידי היפוך כיוון של כל קשתות G.

STRONGLY-CONNECTED-COMPONENTS(G)
1 call DFS(G) to compute the finishing times $f[v]$ for each vertex.
2 compute G^{T}
3 call DFS(G^{T}) considering the vertices in order of decreasing $f[v]$.
4 output the vertices of each tree in the DFS forest of step 3 as a seperate SCC.

דוגמה: (נניח שבהרצה של DFS(G) שבשורה 5 של DFS עוברים על הצמתים בסדר עולה של מספרים, ושבשורה 3 של DFS-VISIT(u) עוברים על השכנים של u בסדר עולה של מספרים!)



תוצאות DFS על G:

v	1	2	3	4	5	6	7	8	9	10	11
$d[v]$	1	2	3	4	5	6	12	13	19	11	7
$f[v]$	22	21	18	17	10	9	15	14	20	16	8

מומלץ לעבור על האלגוריתם צעד, צעד ולבדוק איך מגיעים למיספור הנ"ל. וכן לעבור גם על DFS(G^{T}).

תוצאות DFS על G^{T}:

v	1	2	3	4	5	6	7	8	9	10	11
$d[v]$	1	3	7	9	17	19	12	11	2	8	18
$f[v]$	6	4	16	10	22	20	13	14	5	15	21

הרכיבים: $\{1,2,9\},\{3,4,7,8,10\},\{5,6,11\}$