Finding Strongly Connected Components with DFS

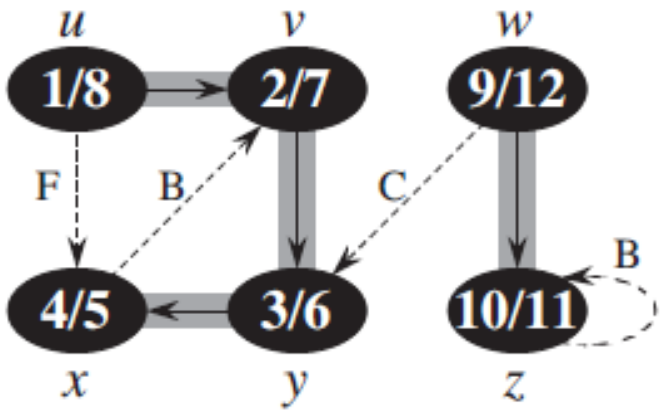
The Sharir-Kosaraju Algorithm

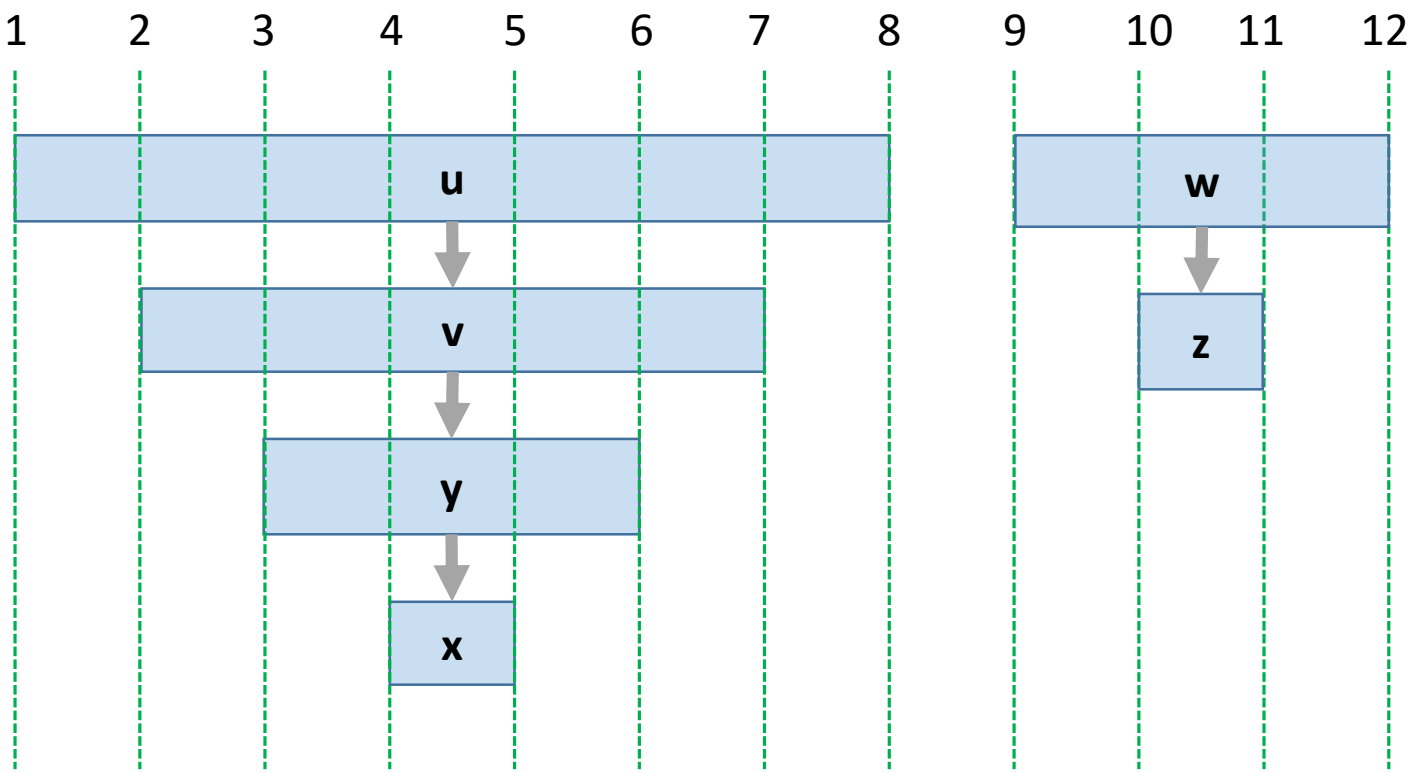
```
DFS(G)
For each vertex u ∈ G.V
    u.color = WHITE
    u.π = NIL
time = 0
For each vertex u ∈ G.V
    if u.color == WHITE
        DFS-Visit(G,u)
```

```
DFS-Visit(G,u)
time = time + 1
u.d = time
u.color = GRAY
For each vertex v ∈ G.Adj[u]
    if v.color == WHITE
        v.π = u
        DFS-Visit(G,v)
u.color = BLACK
time = time + 1
u.f = time
```

Lem 1: The graph $G_\pi = (V, E_\pi)$, $E_\pi = \{ (v_\pi, v) \mid v_\pi \neq NIL \}$ is a forest (DFS-forest).

Each node v is associated with the interval $[v.d, v.f]$





```
DFS(G)
For each vertex u ∈ G.V
    u.color = WHITE
    u.π = NIL
time = 0
For each vertex u ∈ G.V
    if u.color == WHITE
        DFS-Visit(G,u)
```

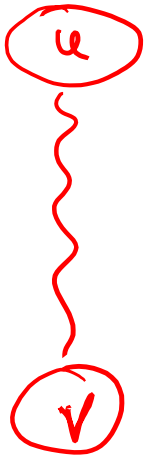
```
DFS-Visit(G,u)
time = time + 1
u.d = time
u.color = GRAY
For each vertex v ∈ G.Adj[u]
    if v.color == WHITE
        v.π = u
        DFS-Visit(G,v)
u.color = BLACK
time = time + 1
u.f = time
```

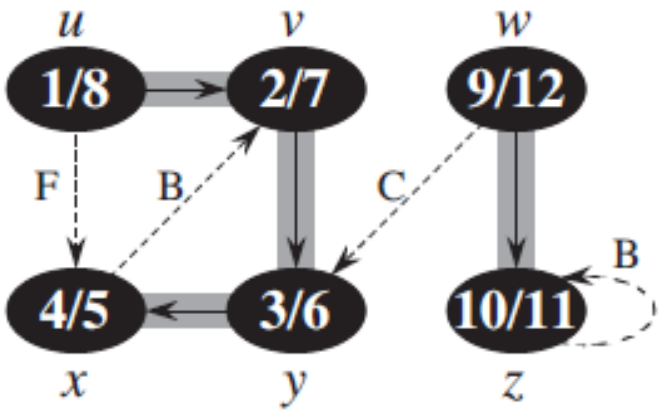
Thm 2: $[v.d, v.f] \subset [u.d, u.f]$ iff v is a descendant of u in the DFS-forest, $[v.d, v.f] \cap [u.d, u.f] = \emptyset$ iff u and v are unrelated in the DFS-forest.

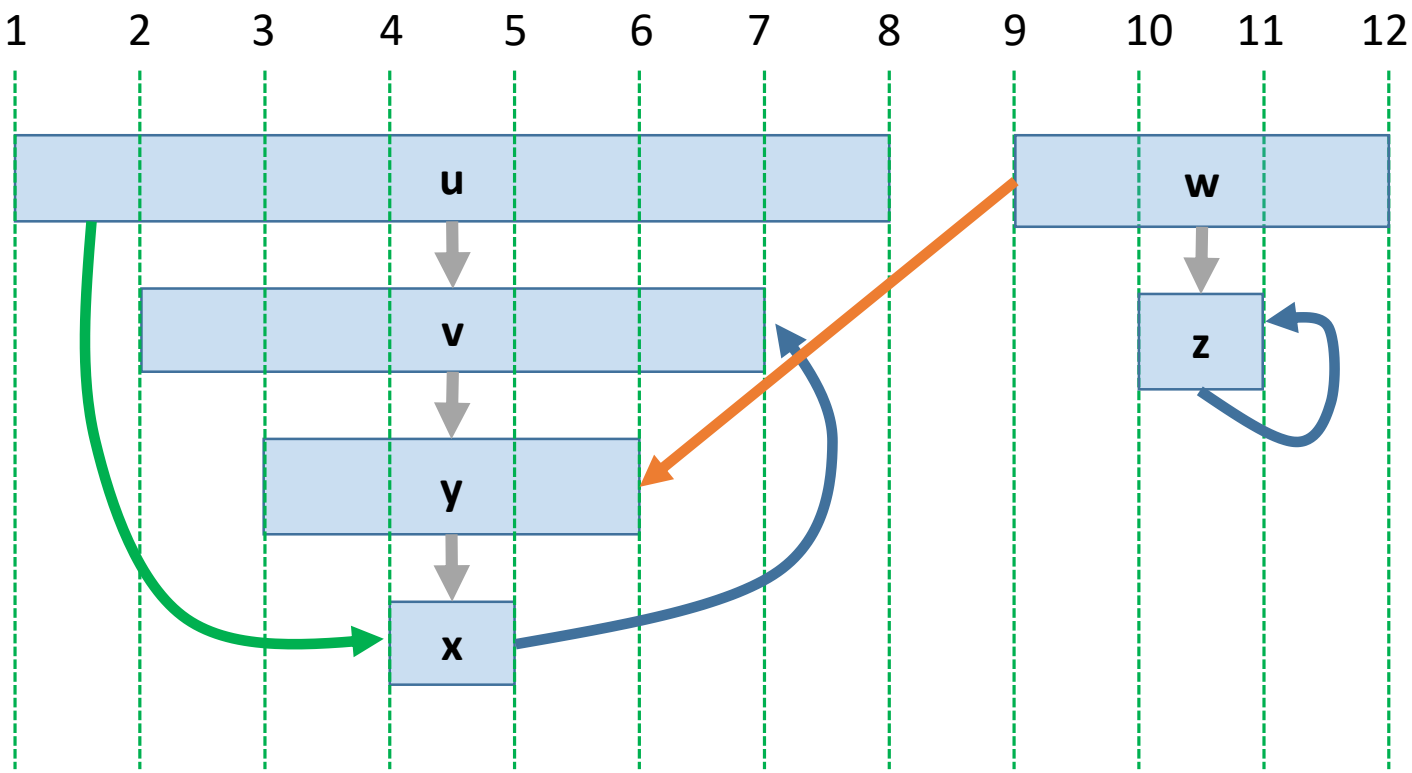
```
DFS(G)
For each vertex u ∈ G.V
    u.color = WHITE
    u.π = NIL
time = 0
For each vertex u ∈ G.V
    if u.color == WHITE
        DFS-Visit(G,u)
```

```
DFS-Visit(G,u)
time = time + 1
u.d = time
u.color = GRAY
For each vertex v ∈ G.Adj[u]
    if v.color == WHITE
        v.π = u
        DFS-Visit(G,v)
u.color = BLACK
time = time + 1
u.f = time
```

Thm 3: v is a descendant of u in the DFS-forest iff when we invoke DFS-Visit(u) there is a white path from u to v.




u
v
w
1/8
2/7
9/12
F
B
C
B
4/5
3/6
10/11
x
y
z


1
2
3
4
5
6
7
8
9
10
11
12
u
w
v
z
y
x

Edge classification

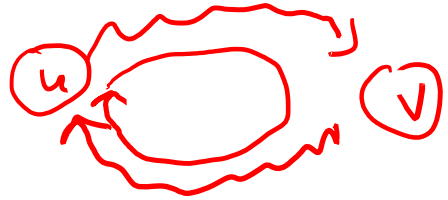
- Tree edges: (u,v), DFS-Visit(v) was called from DFS-visit(u)
- Back edges: (u,v) such that v is an ancestor of u in the DFS-forest
- Forward edges: nontree edges (u,v) such that v is a descendant of u
- Cross edges: (u,v) such that $v.f < u.d$

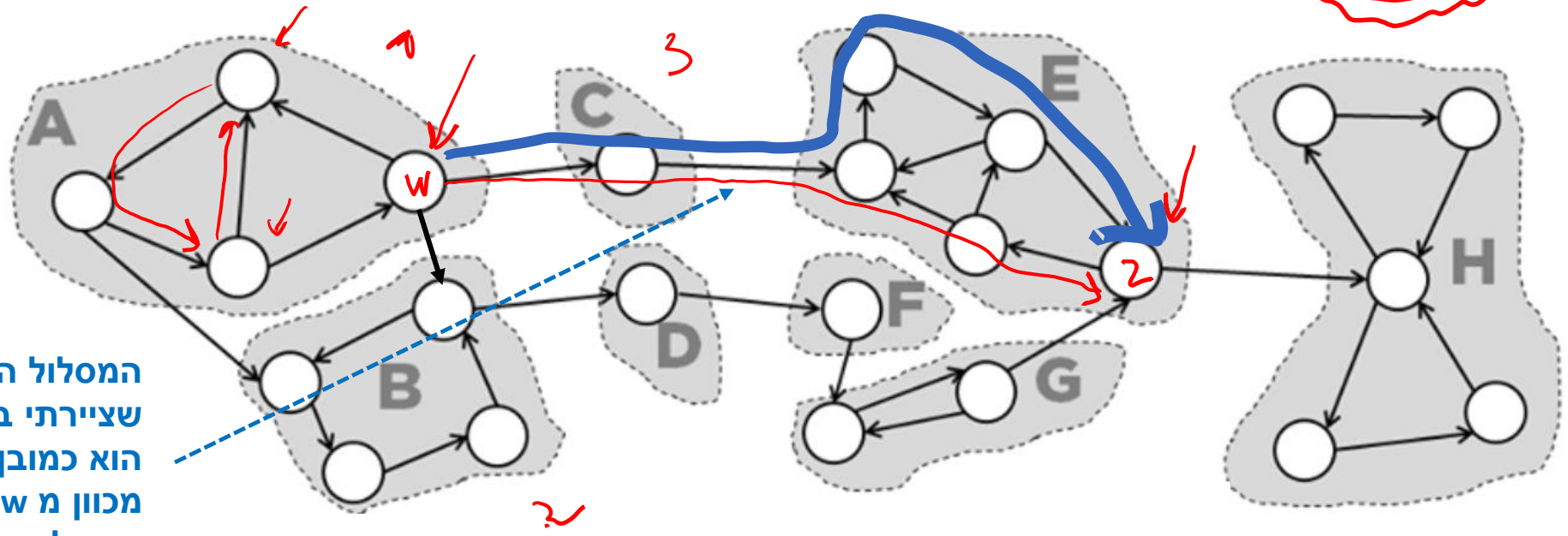
Observe, among non-tree edges:

1. (Only) backward edges go to a vertex with a later finish time
2. (Only) forward edges go to a vertex with later discovery time

Strongly connected components (SCC)

Def: Equivalence classes of the relation (u, v) iff there is a path from u to v and a path from v to u



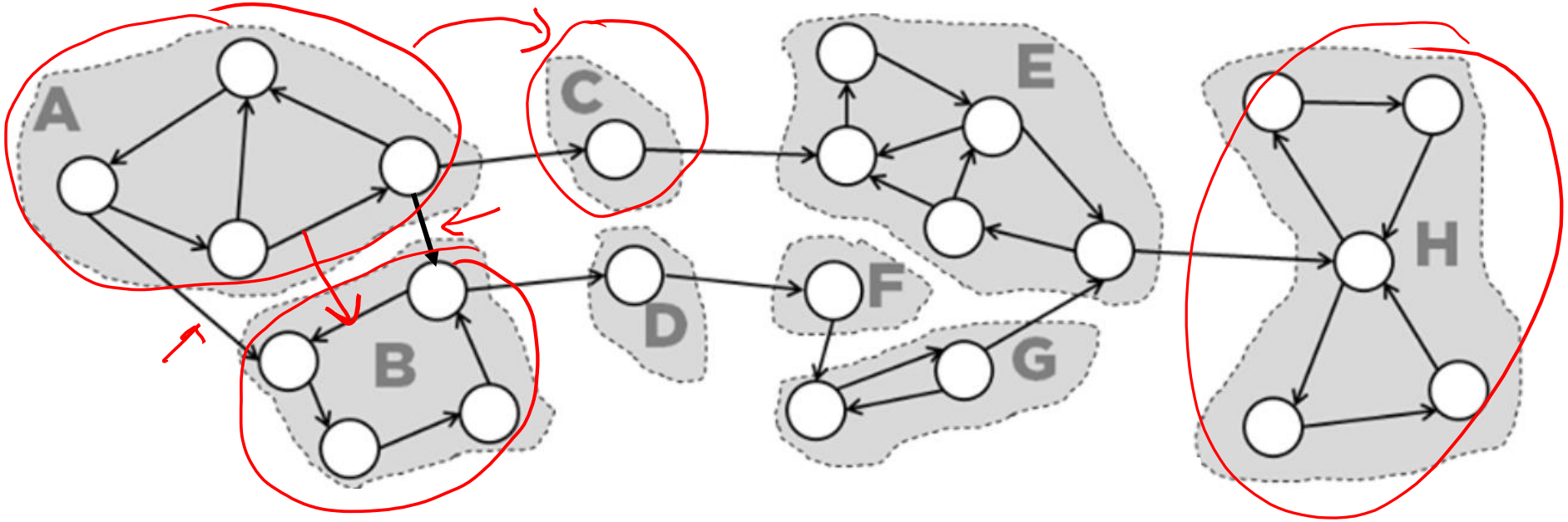


המסלול האדום
שציירתי בזמן ההקלטה
הוא כמובן אינו מסלול
מכוון מ w ל z, המסלול
הכחול הוא מסלול כזה

SCC graph $G^{SCC} = (V^{SCC}, E^{SCC})$

Def: V^{SCC} has a vertex c_i per strong component C_i.

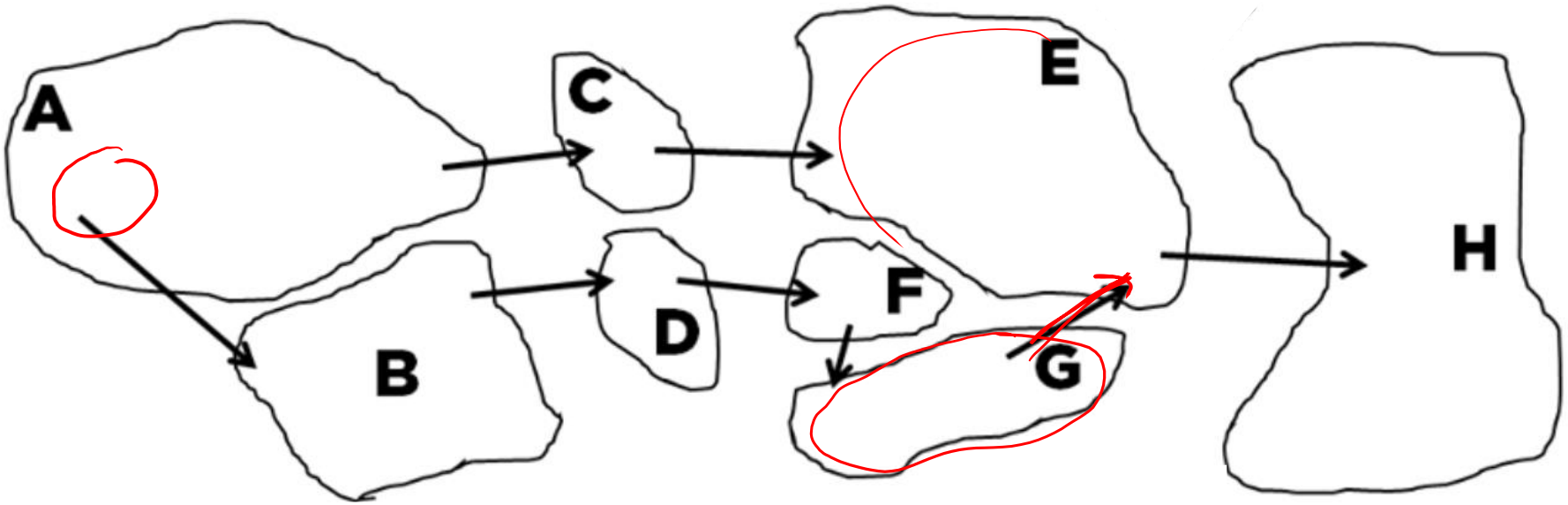
E^{SCC} has a edge (c_i, c_j) if there is an edge in G from a vertex in C_i to a vertex in C_j



SCC graph $G^{SCC} = (V^{SCC}, E^{SCC})$

Def: V^{SCC} has a vertex c per strong component C.

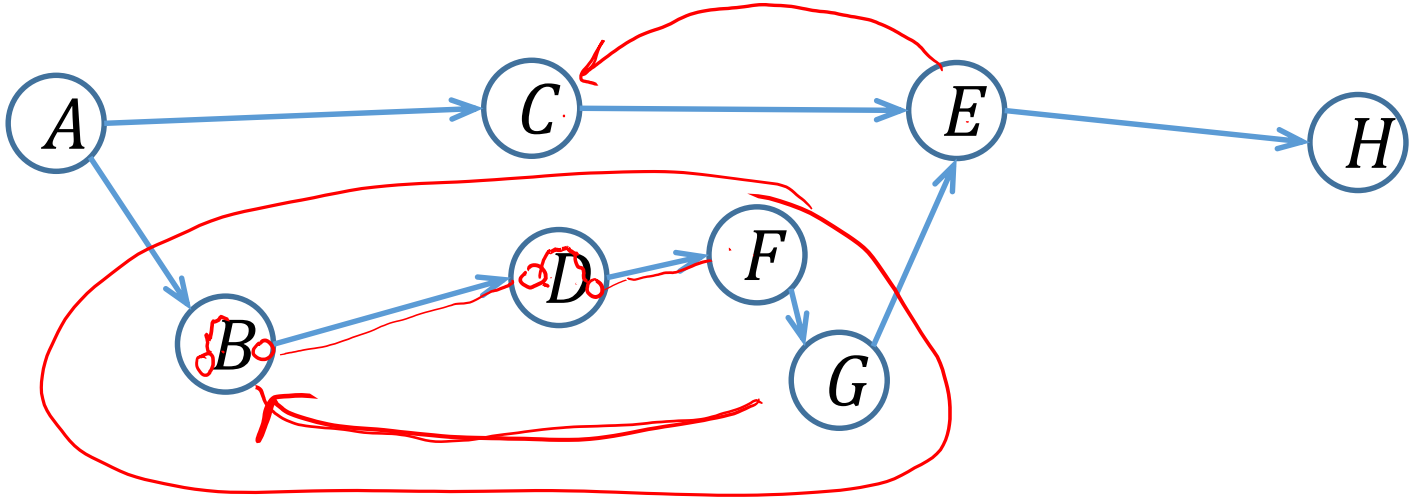
E^{SCC} has a edge (c, c') if there is an edge in G from a vertex in C to a vertex in C'



SCC graph $G^{SCC} = (V^{SCC}, E^{SCC})$

Def: V^{SCC} has a vertex c per strong component C.

E^{SCC} has a edge (c, c') if there is an edge in G from a vertex in C to a vertex in C'



obs: G^{SCC} is a DAG

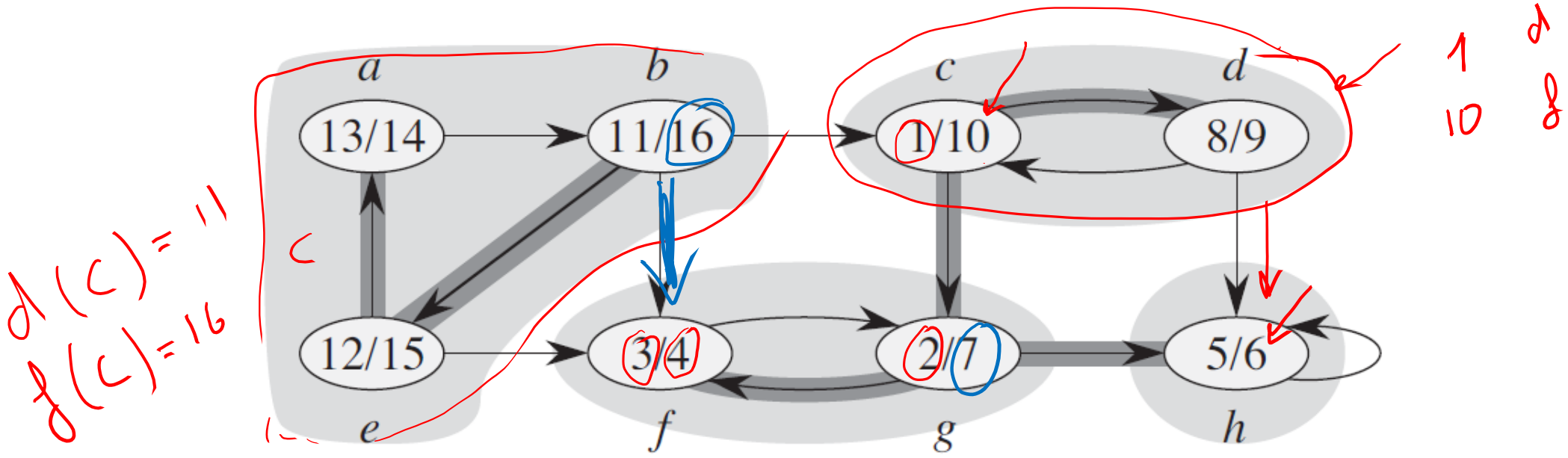
כתוב בתרגיל בלוח: proof
פורמלי.

Computing strongly connected components

איך קיבלנו לחשב את מספר הרכיב

בו הוא נמצא

Lets make a DFS

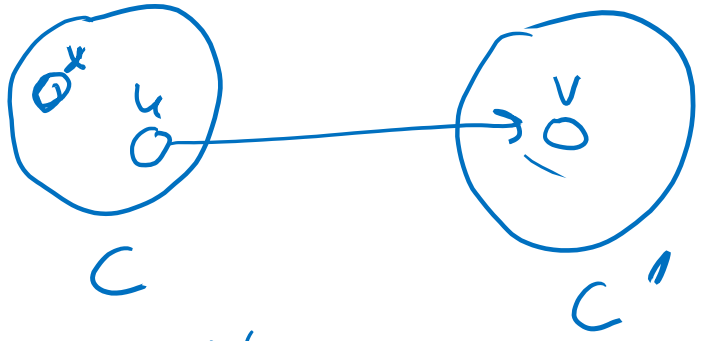


$$\boldsymbol{d(C)} = \min\{\, v.d \mid v \in C \,\} \quad \boldsymbol{f(C)} = \max\{v.f \mid v \in C\}$$

Thm 7: If there is an edge from $u \in C$ to $v \in C'$ then $f(C) \geq f(C')$

10 > 6

$f(C) = x.d$



$f(C) > f(C')$

בשני מקומות כאן
צריך להיות d(C)
ולא f(C)

$x.d = f(C)$

proof

מקרה I $d(C) < d(C')$

יהי x הצומת עבורו

x הראשון שהוסר לה מהמין כך הצמתים ב $C \setminus C'$

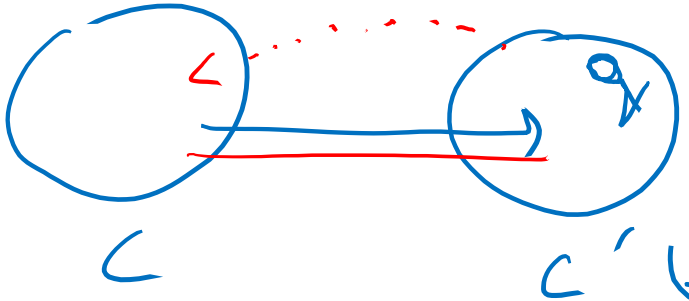
נוכל להניח בהמשך שימי x בעמן שימי x יש מסלול

לכן מצא לכל צומת ב $C \setminus C'$. לפי Thm 3 כל הצמתים

ב $C \setminus C'$ יהיו לאחרים לx ב Forest. $\forall y \in C \setminus C'$ $y.f < x.f$

$\Leftarrow f(C) < f(C')$

טענה II $d(c') < d(c)$



$d(c') = d.y$

בזמן שישנו y כך שלא c'

קיים.

מאחר ו c קיים בזמן שישנו y כך הצמיחה ב c ומבקש שאחרי y שמקיים $d(c) > d(c')$ $d.y$ כך

מעל שיעורים מספרי קיים כך הצמיחה ב c' יהיו גאוגאים גם y בעל Θ בנוסף שישנו

Thm3 $\Leftarrow$ $f(c') = y.f$. אין מספר מכוון מעל c' ~~לצמיחה~~ לצומח ב c (סתירה לכך ש c ו c' רציפי קטיגוריה אבקה) וכן בפרט אין מספר קיים מעל לצומח ב c בזמן שישנו גם y. Thm3 $\Leftarrow$ צמיחה בכל שם גאוגאים גם y בעל Θ $\square$

Reverse the graph

Replace (u, v) by (v, u) ➔ you get G^T

DFS על G^T

כאשר נשלחים לאחור

בזמן לכן מאני להתחיל עם חצים נמחר

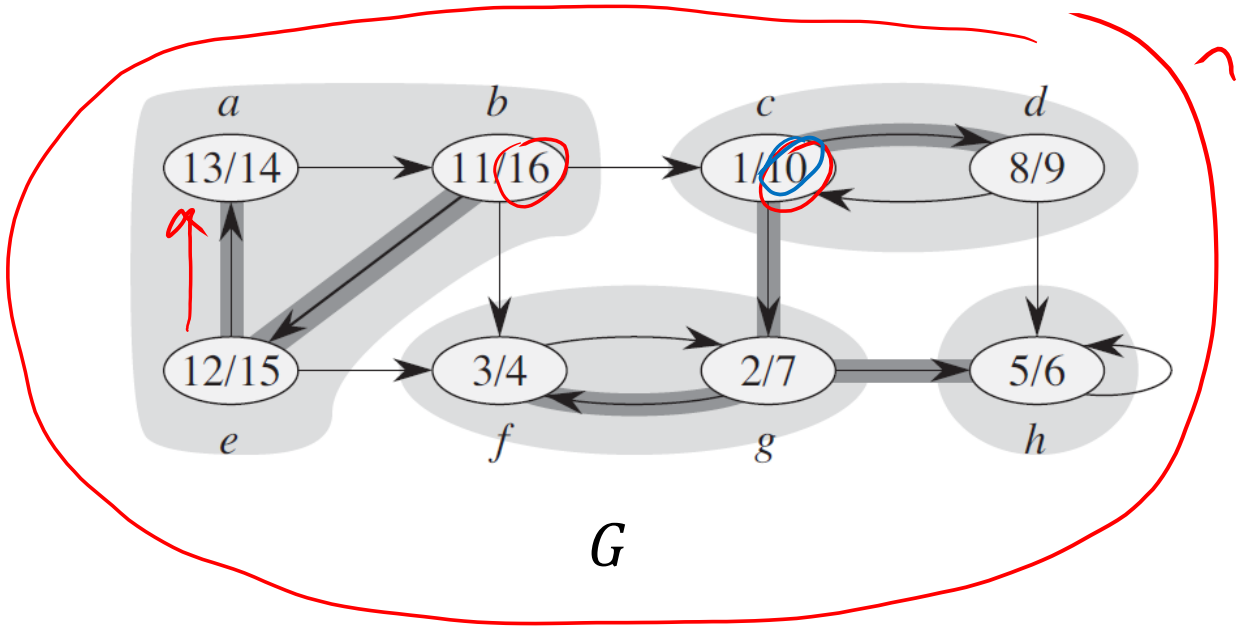
כלב.

לפי

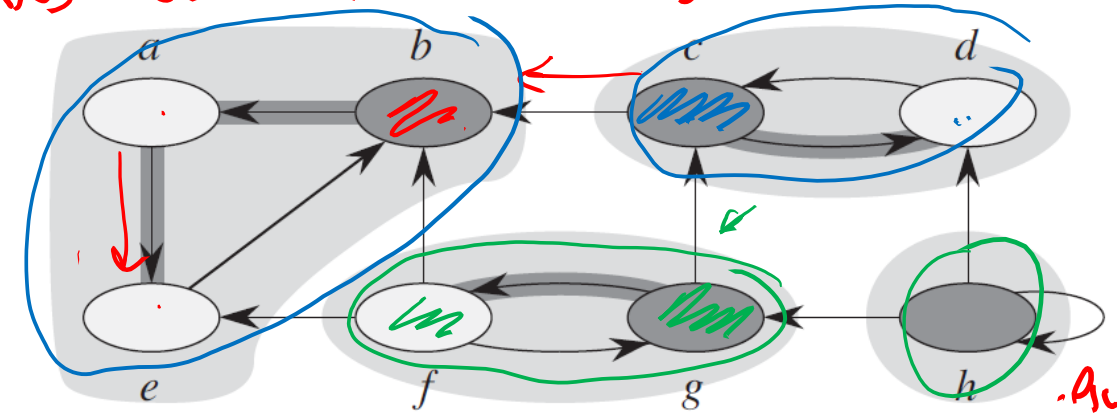
כמן

סוף

הנסיעה.



G



G^T

מה אפשרי ניתן

בגלל הנסיעה בגרף G^T

$f(c) < f(c')$

$u \in C'$

$v \in C'$

(u, v)

An algorithm to compute SCC's

STRONGLY-CONNECTED-COMPONENTS(G)

1 call DFS(G) to compute finishing times $u.f$ for each vertex u

2 compute G^{T}

3 call DFS(G^{T}), but in the main loop of DFS, consider the vertices in order of decreasing $u.f$ ~~(as computed in line 1)~~

4 output the vertices of each tree in the depth-first forest formed in line 3 as a separate strongly connected component

Thm 8: Each DFS tree of $DFS(G^T)$ is a strongly connected component of G

הוכחה

באינדוקציה על העלים שנלמדים ב $DFS(G^T)$

נניח ש k העלים הראשונים ב $DFS(G^T)$ מתאים

כ"א רכיב קשירות מצומצמת של G

נראה שגם העלה ה $k+1$ מתאים לרכיב

קשירות מצומצמת של G

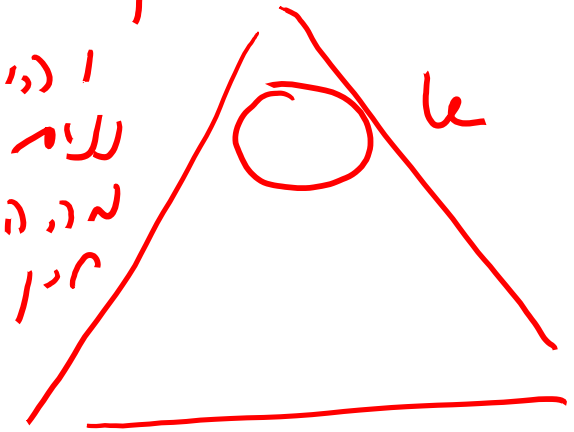
ובפרט u הוא הראש של העלה ה $k+1$

נניח ש u איננו רכיב קשירות מצומצמת וכי

מהנחה את לפחות G עדיין לא שייך

לרכיב קודם אחר מהם מסלול לכל

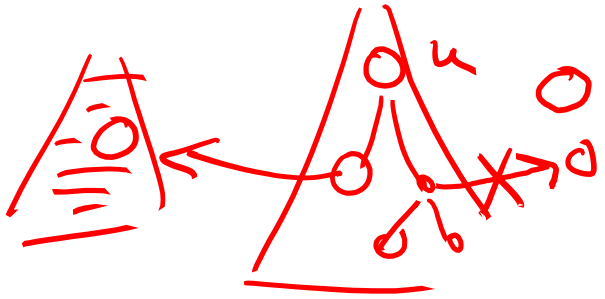
ולכל עליו u יש



בשלמה כל במתי לא יהיו בשבעים לא
בגלל שלא יהיו כלל המשקלות הלבן.

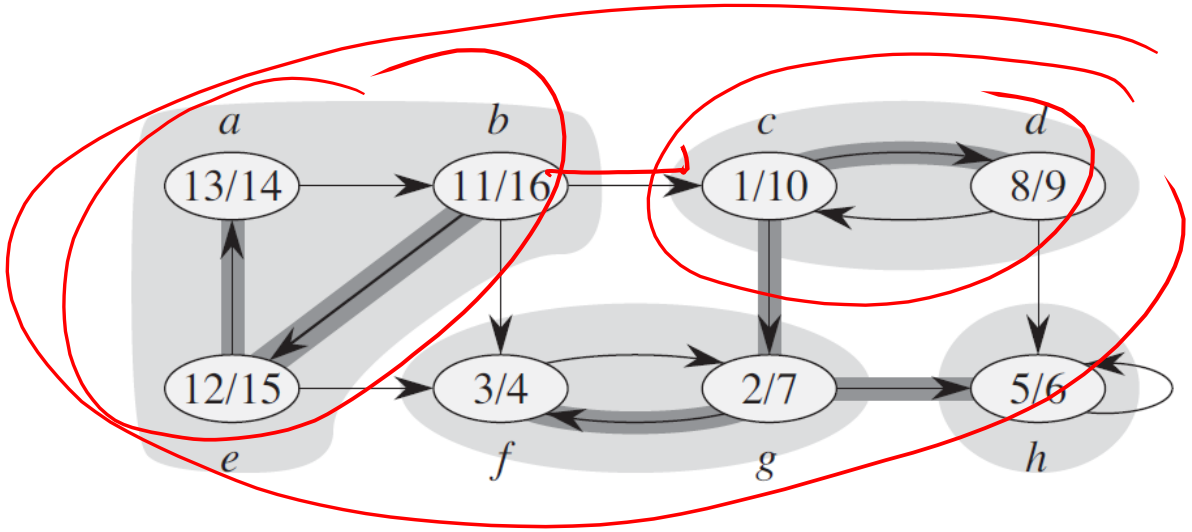
במתים אוינים לא ולמלאים בבכיה קשיים
שבמי הילה כמובן שלא יכנסו לעלות שנעשו לא

במתים אוינים לא אלה בבכיה
קשיים - לא עוד הילה אוינים לשניים
עי משקל מכוון מאל לפי מאפל 7/8.

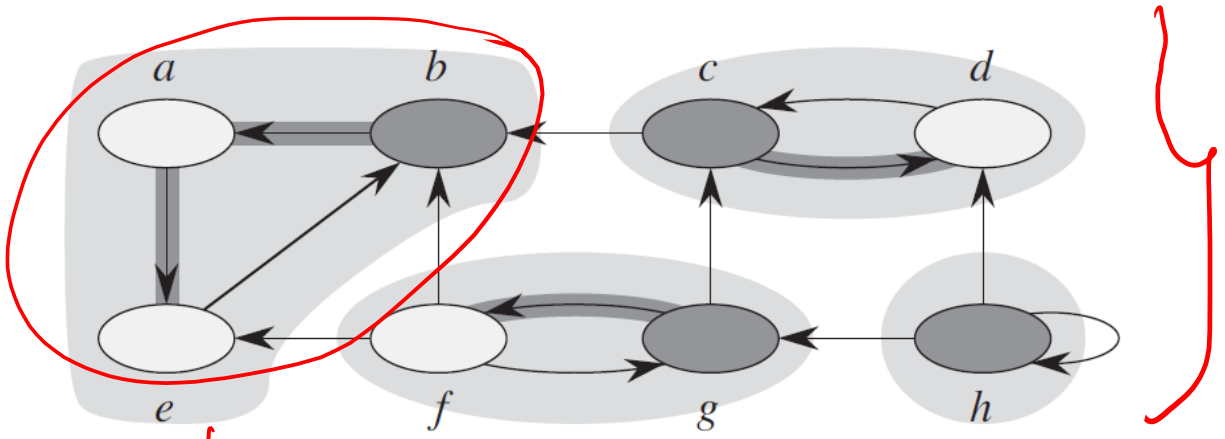


כי אחרי הוא בשר מצומצם בבכיה לא
לעצמות בבכיה $f(c) > f(c)$ לפי מאחרי לכן העלה
$k+1$ ויש בעיה מצומצם לא לא □

Computing $G^{SCC} = (V^{SCC}, E^{SCC})$

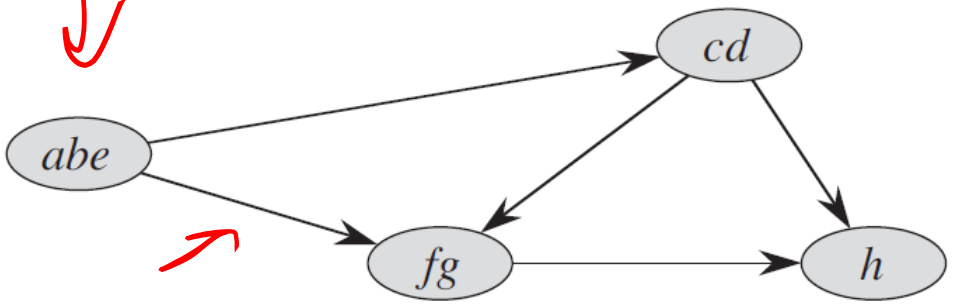


G



G^T

Remove parallel edges using an array..



בעיות רשמו אינם
מחברים אל G^{SCC} בעמן
שקשורים לאותה אותיות בעת יוצעת
רק צומת אל מסוים כבר.