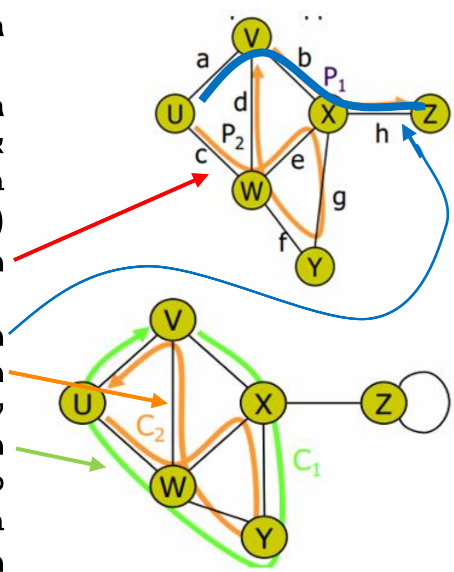
## **<u>גרפים, BFS, DFS:</u>**




**גרף** הוא זוג של 2 קבוצות: קבוצת קודקודים V וקבוצת צלעות $E \subseteq (V \times V)$.
$G=(V,E),\quad V=\{v_1,\cdots,v_n),\quad E=\{e_1,\cdots,e_m\},\ \ e_i=(v_j,v_k)$

**גרף מכוון** הוא גרף שבו לכל צלע יש כיוון מוגדר: קיום צלע מ-i ל-j לא בהכרח אומר שיש צלע מ- j ל- i.

**בגרף לא מכוון**, אין כיוון לצלעות. בגרף לא מכוון עם n קודקודים יכולות להיות עד $\binom{n}{2}$ צלעות (כל זוג צלעות).

**מסלול** בגרף הוא סדרת קודקודים $(v_1,\cdots,v_k)$ כך ש:
$1\le i\le k\le n,\quad (v_i,v_{i+1})\in E$

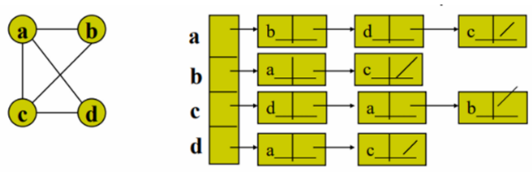
**מסלול פשוט** הוא מסלול שבו כל קודקוד מופיע לכל היותר פעם אחת.

**מעגל** הוא מסלול שמתחיל ונגמר באותו קודקוד (באמצע אפשר לחזור על קודקודים)

**מעגל פשוט** הוא מעגל שבו הקודקוד הראשון מופיע בדיוק פעמיים, וכל השאר עד פעם אחת.

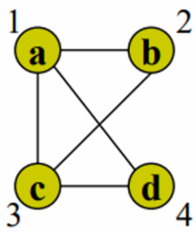
**בגרף קשיר,** $n-1\le m\le\binom{n}{2}\ \Rightarrow\ \Omega(n)\le m\le O(n^2)$

**מסלול קצר** הכוונה המסלול הקצר ביותר מ- s ל- v. נסמן $s\mapsto v$. אם אין מעגלים שליליים, כל מסלול קצר יהיה מסלול פשוט. $\delta(s,v)$ הוא אורך המסלול הקצר.




ייצוג של גרף במחשב:

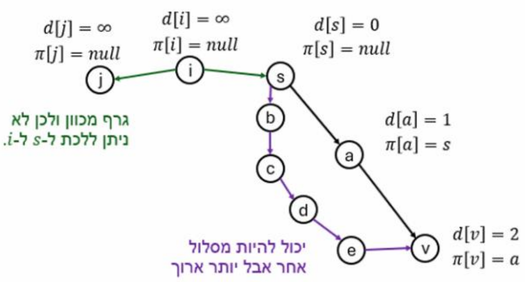
1) **רשימה מקושרת:** כל קודקוד מחזיק רשימה מקושרת של הקודקודים שיש לו צלע אליהם.
סיבוכיות מקום: $O(n+m)$, אם יש מעט צלעות אז זה יותר יעיל.
זמן החיפוש למציאת צלע: $O(maxdegree)$, לפי הקודקוד עם הכי הרבה צלעות.




| | 1 | 2 | 3 | 4 |
|---|---|---|---|---|
| 1 | 0 | 1 | 1 | 1 |
| 2 | 1 | 0 | 1 | 0 |
| 3 | 1 | 1 | 0 | 1 |
| 4 | 1 | 0 | 1 | 0 |

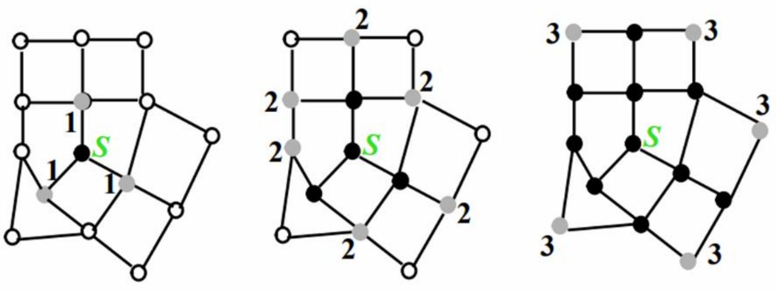
2) **מטריצת שכנויות:** טבלה $n\times n$ לפי הקודקודים.
$$M[i][j]=\begin{cases}0, & \text{no edge}\\ 1, & \text{edge } (v_i,v_j)\in E_G\end{cases}$$
בגרף לא מכוון, המטריצה תהיה סימטרית. סיבוכיות מקום: $O(n^2)$.
זמן חיפוש למציאת צלע: $O(1)$.

### **<u>שיטות למעבר על גרף:</u>**




**<u>BFS</u>:** חיפוש לרוחב, מבוסס על תור. אלגוריתם S.S.S.P – Single Source Shortest Path הכי פשוט. עובד על גרף לא ממושקל. מתחיל מקודקוד מסוים s ונותן לכל קודקוד:
$d(v)$ – מרחק מs,
$\pi(v)$ – הקודקוד הקודם ל- v, במסלול הקצר $s\longmapsto v$.
זמן ריצה: $O(n+m)$.

במהלך הריצה, לכל קודקוד יש **צבע** שמתאר את המצב שלו:
- **לבן:** עוד לא מצאנו את הקודקוד.
- **אפור:** מצאנו את הקודקוד.
- **שחור:** סיימנו את הקודקוד.




מתחילים מs. מוסיפים את השכנים שלו לתור וצובעים אותם אפור.
בכל שלב, עד שתור ריק:
נוציא את הקודקוד הראשון בתור (שנכנס ראשון).
נסמן אותו בשחור, נקבע לו את הערכים ונכניס את הילדים ה**לבנים** שלו לתור.

נגדיר: $N(v)$ = קבוצת השכנים של v.

```
BFS(G, s){
    for all v ∈ V_G:  d(v) ← ∞,   π(v) ← null,   c(v) ← white      O(n)
    c(s) ← gray,   d(s) ← 0,   Q ← s
    while( Q isn't empty ){                    O(n)
        u ← Q.pop
        for( v ∈ N(u) ){                       O(m) amortized
            if( c(v) = white ){
                c(v) ← gray
                d(v) ← d(u) + 1
                π(v) ← u
                Q.push(v)
        c(u) ← black
```

זמן ריצה: אתחול הוא O(n). הלולאת while חוזרת n פעמים, ובתוכה הלולאת for שהיא גם O(n). אבל הפעולה תקרה רק פעם אחת לכל צלע, אז בסה"כ הלולאת while היא **O(n+m)**

**הוכחת נכונות האלגוריתם – טענות:**

1) במהלך הריצה, BFS מגלה את כל הקודקודים שאפשר להגיע אליהם מ s.
2) אחרי BFS, $d(v) = \delta(s,v)$
3) לכל קודקוד שאפשר להגיע אליו מ s, המסלול הקצר $s \mapsto v$ הוא: $s \mapsto \pi(v)$ ואז הצלע $(\pi(v), v)$. (מסלול קצר הוא גם המסלול הקצר מ s לכל אחת מהצלעות בדרך).

**טענות עזר:**

1) אי-שוויון המשולש: $\delta(s,v) \le \delta(s,u) + \delta(u,v)$. ההוכחה נובעת באינדוקציה מהגדרת δ.
2) אחרי BFS, $d(v) \ge \delta(s,v)$ (זה בעצם שווה, אבל יותר קל להוכיח קטן-שווה וגם גדול-שווה ומכאן ינבע שוויון).
הוכחה באינדוקציה על סדר צביעת הקודקודים: בהתחלה $d(s) = 0$, $d(v) = \infty$, מתקיים.
נניח שמתקיים בשלב מסוים, נתבונן בקודקוד u כלשהו. נתבונן ב $v \in N(u)$ כלשהו. כשצובעים את v באפור, u עדיין אפור. מתקיים:
   1) $d(u) \ge \delta(s,u)$ מהנ"א,
   2) $d(v) = d(u) + 1$ מהגדרת האלגוריתם,
   3) $\delta(s,v) \le \delta(s,u) + 1$ מטענה 1.
בסה"כ: $\delta(s,v) \le \delta(s,u) + 1 \le d(u) + 1 = d(v)$
3) אם $(v_1 \cdots v_n) \in Q$ (בסדר הזה) אז: $d(v_i) \le d(v_{i+1})$ and $d(v_n) \le d(v_1) + 1,$
כלומר d מונוטוני בתוך Q, ויש לכל היותר הפרש של 1 מקצה לקצה.
הוכחה: באינדוקציה על אינווריאנטה של הלולאה: (בהתחלה מתקיים). נניח שמתקיים בשלב מסוים. נוציא את $v_1$, נכניס את השכנים הלבנים שלו. לכולם מתקיים $d(v_i) = d(v_1) + 1$.

מסקנה 1: אם $v_i$ הוכנס לתור לפני $v_j$, אז $d(v_i) \le d(v_j)$.

**הוכחת האלגוריתם:** נב"ש שקיים קודקוד כך ש: $d(v) \ne \delta(s,v)$. (אם יש יותר מאחד, ניקח את זה עם δ מינימלי). מטענה 2, נקבל שמתקיים $d(v) > \delta(s,v)$.
וגם, בהכרח $v \ne s$ (כי $d(s) = 0 = \delta(s,s)$ לפי הגדרה), ואפשר להגיע ל- v מ- s (כי אחרת $\delta(v) = \infty$).
נגדיר u הקודקוד הקודם ל- v במסלול הקצר $s \mapsto v$. מתקיים: $\delta(s,u) < \delta(s,v)$, וגם $d(u) = \delta(s,u)$ (כי לקחנו v עם הδ הכי מינימלי). אז: $d(v) > \delta(s,v) = \delta(s,u) + 1 = d(u) + 1 \quad (1)$
נתבונן בצבע של v ברגע ש u מוצא מהתור:

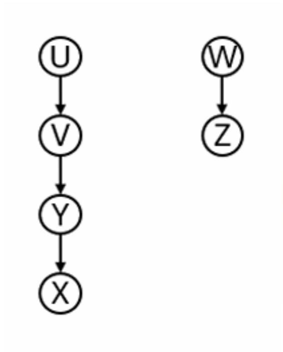
- לבן: אז $d(v) = d(u) + 1$
- אפור: אז קיים w שיצא מהתור לפני u, ש v הוא אחד השכנים שלו [$v \in N(w)$, $d(w) \le d(u)$].
אז $d(v) = d(w) + 1 \Rightarrow d(v) \le d(u) + 1$.
- שחור: אז v כבר היה מחוץ לתור, אז $d(v) \le d(u)$ (ממסקנה 1)

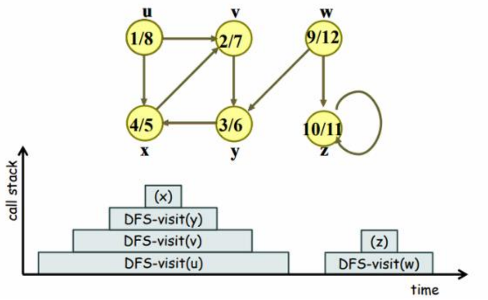
בכל המקרים, זו סתירה ל (1).
מסקנה: לכל v מתקיים $d(v) = \delta(s,v)$.
ובגלל ש δ משתנה רק כשמגיעים לקודקוד, זה אומר שBFS מגיע לכל קודקוד שאפשרי.
ומכיוון ש $d(v) = d(u) + 1 \Longrightarrow \pi(v) = u$, המסלול הקצר $s \mapsto v$ הוא: $s \mapsto \pi(v)$ ואז הצלע $(\pi(v), v)$.

```
printPath(s, v){
    if(s == v) print(s)
    else{ if(π(v) == null) print(no path)
        else{ printPath(s, π(v)
            print(v)}}
```

**__עץ BFS – תת-גרף קודמים (predecessor subgraph):__**
תת-גרף של G, שמתאר את המסלולים הקצרים: $G_\pi = (V_\pi, E_\pi)$.
הצלעות ב- $E_\pi$ נקראות **צלעות עץ – tree edges.**
מתקיים (כמו בכל עץ): $|E_\pi| = |V_\pi| - 1$

**<u>DFS</u>:** חיפוש לעומק, מבוסס על מחסנית (רקורסיה).
מוצא את כל הקודקודים (גם אם הגרף לא קשיר). אין חשיבות לקודקוד ההתחלה. נותן לנו מידע לגבי הקשרים בין קודקודים. צובעים קודקודים כמו בBFS.
נותן לכל קודקוד: $d(v)$ – זמן גילוי, הרגע שבו v נצבע אפור. $f(v)$ – זמן סיום, הרגע שבו v נצבע שחור. $\pi(v)$ – הקודקוד שממנו גילינו את v.




**יעד DFS:**
$G_\pi = (V, E_\pi)$
$E_\pi = \{(\pi(v), v) \mid v \in V \text{ and } \pi(v) \neq null\}$
הצלעות הן הצעדים שעשינו בDFS.

כמו בBFS, הפעולה העיקרית תקרה רק פעם אחת לכל קודקוד, אז זמן הריצה הוא סה"כ **O(n+m)**

```
DFS(G)
  for(u ∈ V)
    c(u) ← 0, π(u) ← null
  t ← 0
  for(u ∈ V)
    if( c(u) == white )
      visit(u, t)
```

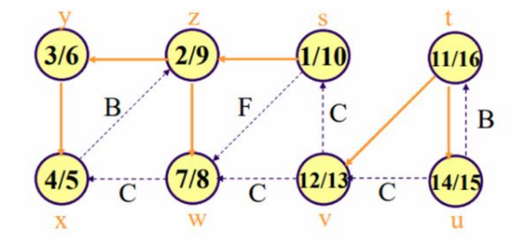
```
visit(u, t)
  c(u) ← gray,  d(u) ← t,  t + +
  for( v ∈ N(u))
    if( c(v) == white)
      π(v) ← u, visit(v, t)
  c(u) ← black, f(u) ← t + +
```

**משפט הסוגריים:** לכל שני קודקודים $u, v$, מתקיים בדיוק אחד מהבאים:
1) $d(u) < f(u) < d(v) < f(v) \quad or \quad d(v) < f(v) < d(vu) < f(u)$, אף קודקוד הוא לא צאצא של השני ב $G_\pi$.
2) $d(u) < d(v) < f(v) < f(u)$, v הוא צאצא של u ב $G_\pi$.
3) $d(v) < d(u) < f(u) < f(v)$, u הוא צאצא של v ב $G_\pi$.
הוכחה: ראשית נשים לב שמהגדרת האלגוריתם, אין שני זמנים שווים, ולכל קודקוד מתקיים $d(v) < f(v)$.
אם $d(u) < d(v) < f(u)$, זה אומר שביקרנו ב v לפני שסיימנו את u. אז v יהיה צאצא של u, ונסיים את v לפני u. אז $d(u) < d(v) < f(v) < f(u)$. בה"כ באופן דומה עבור $d(v) < d(u) < f(v)$.
אם (בה"כ) $d(v) < f(v) < f(u)$, זה המקרה הראשון. וכל מצב אחר לא אפשרי.

**משפט המסלול הלבן:** v הוא צאצא של u בעץ אמ"מ בזמן $d(u)$ היה מסלול $v \mapsto u$ שמורכב כולו מקודקודים לבנים.
הוכחה: כיוון ראשון: נניח ש v צאצא של u בעץ. האלגוריתם מתקדם רק על קודקודים לבנים, אז המסלול $v \mapsto u$ היה כולו לבן.
כיוון שני: נניח שקיים מסלול $v \mapsto u$ שמורכב כולו מקודקודים לבנים בזמן $d(u)$. נוכיח ש v צאצא של u בעץ, באינדוקציה על אורך המסלול: אם $n = 1$, זה אומר ש $v \in N(u)$. אז חייבים לסיים עם v לפני שמסיימים עם u.
נניח שמתקיים עבור מסלול באורך $n - 1$, ונוכיח עבור n:
יהי w הקודקוד הקודם ל- v במסלול הלבן. אז מהנ"א, w צאצא של u בעץ. נתבונן בצבע של v בזמן $d(w)$:
- לבן – v צאצא של w, שהוא צאצא של u.
- אפור/שחור – w הוא צאצא של u, אז מתקיים $d(u) < d(w) < f(w) < f(u)$, $d(v) < d(w)$. ומכיוון ש v היה לבן בזמן $d(u)$, $d(u) < d(v)$.אז ממשפט הסוגריים נקבל ש $d(u) < d(v) < f(v) < f(u)$. v צאצא של u בעץ.

**סוגי צלעות:** נחלק את הצלעות של הגרף המקורי לקבוצות לפי המצב של הקודקודים של כל צלע בזמן הגילוי של הצלע:
- צלע עץ – tree edge: צלע ביער, $e \in E_\pi$. $(u, v)$ כך שגילינו את v במהלך בדיקה של u.
- צלע אחורה – back edge: $(u, v)$ כך ש u צאצא של v בעץ.
- צלע קדימה – forward edge: $(u, v)$ כך ש v צאצא של u בעץ, אבל לא צלע ביער.
- צלע חוצה – cross edge: כל צלע אחרת. (הצלע קיימת בגרף אבל אף קודקוד לא צאצא של השני ביער.)

הגדרת צבעים במהלך ריצת DFS: בזמן $visit(u)$, כשבודקים את $N(u)$.
נסווג את $(u, v)$ לפי הצבע של v:
לבן – צלע עץ.   אפור – צלע אחורה.
שחור: אם $d(u) < d(v)$ – קדימה.   אם $d(v) < d(u)$ – חוצה.



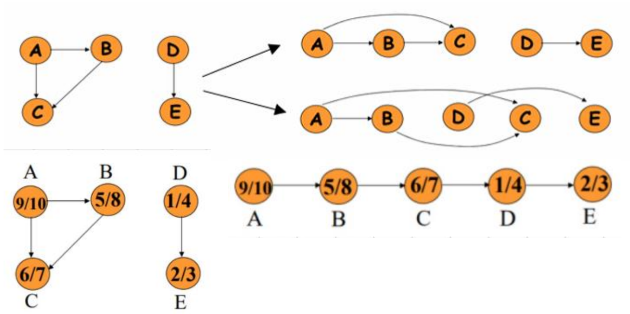
(s (z (y (x x) y) (w w) z) s) (t (v v) (u u) t)

**לגרף לא מכוון יש רק צלעות עץ/אחורה:**
הוכחה: נתבונן ב $(u, v)$, בה"כ $d(u) < d(v)$. ממשפט הסוגריים, $f(v) < f(u)$.
אם בדקנו קודם את הצלע בכיוון $v \to u$, זה אומר שv היה לבן עד אותו רגע, אז זה צלע עץ.
אם בדקנו קודם $u \to v$, אז u היה אפור באותו רגע (כי לא סיימו לבדוק את כל הצאצאים שלו). אז זה צלע אחורה.

**<u>גרף מכוון ללא מעגלים – DAG – Directed Acyclic Graph:</u>**

טענה: **גרף הוא DAG אמ"מ DFS לא מחזיר צלעות אחורה**. הוכחה: יהי גרף מכוון. כיוון ראשון – נניח שהגרף חסר מעגלים. נב"ש שDFS נתן צלע אחורה. כלומר, הגענו ל v מ u, ועכשיו יש צלע $u \to v$. זה מעגל. סתירה. כיוון שני: נניח שאין צלעות אחורה. נב"ש שיש מעגל C כלשהו בגרף. ניקח את u, הקודקוד הראשון ב-C שמתגלה. נקרא לקודקוד הקודם במעגל v. בזמן $d(u)$, היה מסלול לבן $u \mapsto v$. (כי u הוא הקודקוד הראשון שמגלים במעגל). אז v צאצא של u בעץ. אז מהגדרה, $(v, u)$ תהיה צלע אחורה. סתירה.

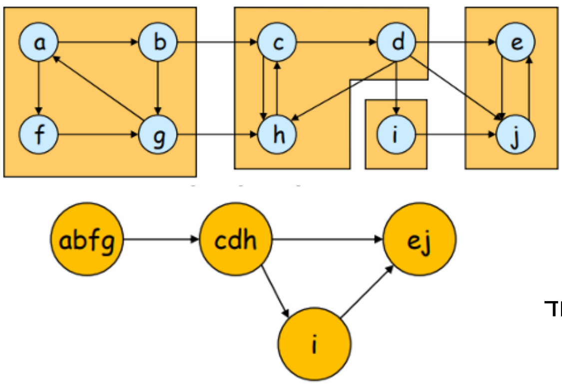
**<u>מיון טופולוגי:</u>** מיון של הקודקודים כך שכל הצלעות הולכות "ימינה". (יכולות להיות כמה דרכים נכונות). זה אפשרי רק בDAG, כי אם יש מעגל אי אפשר לתת לגרף "כיוון".




שיטה: נריץ DFS, ונשמור את הקודקודים לפי $f(v)$, מהגדול לקטן.
הוכחת נכונות: צ"ל שאם $(u, v) \in E$, אז $f(u) > f(v)$.
נתבונן ב$c(v)$ בזמן בדיקת $(u, v)$:
לבן – ממשפט הסוגריים נקבל ש $d(u) < d(v) < f(v) < f(u)$.
אפור – אז $(u, v)$ היא צלע אחורה וזה לא אפשרי בDAG.
שחור – כבר סיימנו את v לפני u, אז $f(u) > f(v)$.

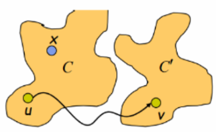
**<u>רכיבי קשירות חזקה – SCC – Strongly Connected Components:</u>**

קבוצת קשירות חזקה היא קבוצת קודקודים שבה אפשר להגיע מכל קודקוד לכל קודקוד אחר. **רכיב קשירות חזקה** הוא קבוצת קשירות שהיא מקסימלית, כלומר אין עוד קודקוד שאפשר להוסיף לקבוצה והיא עדיין תקיים קשירות חזקה.

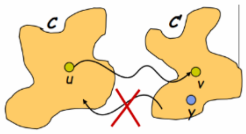



**גרף רכיבים – component graph:** $G^{SCC} = (V^{SCC}, E^{SCC})$.
כל קודקוד הוא רכיב קשירות חזקה.
צלעות הן הצלעות בגרף המקורי שמחברות בין רכיבי קשירות.
$G^{SCC}$ הוא DAG – אם היה מעגל, כל הקודקודים בשני הרכיבים היו נגישים אחד מהשני וזה היה צריך להיות רכיב קשירות אחד.

**אלגוריתם למציאת $G^{SCC}$:** נעשה מיון טופולוגי (DFS אחד), ונריץ DFS שני על $G^T$ – הגרף המקורי אבל צלעות הפוכות. בלולאה הראשית נרוץ לפי הסדר של המיון. ביער DFS שנקבל, כל עץ הוא רכיב קשירות.
כדי להוכיח את נכונות האלגוריתם, נשתמש בטענת עזר:
**יהיו $C, C'$ רכיבי קשירות שונים. נניח שיש מסלול $u \mapsto v$ כך ש $u \in C$, $v \in C'$. אז $f_{max}(C) > f_{max}(C')$.**
הוכחת טענת העזר:

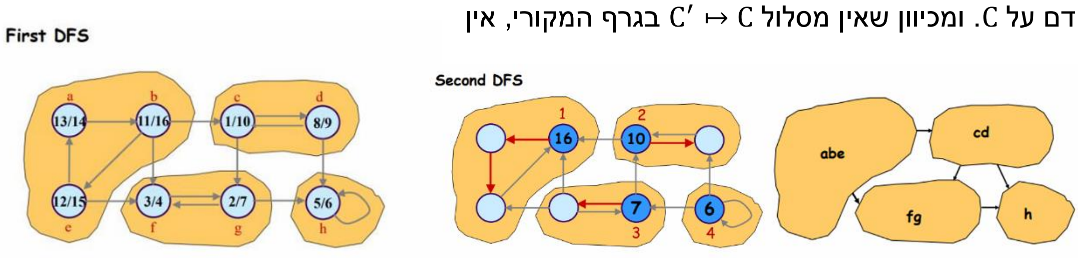



מקרה 1: $d_{min}(C) < d_{min}(C')$. יהי x הקודקוד הראשון שמתגלה בC. בזמן $d(x)$, כל הקודקודים ב$C, C'$ לבנים. אז ממשפט המסלול הלבן, כולם צאצאים של x ביער. אז ממשפט הסוגריים, $f(x) > f(y)$ לכל $y \in \{C \cup C'\}$. אז $f_{max}(C) > f_{max}(C')$.




מקרה 2: $d_{min}(C) > d_{min}(C')$. יהי y הקודקוד הראשון שמתגלה ב$C'$. כל הקודקודים ב $C'$ הם צאצאים של y בעץ (באופן דומה למקרה 1). אז $f(y) = f_{max}(C')$. מכיוון שאין מסלול $C' \mapsto C$, (כי אחרת היה מעגל), בזמן $f(y)$ כל הקודקודים ב C לבנים, אז $f_{max}(C) > f_{max}(C')$.

אז בDFS השני, נעבור קודם על C. ומכיוון שאין מסלול $C' \mapsto C$ בגרף המקורי, אין מסלול $C \mapsto C'$ ב $G^T$.
אז $C'$ יהיה עץ חדש.
זה מתקיים עבור כל שני רכיבי קשירות חזקה. אז כל רכיב יהיה עץ נפרד.



**מסלול קצר דרך קודקוד:** בהינתן גרף לא מכוון, נבדוק אם קיים מסלול קצר $u \mapsto v$ שעובר דרך $w$: נעשה
$\mathrm{BFS}(G,u),\ \ \mathrm{BFS}(G,w)$
ונבדוק אם $\delta(u,v) = \delta(u,w) + \delta(w,v)$.

נתון DAG בייצוג רשימת שכנויות. צ"ל אלגוריתם שבודק **האם קיימים קודקודים s, t כך שקיים מסלול מ-s לכל קודקוד, וקיים מסלול מכל קודקוד ל-t.**
נעשה מיון טופולוגי. אם קיימים קודקודים כאלה הם יהיו הראשון והאחרון. נעשה BFS מהראשון. אם אפשר להגיע ממנו לכל הקודקודים, זה s. נעשה BFS על $G^T$ מהאחרון, זה יגיד לנו אם זה t. (חישוב הגרף ההפוך לוקח O(n+m) ברשימת שכנויות)

**אלגוריתם שבודק אם גרף הוא דו"צ:**
גרף מוגדר דו"צ אם קיימת חלוקה A,B של הקודקודים כך ש: $V = A \cup B,\ \ A \cup B = \emptyset$ , ואין צלעות בין A ל- B.
נעשה BFS, ונצבע את הקודקודים בצורה שונה:

```
Bipitrate(G, s)
  color all nodes white
  c(s) ← blue,      Q ← s
  while(Q ≠ ∅)
    u ← Q.pop
    for(v ∈ N(u))
      if(c(v) == c(u)) return false
      if(c(v) == white)
        if(c(u) == blue) c(v) ← red
        if(c(u) == red) c(v) ← blue
      Q ← v
  return true
```

הוכחת נכונות האלגוריתם: כיוון ראשון: נניח שהאלגוריתם החזיר true. ניקח קבוצות לפי הצבעים של הקודקודים. אין קודקוד שהוא באותו צבע כמו הבן שלו. אז זאת חלוקה דו"צ.
כיוון שני: טענות עזר:
טענה 1: אם יש לגרף מעגל אי זוגי, הוא לא 2-צביע. הוכחה: נלך על המעגל וניתן צבעים לסירוגין. כשנחזור להתחלה נהיה באותו צבע.
טענה 2: אם האלגוריתם החזיר false, זה אומר שיש מעגל אי"ז. הוכחה: האלגוריתם מחזיר false רק במקרה שהגענו לקודקוד כחול (בה"כ) דרך קודקוד כחול. כלומר הגענו חזרה למקום שהיינו – זה מעגל. אם הוא היה זוגי, הצבעים היו מתחלפים והיינו בצבע השני. אז הוא אי"ז.
מסקנה: נניח שהאלגוריתם החזיר false. אז יש מעגל אי"ז, אז הגרף לא 2-צביע, אז הוא לא דו"צ.

הגדרה: **שורש** של גרף הוא קודקוד שיש ממנו מסלול לכל קודקוד אחר. $R(G) \subseteq V_G$ היא קבוצת השורשים של G.
טענה: **בכל DFS, R(G) תהיה מוכלת כולה בעץ האחרון ביער.** הוכחה: נגדיר את העצים $T_1 \cdots T_i \cdots T_n$. נב"ש שיש שורש $r$ שנמצא ב$T_i$. בגלל שהוא שורש, קיים מסלול ממנו אל השורש של העץ האחרון, $r_t$. נתבונן בw, הקודקוד הראשון במסלול הזה שנמצא ב$T_n$. הקודקוד הקודם לו במסלול הוא u, $u \in T_i$. ממשפט המסלול הלבן, הם צריכים להיות באותו עץ. סתירה.

טענה: **נגדיר את r השורש של העץ האחרון. אם $\mathbf{R(G) \neq \emptyset}$, אז $\mathbf{r \in R(G)}$.** הוכחה: מהטענה הקודמת נקבל ש $R(G) \subseteq T_n$, ומהשורש של העץ יש מסלול לכל קודקוד בעץ, אז יש מסלול לכל הקודקודים בגרף. אז השורש של העץ הוא שורש בגרף.
נובע מכך – **אם נתחיל DFS מקודקוד שהוא שורש של הגרף, נקבל עץ יחיד.**

הגדרה: **יעד גלובלי** הוא קודקוד שיש מסלול אליו מכל קודקוד אחר.
גרף נקרא **גרף מעקפים** אם לכל שני קודקודים $u, v$ קיים קודקוד $w$ כך שיש מסלולים $u \mapsto w, v \mapsto w$.
טענה: גרף הוא גרף מעקפים אמ"מ יש בו יעד גלובלי. הוכחה: כיוון ראשון: אם יש י"ג, הוא ה- w. כיוון שני: יהי גרף מעקפים, נב"ש שאין בו יעד גלובלי. נגדיר: $p(v) \coloneqq \{u \in V | \exists \text{a path } u \mapsto v\}$ מספר הקודקודים שאפשר להגיע אליהם מ v.
נשים לב שאם v הוא י"ג, אז $|V| = |p(v)|$, אז מההנחה, $k \coloneqq \max(|p(v)|) < |V|$. ניקח קודקוד u שמקיים $|p(u)| = k$. ניקח קודקוד $z \notin p(u)$ (בהכרח קיים כי u לא יעד גלובלי), אם יש מסלול $u \mapsto z$ אז אפשר להגיע לz מכל קודקוד שאפשר להגיע ממנו לu, וגם אפשר להגיע מu ל- z כלומר $|p(z)| > |p(u)|$. סתירה להנחת המקסימליות. אם אין מסלול $u \mapsto z$ אז בגלל שזה גרף מעקפים קיים w שאפשר להגיע אליו מ-u וגם מ-z, גם סתירה למקסימליות של $p(u)$.

טענה: **קודקוד הוא יעד גלובלי אמ"מ הוא שורש של $\mathbf{G^T}$.**
הוכחה: v שורש בG אמ"מ יש מסלול ממנו לכל קודקוד, אמ"מ יש מסלול מכל קודקוד אליו ב$G^T$, אמ"מ הוא יעד גלובלי ב$G^T$.

**אלגוריתם שבודק אם גרף הוא גרף מעקפים:** נריץ $\mathrm{DFS}(G^T)$, אם יש יותר מעץ אחד ביער נריץ DFS מהשורש שלו.
אם בDFS הזה מקבלים עץ יחיד זה גרף מעקפים, אחרת לא. סה"כ O(m+n).
הוכחת נכונות: G גרף מעקפים אמ"מ יש לו י"ג, אמ"מ יש שורש ב$G^T$, אמ"מ כשמריצים DFS מהקודקוד הזה מקבלים עץ יחיד.

**מציאת מסלול קצר מקבוצה לקודקוד מסוים:** נתונה קבוצה $V' = \{v_1, \cdots v_k\} \subseteq V$. וקודקוד $w$. צריך למצוא את המסלול הקצר ביותר מאחד מקודקודי $V'$, ל-w. אפשר לעשות $\mathrm{BFS}(G^T, w)$, ולהחזיר את המינימלי מבין $V'$. או להוסיף קודקוד s שמחובר לכל $V'$, ולעשות $\mathrm{BFS}(G', s)$. שני הדרכים שווים אסימפטוטית.

הגדרה: **גשר** הוא צלע שאם נוריד אותו נקבל עוד רכיב קשירות. **מציאת גשרים בגרף לא מכוון:** אבחנה: צלע היא גשר אמ"מ היא לא נמצאת על אף מעגל. (אם היא על מעגל אפשר להוריד אותה ופשוט ללכת "מסביב"). צלע back edge מסמלת דרך להגיע לקודקוד שדרכו הגענו לקודקוד הנוכחי – כלומר מעגל.

נגדיר לכל קודקוד קבוצה B – כל הקודקודים שהם אב קדמון של v בעץ DFS, שמחוברים לv **או צאצא של** v ע"י back edge.

נוסיף לכל קודקוד שדה low – $\min\{d(u)|u \in B(v)\}$. נעדכן את low תוך כדי הריצה. מה שצריך לעשות זה רק לשנות את המימוש של visit:

$$
\begin{array}{l}
\mathrm{visit}(u) \\
\quad c(u) \leftarrow \mathrm{gray}, d \leftarrow ++t, \mathrm{low}(u) \leftarrow t \\
\quad \mathrm{for}\big(v \in N(u)\big) \\
\qquad \mathrm{if}(c(v) == \mathrm{white}) \\
\qquad\quad \pi(v) \leftarrow u, \mathrm{visit}(v) \\
\qquad\quad \mathrm{if}\big(\mathrm{low}(v) < \mathrm{low}(u)\big)\mathrm{low}(u) \leftarrow \mathrm{low}(v) \\
\qquad \mathrm{else\ if}\big(c(v) == \mathrm{gray}\ \&\ \pi(u) \neq v\ \&\ d(v) < \mathrm{low}(u)\big)\ \mathrm{low}(u) \leftarrow d(v) \\
\quad c(u) \leftarrow \mathrm{black}
\end{array}
$$