

Probleme de conectivitate în grafuri neorientate



Muchii critice

Muchii critice

- ▶ G – graf neorientat
- ▶ $e \in E(G)$ **critică (punte, muchie de articulație)** = prin eliminarea ei crește numărul de componente conexe ale grafului

$$\text{nr componente } (G - e) > \text{nr. componente } (G)$$

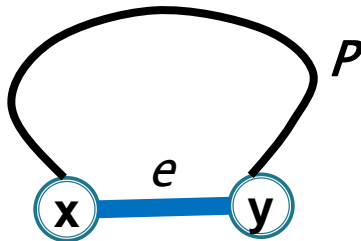
- ▶ Un graf conex fără punți se numește **2-muchie conex**.

Muchii critice

- ▶ O muchie este critică \Leftrightarrow nu este conținută într-un ciclu

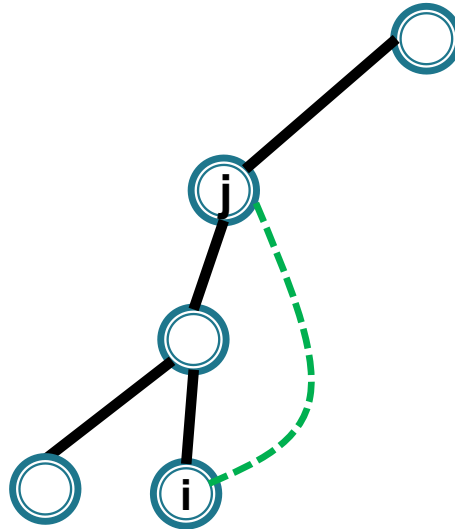
Demonstrație

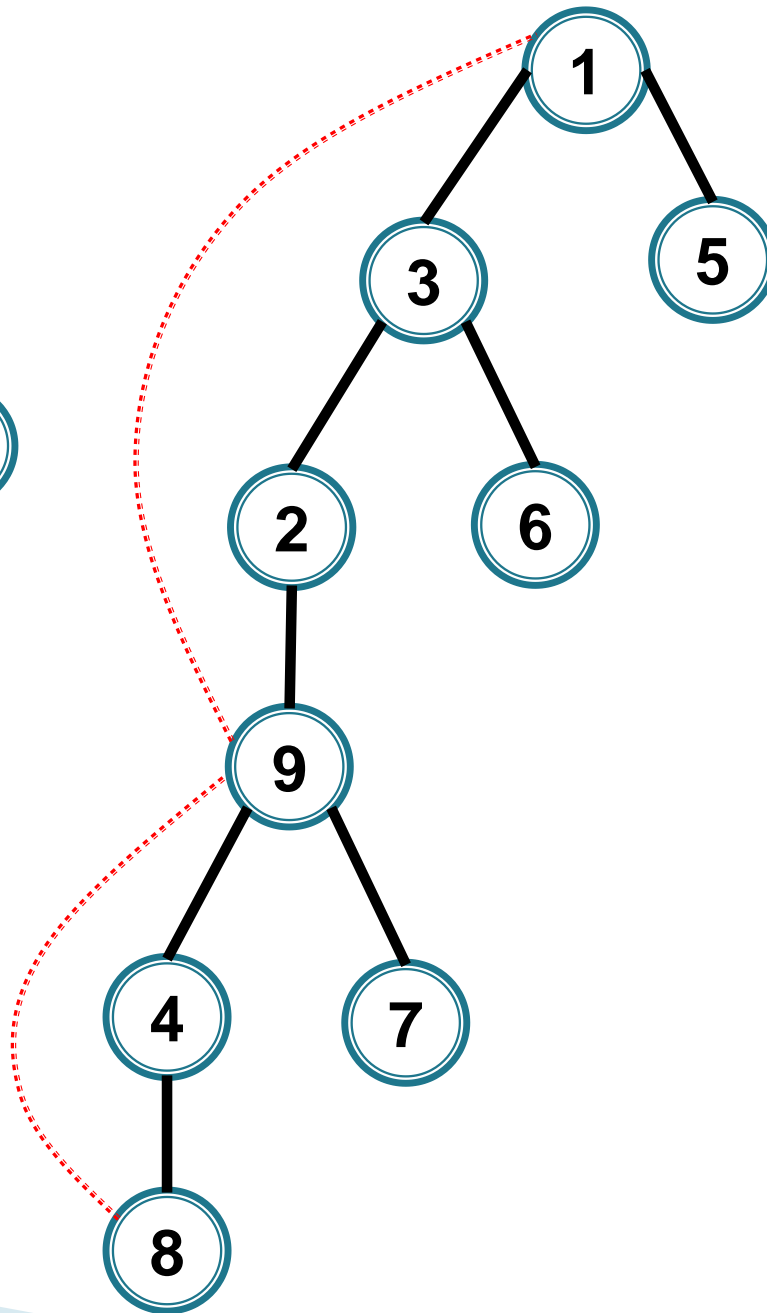
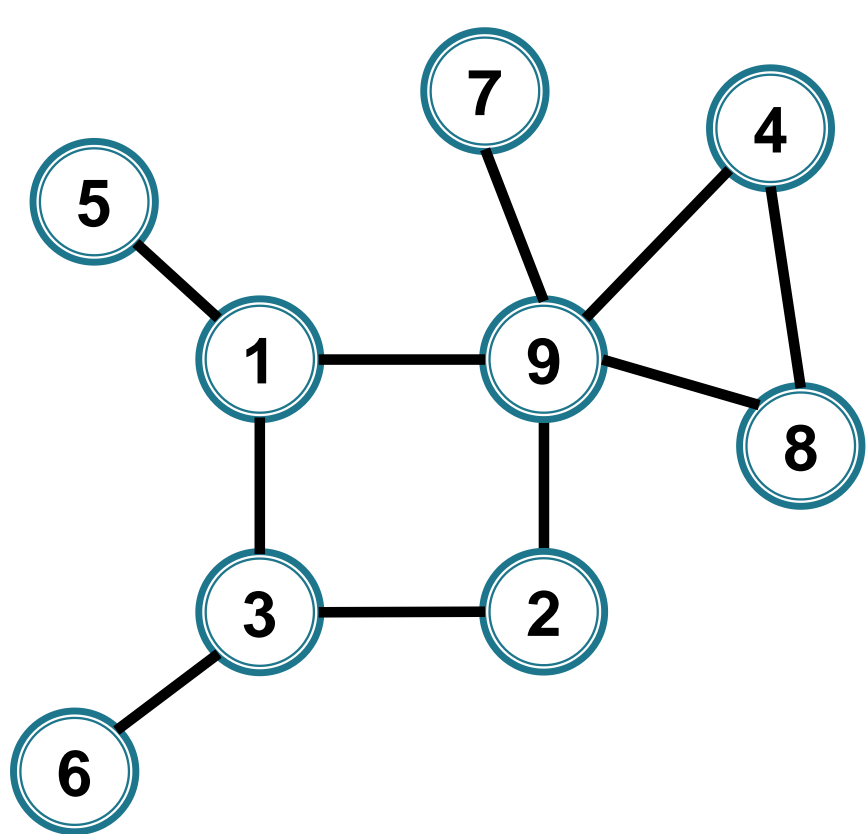
O muchie nu este critică \Leftrightarrow este conținută într-un ciclu



Muchii critice

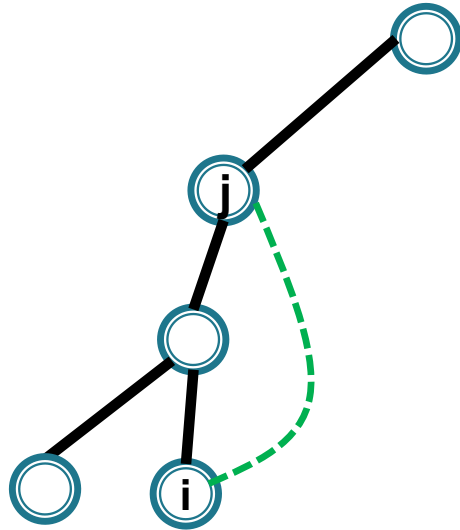
- ▶ Găsirea unui ciclu – parcurgere DF
 - **muchii de avansare** – ale arborelui DF (memorat cu vector tata), prin care se descoperă vârfuri noi
 - **muchii de întoarcere** – închid ciclu, nu pot fi critice





Muchii critice

- ▶ Găsirea unui ciclu – parcurgere DF – $O(n)$
 - muchii de avansare – ale arborelui DF (memorat cu vector tata), prin care se descoperă vârfuri noi
 - muchii de întoarcere – închid ciclu, nu pot fi critice



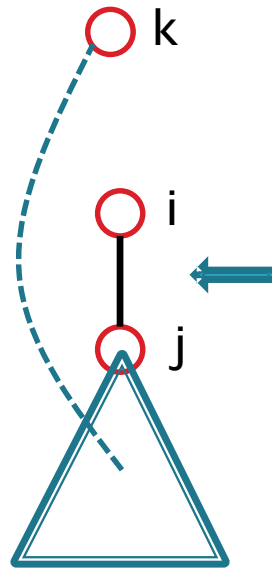
Doar muchiile de avansare pot fi critice

Muchii critice

Cum testăm dacă o muchie de avansare (i,j) este critică?



- nu este conținută într-un ciclu închis de o muchie de întoarcere



Muchii critice

O muchie de avansare (i,j) este critică

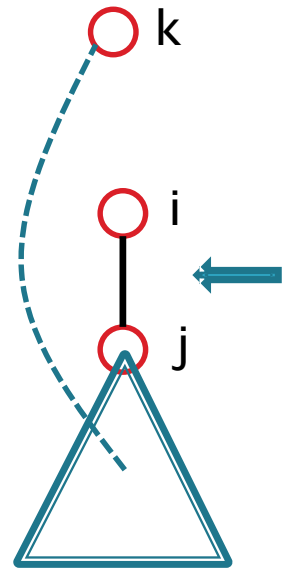
\Leftrightarrow

nu este conținută într-un ciclu închis de o muchie de întoarcere

\Leftrightarrow

nu există nicio muchie de întoarcere cu

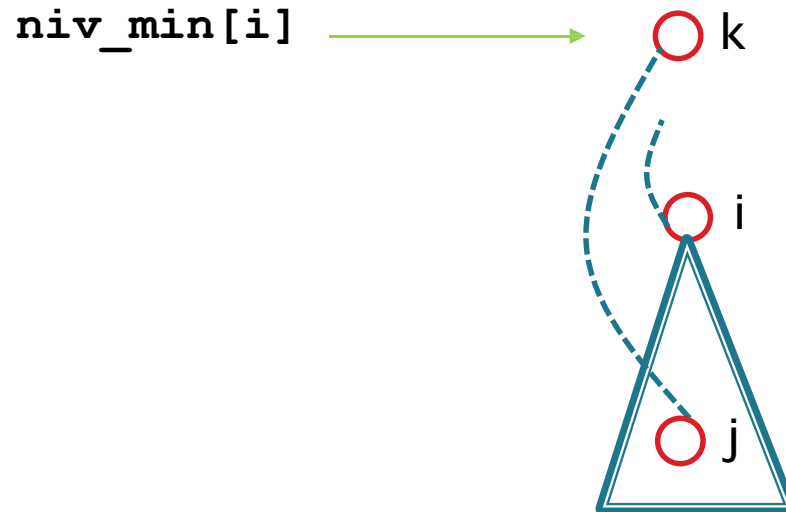
- o extremitate **în j sau într-un descendent al lui j** și
- cealaltă extremitate **în i sau într-un ascendent al lui i** (într-un vârf de pe un nivel mai mic sau egal cu nivelul lui i)



Muchii critice

Memorăm pentru fiecare vârf i :

$\text{niv_min}[i]$ = *intuitiv: cât de sus putem ajunge din i mergând în sensul parcurgerii DF*

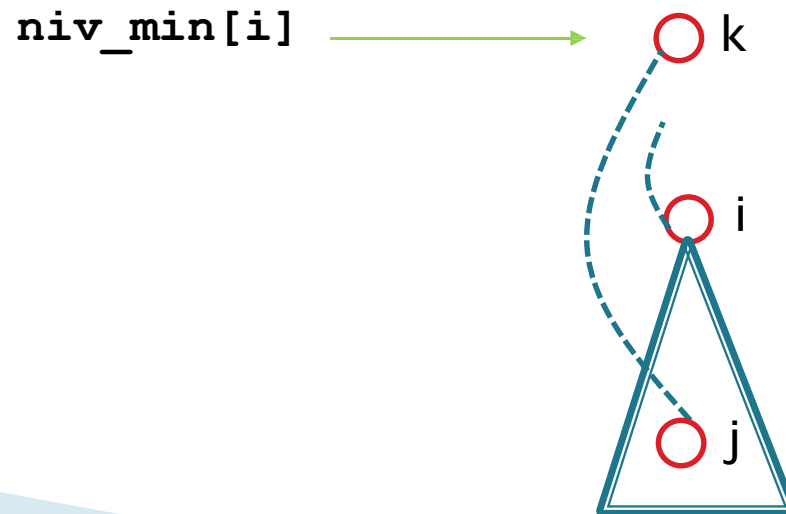


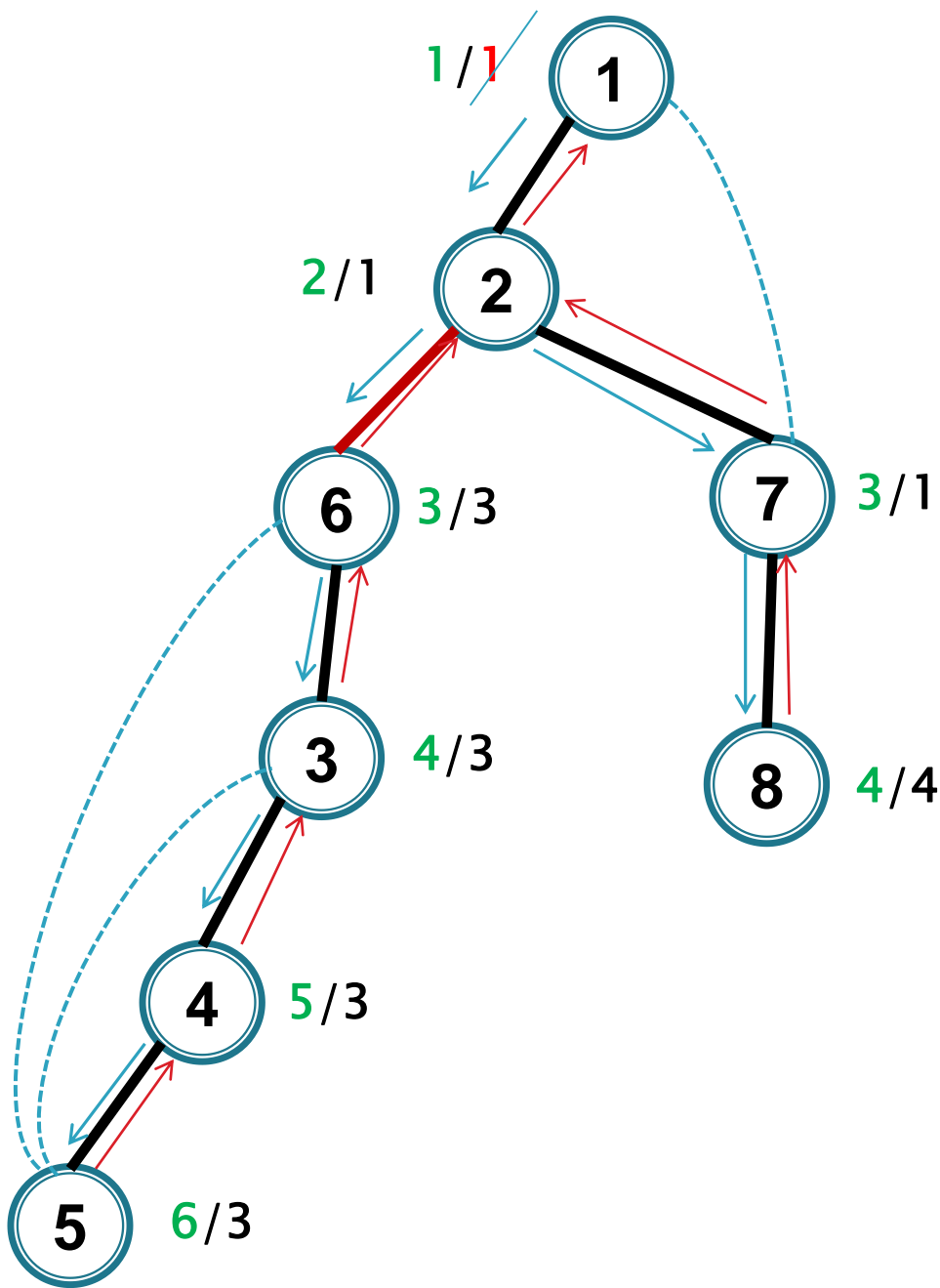
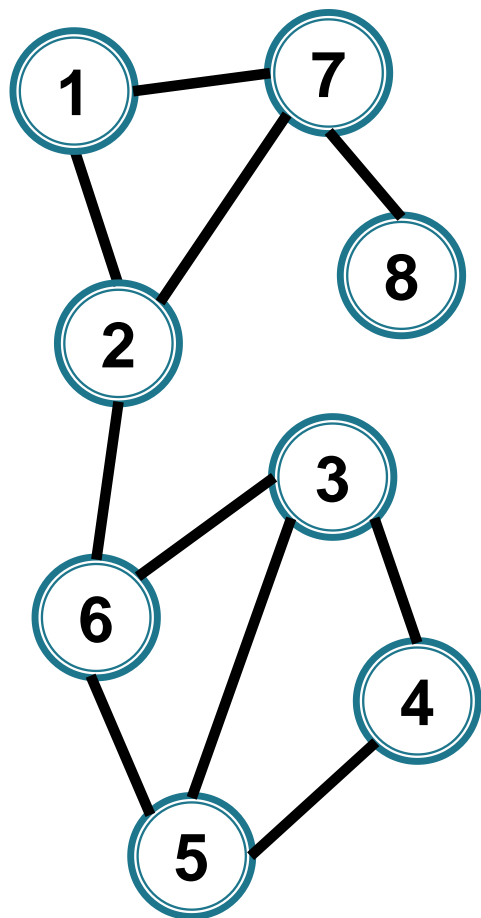
Muchii critice

Memorăm pentru fiecare vârf i :

$niv_min[i]$ = nivelul minim al unui vârf care este extremitate a unei muchii de întoarcere din i sau dintr-un descendent al lui i

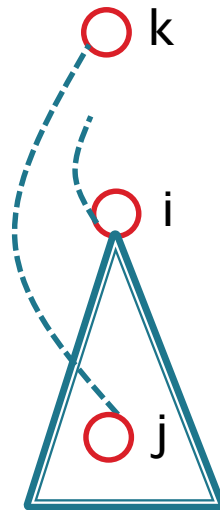
= nivelul minim la care se închide un ciclu elementar care conține vârful i (printr-o muchie de întoarcere)





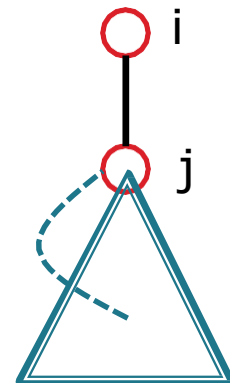
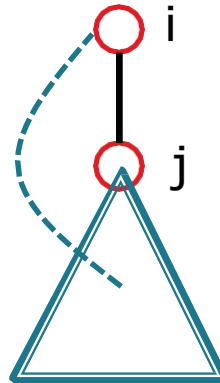
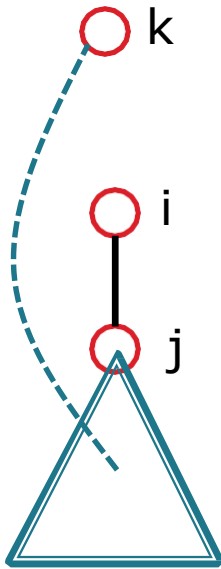
Muchii critice

- ▶ $\text{nivel}[i] = \text{nivelul lui } i \text{ în arborele DF}$
- ▶ $\text{niv_min}[i] = \min \{ \text{nivel}[i], A, B \}$
 - $A = \min \{ \text{nivel}[k] \mid i \text{ k muchie de întoarcere} \}$
 - $B = \min \{ \text{nivel}[k] \mid j \text{ descendent al lui } i, j \text{ k muchie de întoarcere} \}$



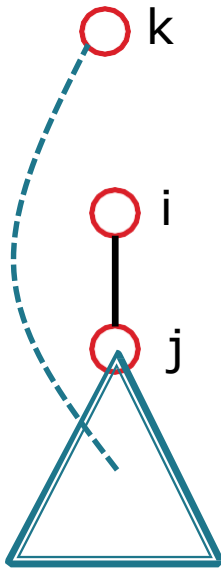
Muchii critice

O muchie de avansare ij este critică \Leftrightarrow



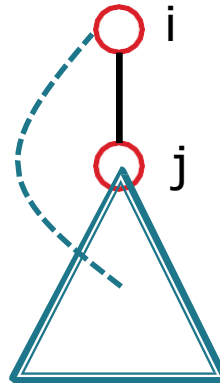
Muchii critice

O muchie de avansare ij este critică \Leftrightarrow



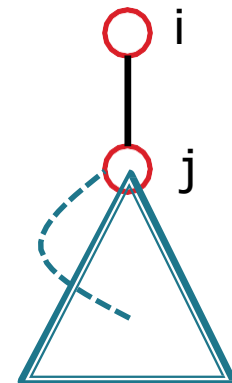
NU este critică

$\text{niv_min}[j] < \text{nivel}[i]$



NU este critică

$\text{niv_min}[j] = \text{nivel}[i]$

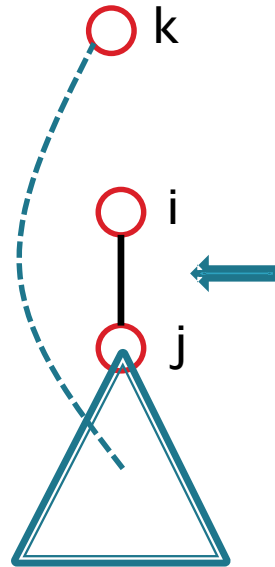


ESTE critică

$\text{niv_min}[j] > \text{nivel}[i]$

Muchii critice

O muchie de avansare ij este critică $\Leftrightarrow \text{niv_min}[j] > \text{nivel}[i]$



Muchii critice

Cum calculăm eficient $\text{niv_min}[i]$?

$$\text{niv_min}[i] = \min \{ \text{nivel}[i], A, B \}$$

$$A = \min \{ \text{nivel}[k] \mid i \text{ k muchie de întoarcere} \}$$

$$B = \min \{ \text{nivel}[k] \mid j \text{ descendent al lui } i, \\ j \text{ k muchie de întoarcere} \}$$



B se poate calcula recursiv

Muchii critice

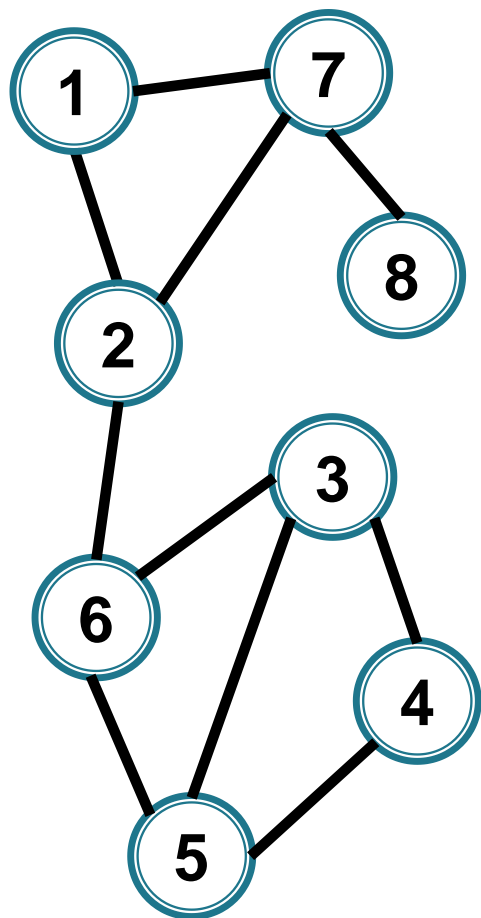
Cum calculăm eficient $niv_min[i]$?

$$niv_min[i] = \min \{ nivel[i], A, B \}$$

$$A = \min \{ nivel[k] \mid i \rightarrow k \text{ muchie de întoarcere} \}$$

$$B = \min \{ nivel[k] \mid j \text{ descendent al lui } i, \\ j \rightarrow k \text{ muchie de întoarcere} \}$$

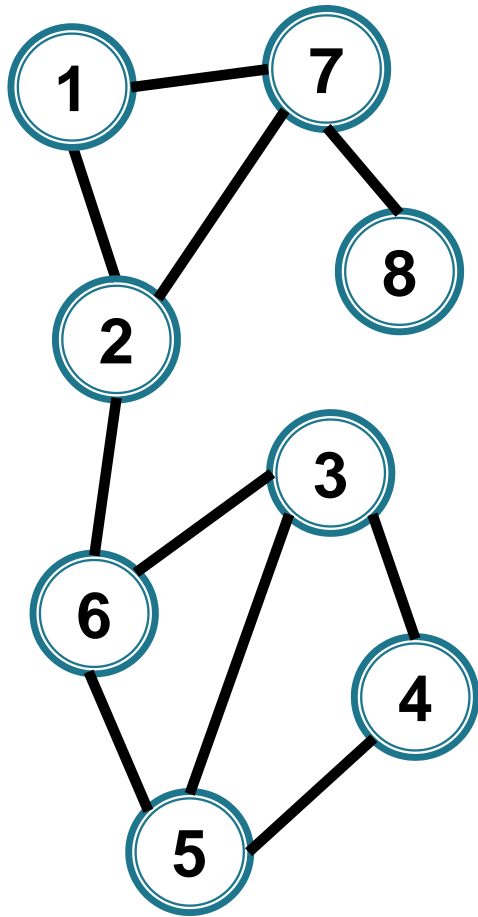
$$B = \min \{ niv_min[j] \mid j \text{ fiu al lui } i \}$$



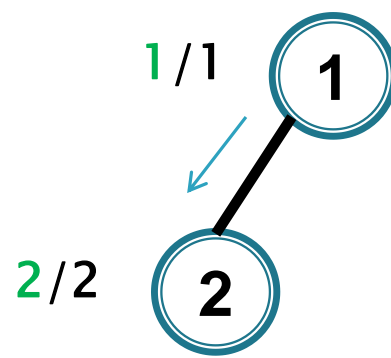
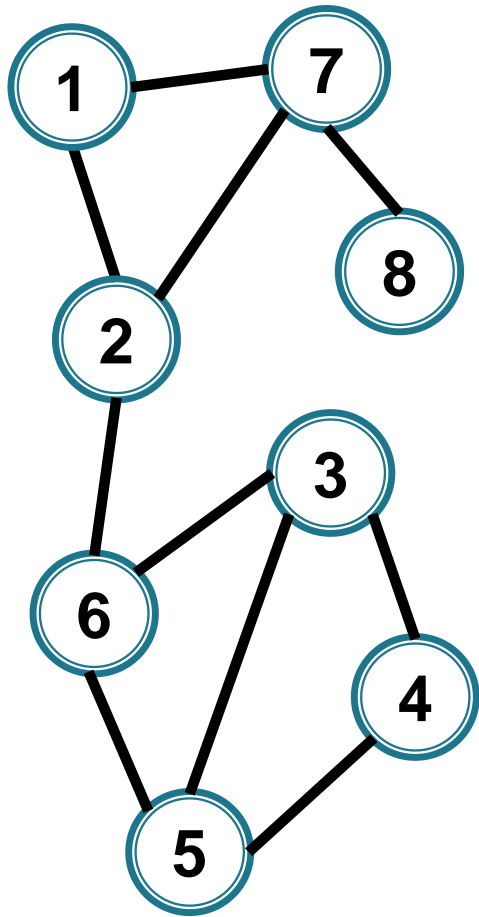
nivel/niv_min

1/1

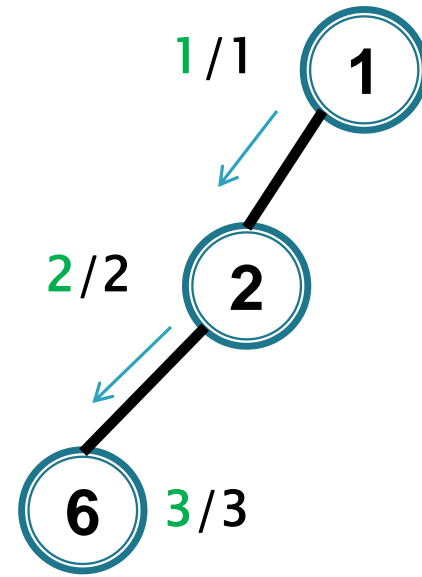
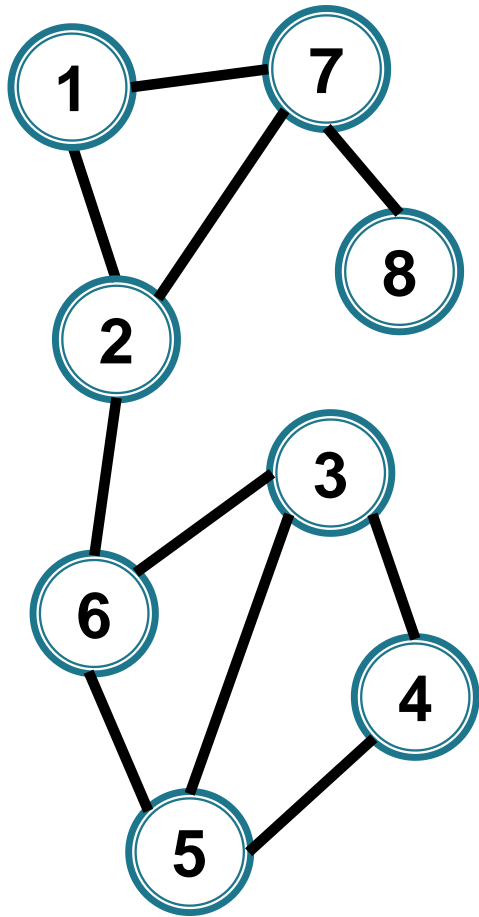
1



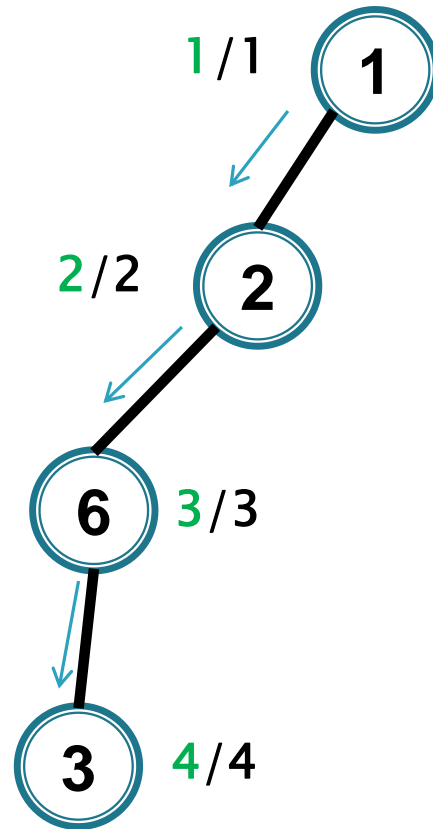
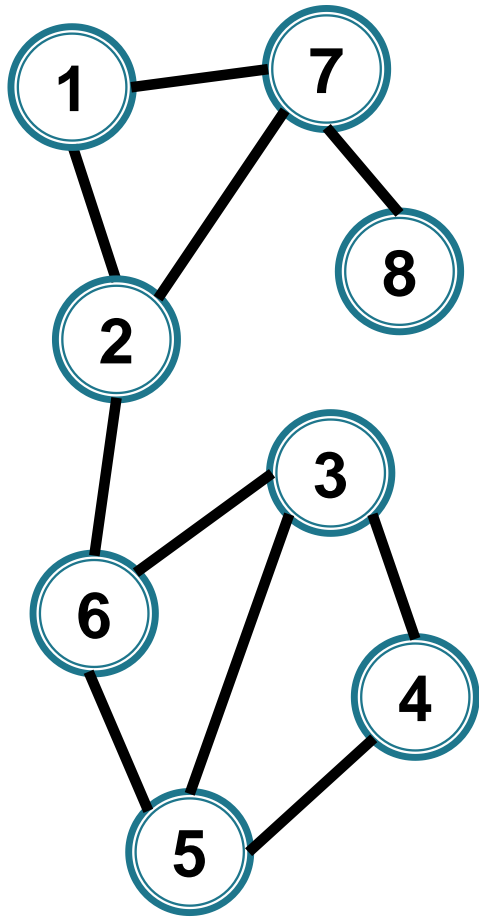
nivel/niv_min



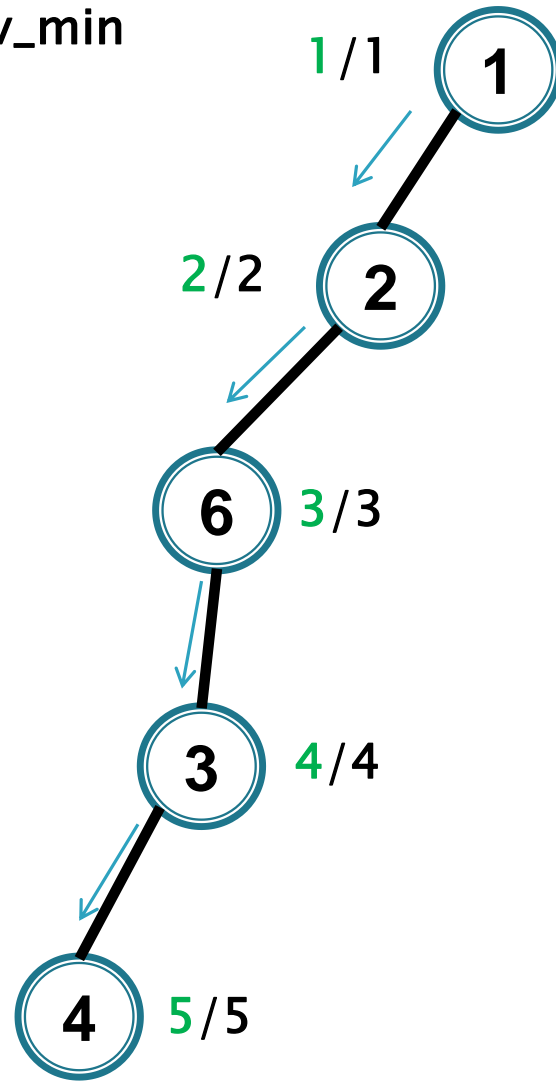
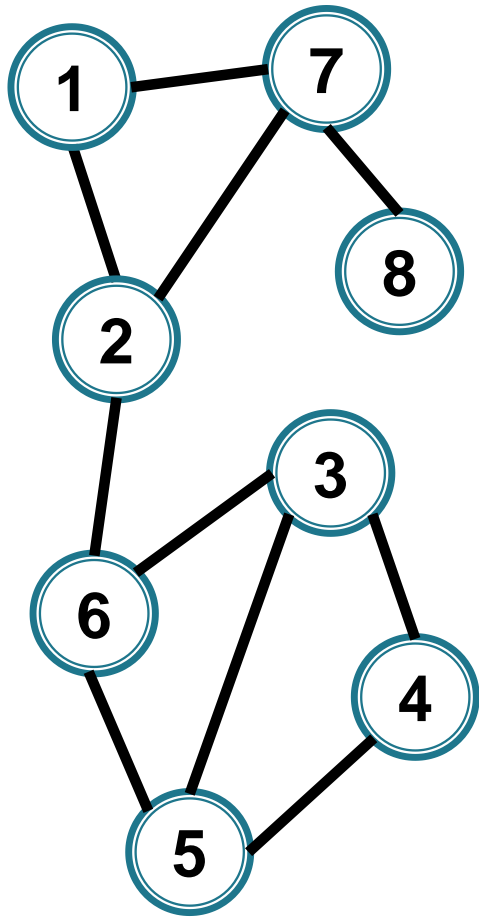
nivel/niv_min

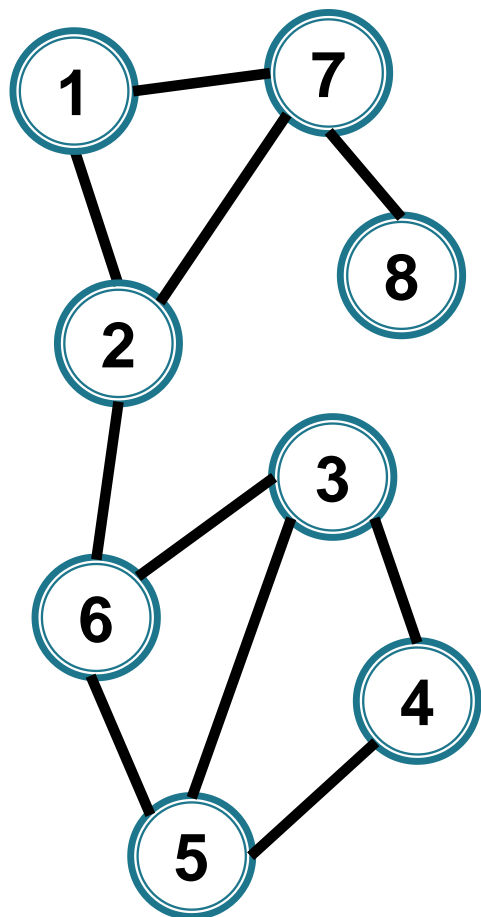


nivel/niv_min

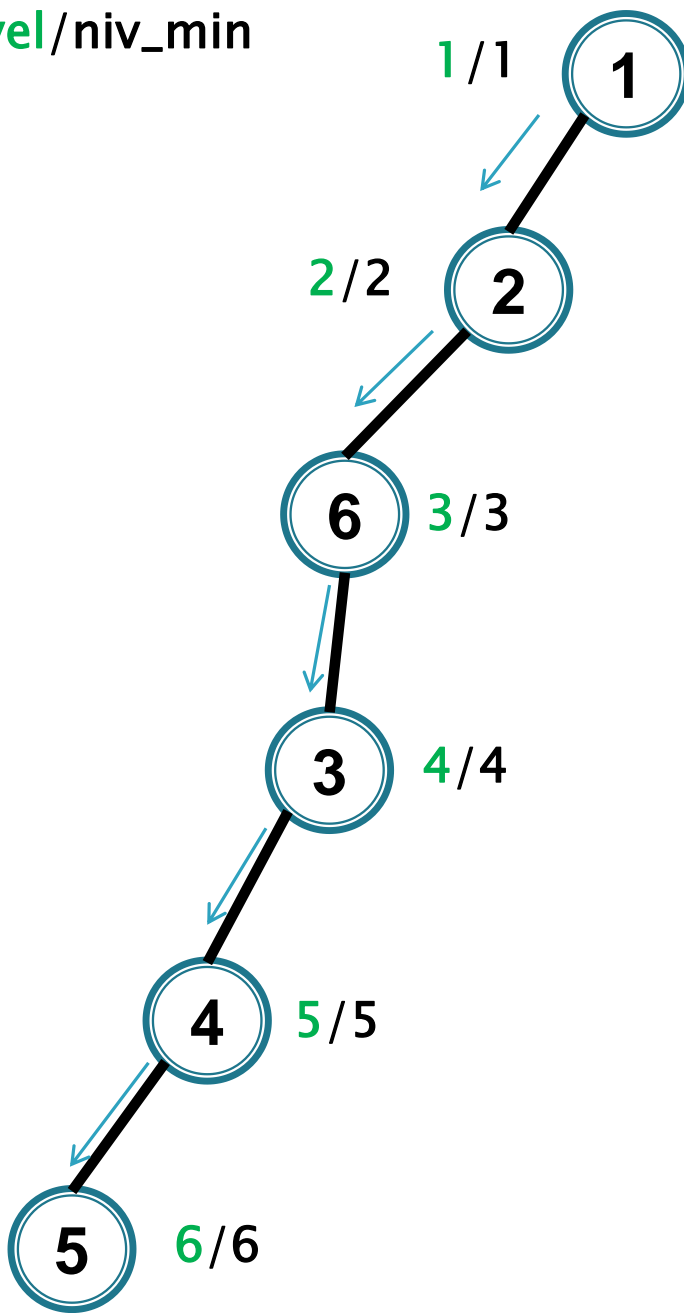


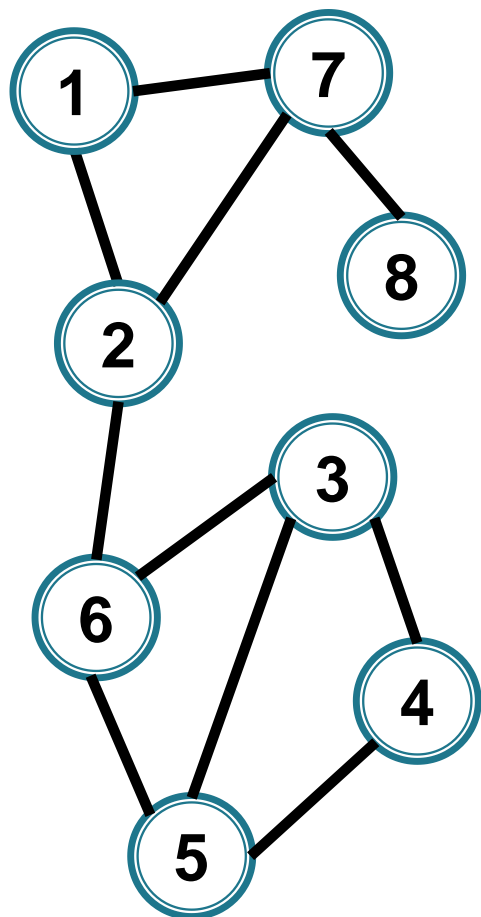
nivel/niv_min



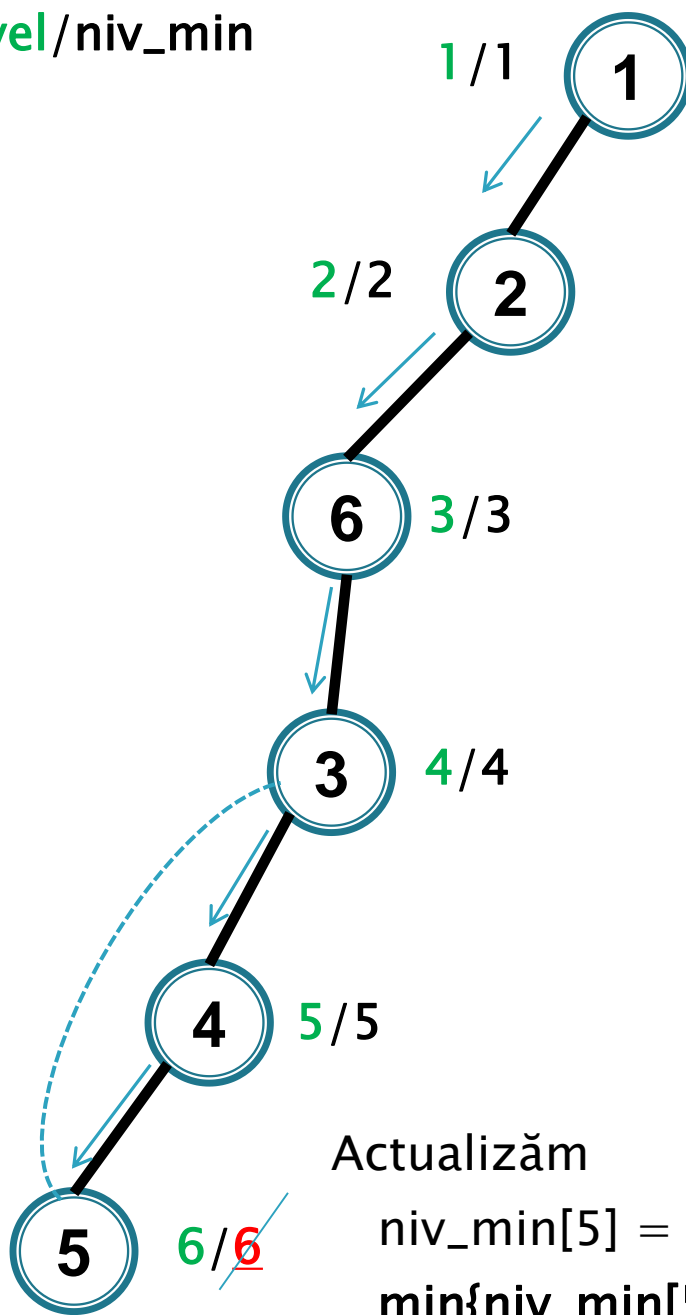


nivel/niv_min





nivel/niv_min

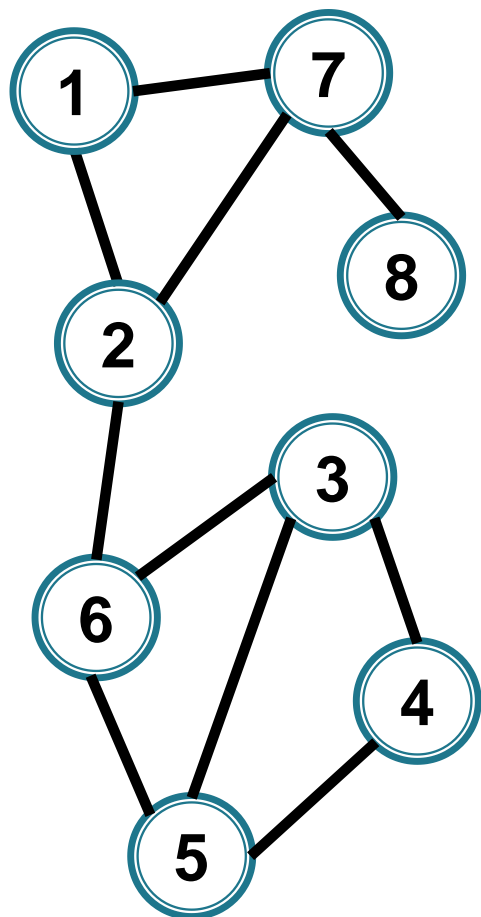


Actualizăm

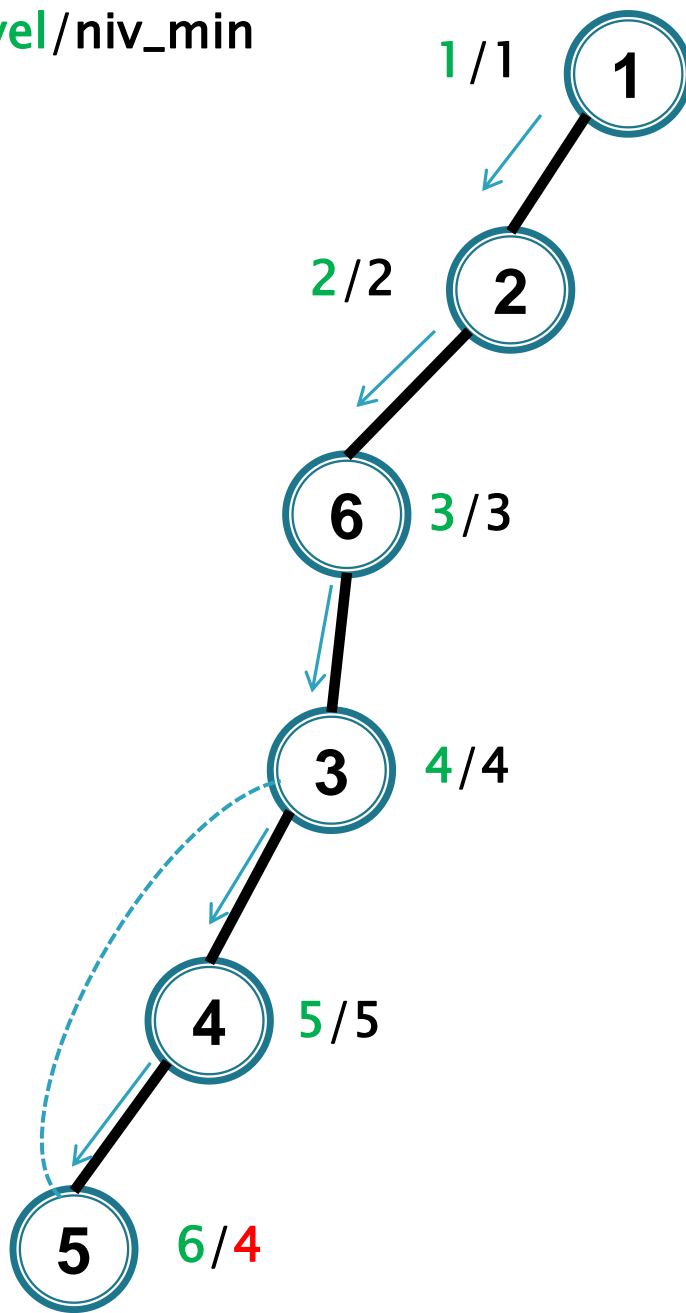
$\text{niv_min}[5] =$

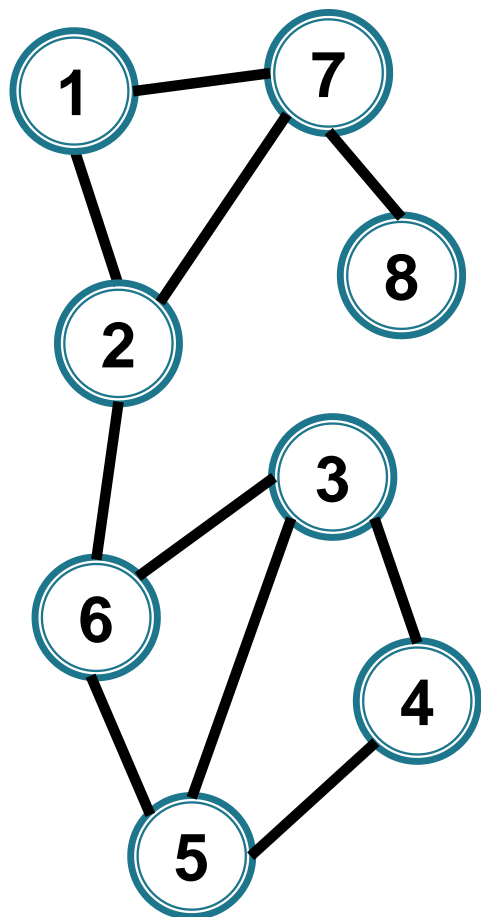
$\min\{\text{niv_min}[5], \text{nivel}[3]\}$

(cazul A)

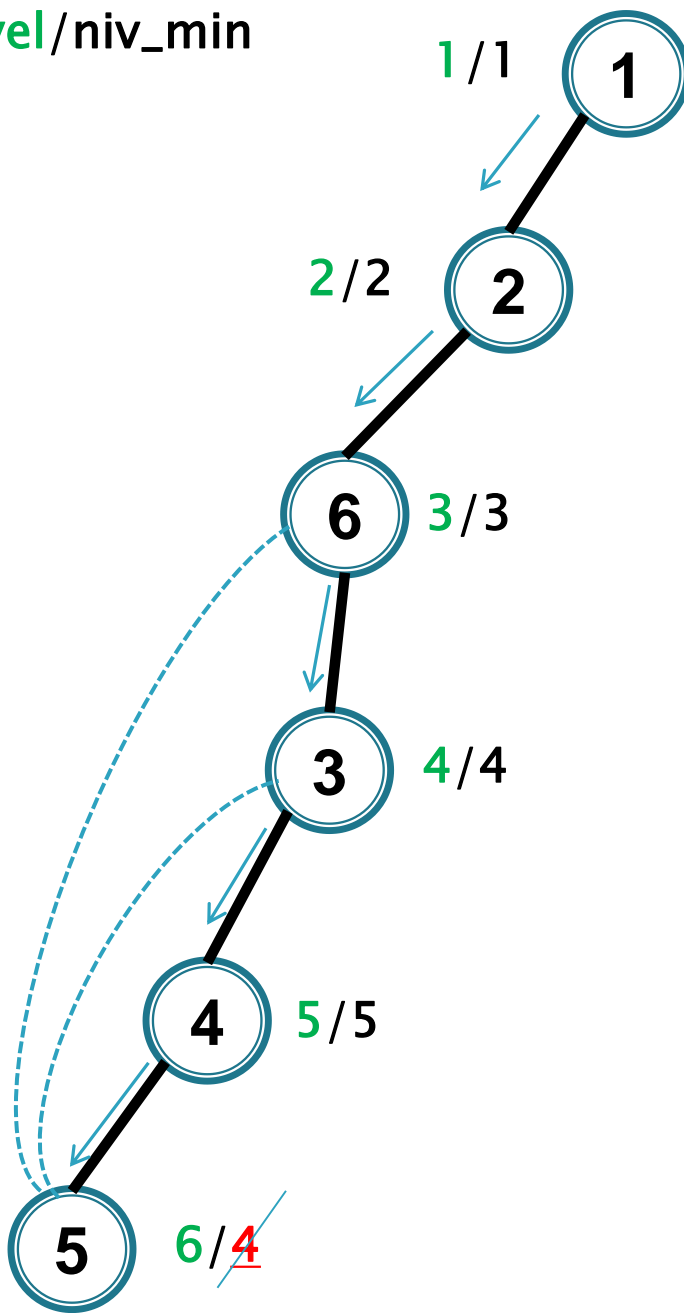


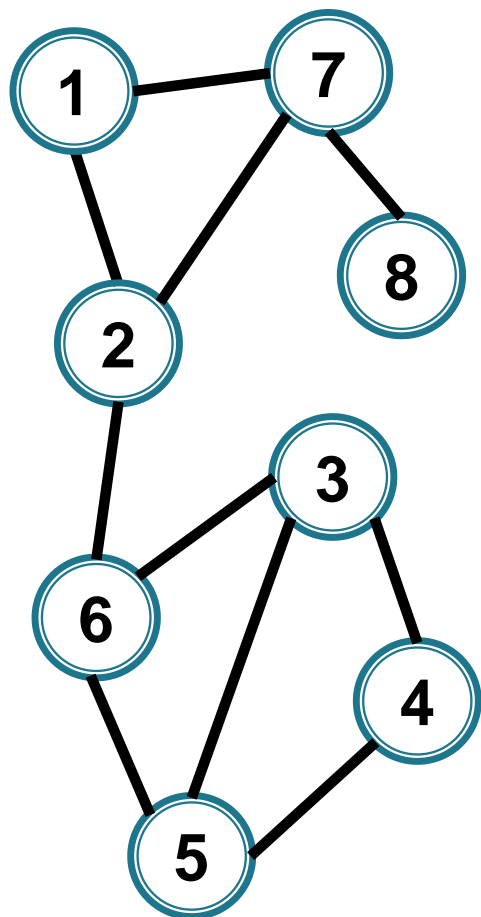
nivel/niv_min



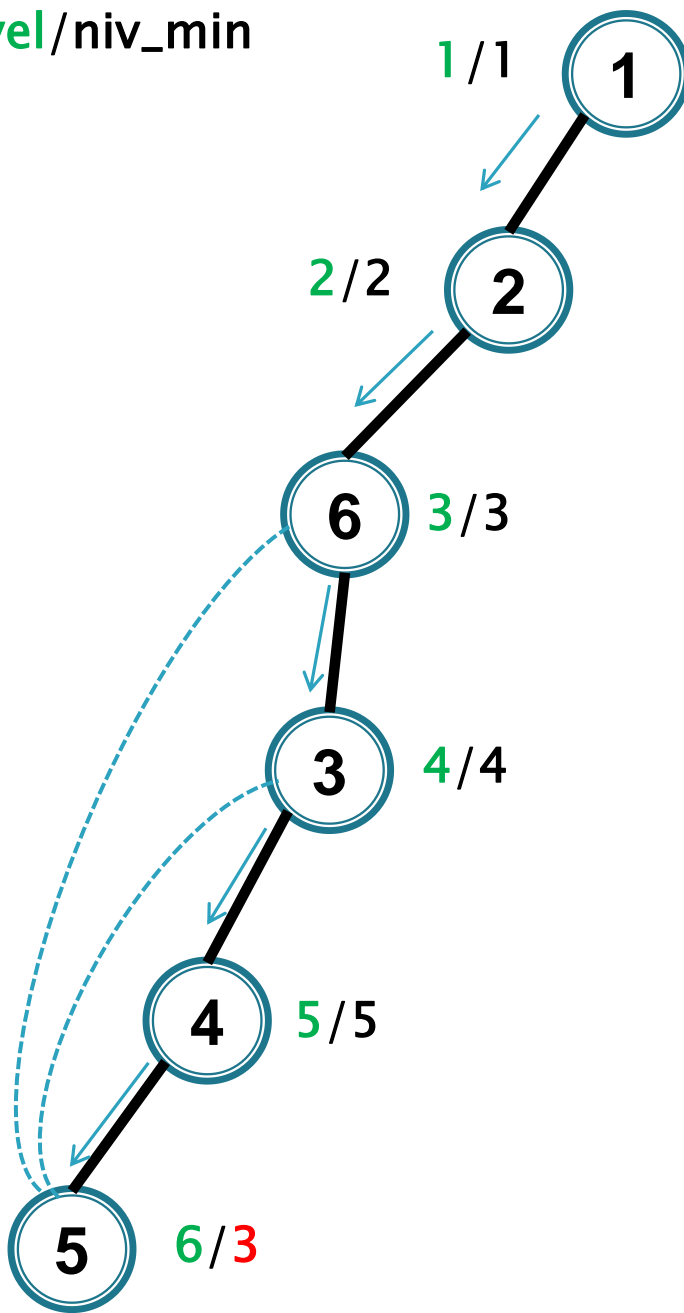


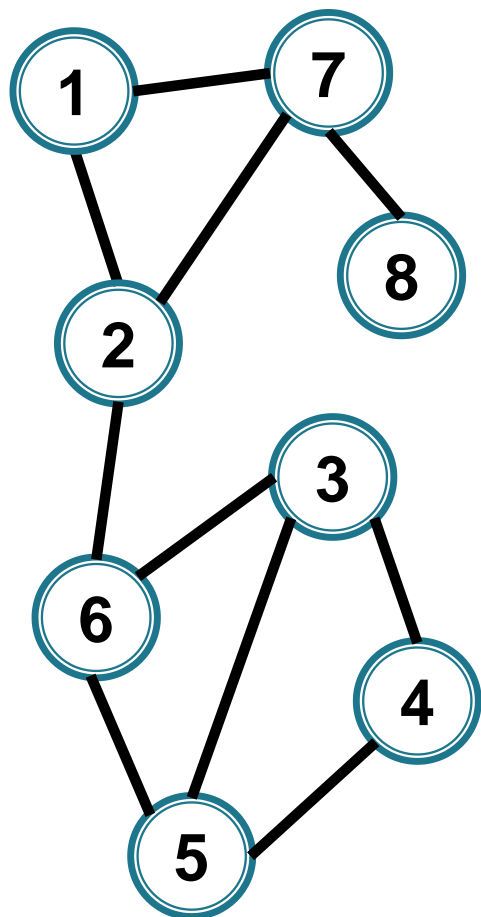
nivel/niv_min



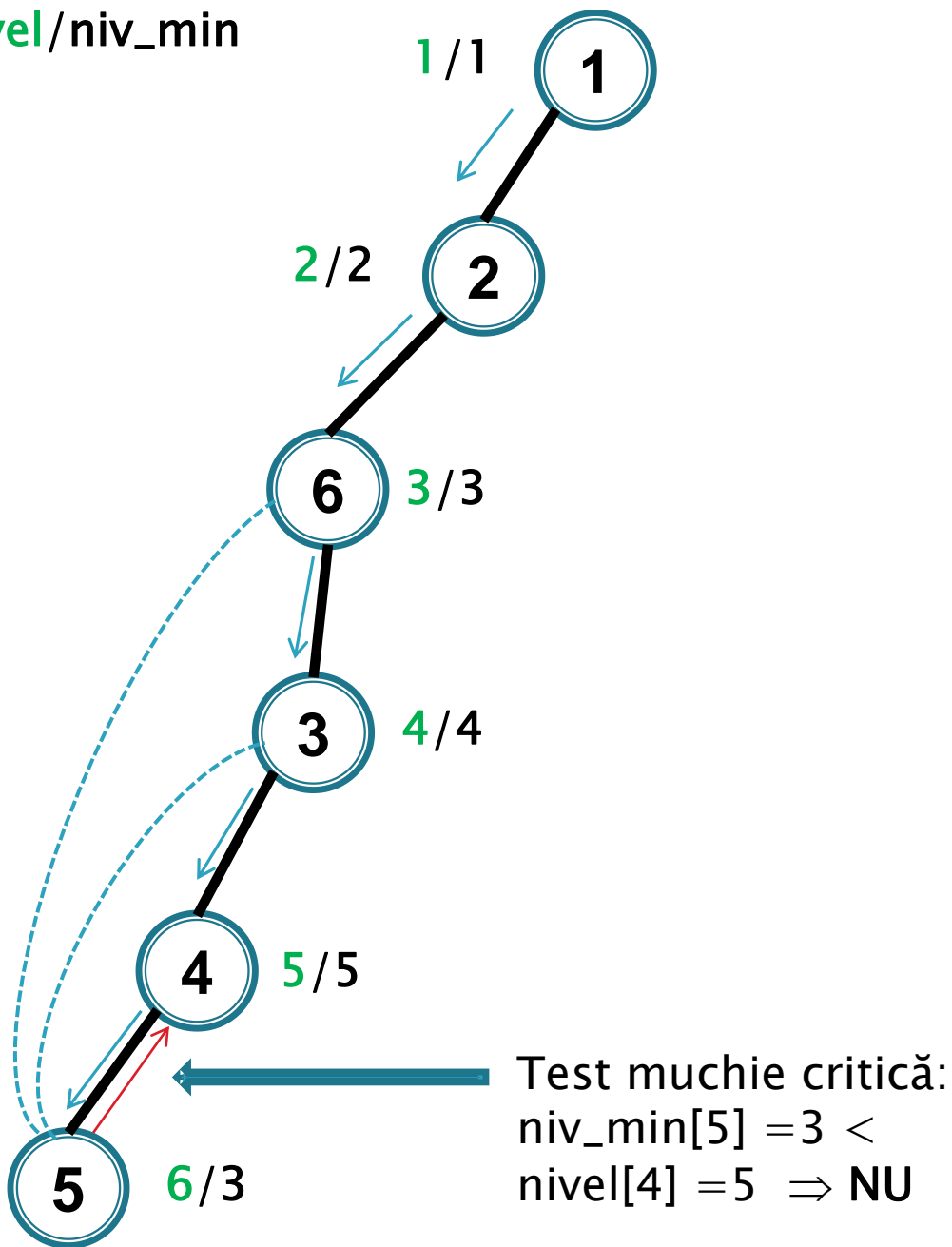


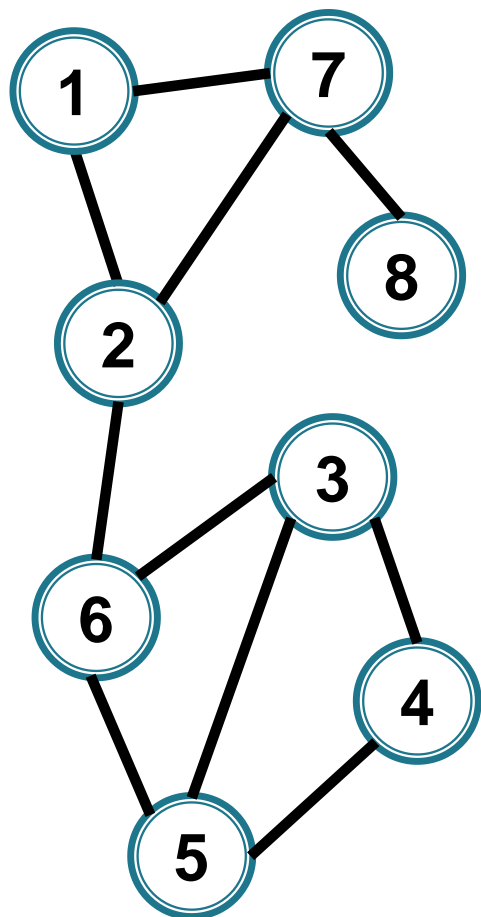
nivel/niv_min



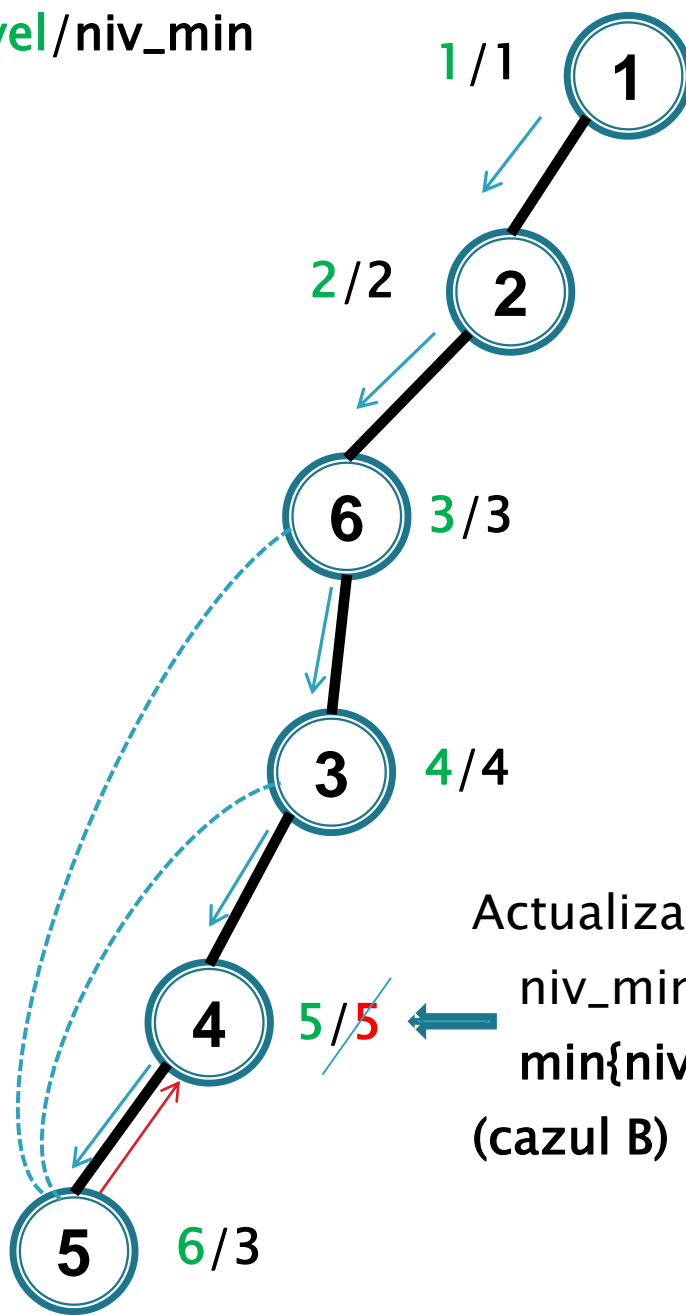


nivel/niv_min





nivel/niv_min

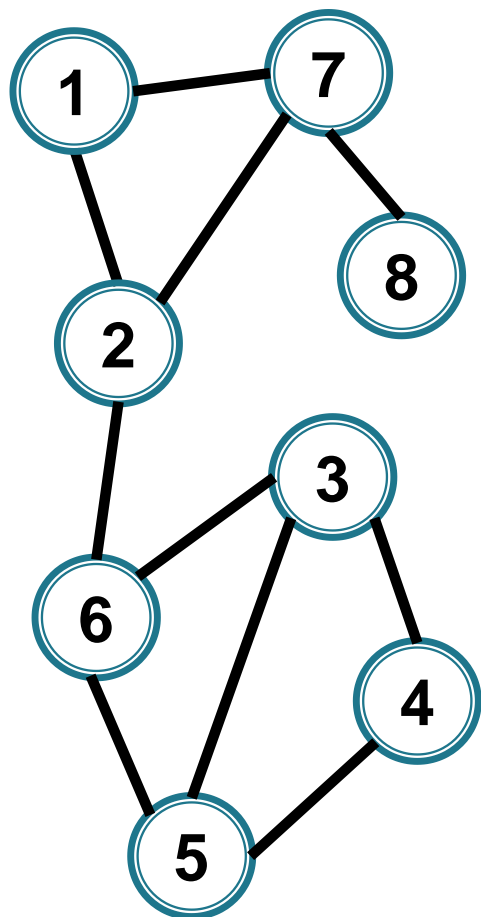


Actualizam

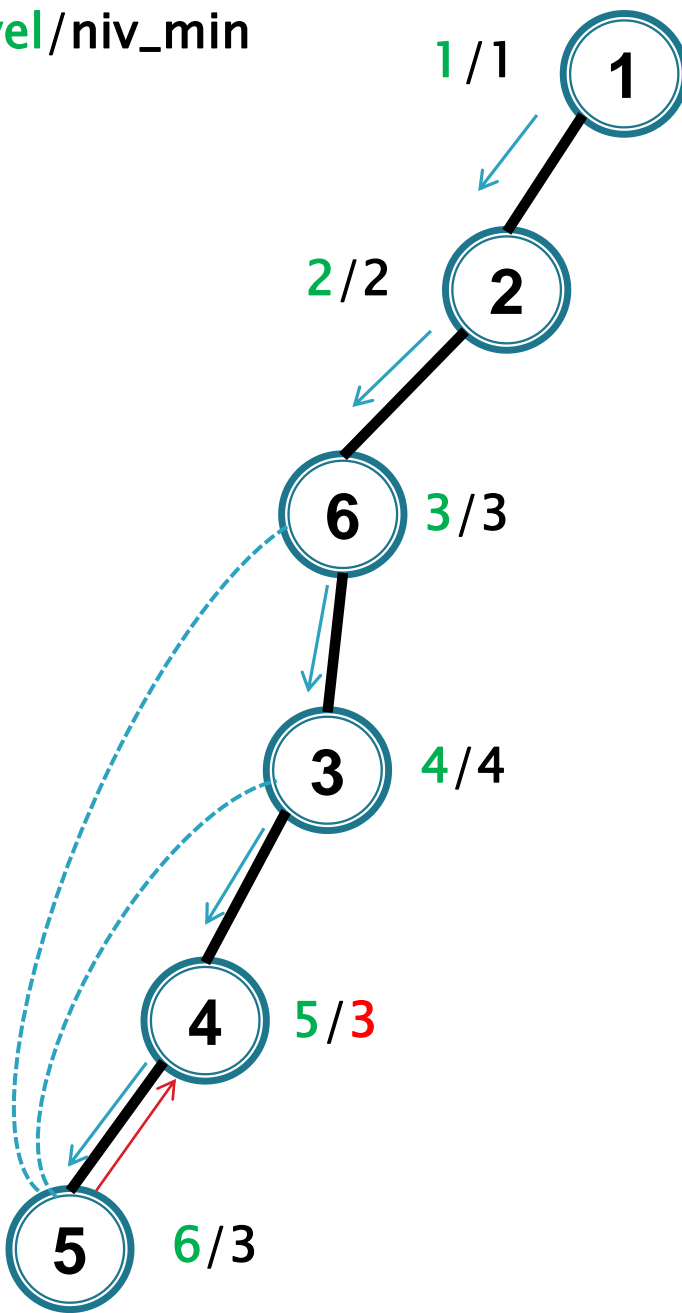
$niv_min[4] =$

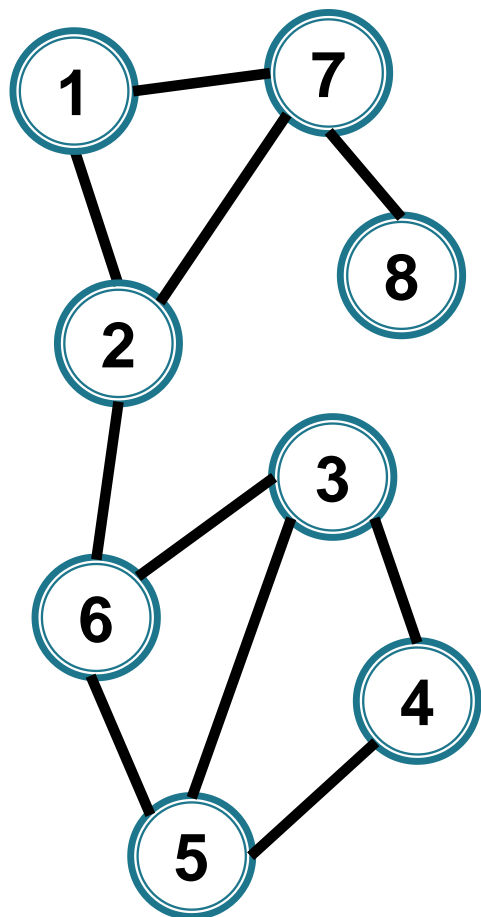
$\min\{niv_min[4], niv_min[5]\}$

(cazul B)

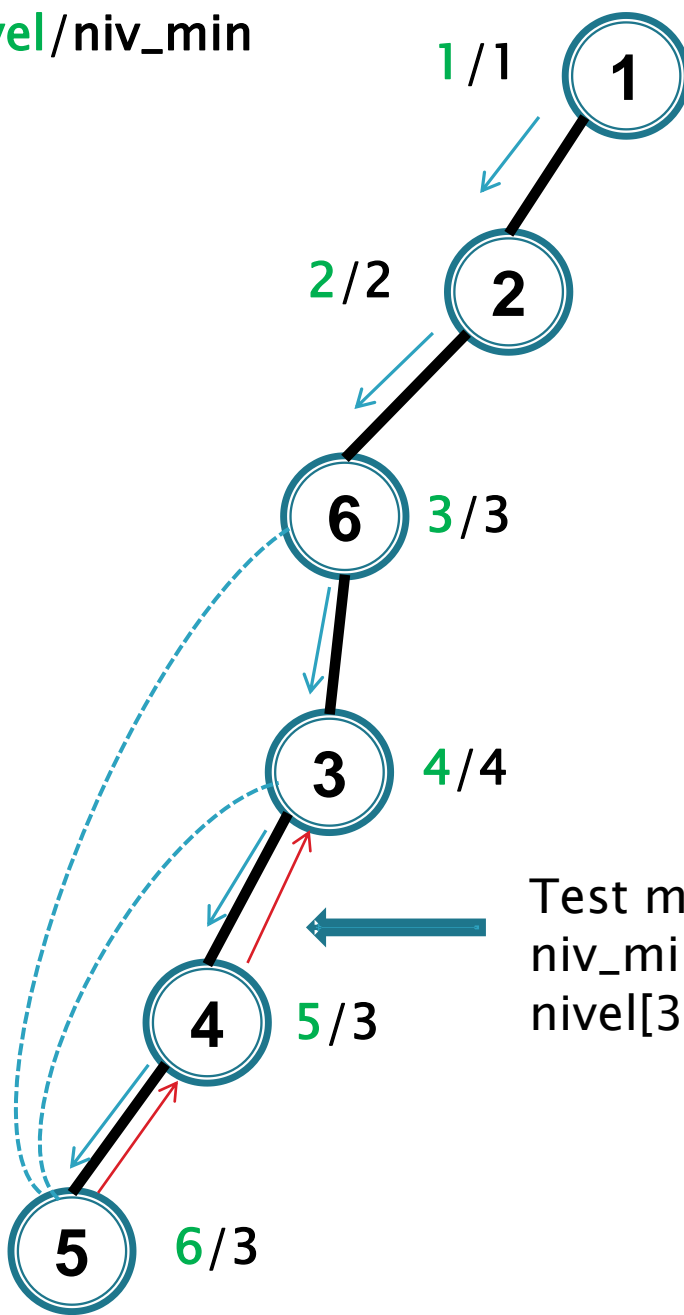


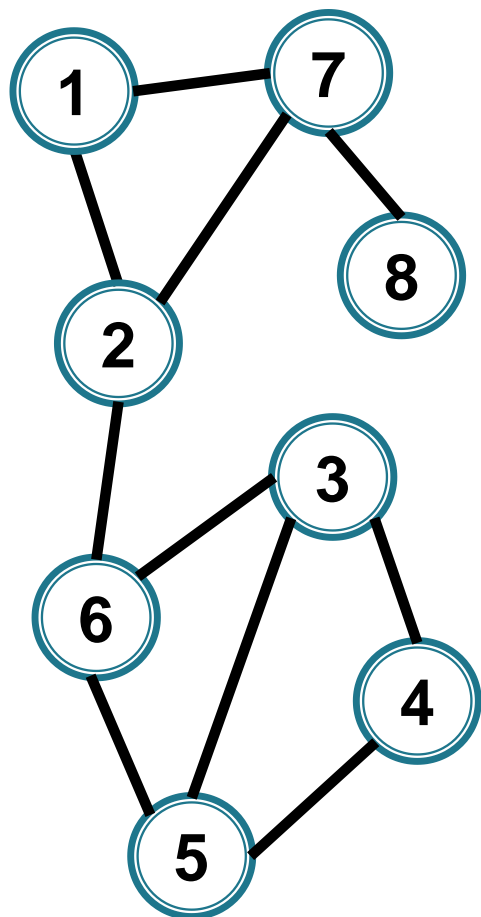
nivel/niv_min



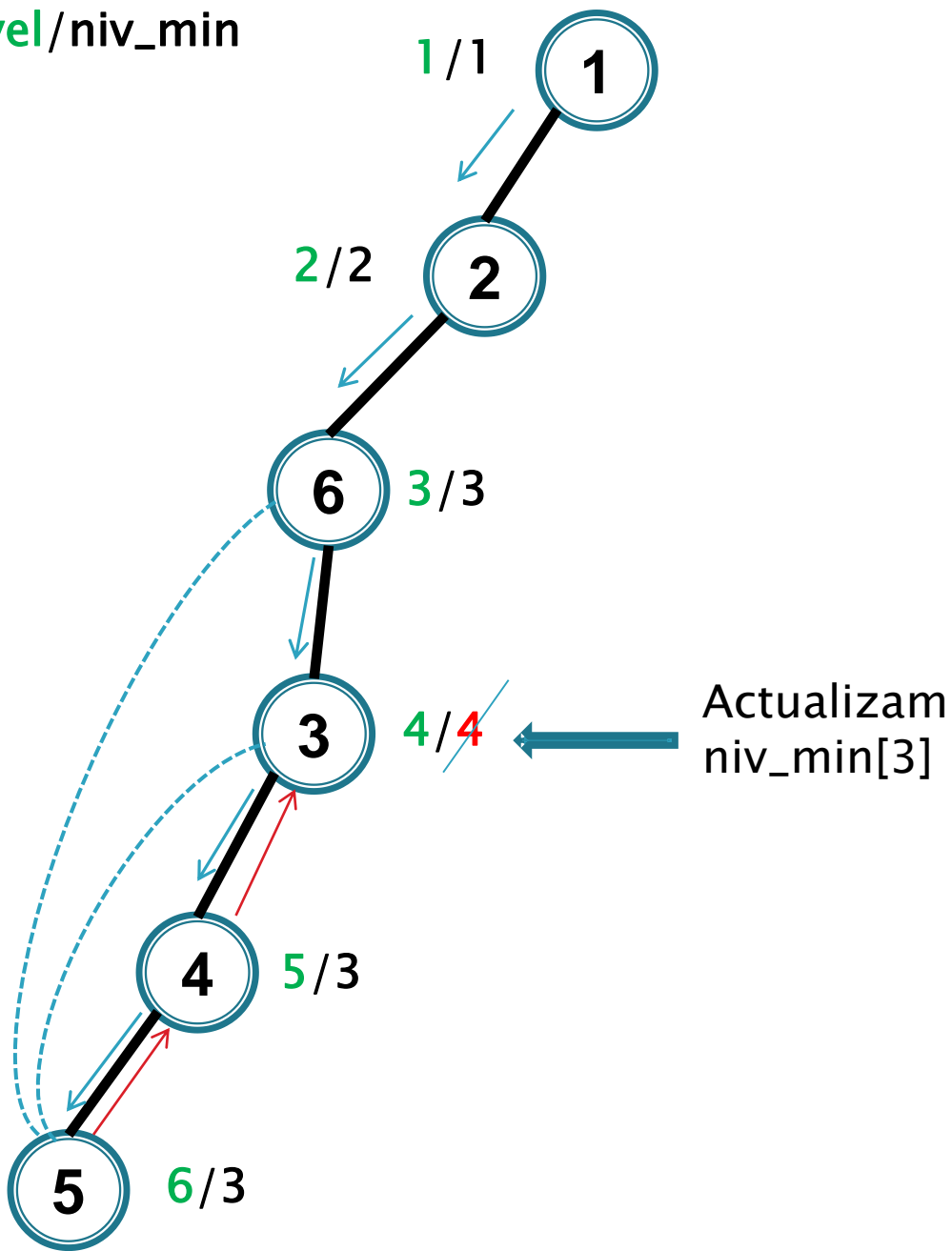


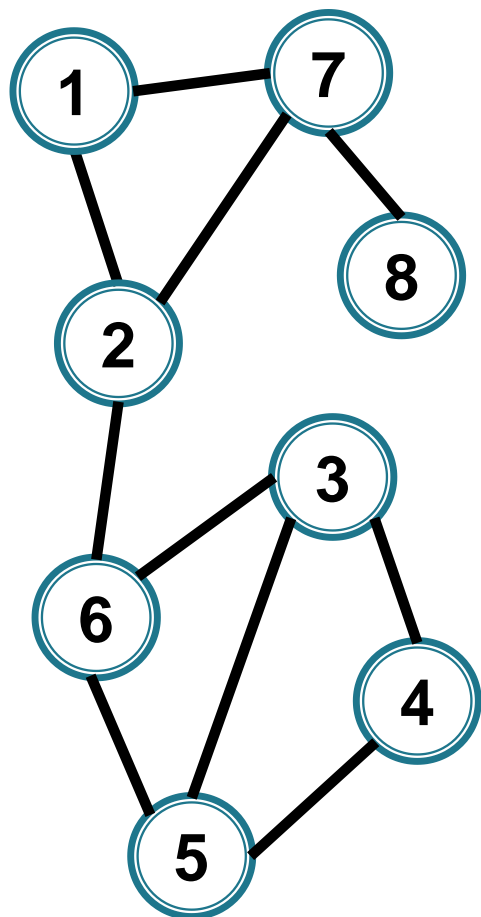
nivel/niv_min



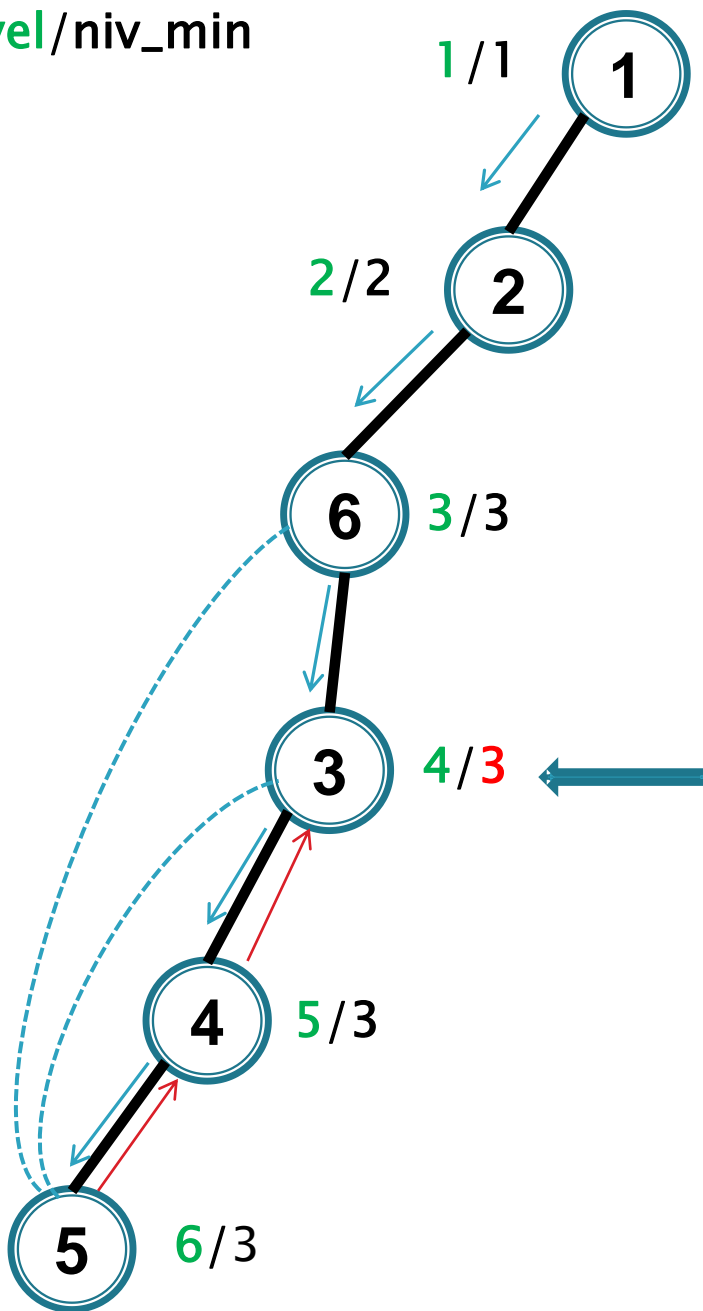


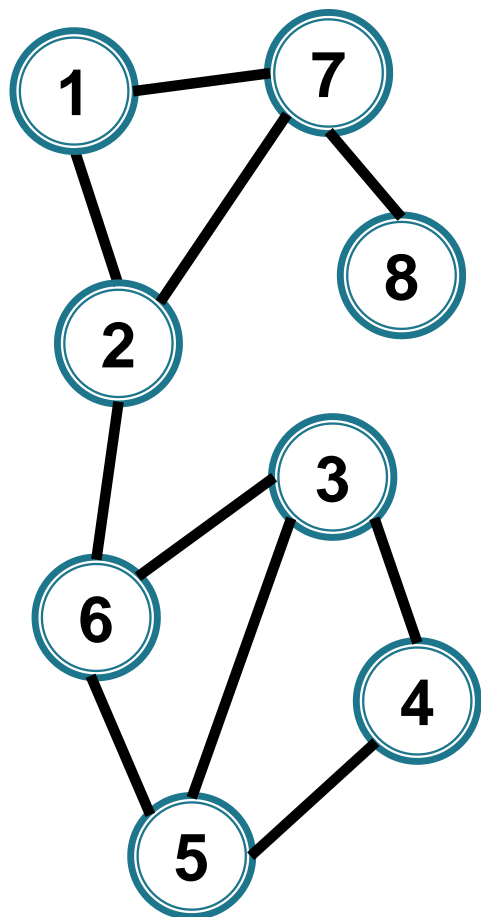
nivel/niv_min



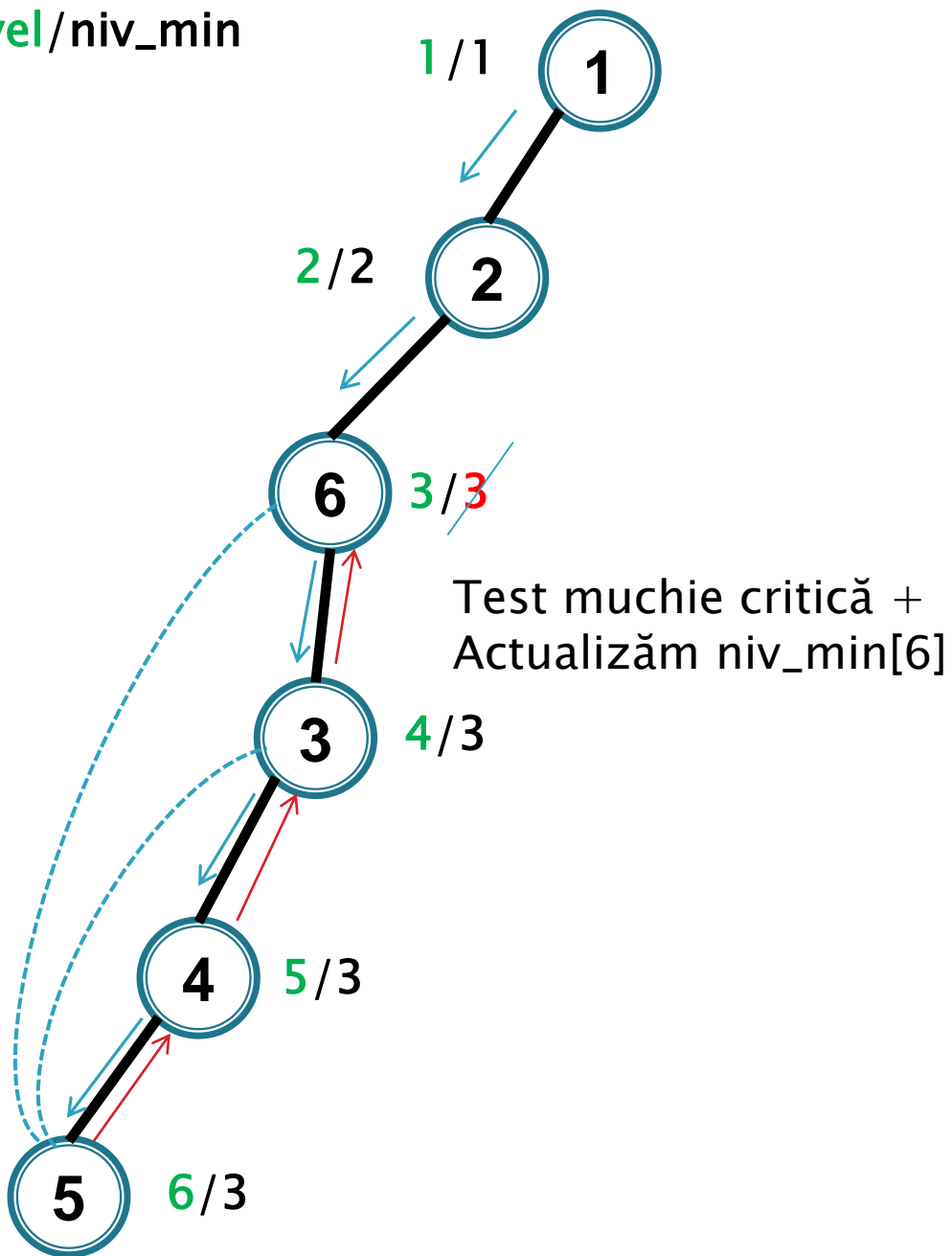


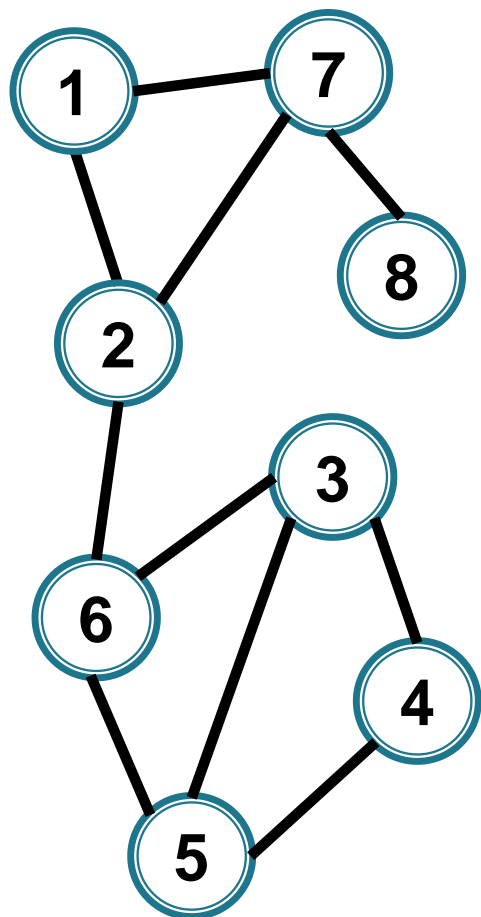
nivel/niv_min



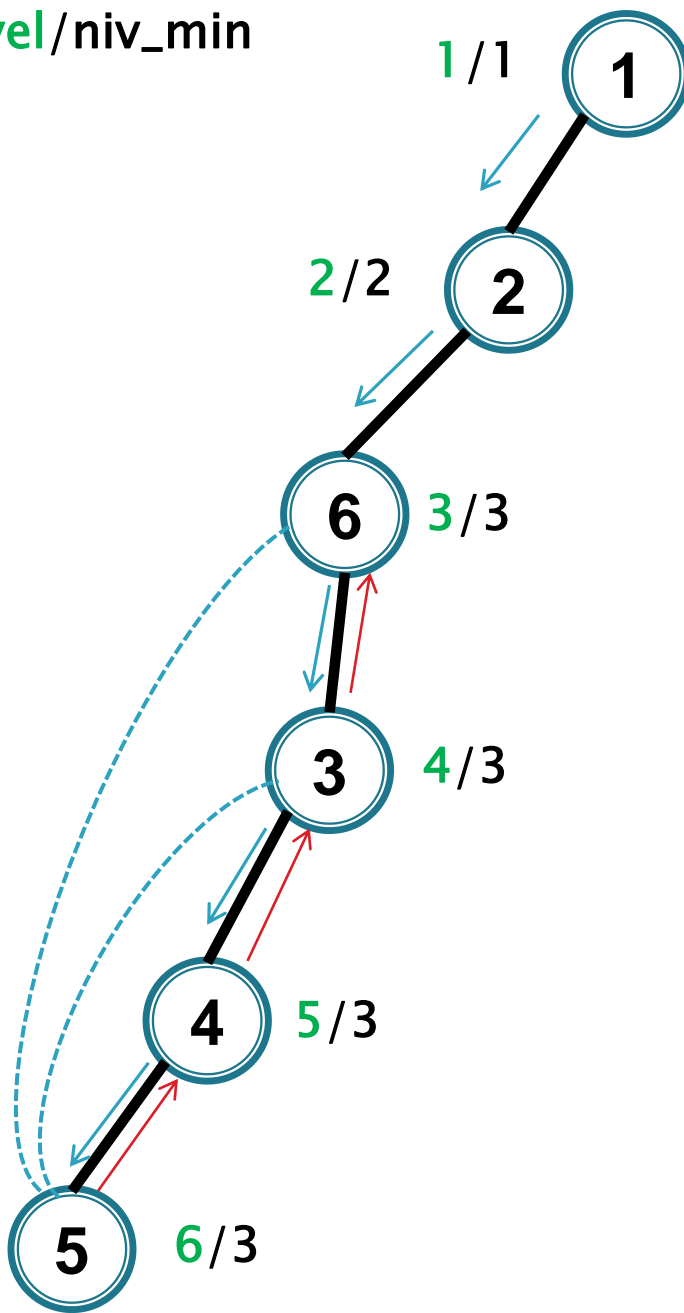


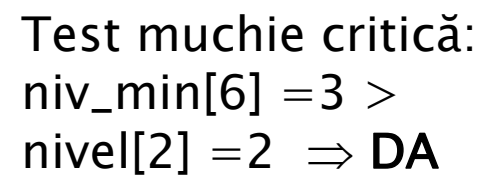
nivel/niv_min

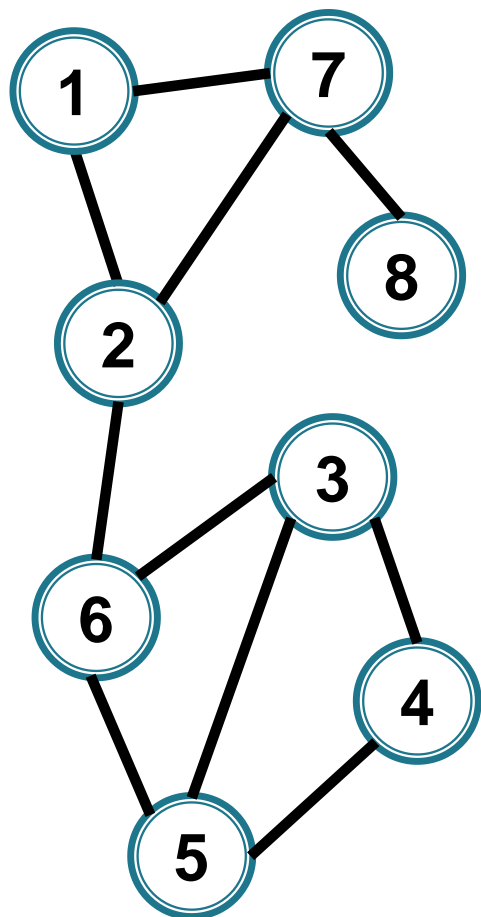




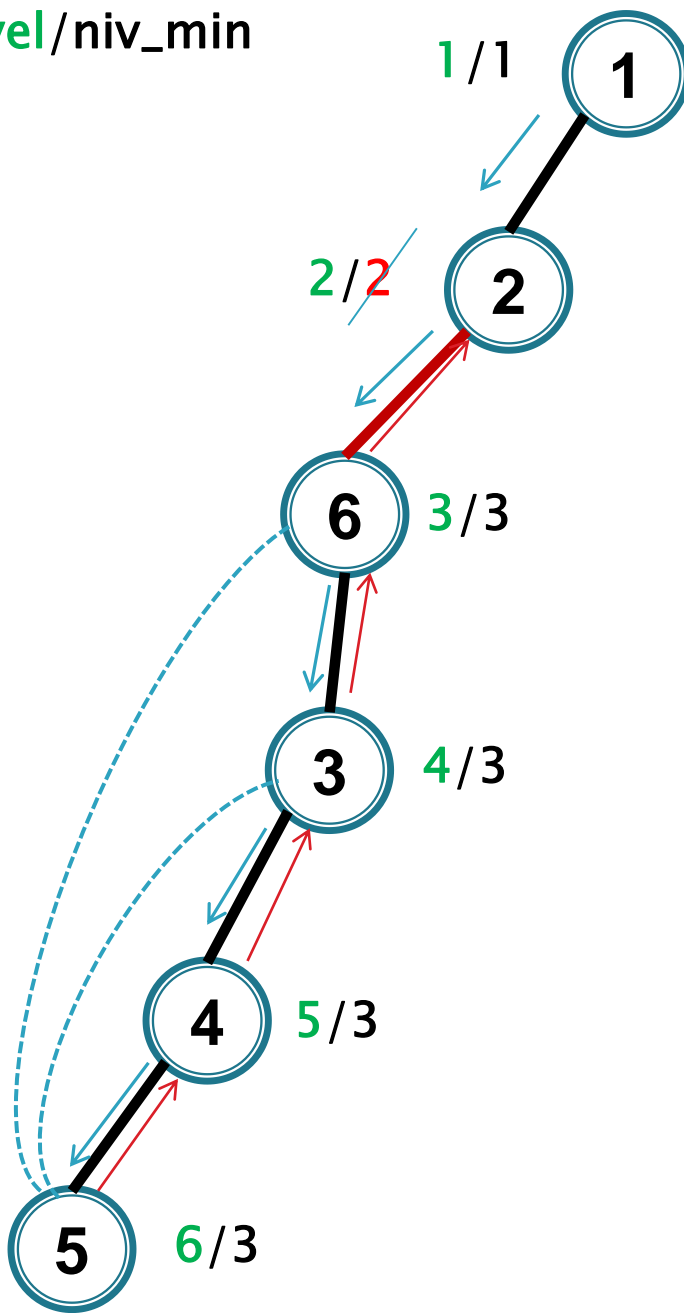
nivel/niv_min

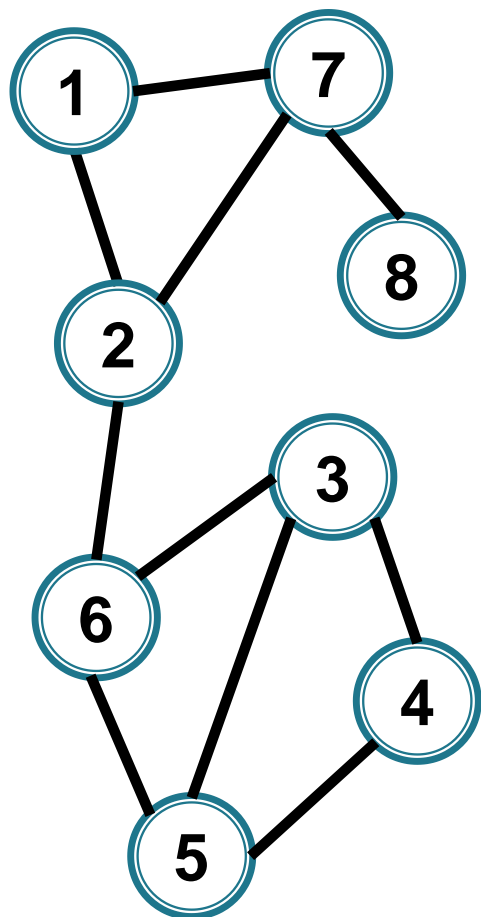




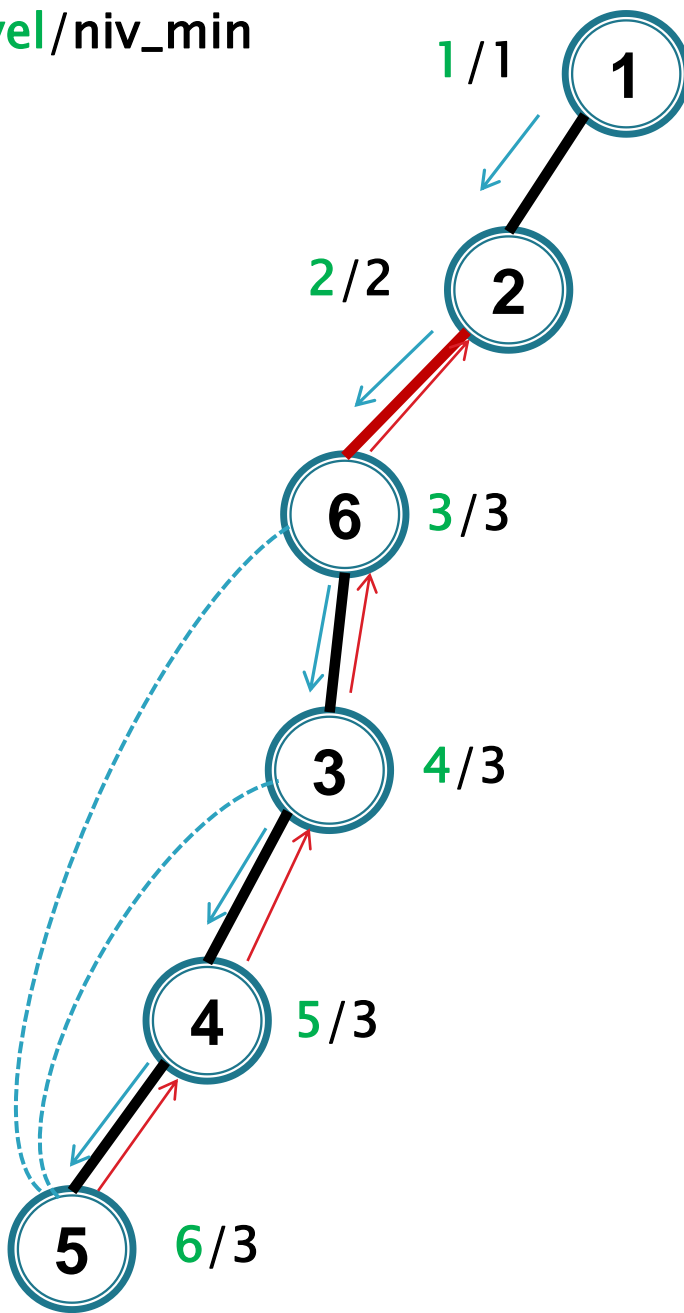


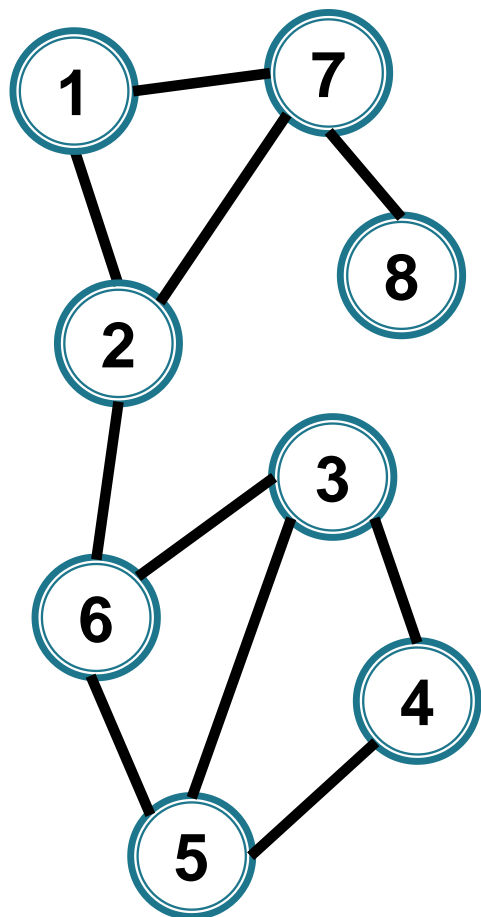
nivel/niv_min



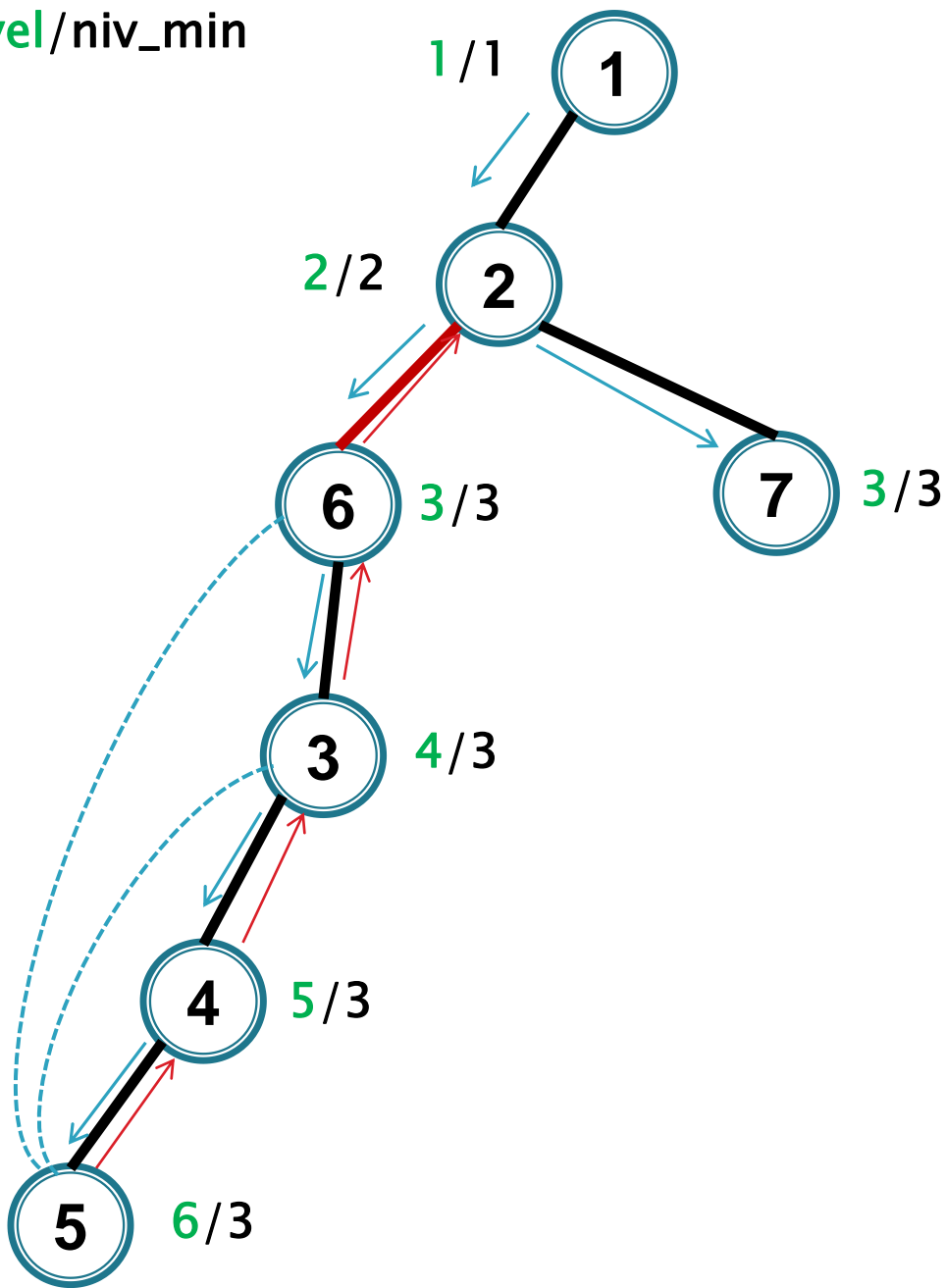


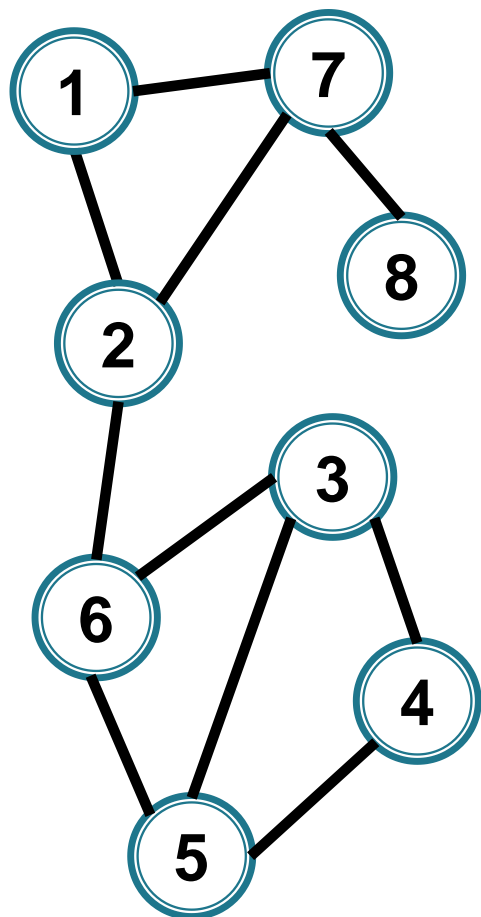
nivel/niv_min



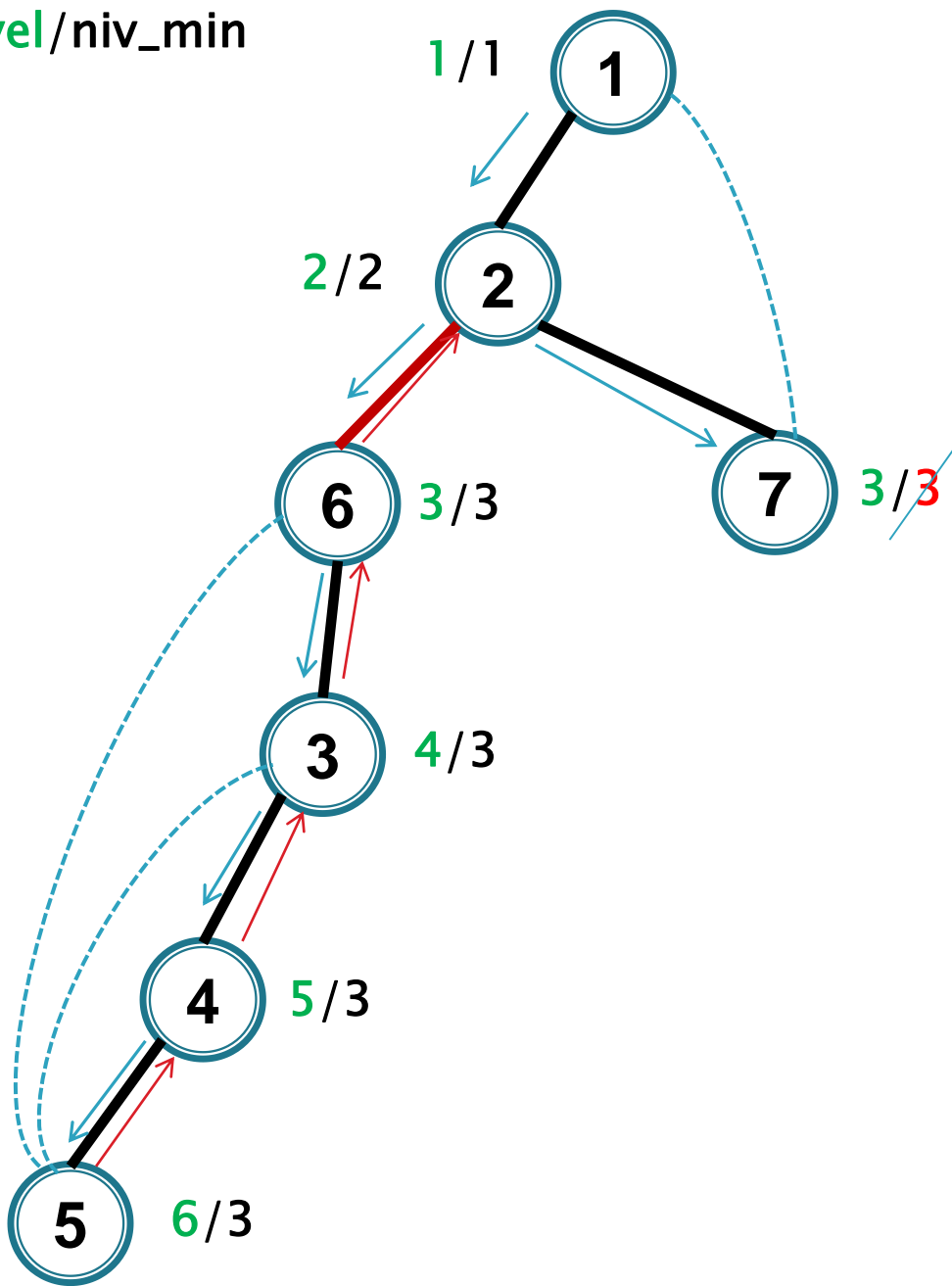


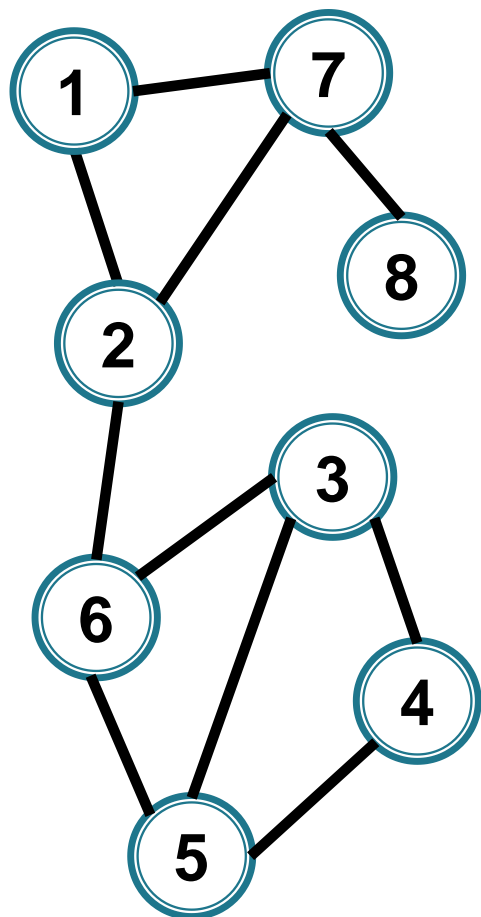
nivel/niv_min



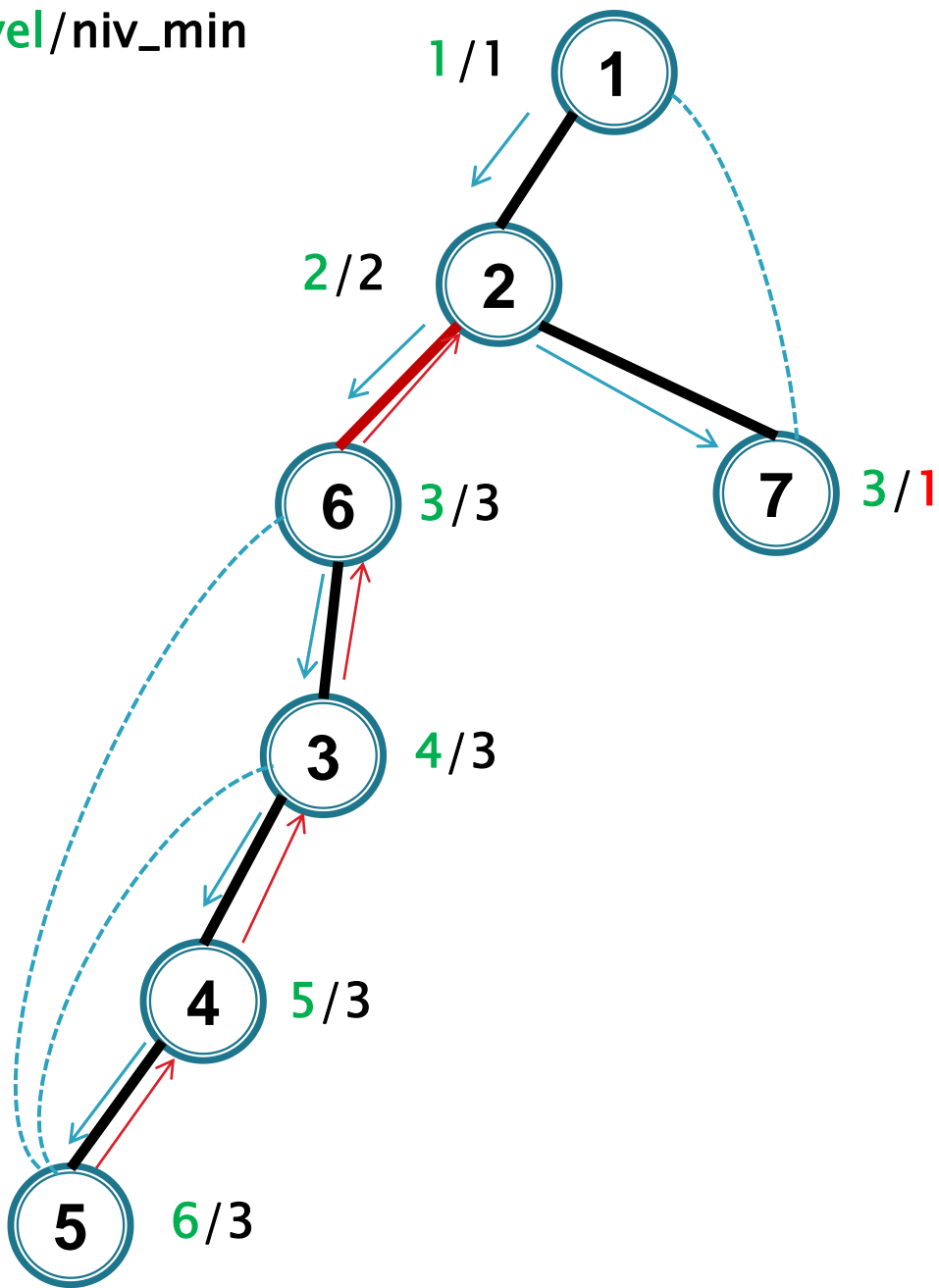


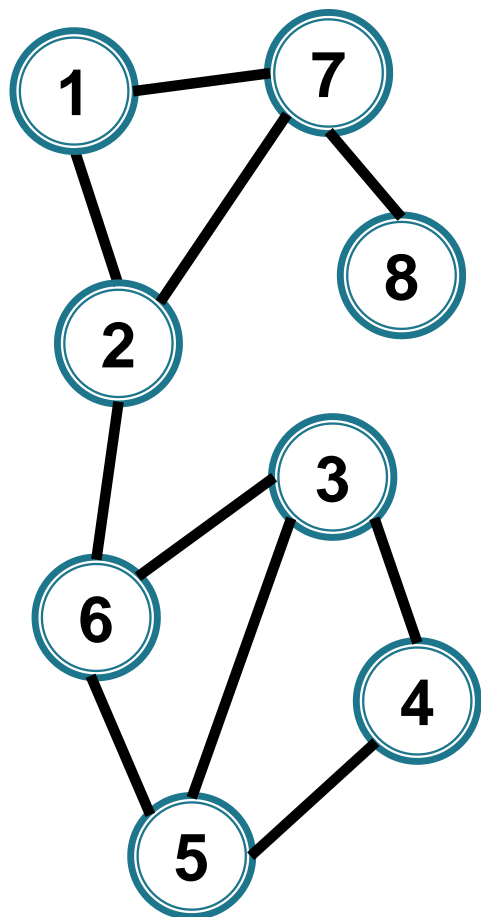
nivel/niv_min



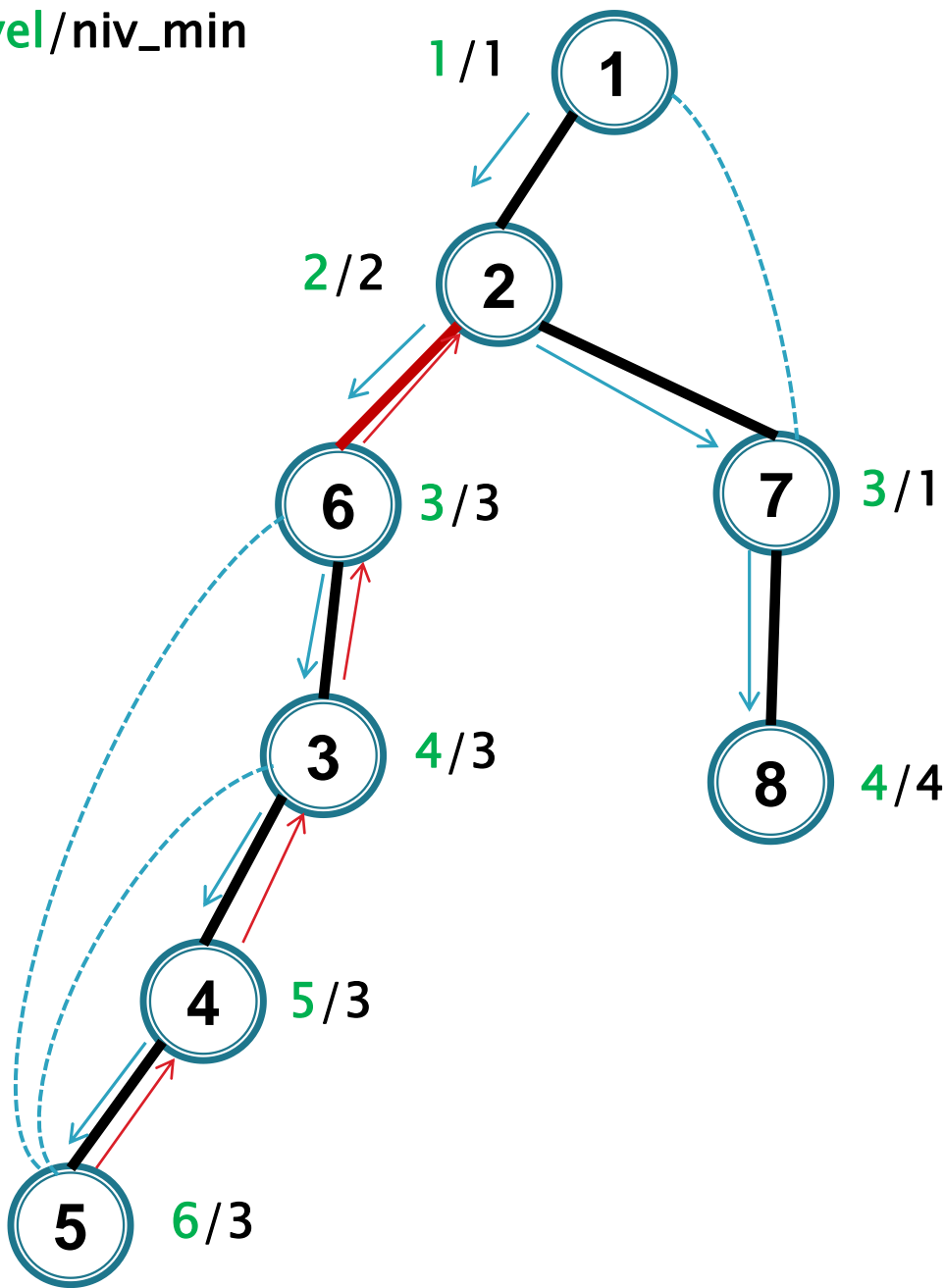


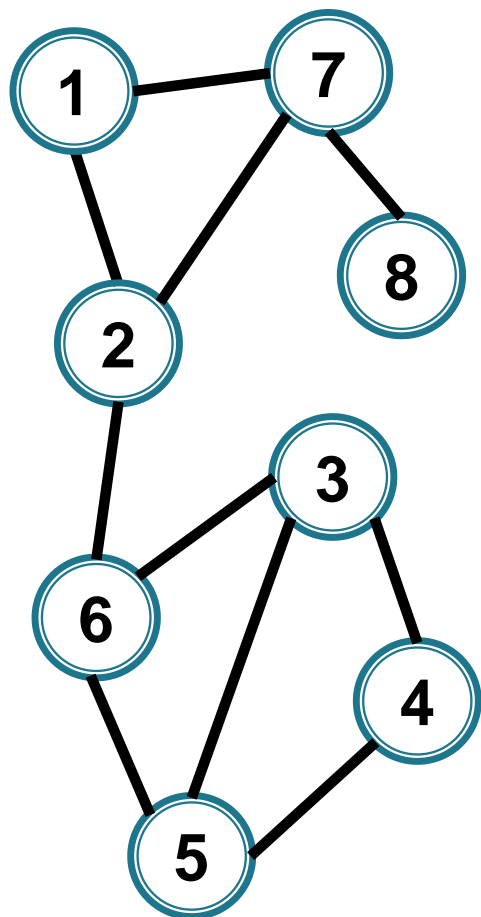
nivel/niv_min



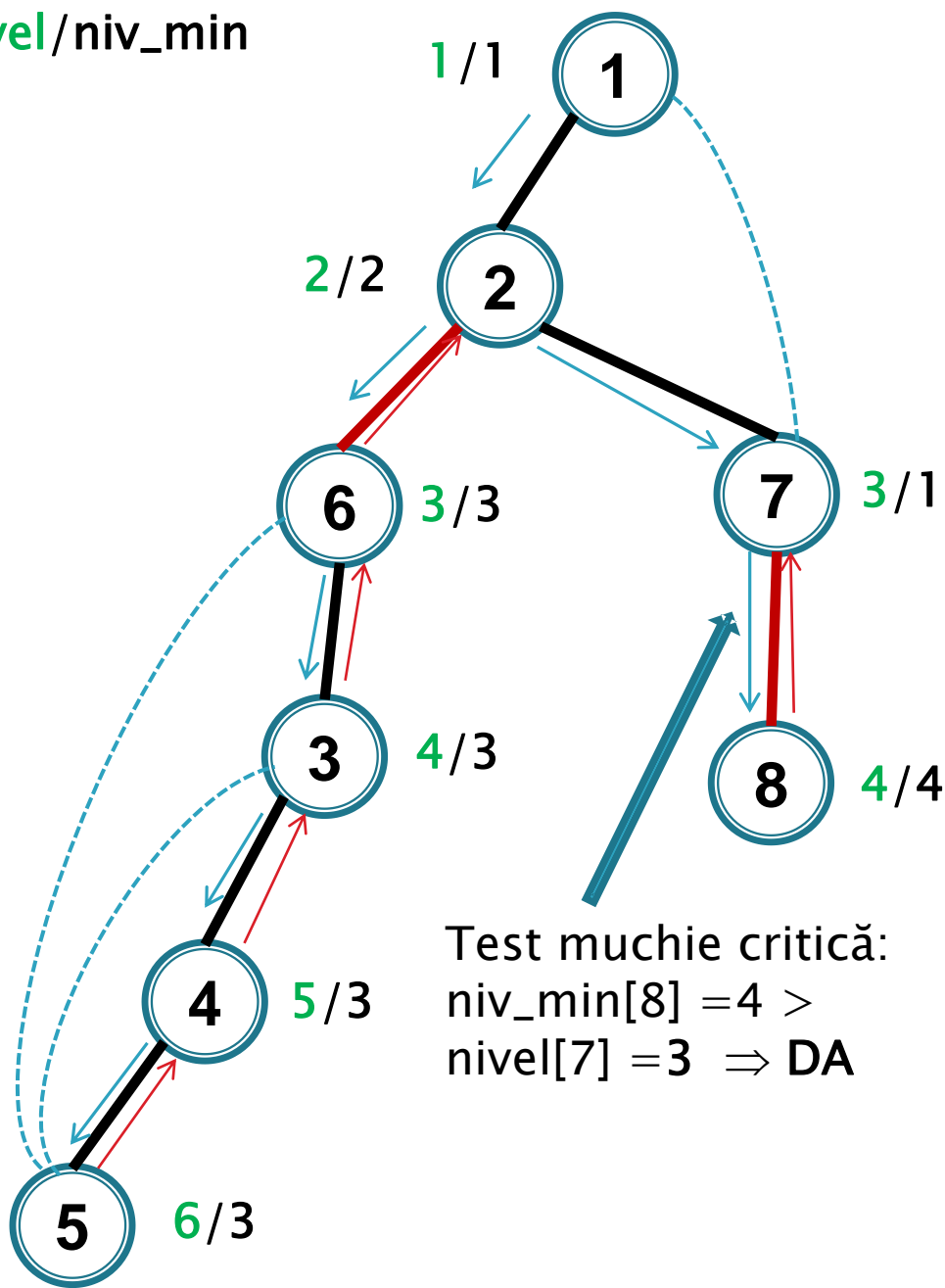


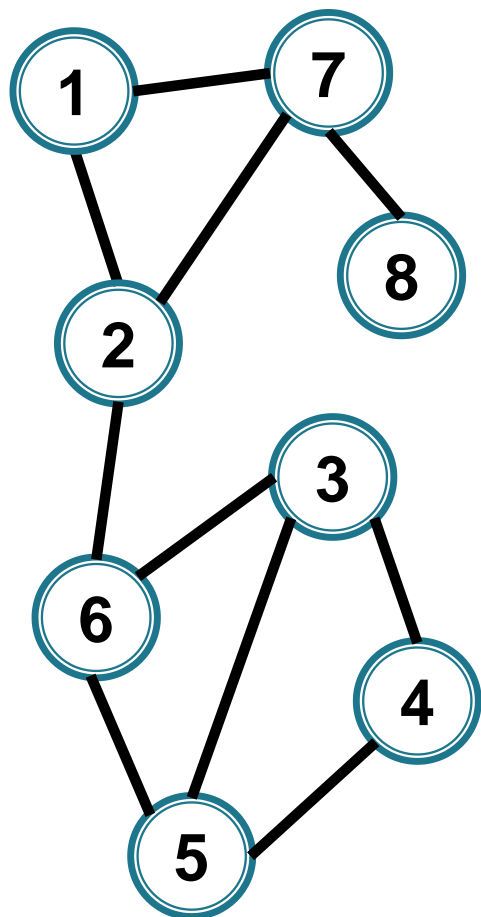
nivel/niv_min



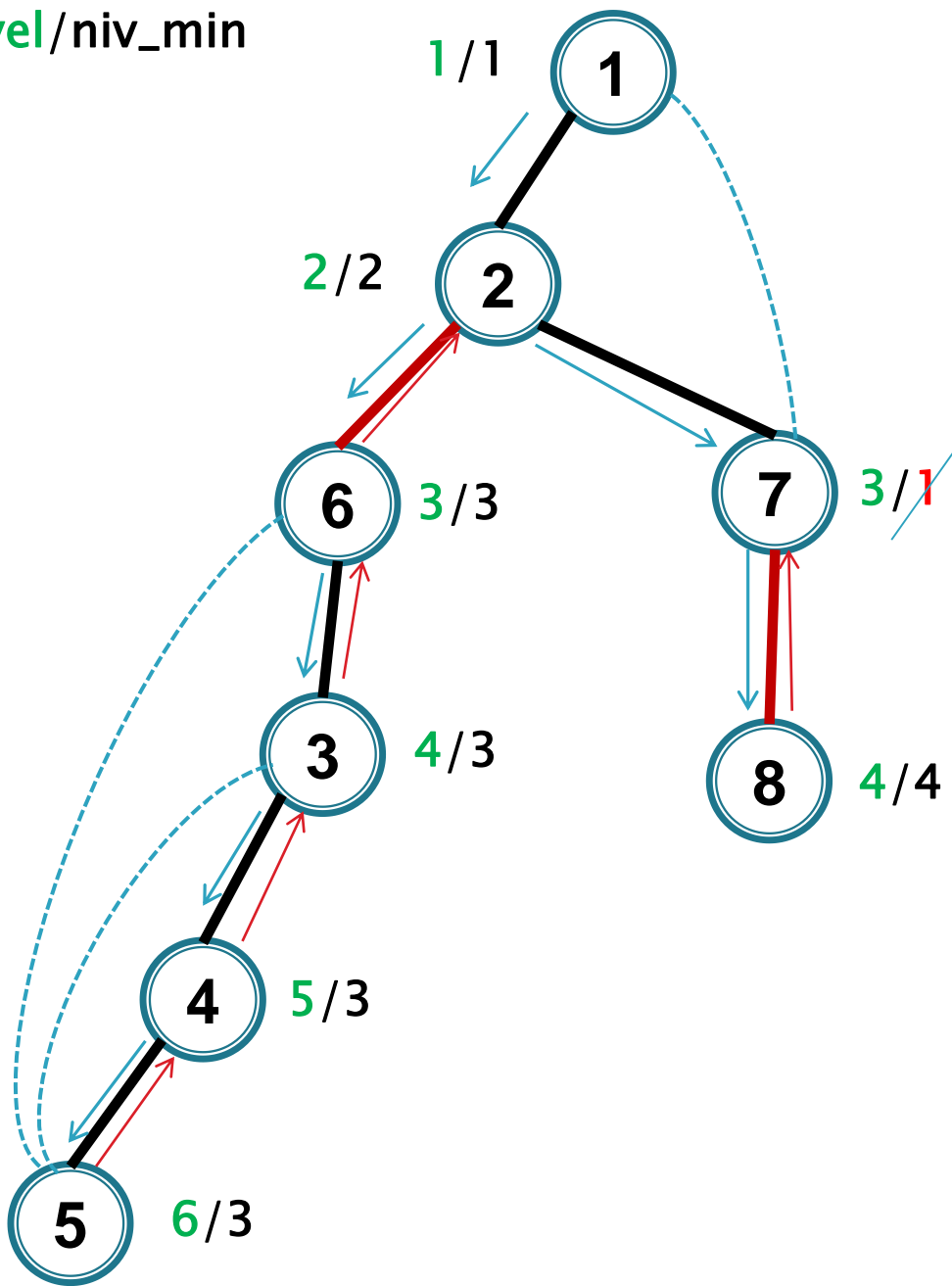


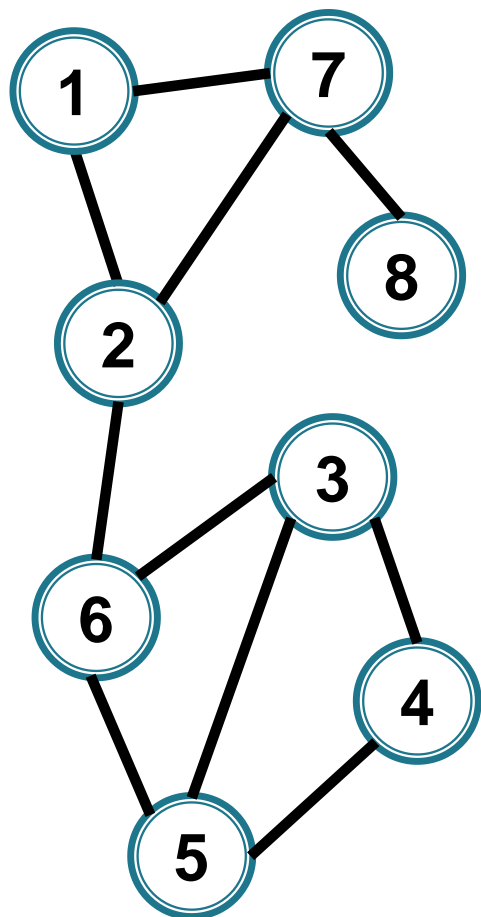
nivel/niv_min



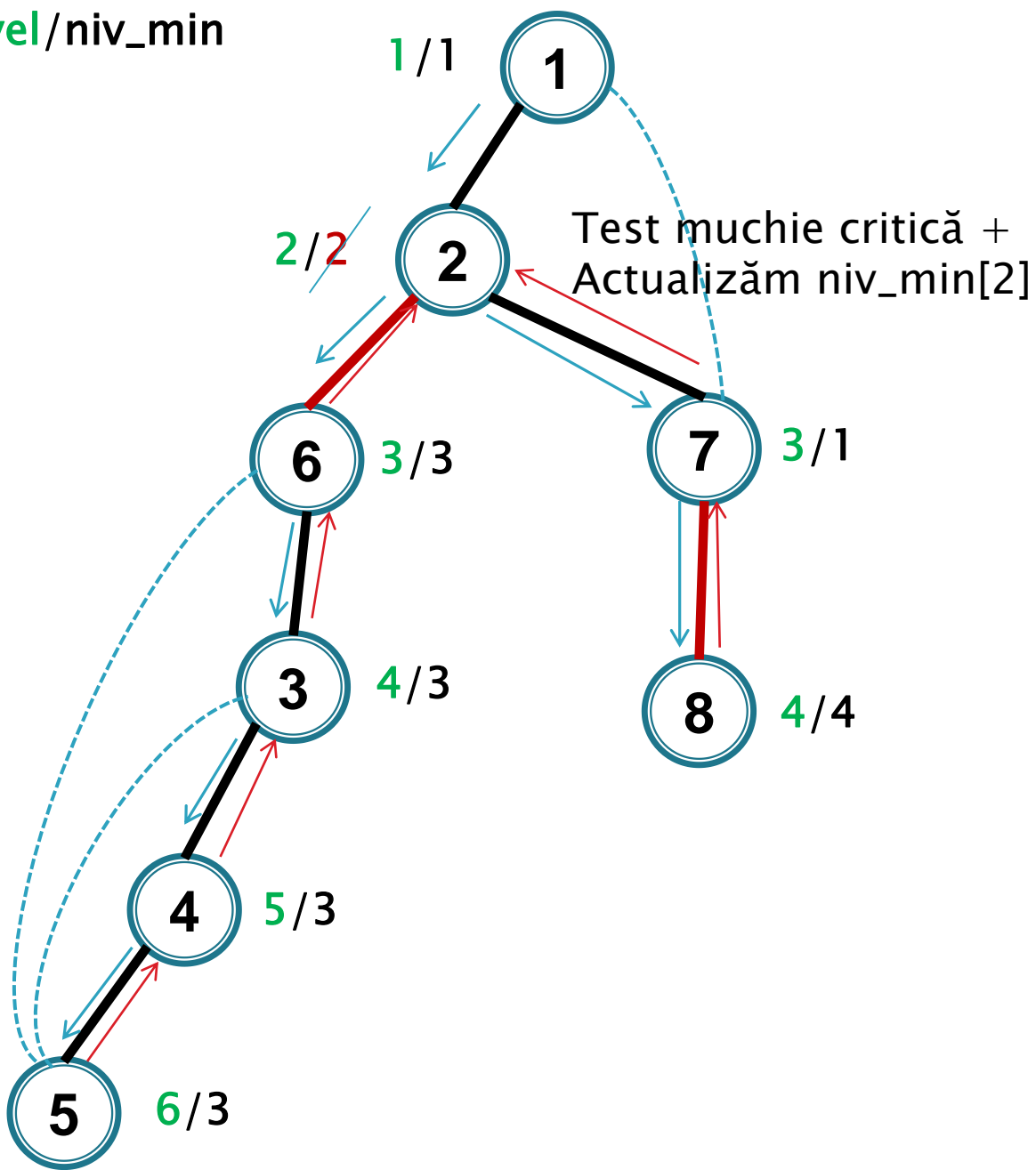


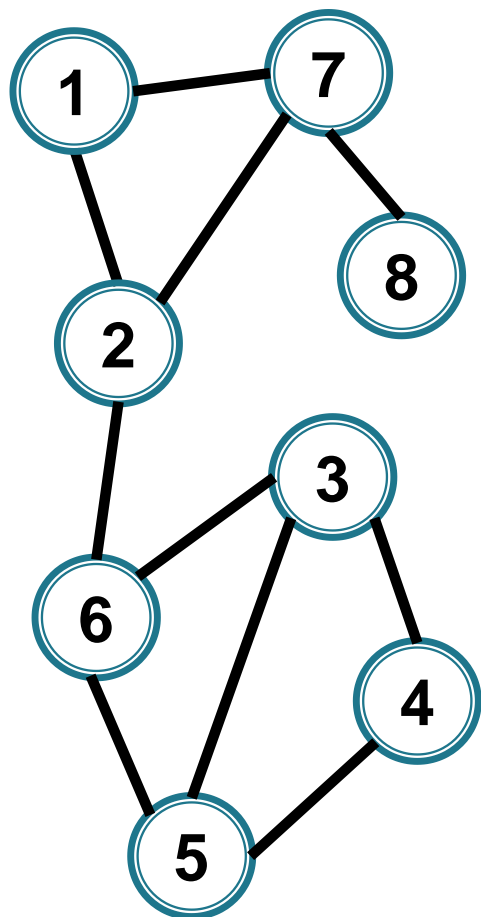
nivel/niv_min



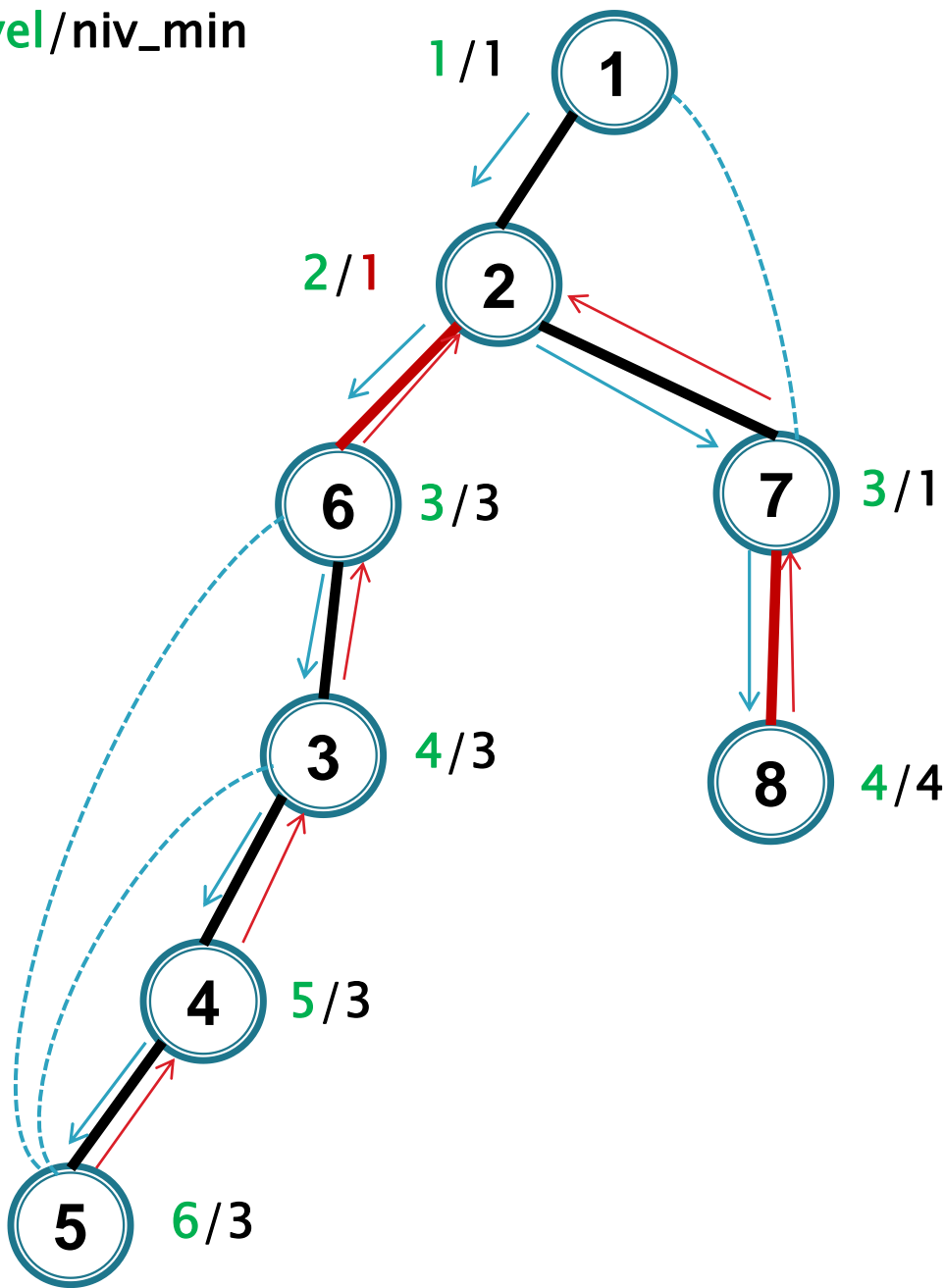


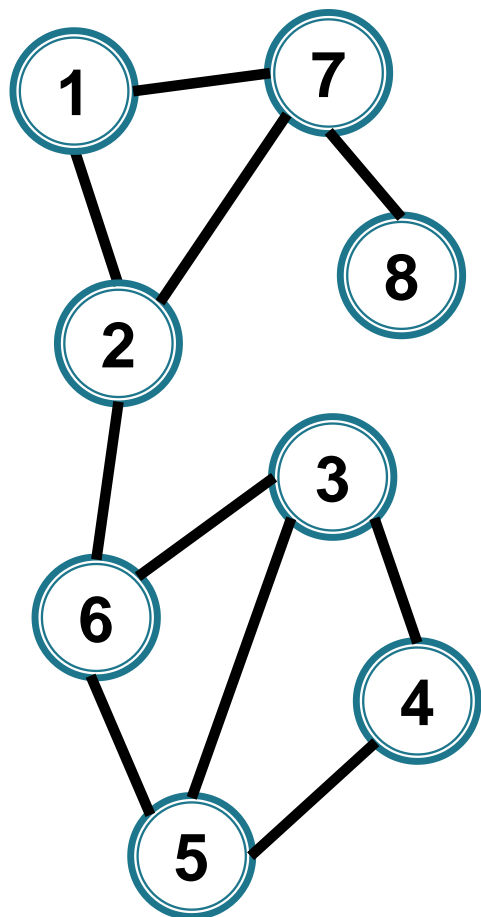
nivel/niv_min





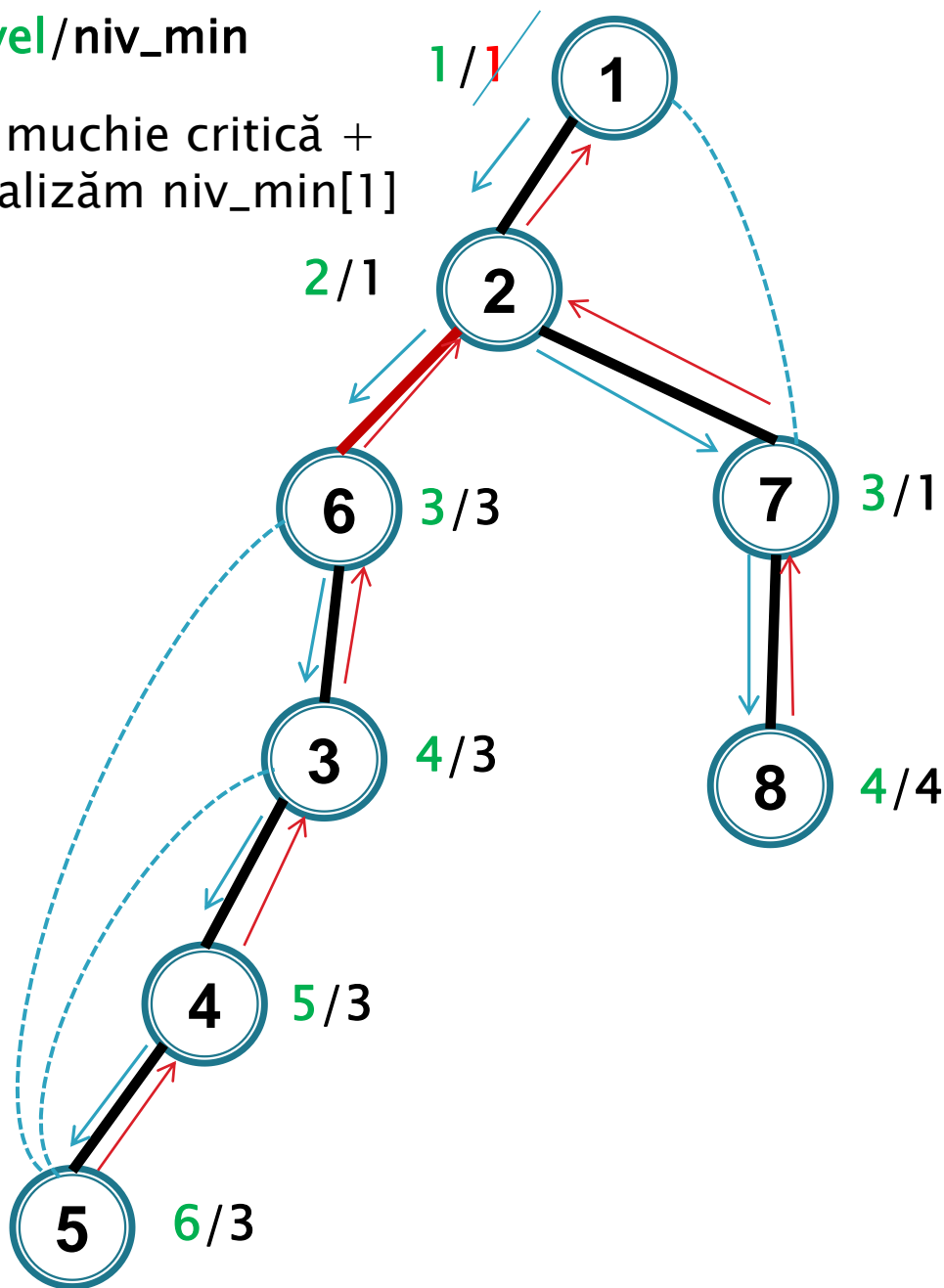
nivel/niv_min

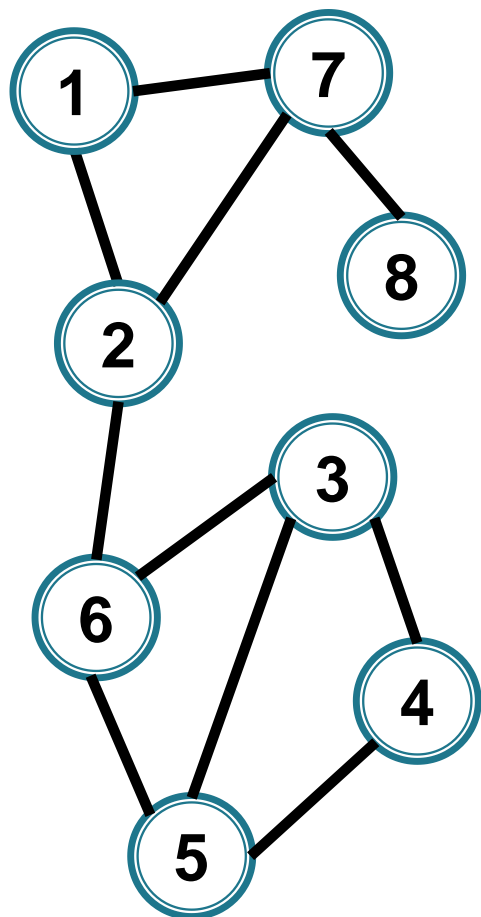




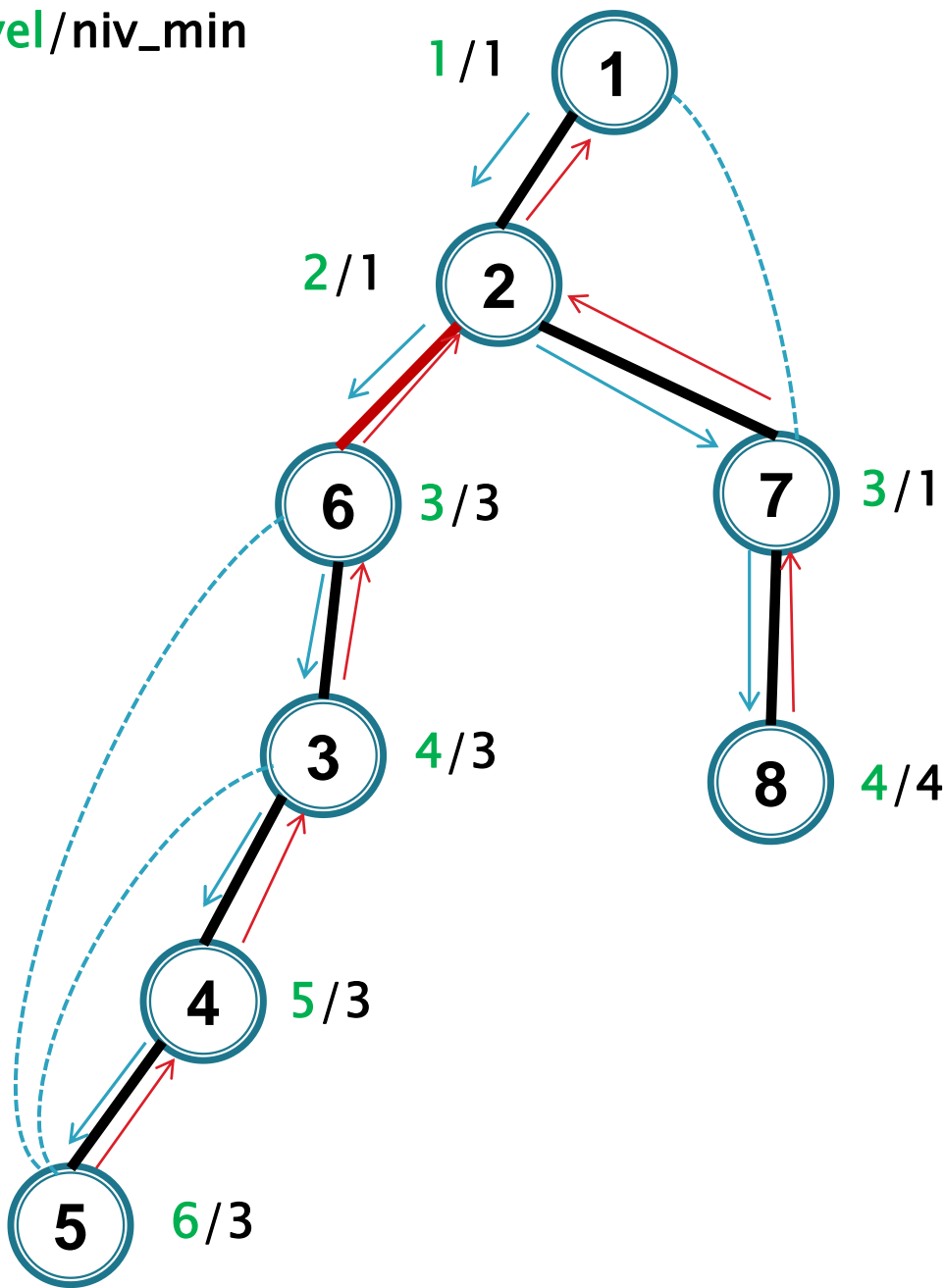
nivel/niv_min

Test muchie critică +
Actualizăm niv_min[1]





nivel/niv_min



Indicații implementare

```
void df(int i){
    viz[i] = 1;
    niv_min[i] = nivel[i];
    for(j vecin al lui i)
        if(viz[j]==0){ //ij muchie de avansare
            nivel[j] = nivel[i]+1;
            df(j);

            //actualizare niv_min[i]- formula B
            ...

            //test ij este muchie critica
            ...
        }
    else
        if(nivel[j]<nivel[i]-1) //ij muchie de intoarcere
            //actualizare niv_min[i]- formula A
            ...
}
```

Indicații implementare

```
void df(int i){
    viz[i] = 1;
    niv_min[i] = nivel[i];
    for(j vecin al lui i)
        if(viz[j]==0){ //ij muchie de avansare
            nivel[j] = nivel[i]+1;
            df(j);

            //actualizare niv_min[i]- formula B
            niv_min[i] = min{niv_min[i], niv_min[j] }

            //test ij este muchie critica
            if (niv_min[j]>nivel[i]) scrie muchia ij
        }
    else
        if(nivel[j]<nivel[i]-1) //ij muchie de intoarcere
            //actualizare niv_min[i]- formula A
            niv_min[i] = min{niv_min[i], nivel[j] }
}
```

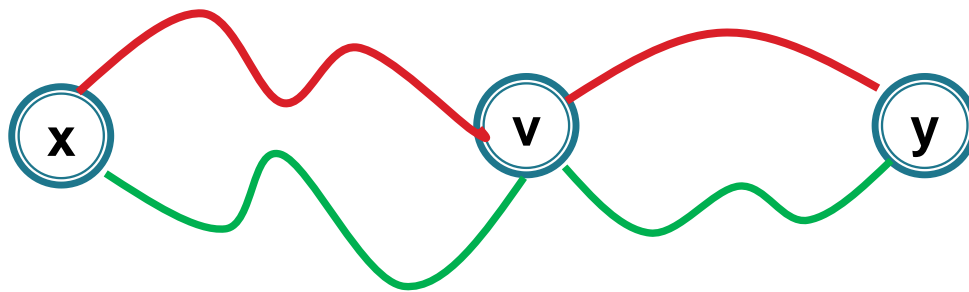
Puncte critice

Puncte critice

- ▶ Un vârf v este punct critic \Leftrightarrow

există două vârfuri $x, y \neq v$ astfel

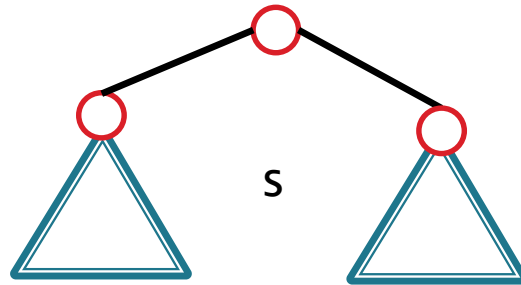
încât v aparține oricărui x, y -lanț



Puncte critice

- ▶ Arborele DF

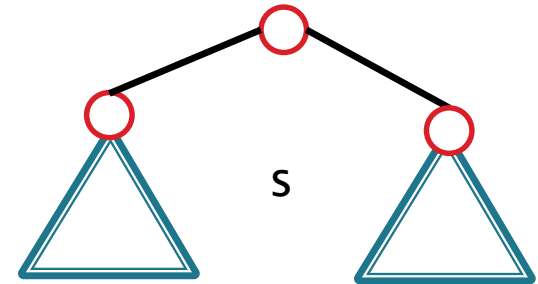
- rădăcina s este punct critic \Leftrightarrow



Puncte critice

▶ Arborele DF

- rădăcina s este punct critic \Leftrightarrow

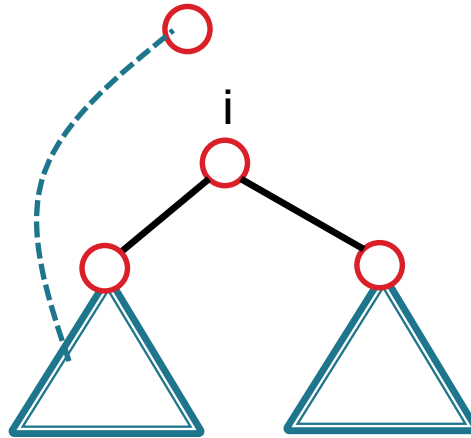


are cel puțin 2 fii în arborele DF

Puncte critice

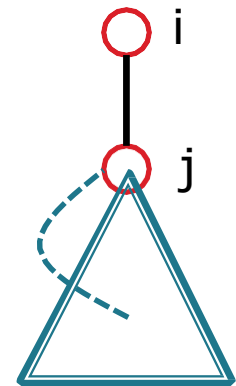
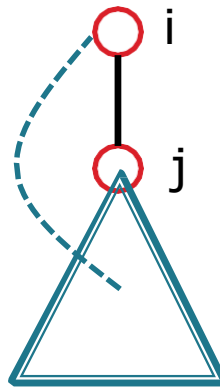
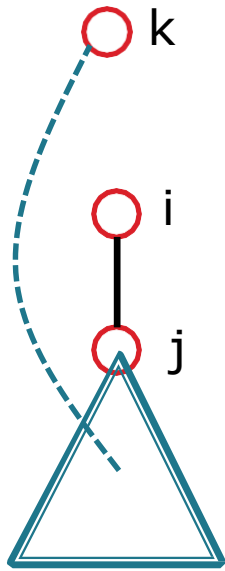
▶ Arborele DF

- un alt vârf i din arbore este critic \Leftrightarrow



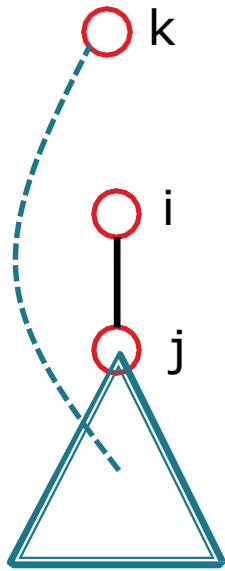
Puncte critice

Pentru $i \neq s$



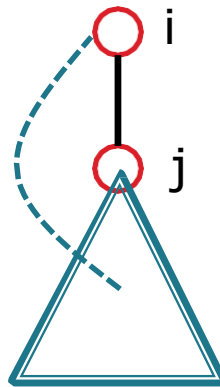
Puncte critice

Pentru $i \neq s$



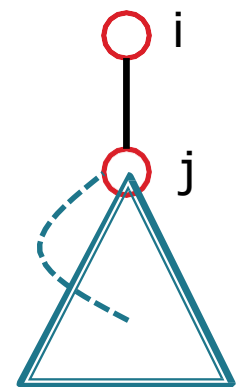
i NU este critic

$niv_min[j] < nivel[i]$



i ESTE critic

$niv_min[j] = nivel[i]$



i ESTE critic

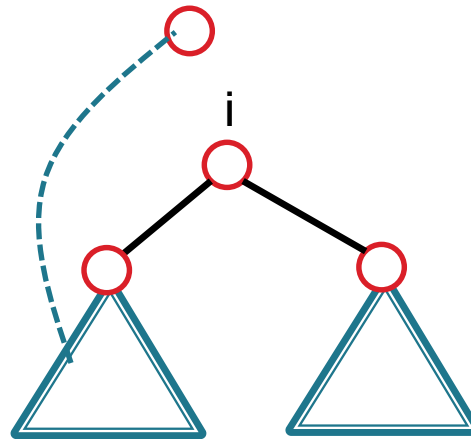
$niv_min[j] > nivel[i]$

Puncte critice

▶ Arborele DF

- un alt vârf i din arbore este critic \Leftrightarrow

are cel puțin un fiu j cu
 $niv_min[j] \geq nivel[i]$



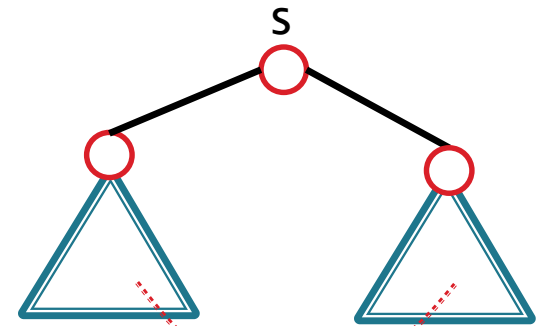
Puncte critice

- ▶ Un vârf v este punct critic \Leftrightarrow există două vârfuri $x, y \neq v$ astfel încât v aparține oricărui x, y -lanț

- ▶ Arborele DF

- rădăcina s este punct critic \Leftrightarrow

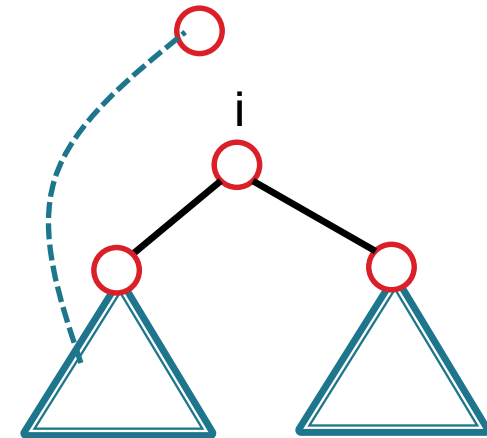
are cel puțin 2 fii în arborele DF



nu există muchii între subarbori
(de traversare)

- un alt vârf i din arbore este critic \Leftrightarrow

are cel puțin un fiu j cu
 $niv_min[j] \geq nivel[i]$

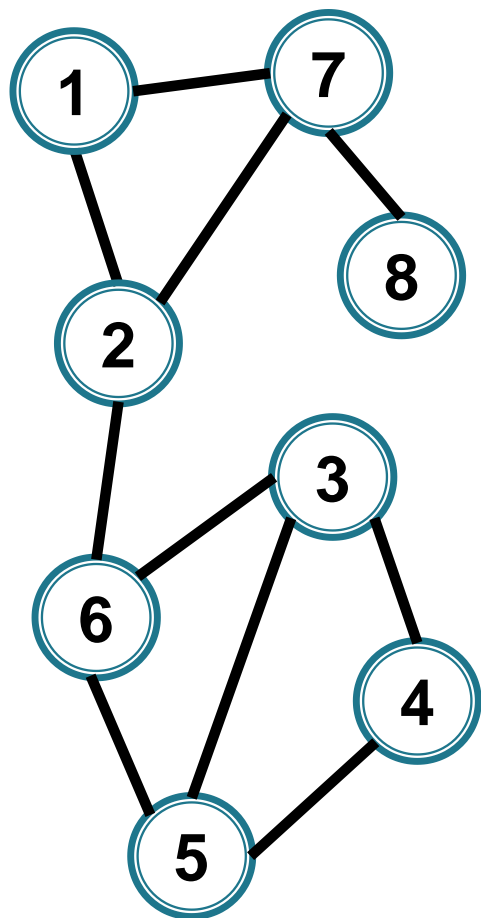


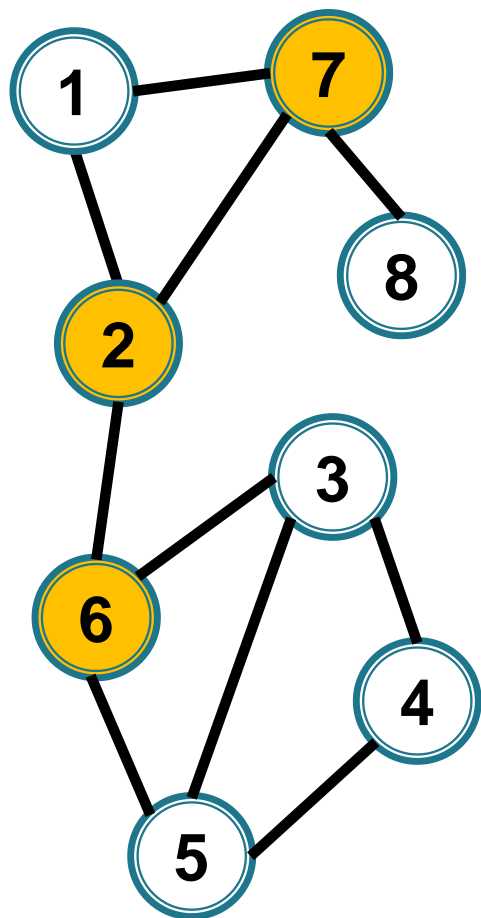
Componente biconexe

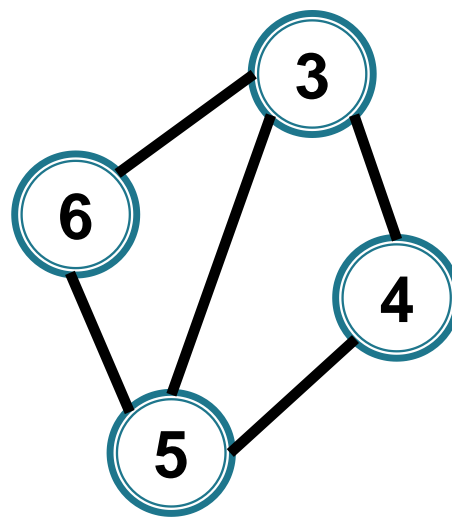
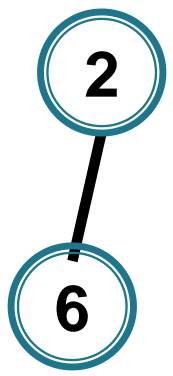
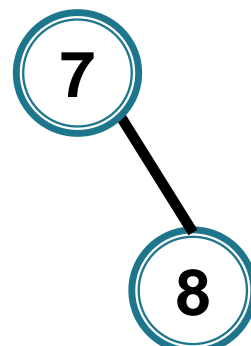
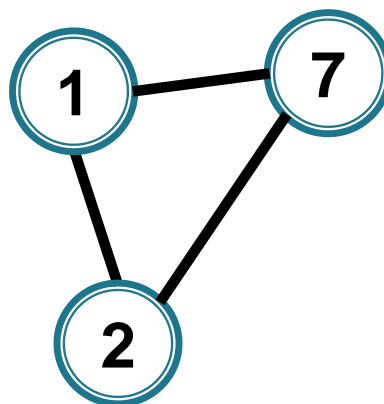
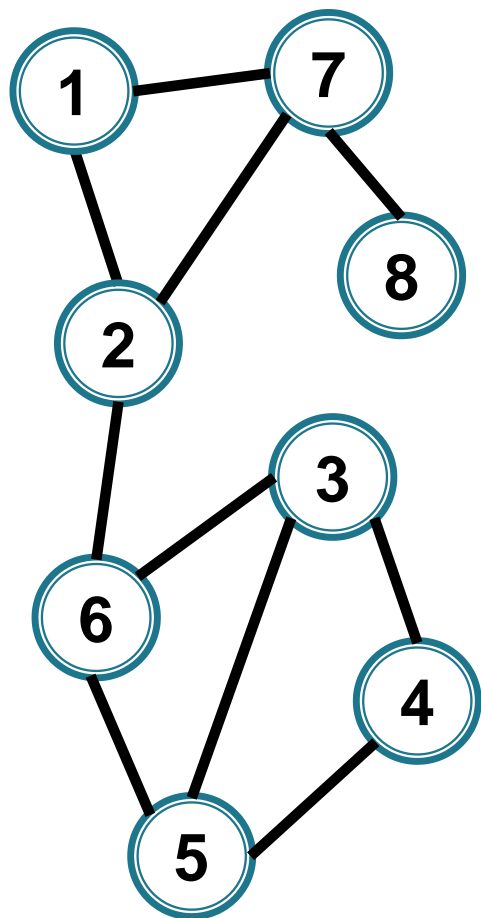


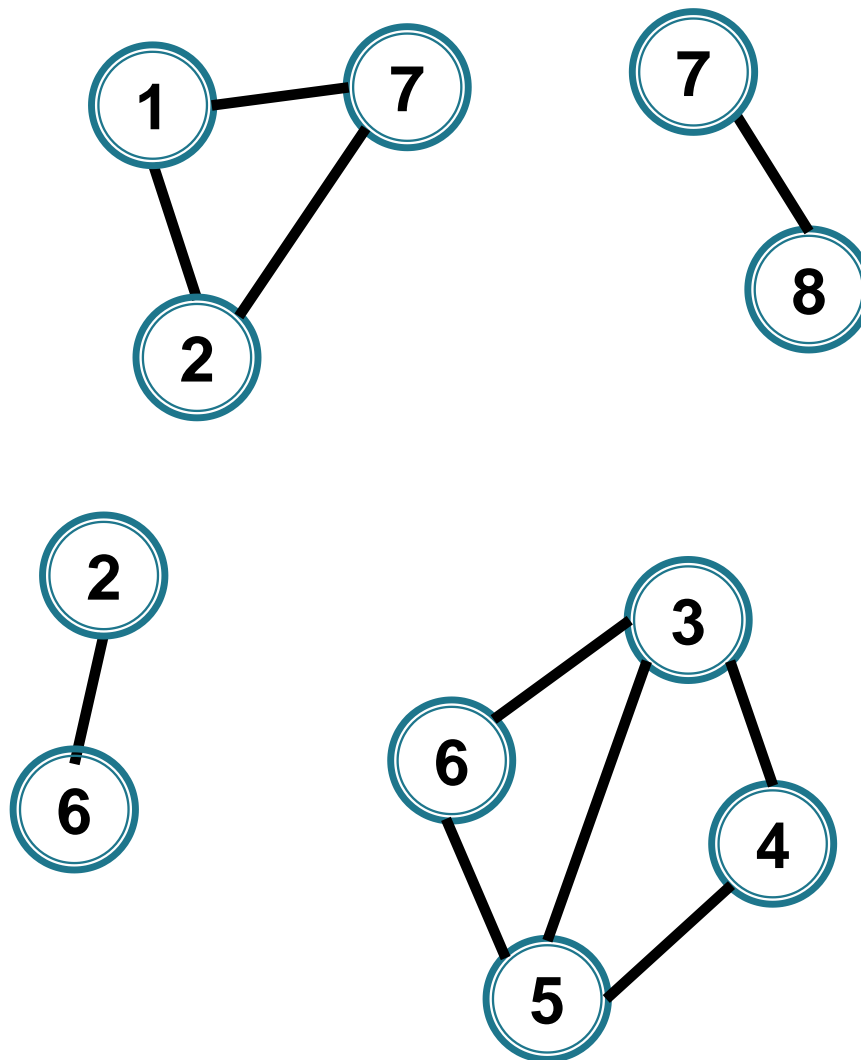
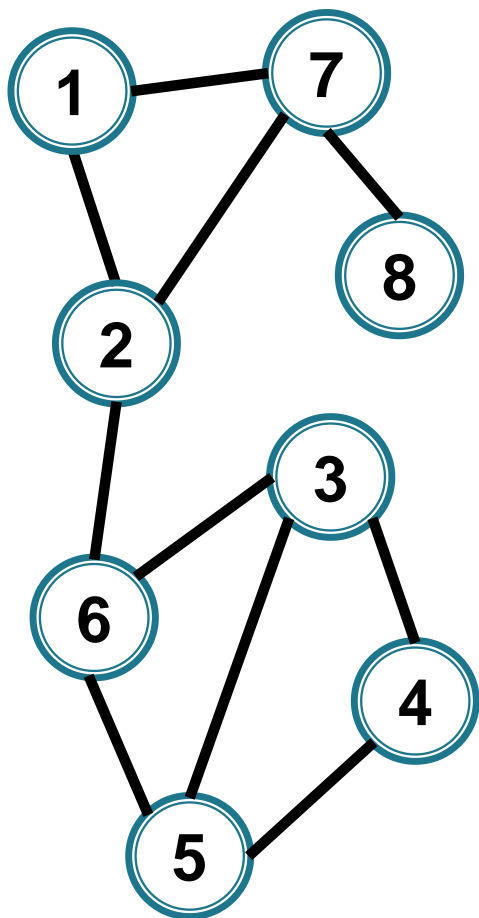
Componente biconexe

- ▶ G – graf neorientat
- ▶ $G=(V,M)$ biconex = nu are puncte de articulație.
- ▶ Componentă biconexă (bloc) a lui G = subgraf biconex maximal.







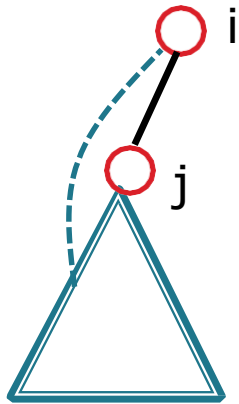


Muchie-disjuncte, nu vârf-disjuncte

Componente biconexe

► Algoritm

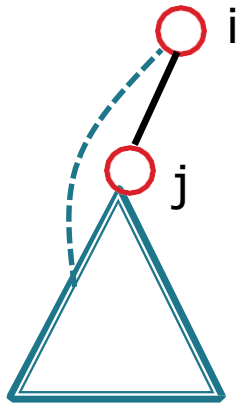
Intuitiv: când un fiu j semnalează că vârful curent i este critic, se poate afișa componenta care conține ij (care “se rupe” din vârful j)



Componente biconexe

► Algoritm

Intuitiv: când un fiu j semnalează că vârful curent i este critic, se poate afișa componenta care conține ij (care “se rupe” din vârful j)

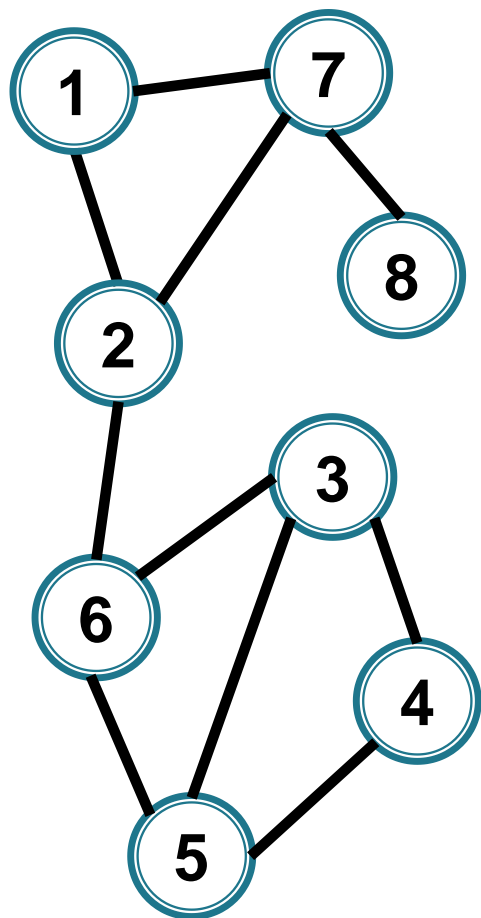


Memorăm muchiile într-o **stivă**

Când detectăm o componentă biconexă –
scoatem muchiile din stivă până la muchia ij
 \Rightarrow acestea formează o componentă

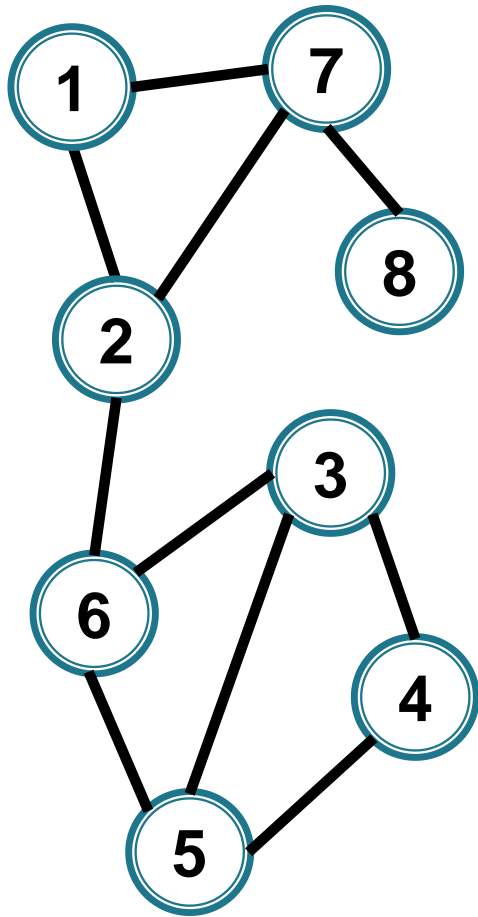
Indicații implementare

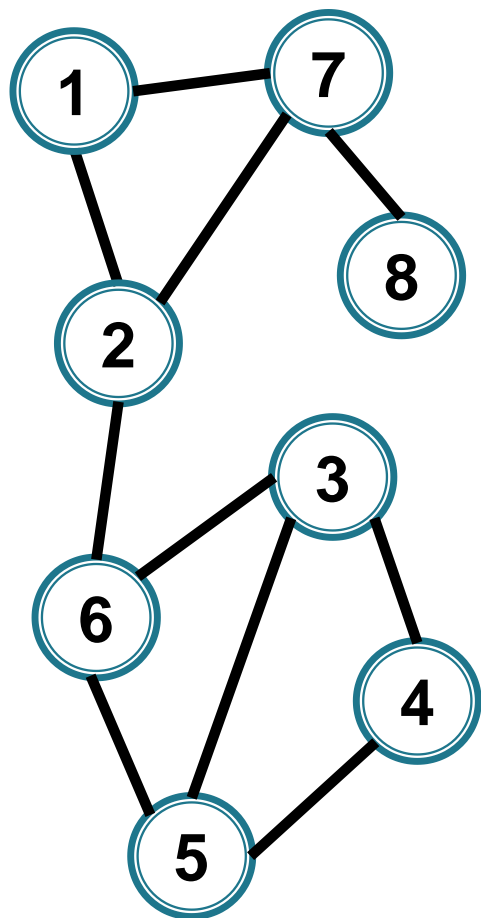
```
void df(int i){
    viz[i] = 1;
    niv_min[i] = nivel[i];
    for(j vecin al lui i)
        if(viz[j]==0){ //ij muchie de avansare
            nivel[j] = nivel[i]+1;
            adauga(S,ij)
            df(j);
            niv_min[i] = min{niv_min[i], niv_min[j] }
            if (niv_min[j] >= nivel[i])
                elimina din S toate muchiile pana la ij
        }
    else
        if(nivel[j]<nivel[i]-1) //ij muchie de intoarcere
            //actualizare niv_min[i]- formula A
            niv_min[i] = min{niv_min[i], nivel[j]}
            adauga(S,ij)
        }
}
```



nivel/niv_min

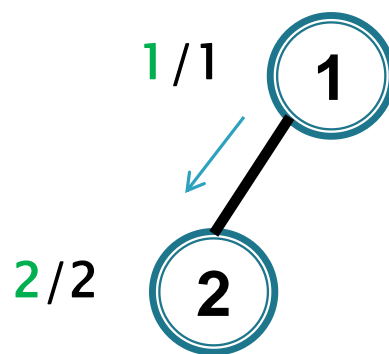
1/1

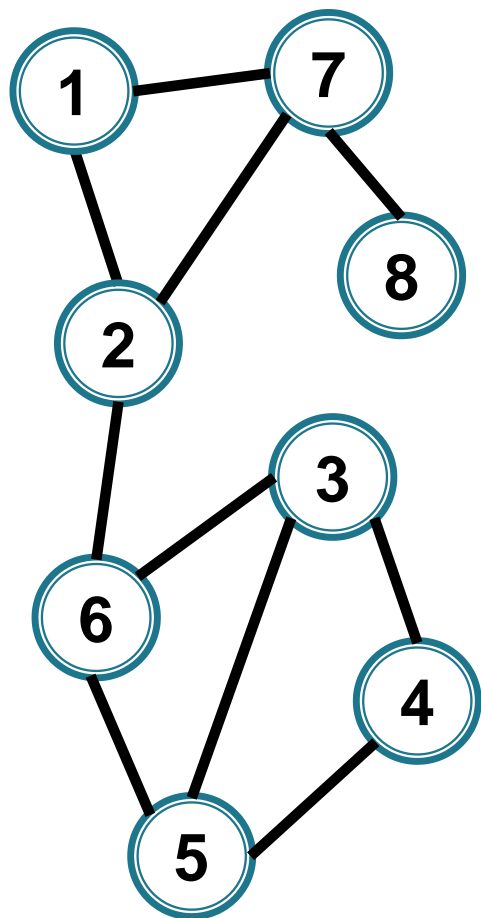




S:
1 2

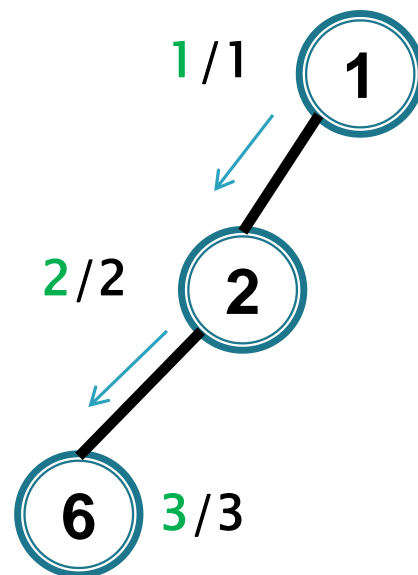
nivel/niv_min

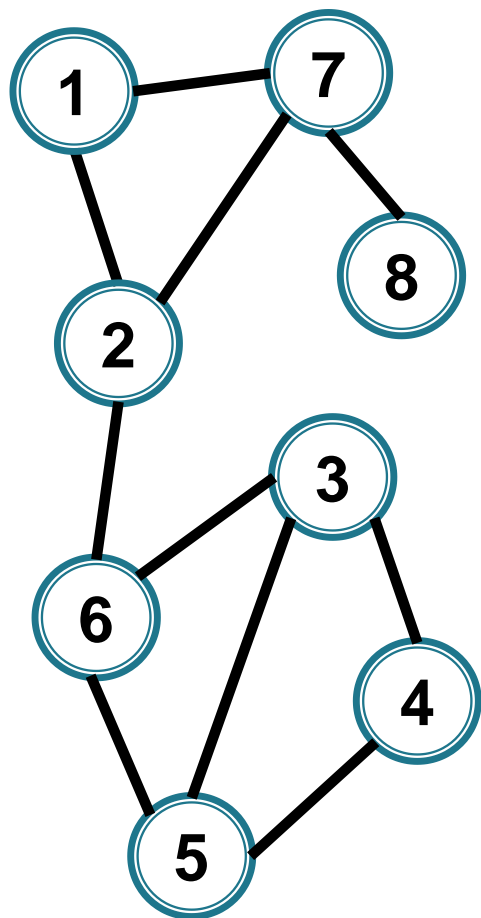




S:
1 2
2 6

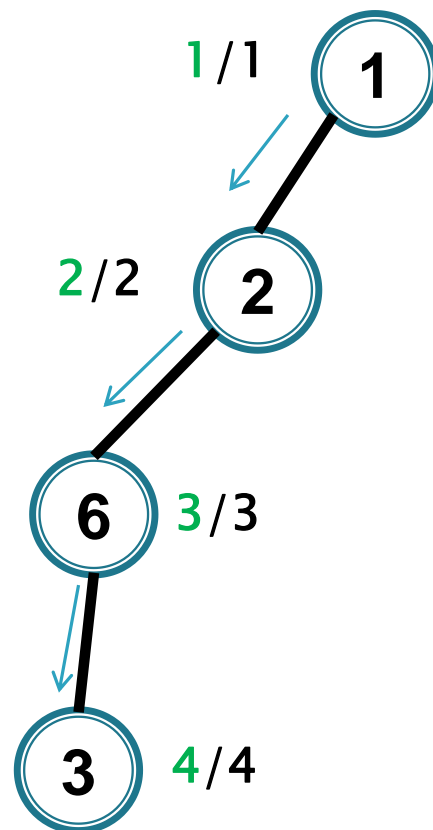
nivel/niv_min

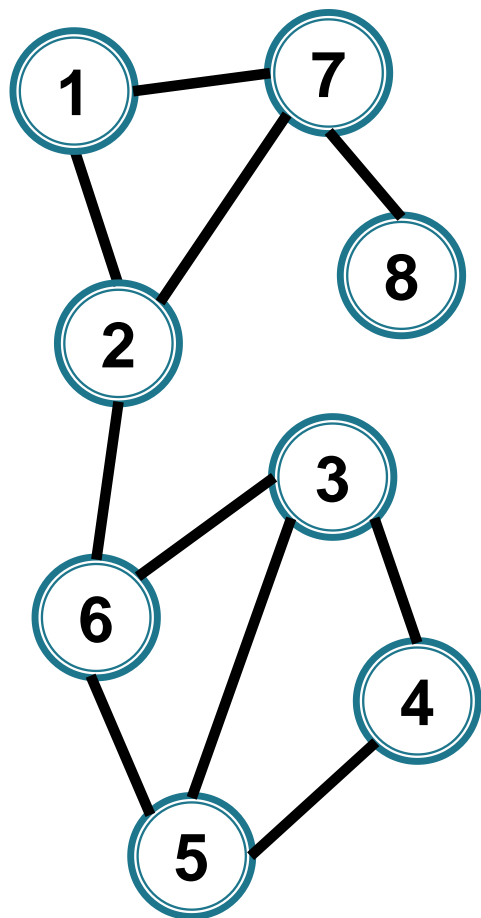




S:
1 2
2 6
6 3

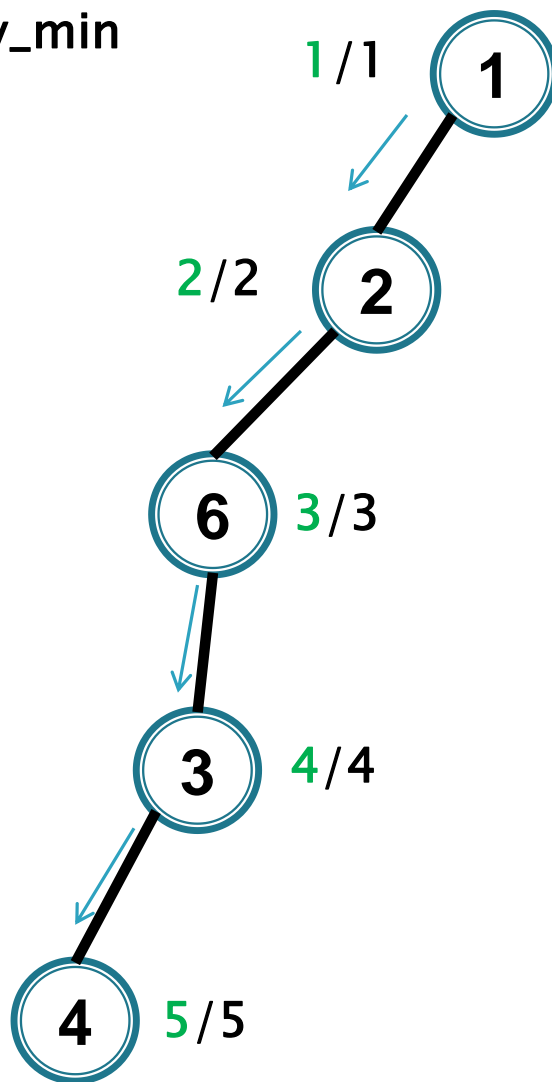
nivel/niv_min

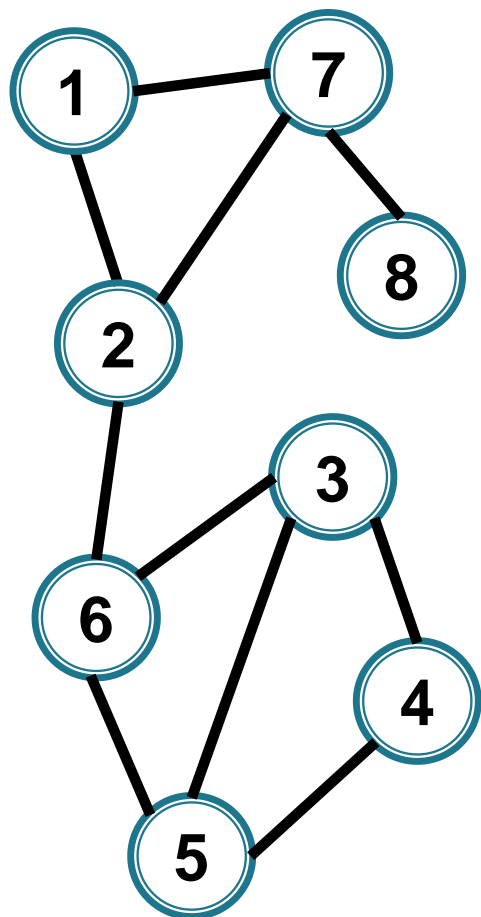




S:
 1 2
 2 6
 6 3
 3 4

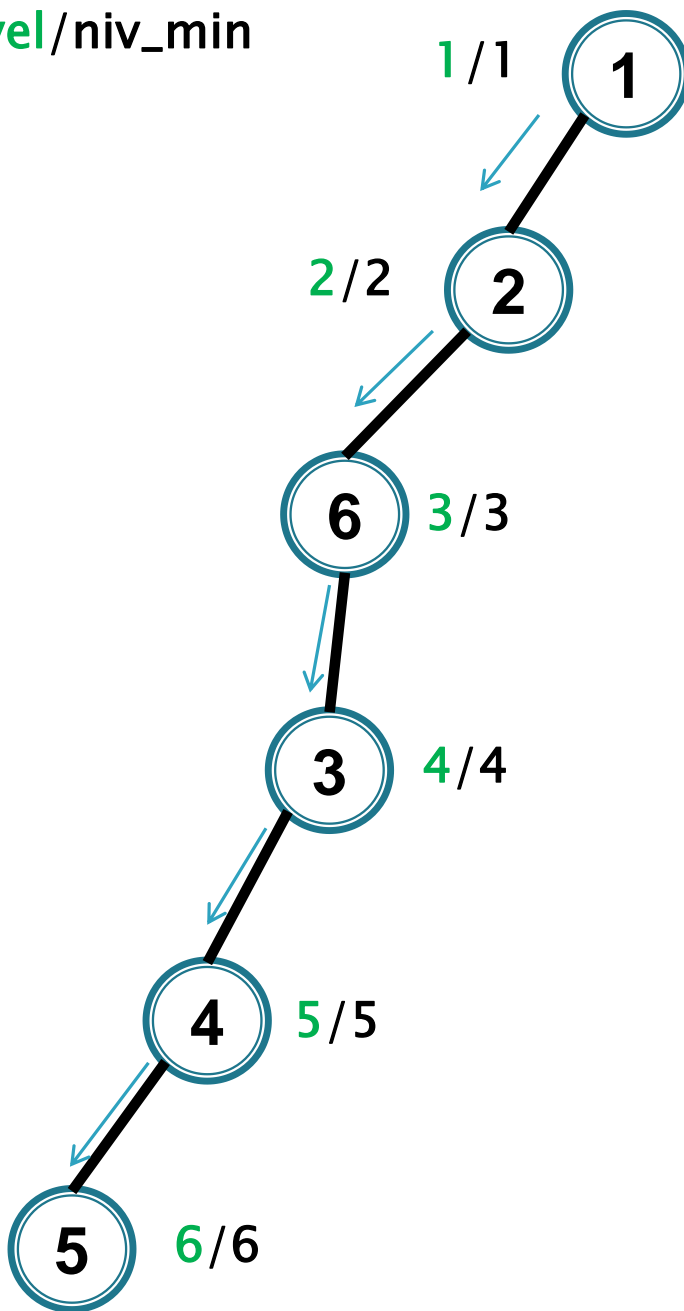
nivel/niv_min

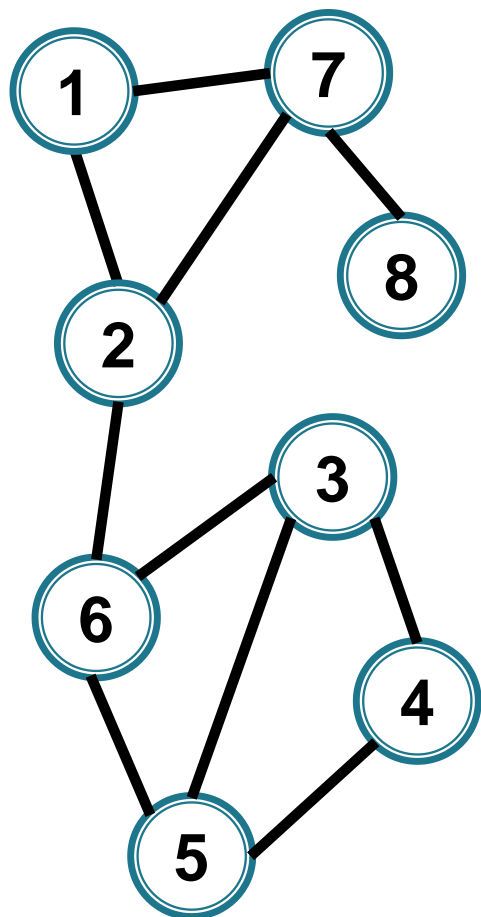




S:
 1 2
 2 6
 6 3
 3 4
 4 5

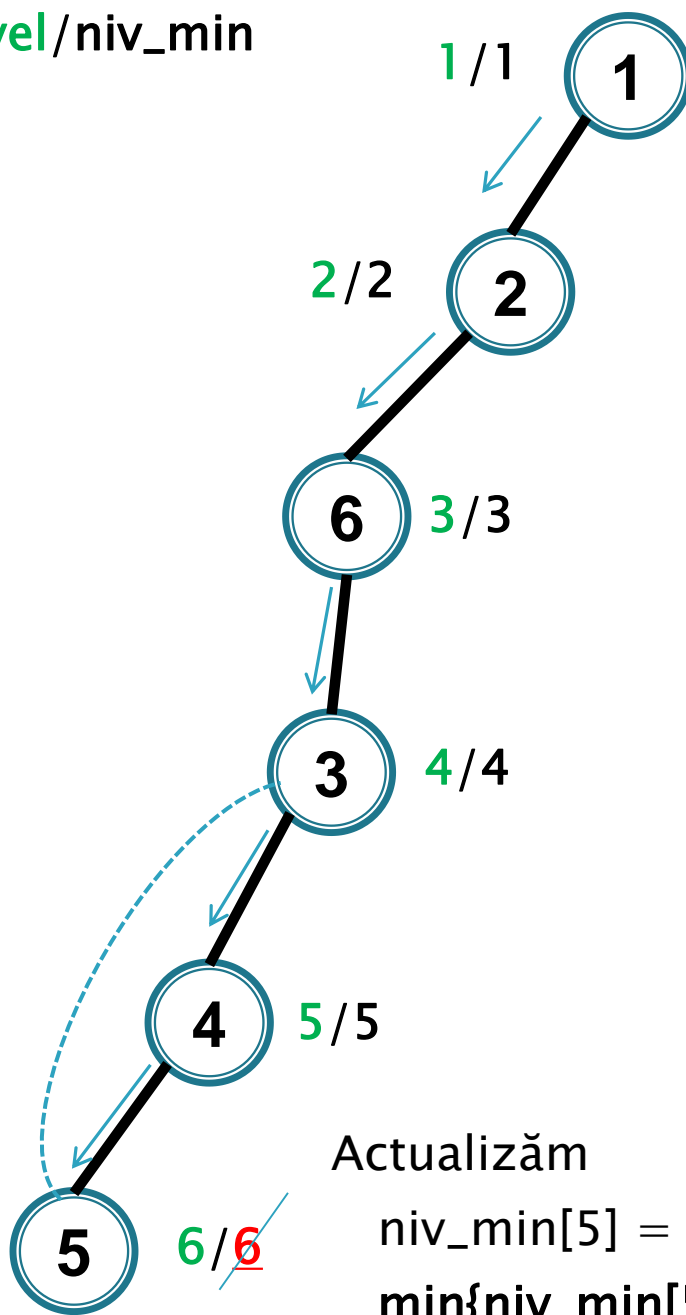
nivel/niv_min





S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3

nivel/niv_min

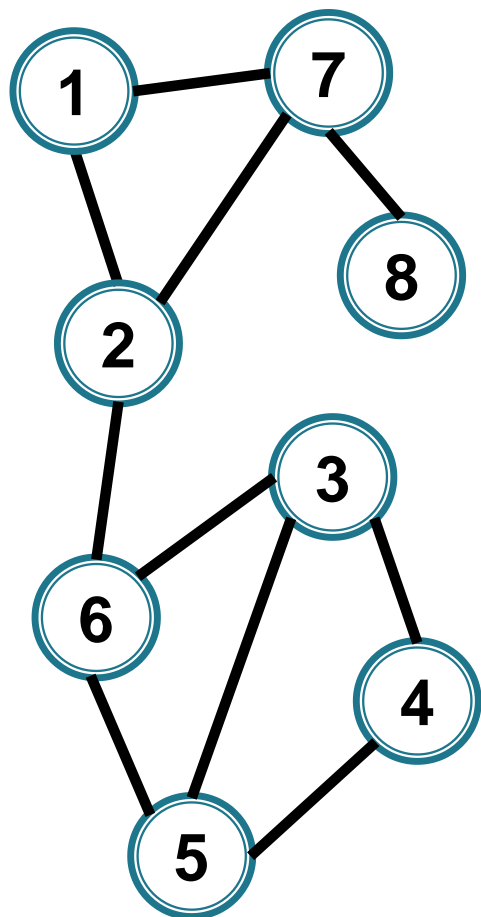


Actualizăm

$\text{niv_min}[5] =$

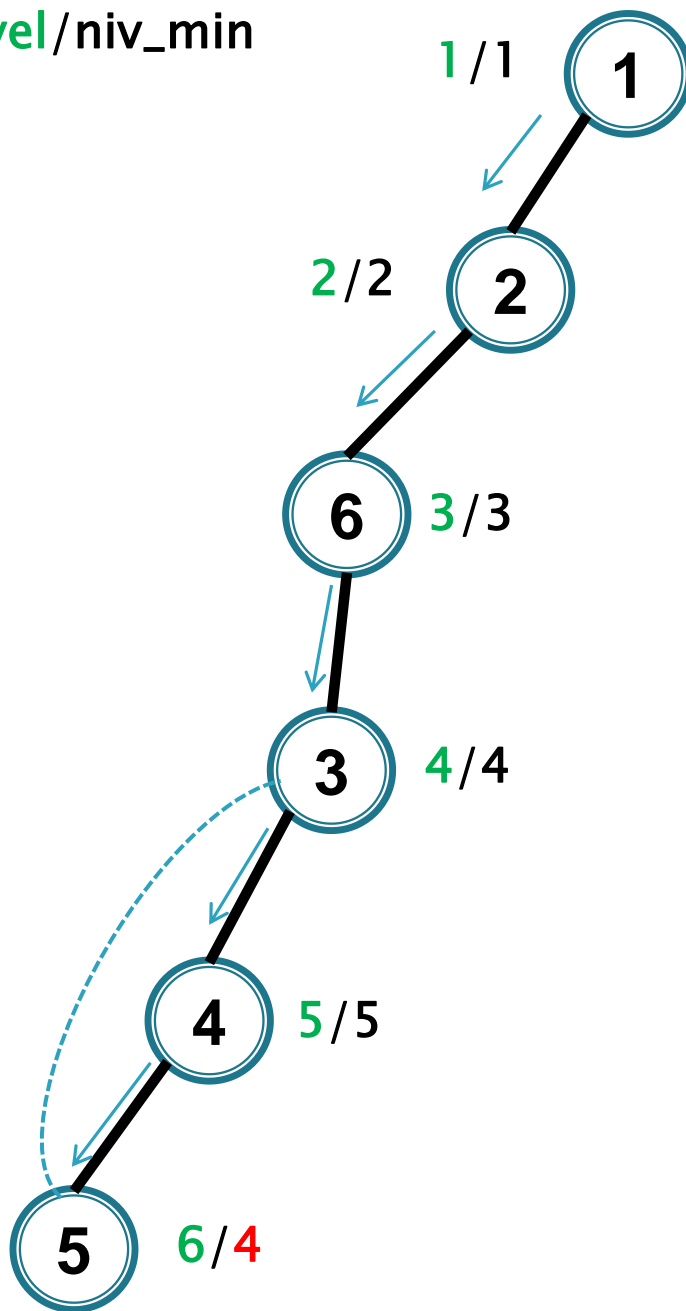
$\min\{\text{niv_min}[5], \text{nivel}[3]\}$

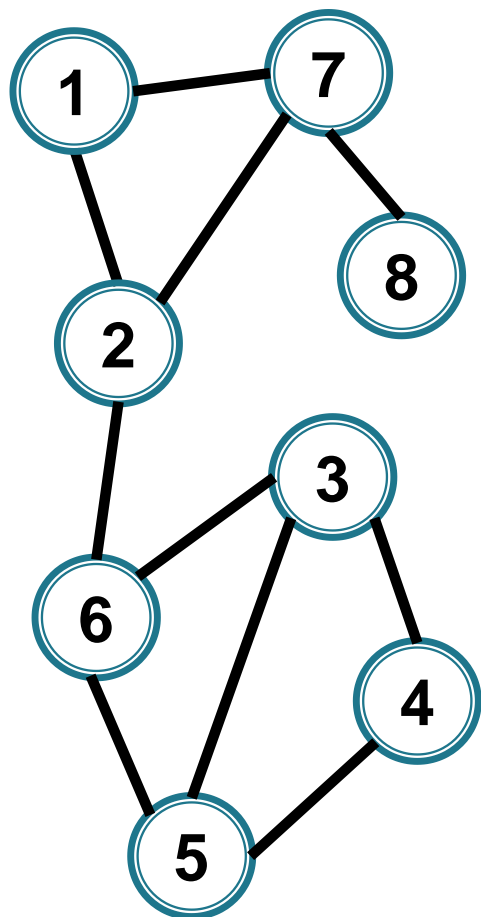
(cazul A)



S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3

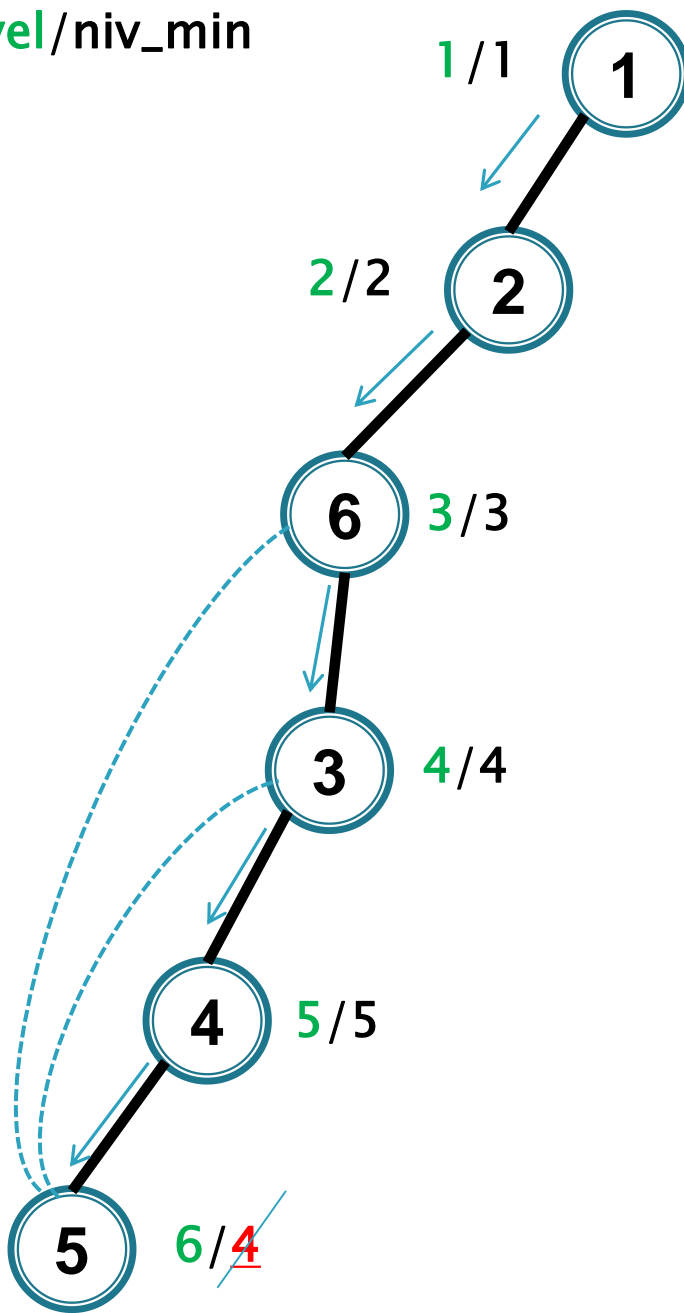
nivel/niv_min

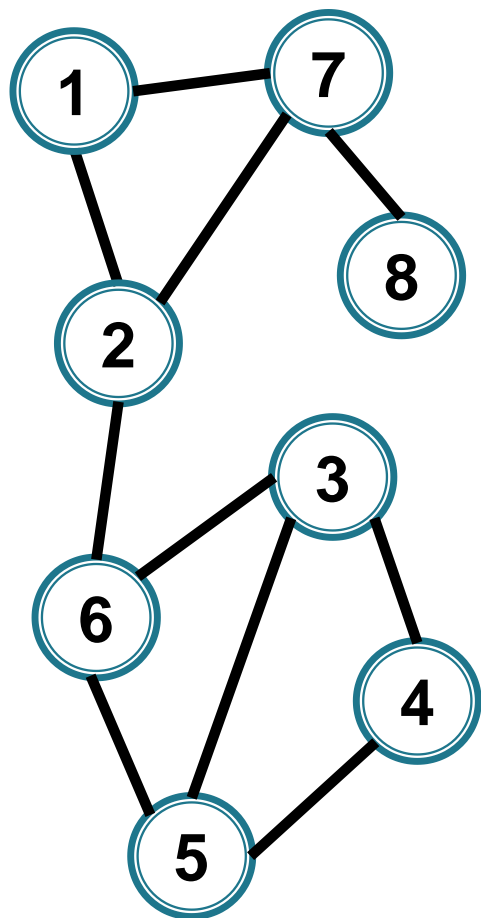




S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

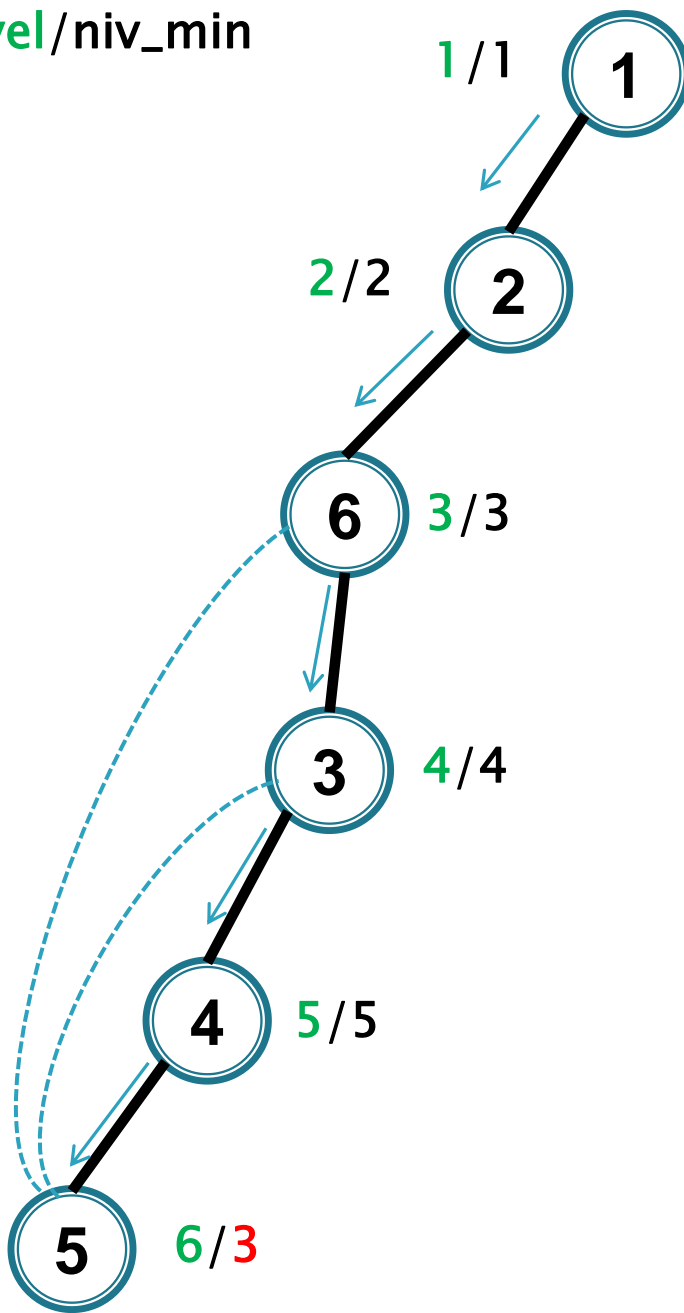
nivel/niv_min

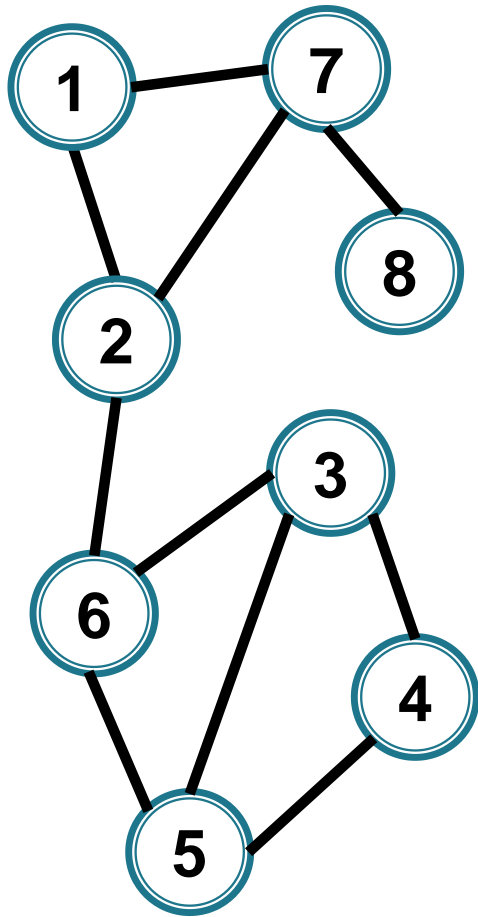




S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

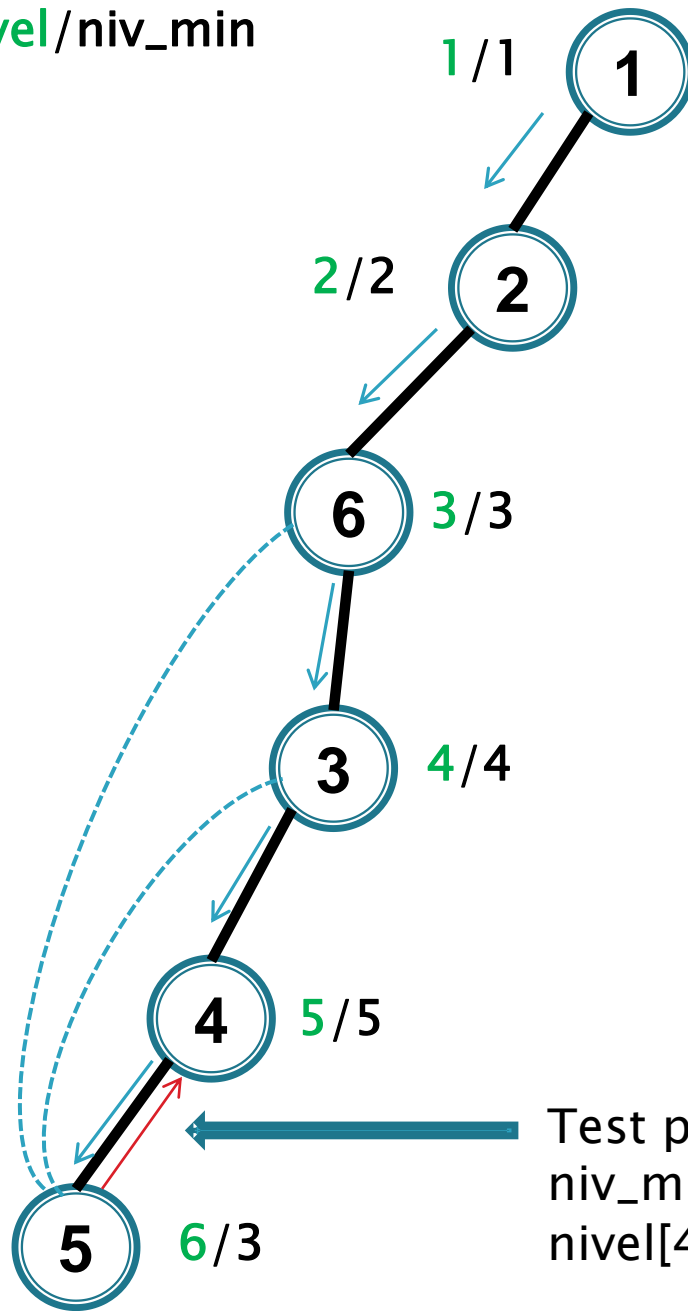
nivel/niv_min



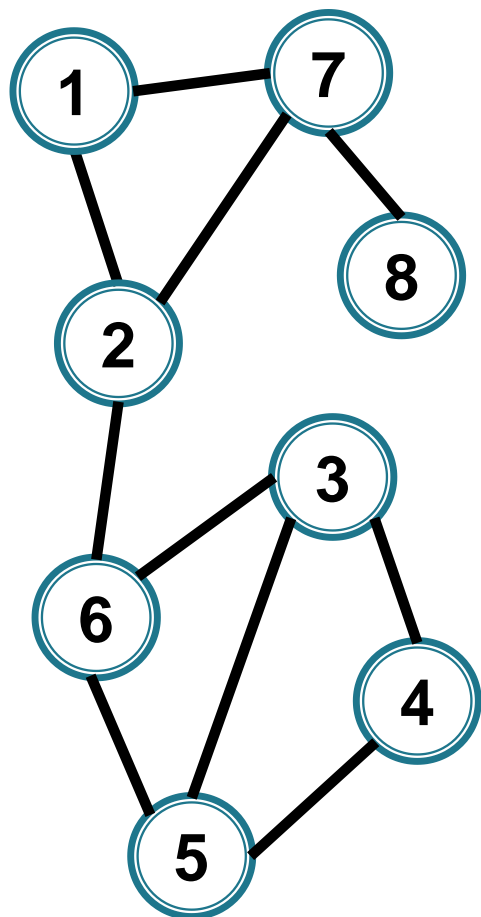


S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

nivel/niv_min

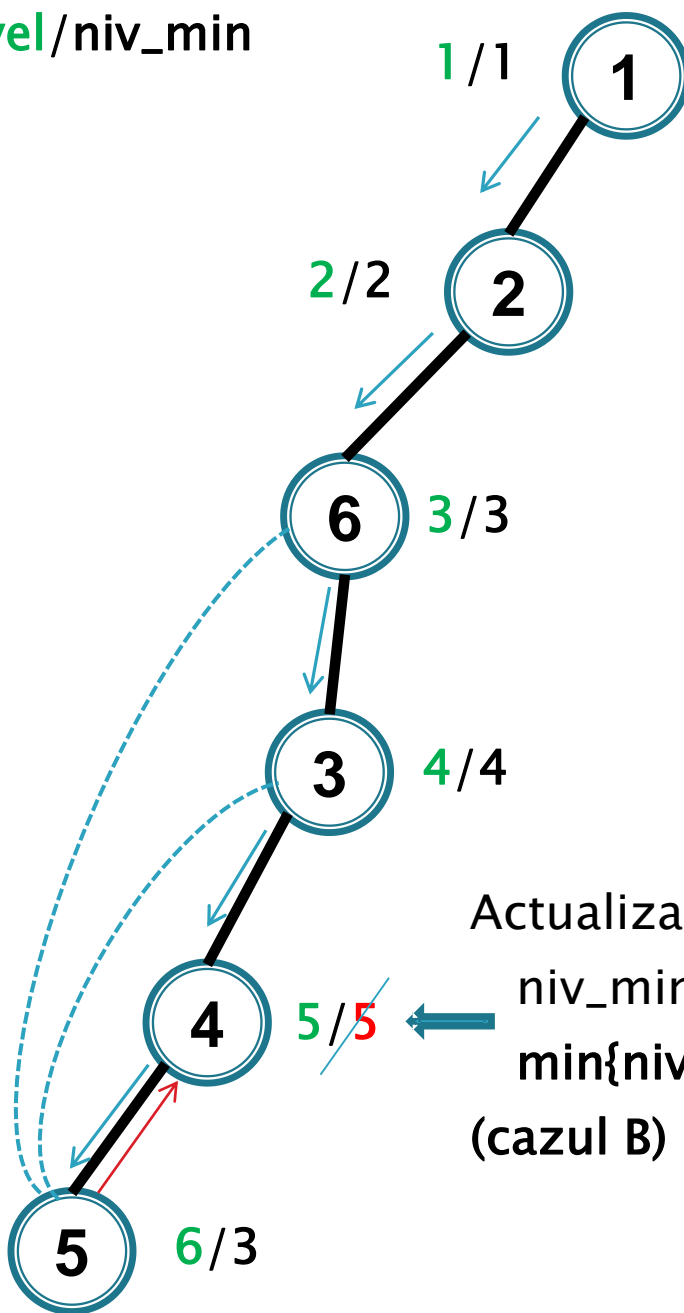


Test punct critic:
 $\text{niv_min}[5] = 3 < \text{nivel}[4] = 5 \Rightarrow \text{NU}$



S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

nivel/niv_min

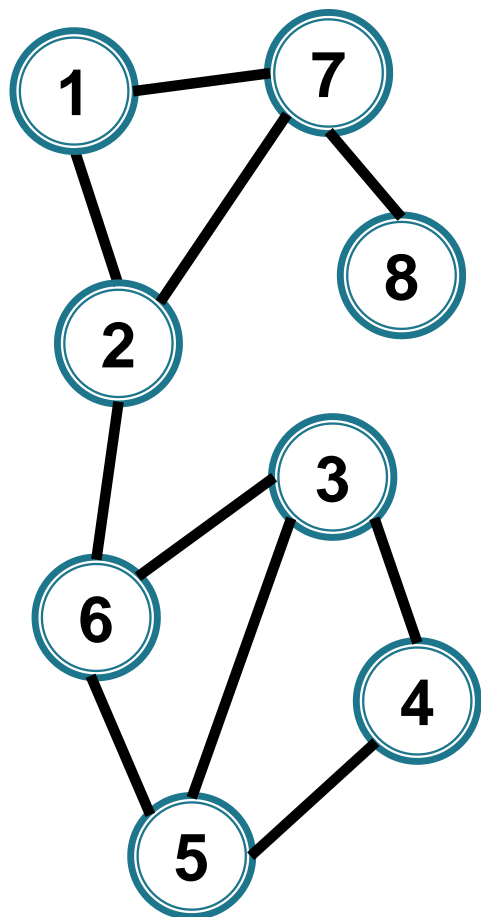


Actualizam

$\text{niv_min}[4] =$

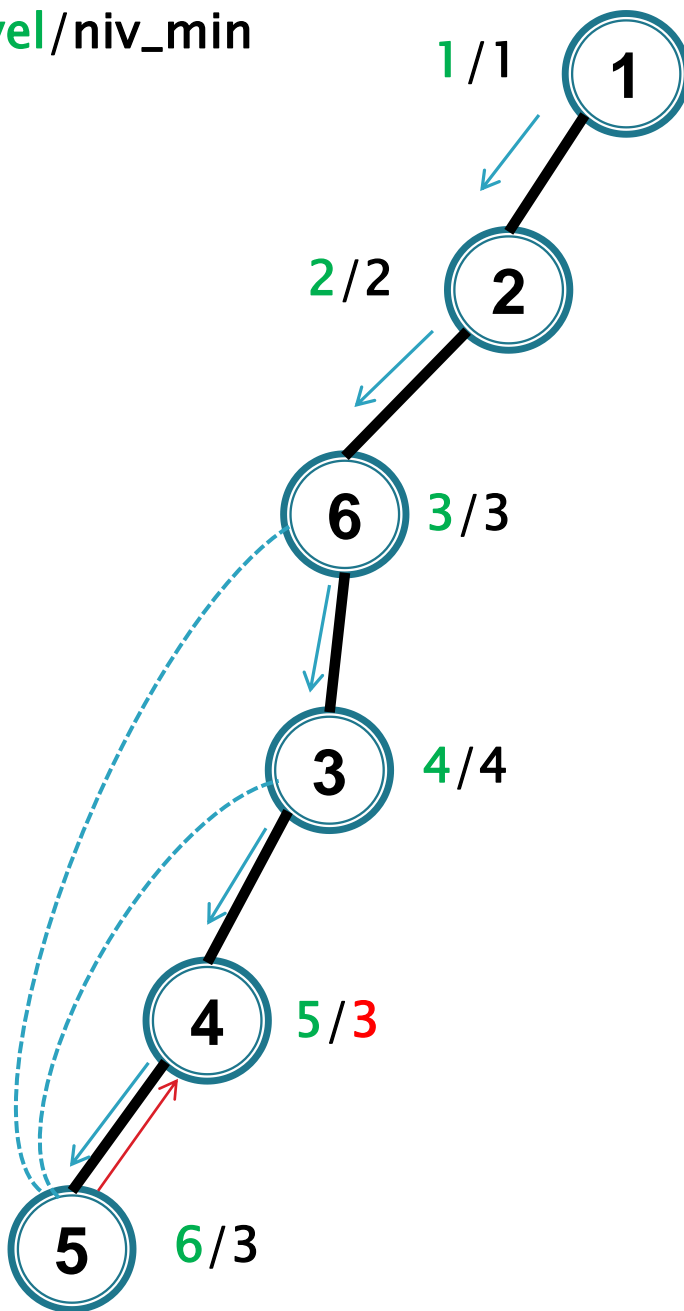
$\min\{\text{niv_min}[4], \text{niv_min}[5]\}$

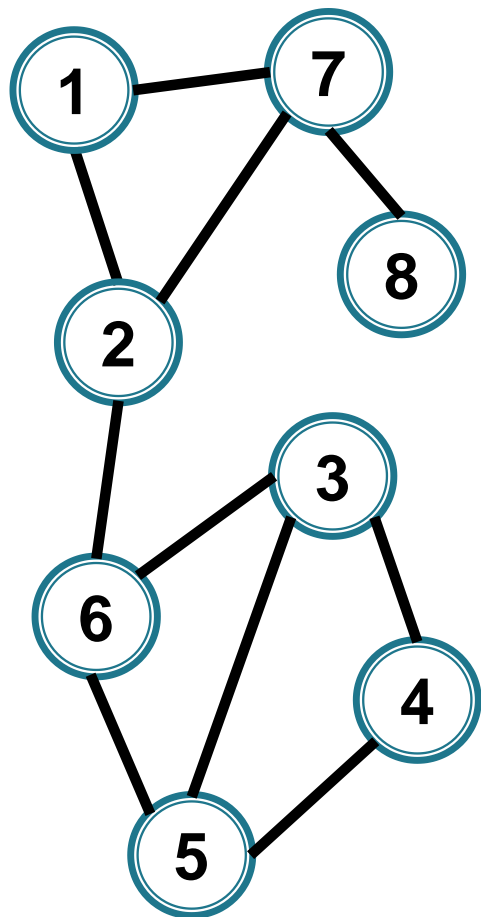
(cazul B)



S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

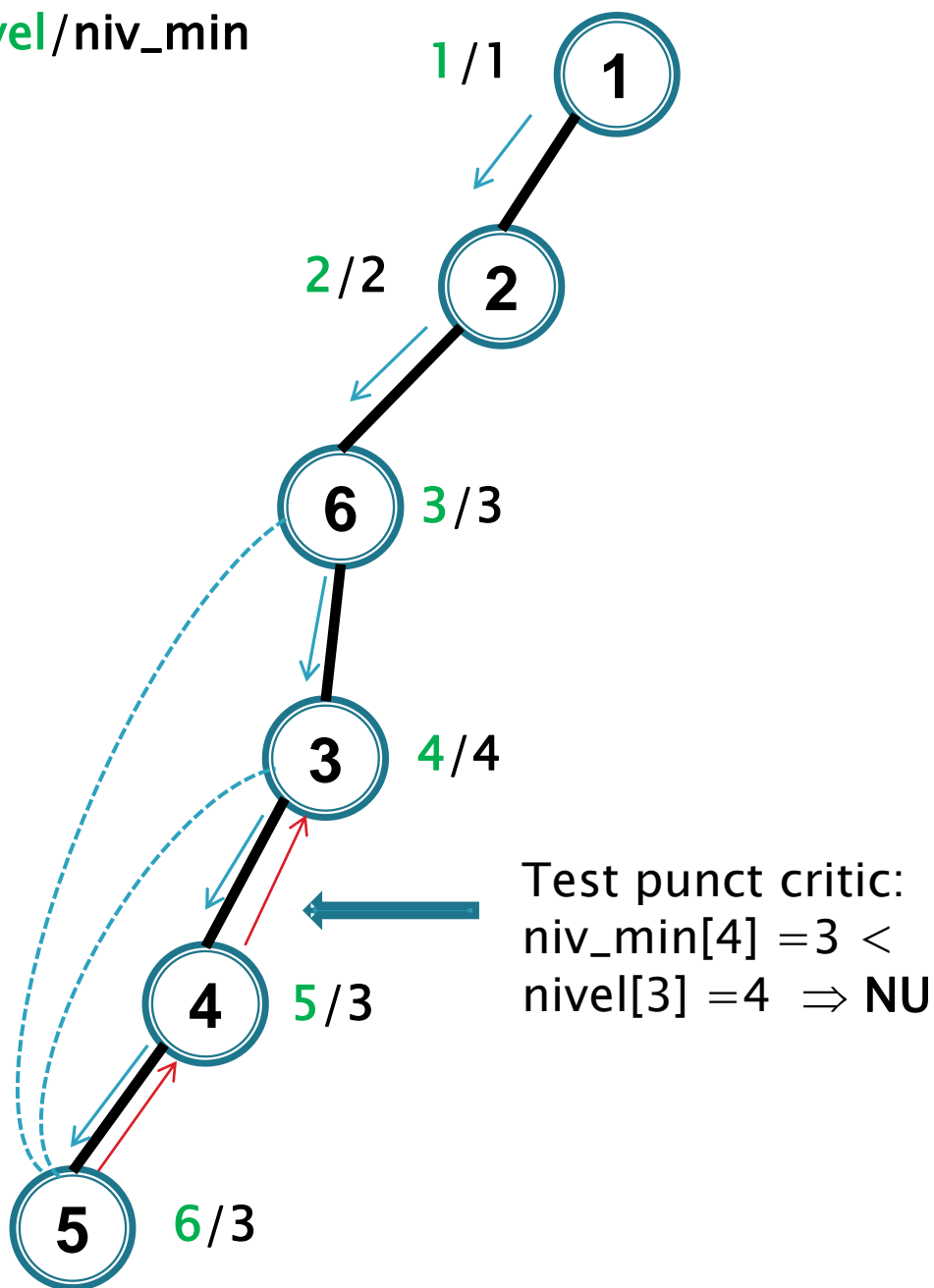
nivel/niv_min

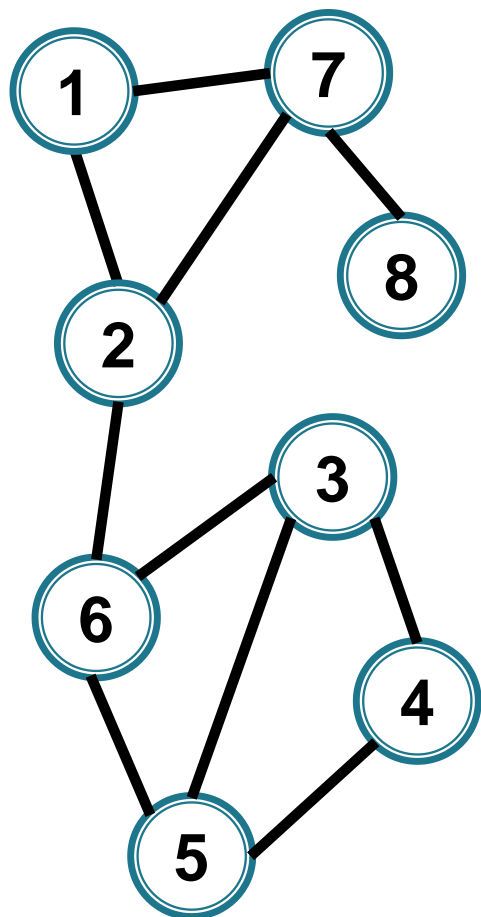




S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

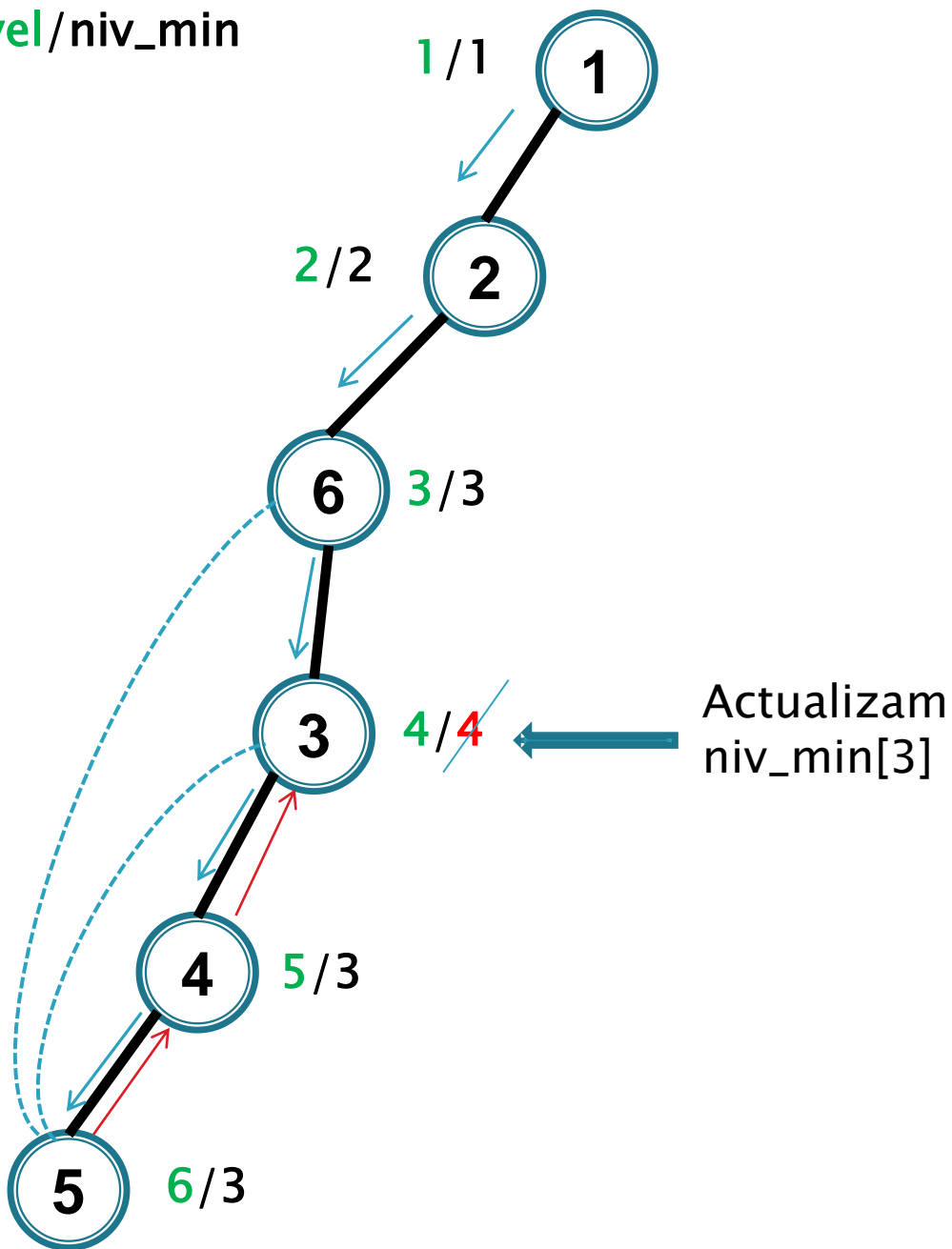
nivel/niv_min

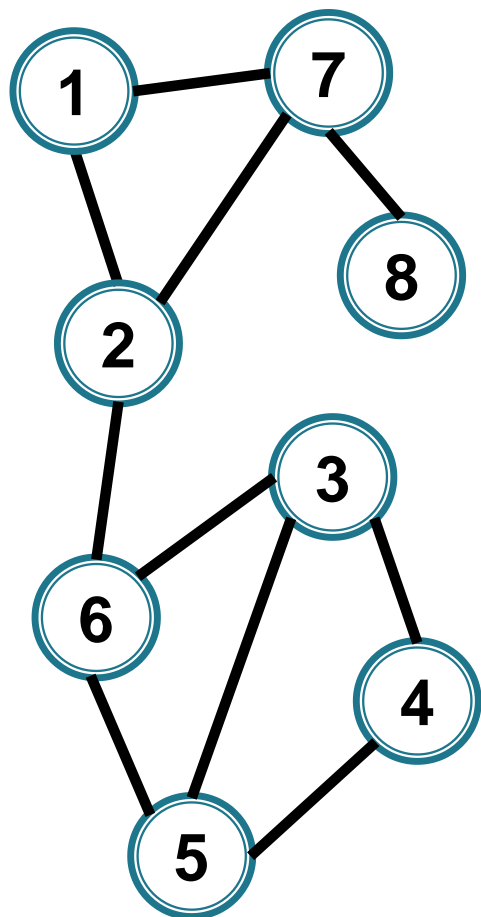




S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

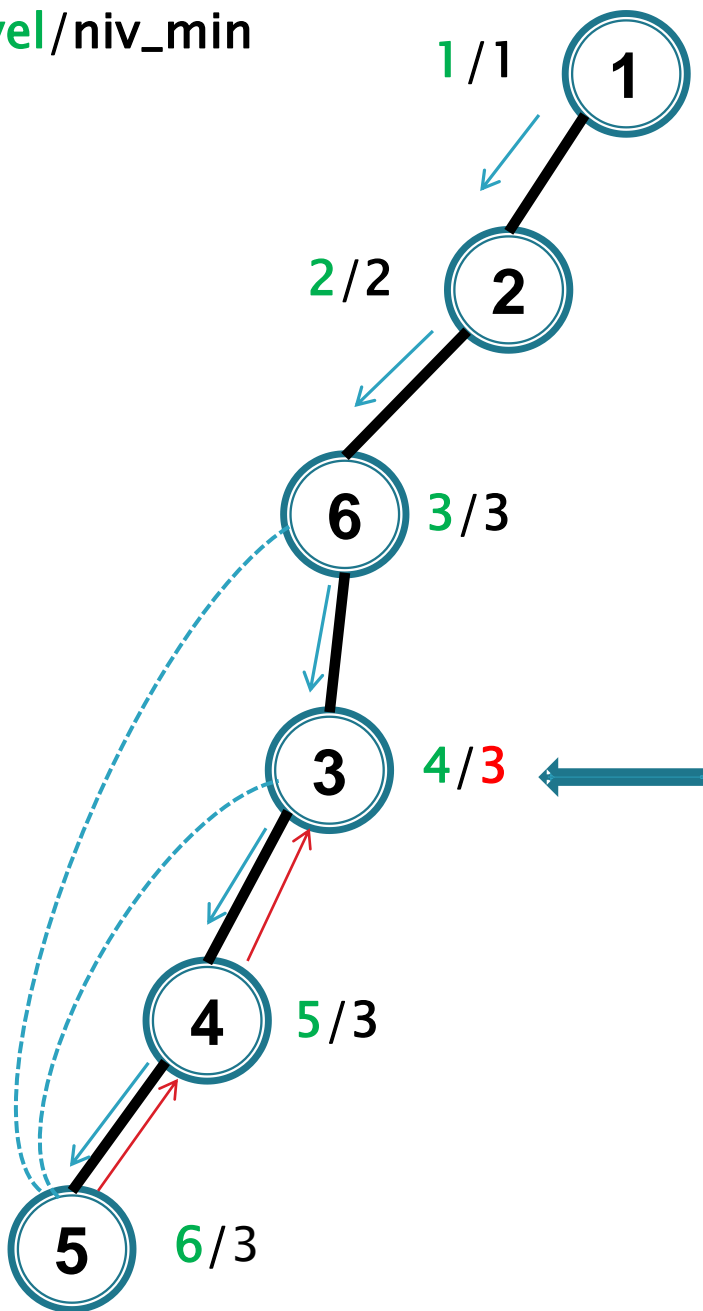
nivel/niv_min

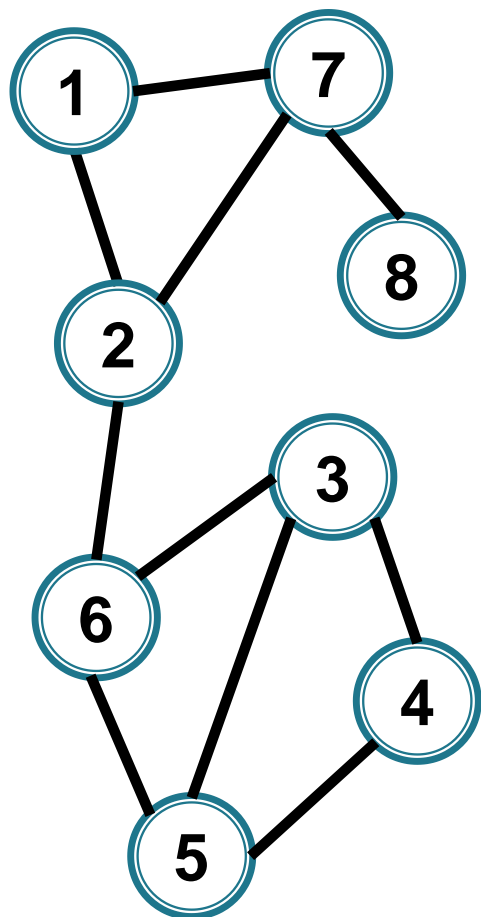




S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

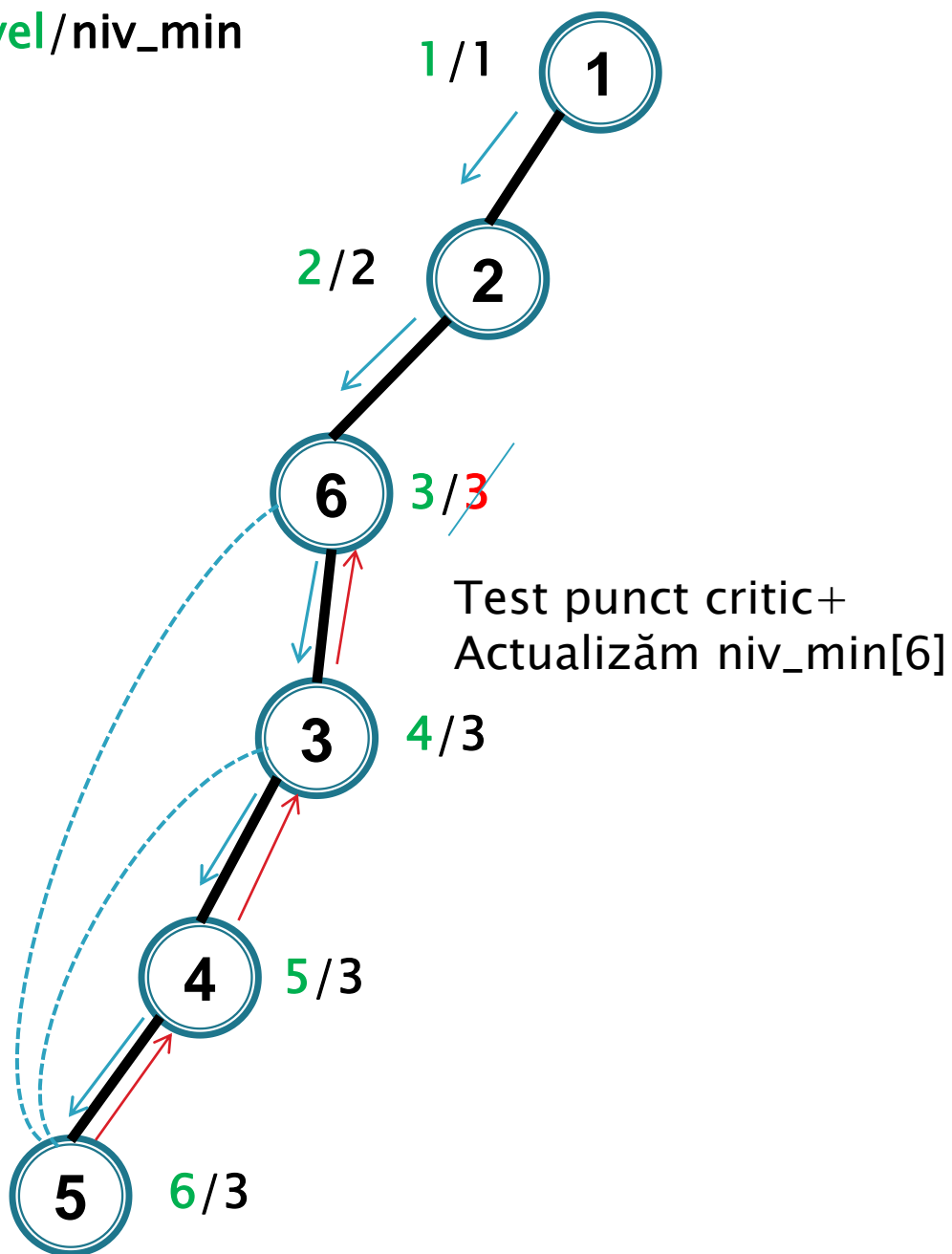
nivel/niv_min

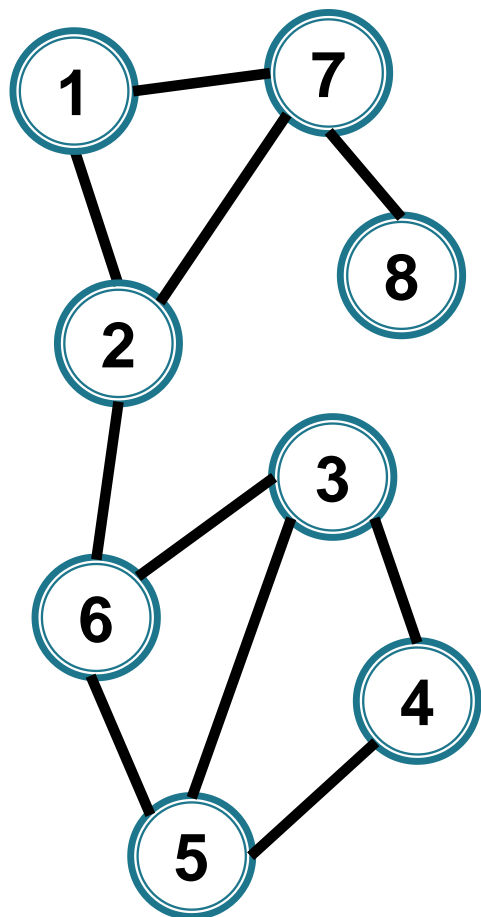




S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

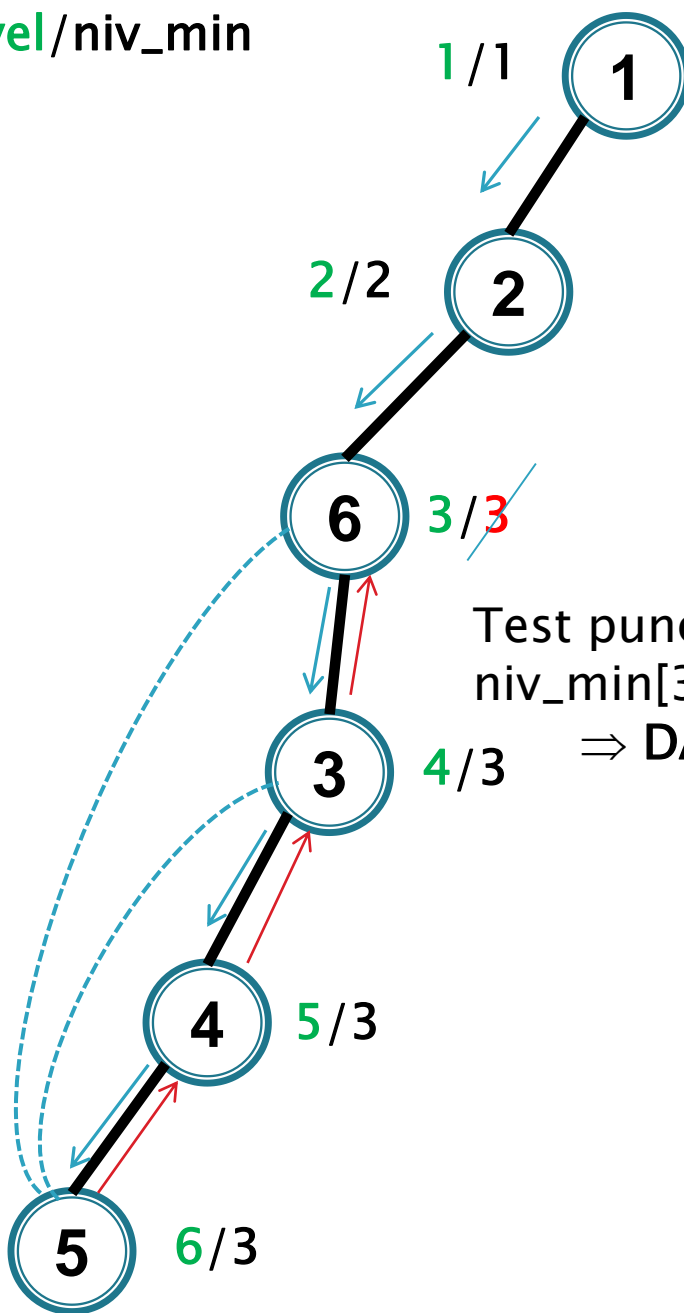
nivel/niv_min



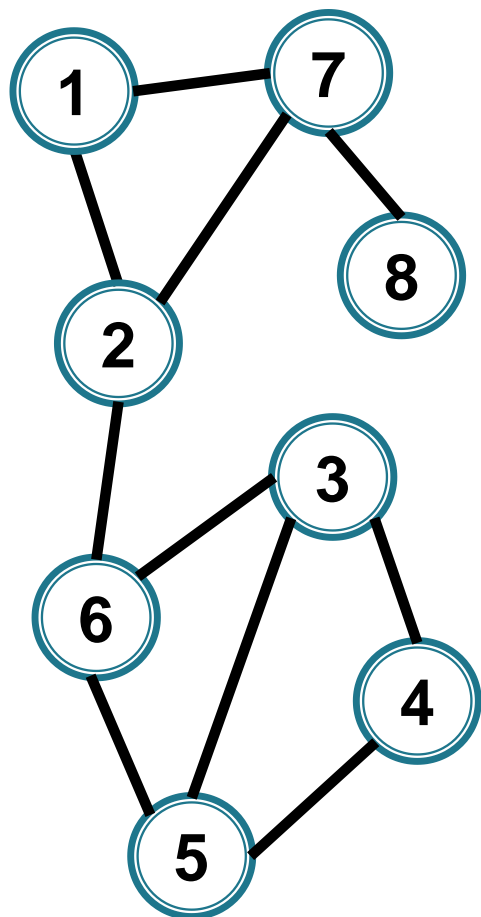


S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

nivel/niv_min

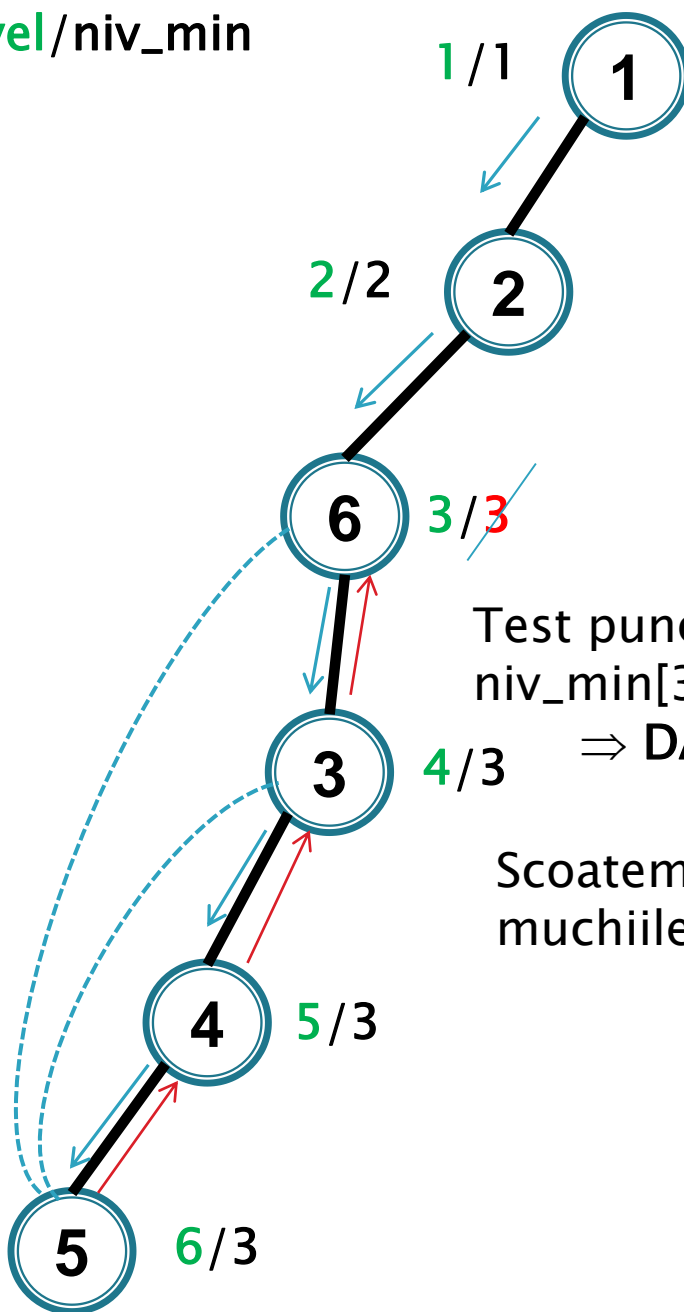


Test punct critic:
 $\text{niv_min}[3] = 3 = \text{nivel}[6]$
 $\Rightarrow \text{DA}$



S:
 1 2
 2 6
 6 3
 3 4
 4 5
 5 3
 5 6

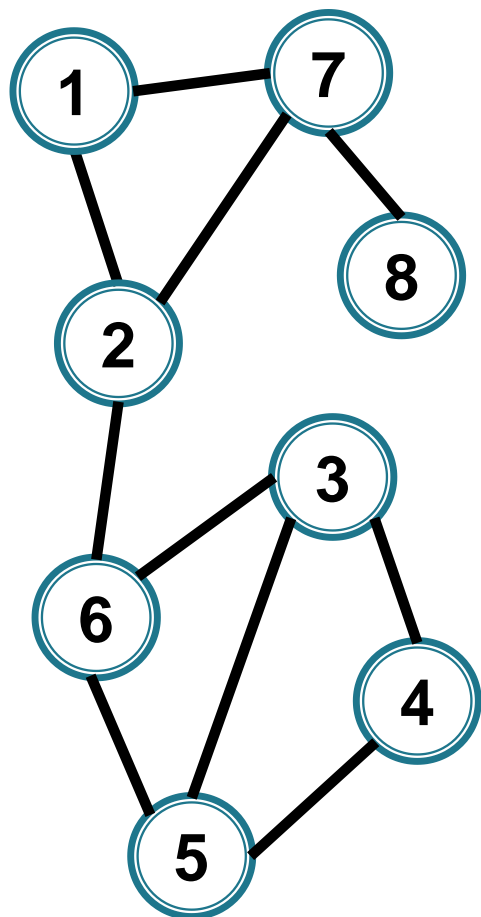
nivel/niv_min



Test punct critic:
 $\text{niv_min}[3] = 3 = \text{nivel}[6]$

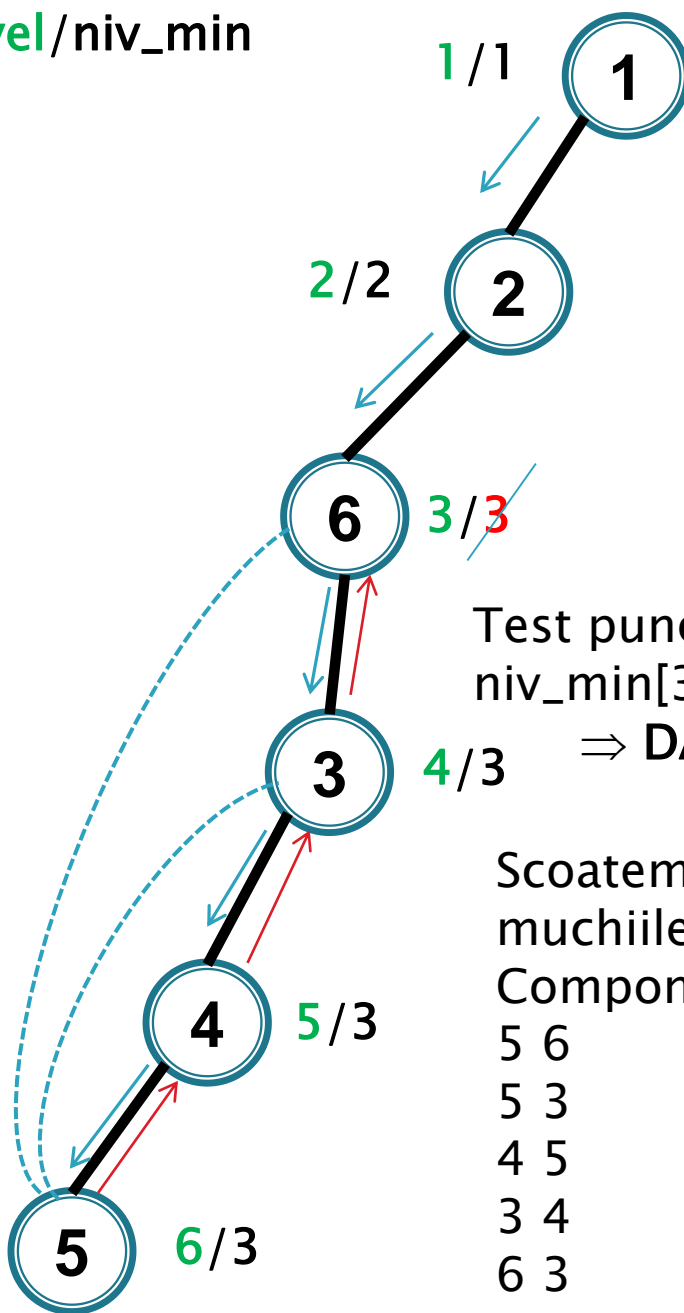
$\Rightarrow \text{DA}$

Scoatem din stiva toate
 muchiile pana la 3 6 \Rightarrow



S:
1 2
2 6

nivel/niv_min

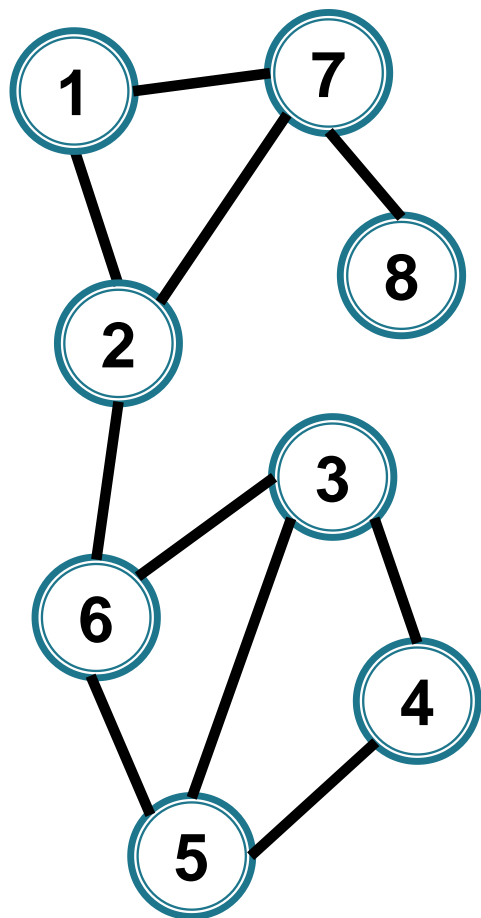


Test punct critic:
 $\text{niv_min}[3] = 3 = \text{nivel}[6]$

\Rightarrow DA

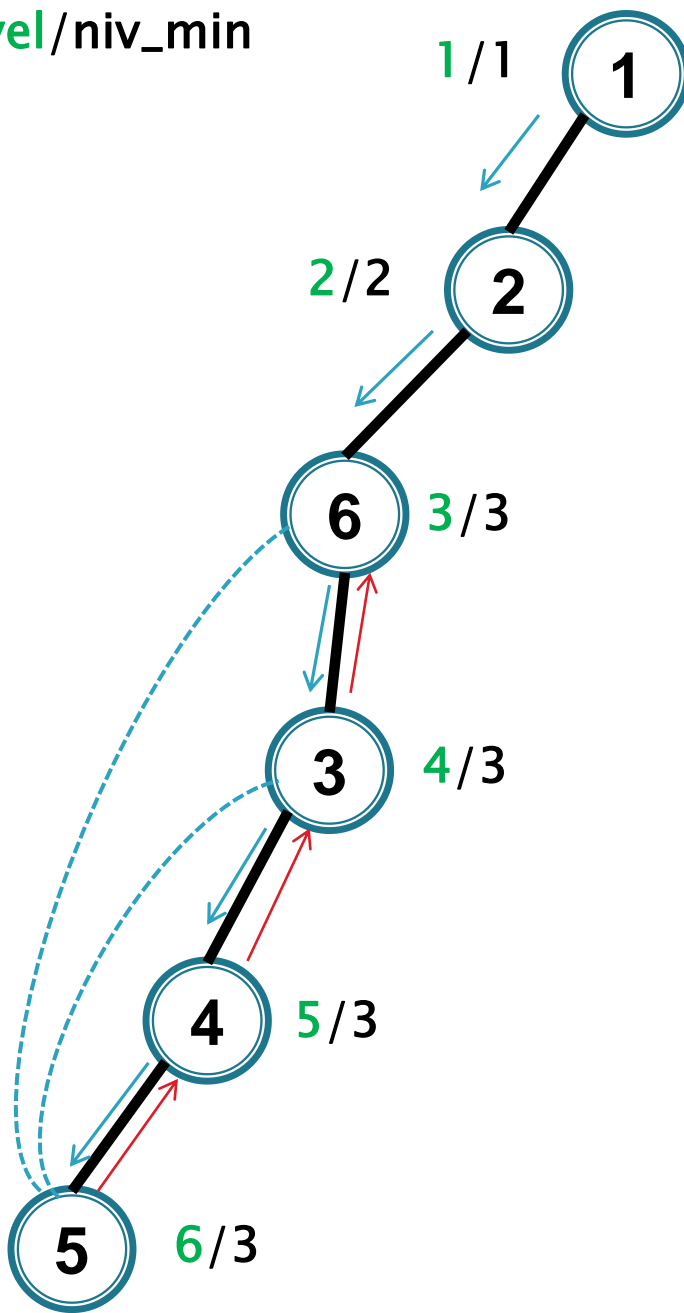
Scoatem din stiva toate
muchiiile pana la 3 6 \Rightarrow
Componenta cu muchiiile

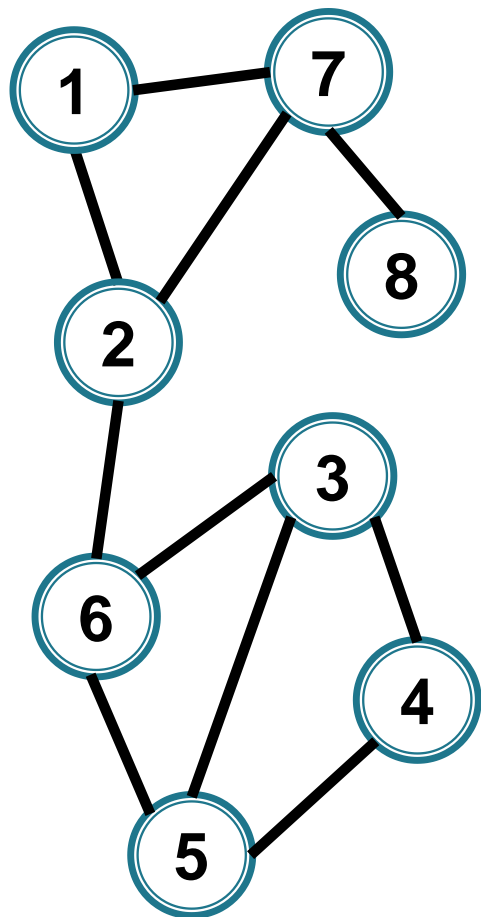
5 6
5 3
4 5
3 4
6 3



S:
1 2
2 6

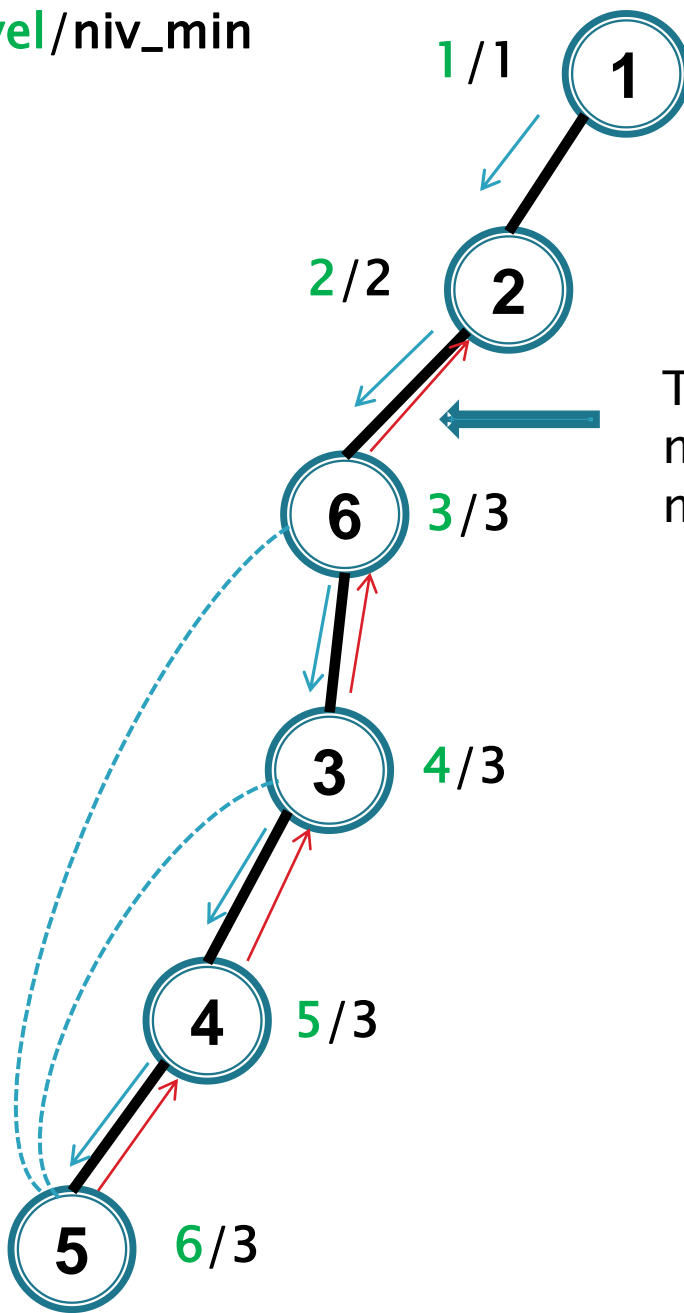
nivel/niv_min



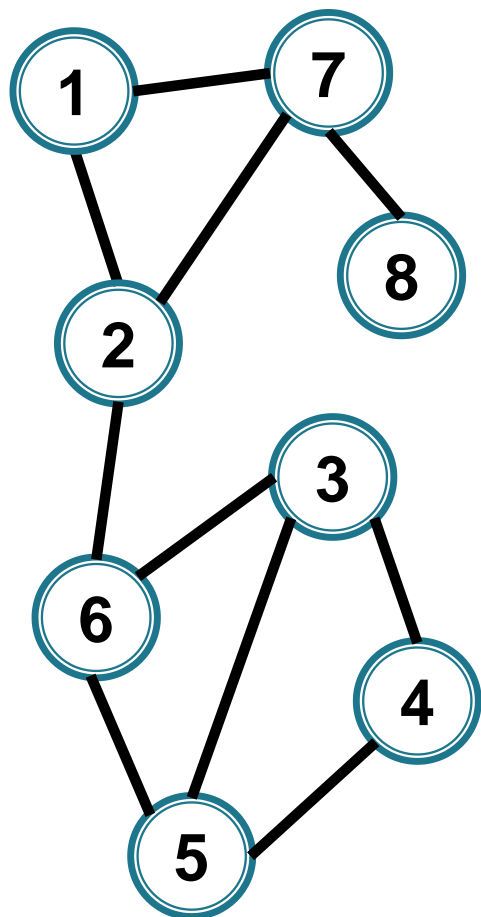


S:
1 2
2 6

nivel/niv_min

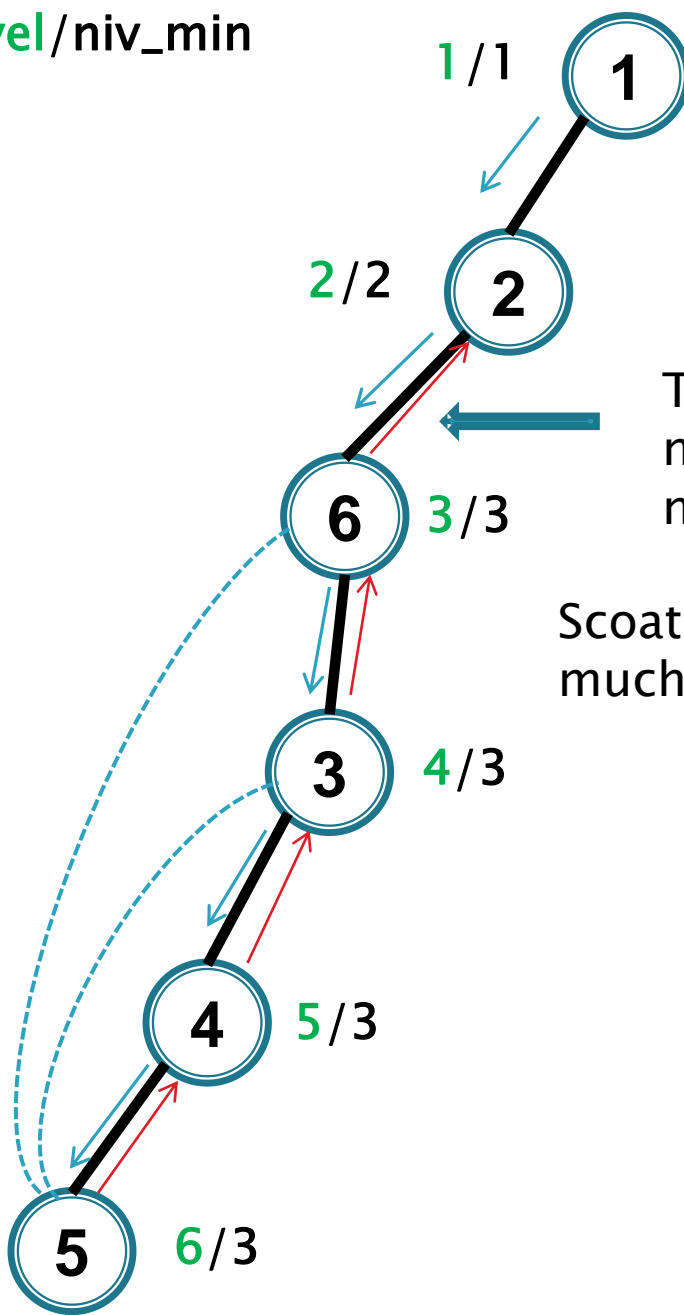


Test punct critic:
 $\text{niv_min}[6] = 3 >$
 $\text{nivel}[2] = 2 \Rightarrow \text{DA}$



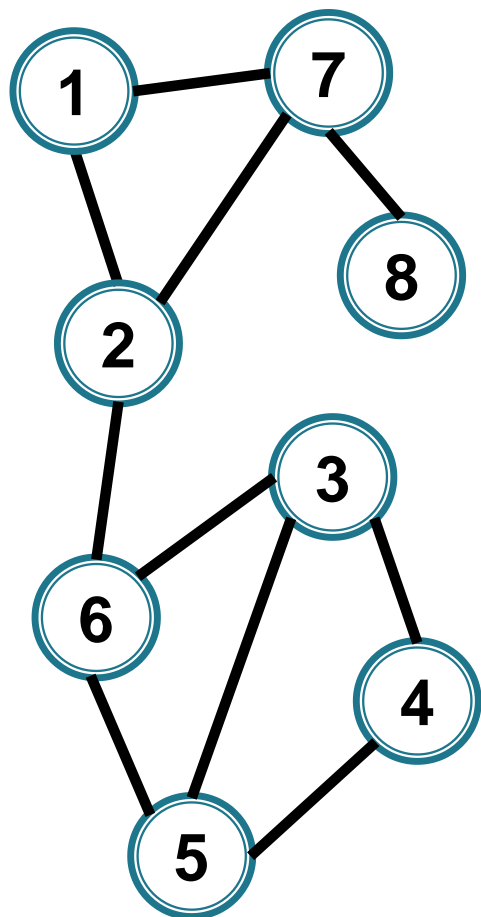
S:
1 2
2 6

nivel/niv_min



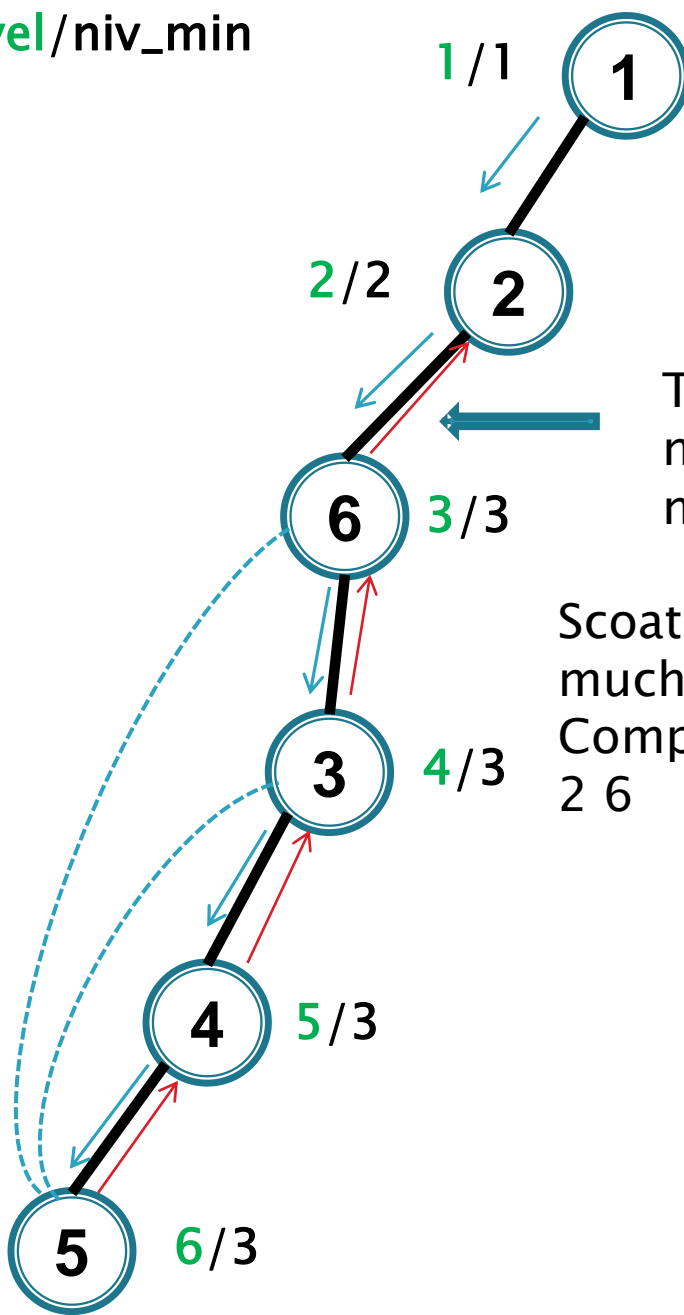
Test punct critic:
 $\text{niv_min}[6] = 3 >$
 $\text{nivel}[2] = 2 \Rightarrow \text{DA}$

Scoatem din stiva toate
muchiiile pana la 2 6 =>



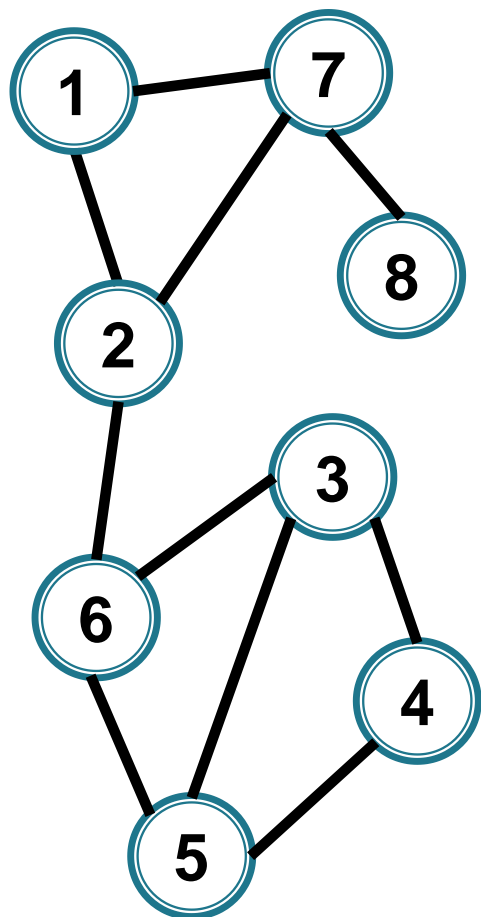
S:
1 2

nivel/niv_min



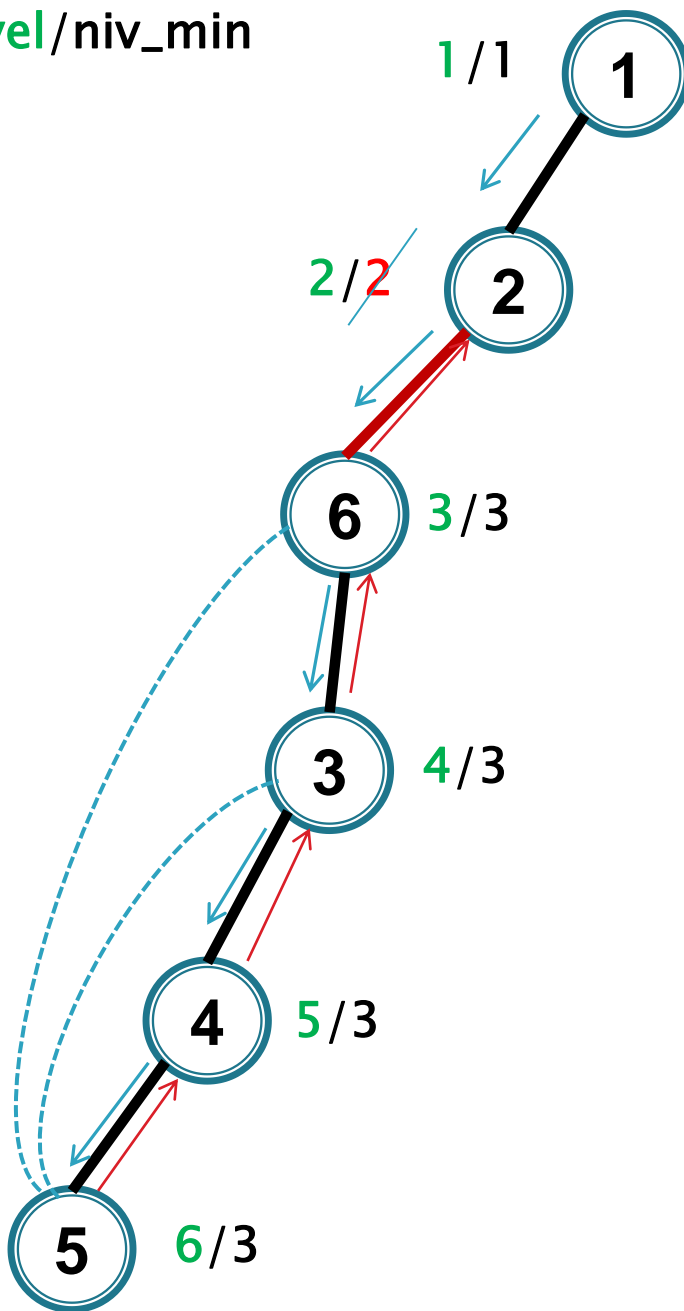
Test punct critic:
niv_min[6] = 3 >
nivel[2] = 2 ⇒ **DA**

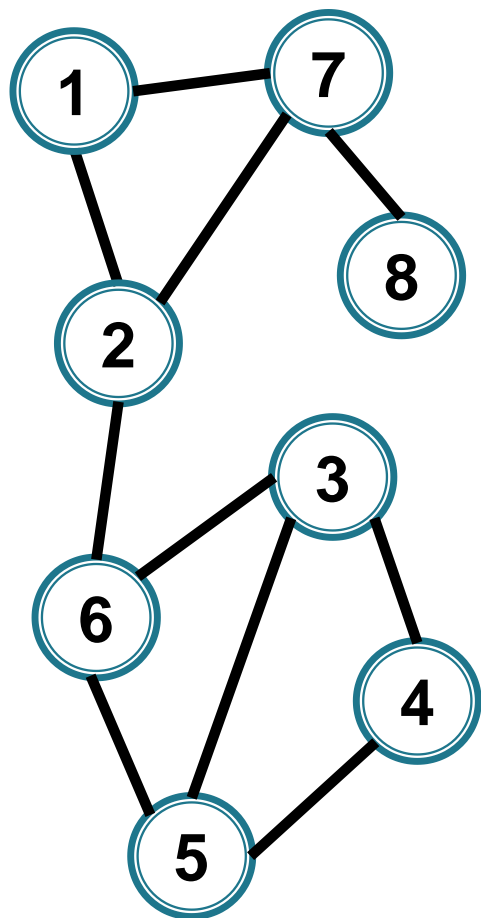
Scoatem din stiva toate
muchile pana la 2 6 =>
Componenta cu muchiile
2 6



S:
1 2

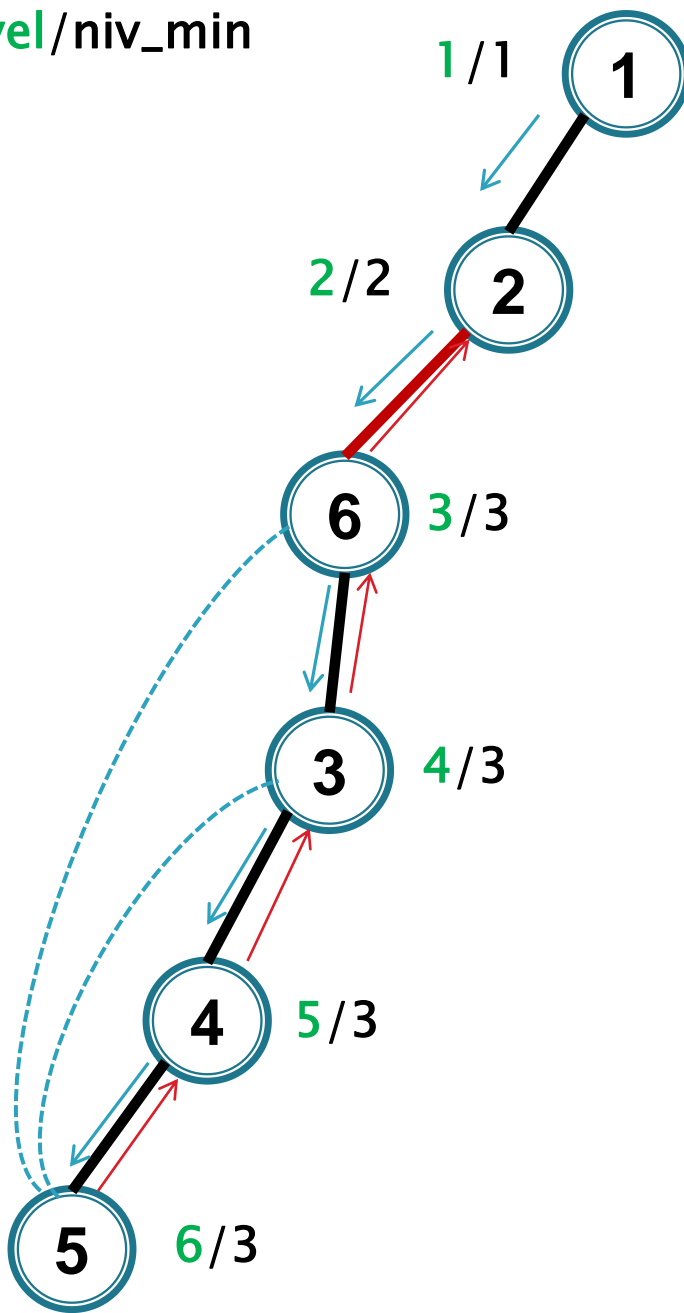
nivel/niv_min

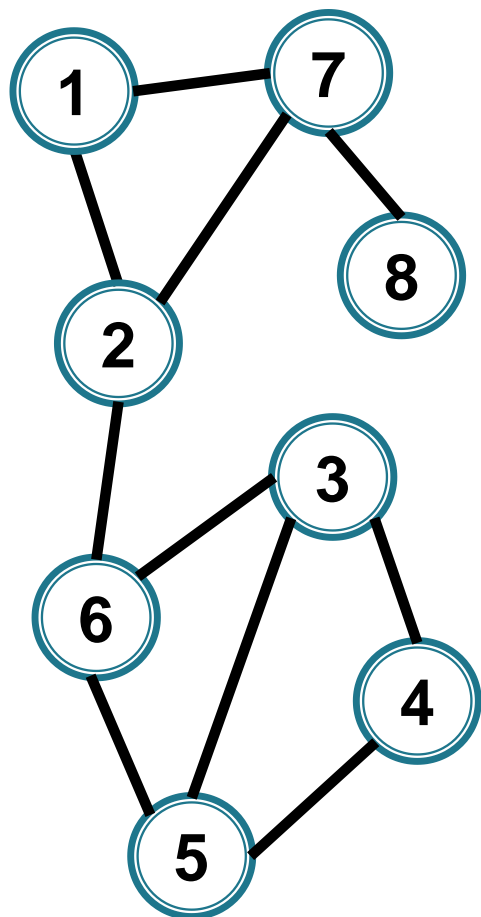




S:
1 2

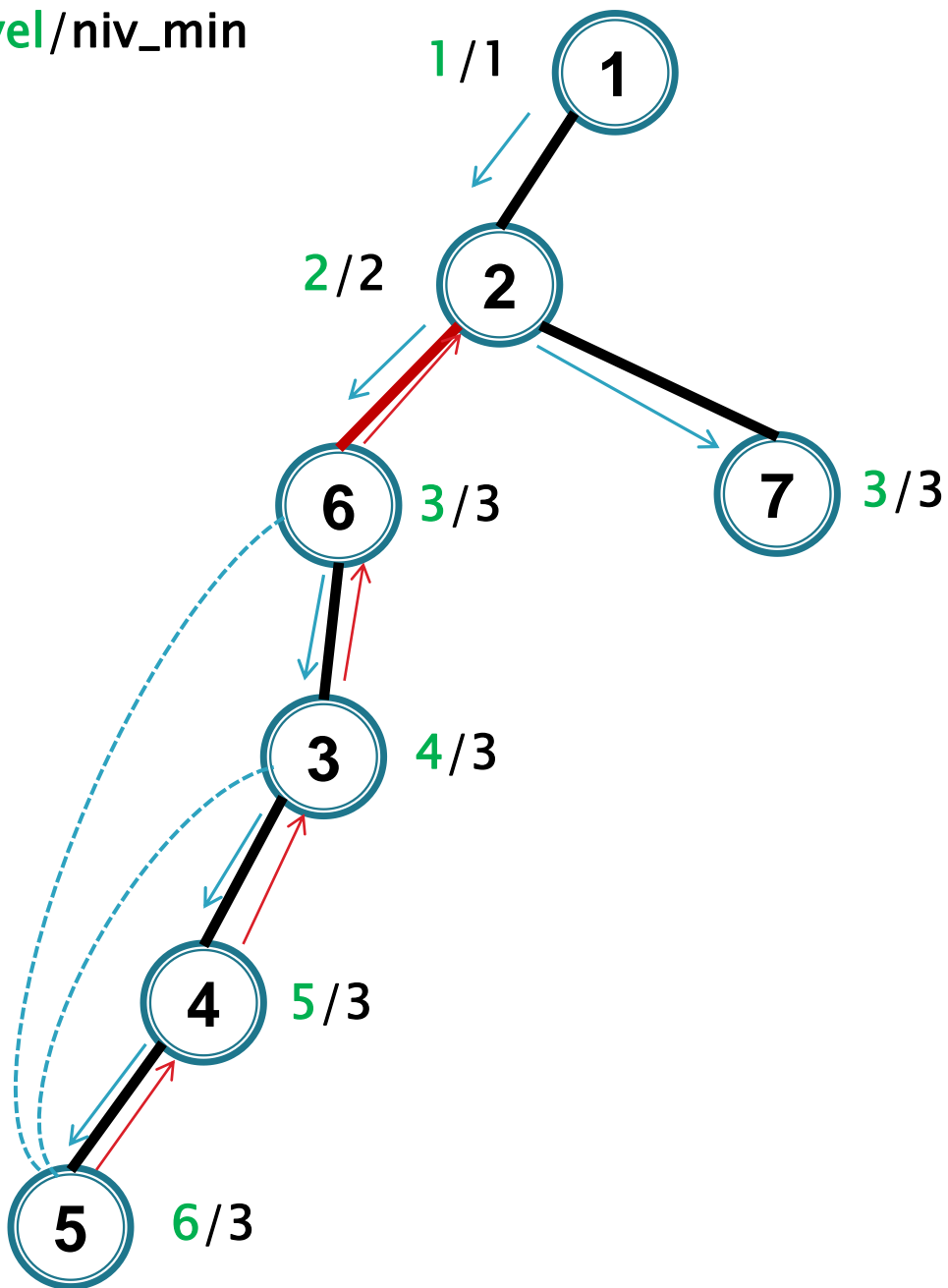
nivel/niv_min

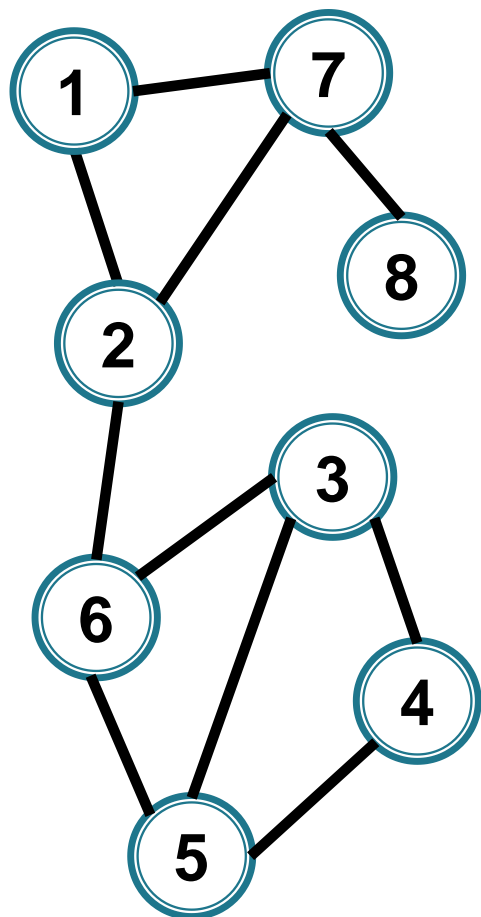




S:
1 2
2 7

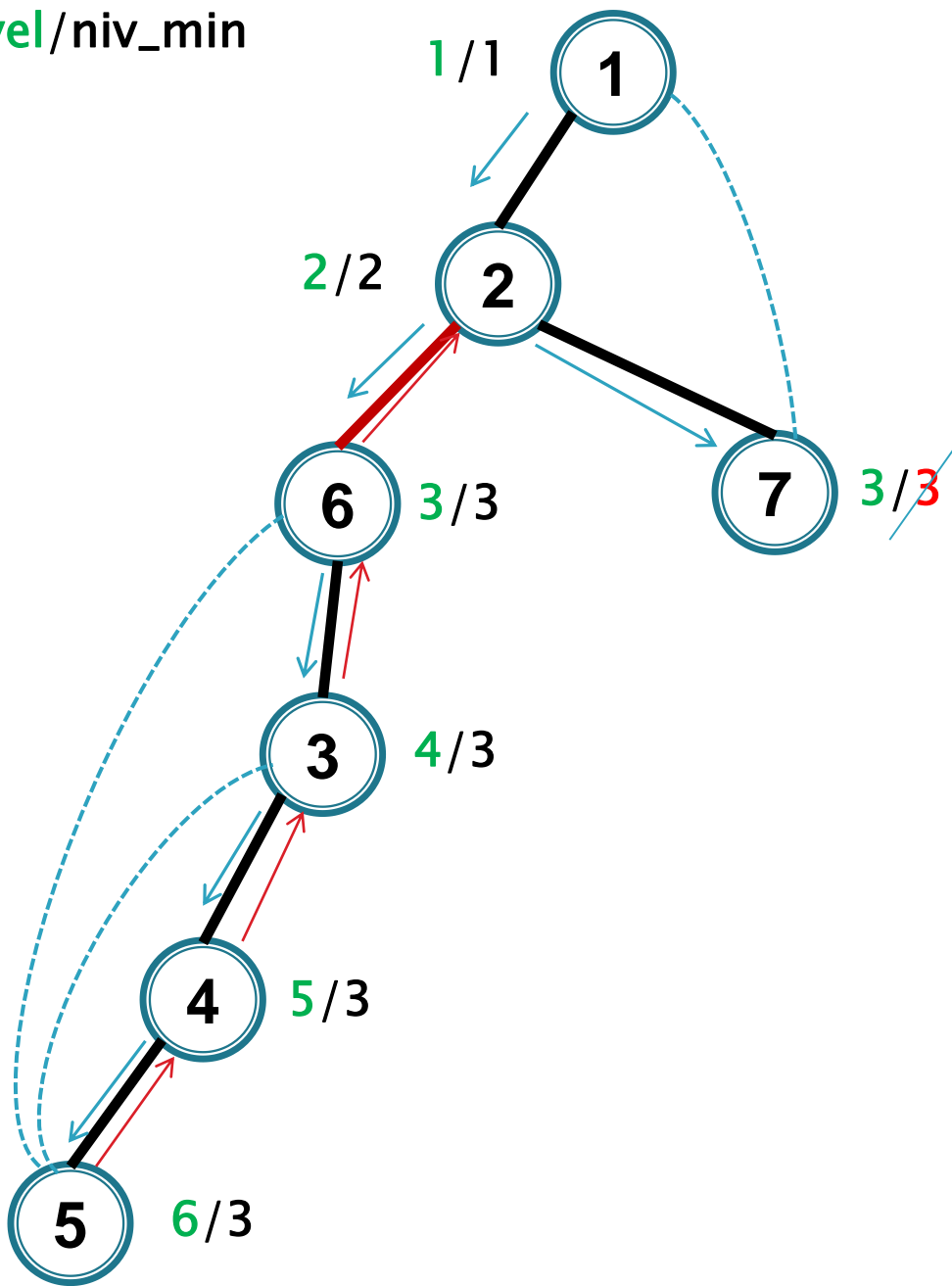
nivel/niv_min

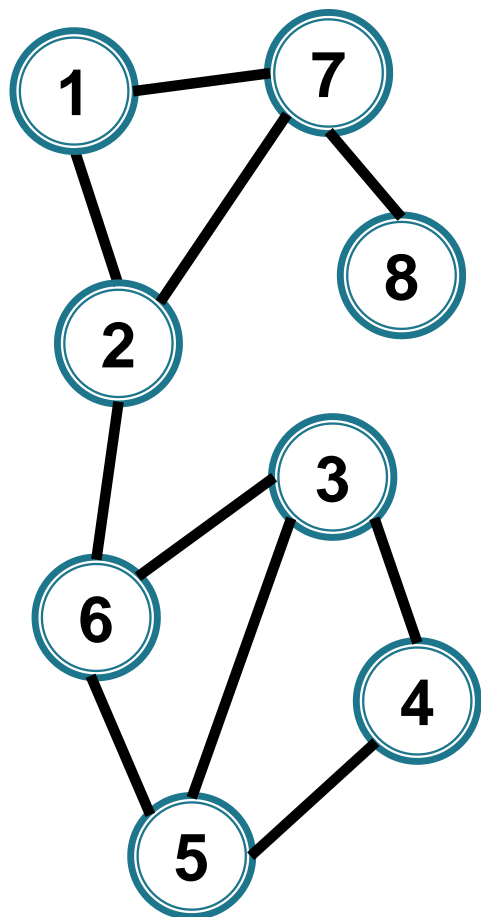




S:
1 2
2 7
1 7

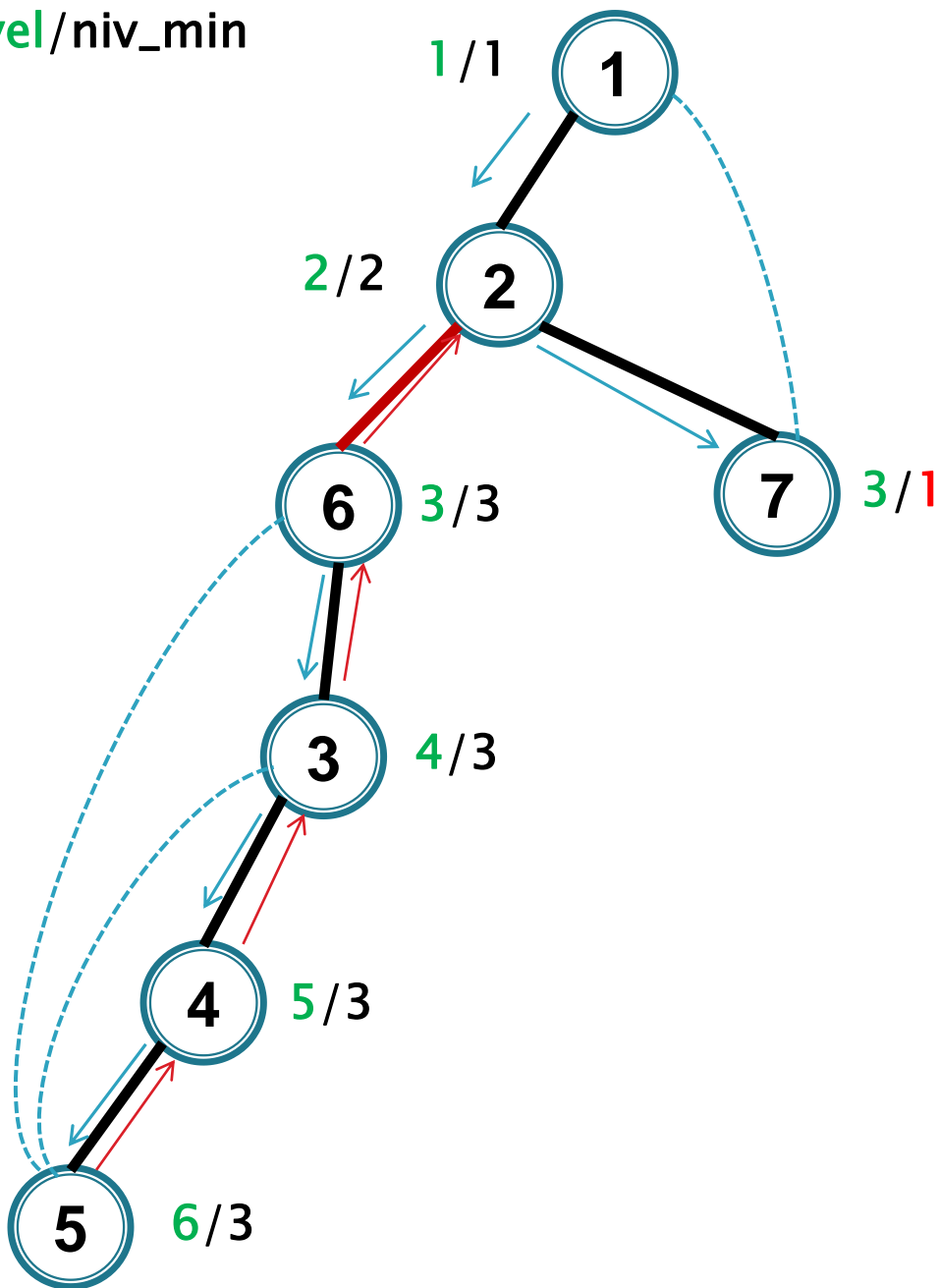
nivel/niv_min

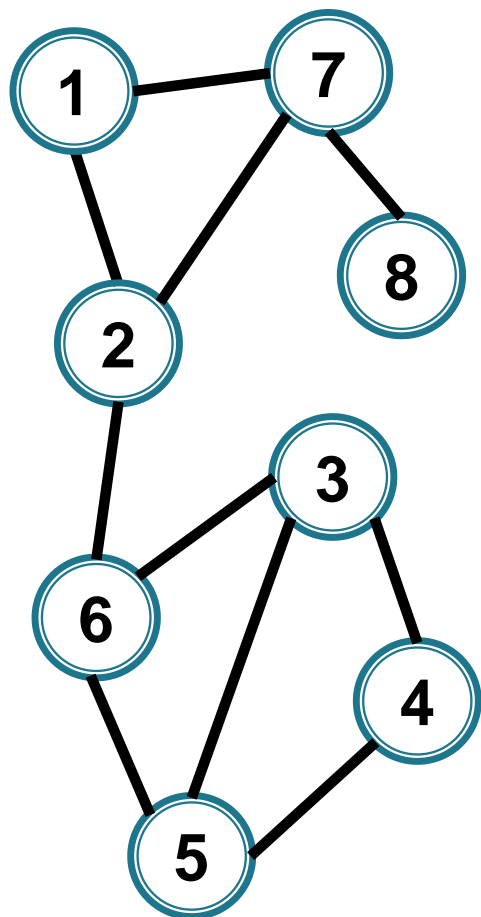




S:
 1 2
 2 7
 1 7

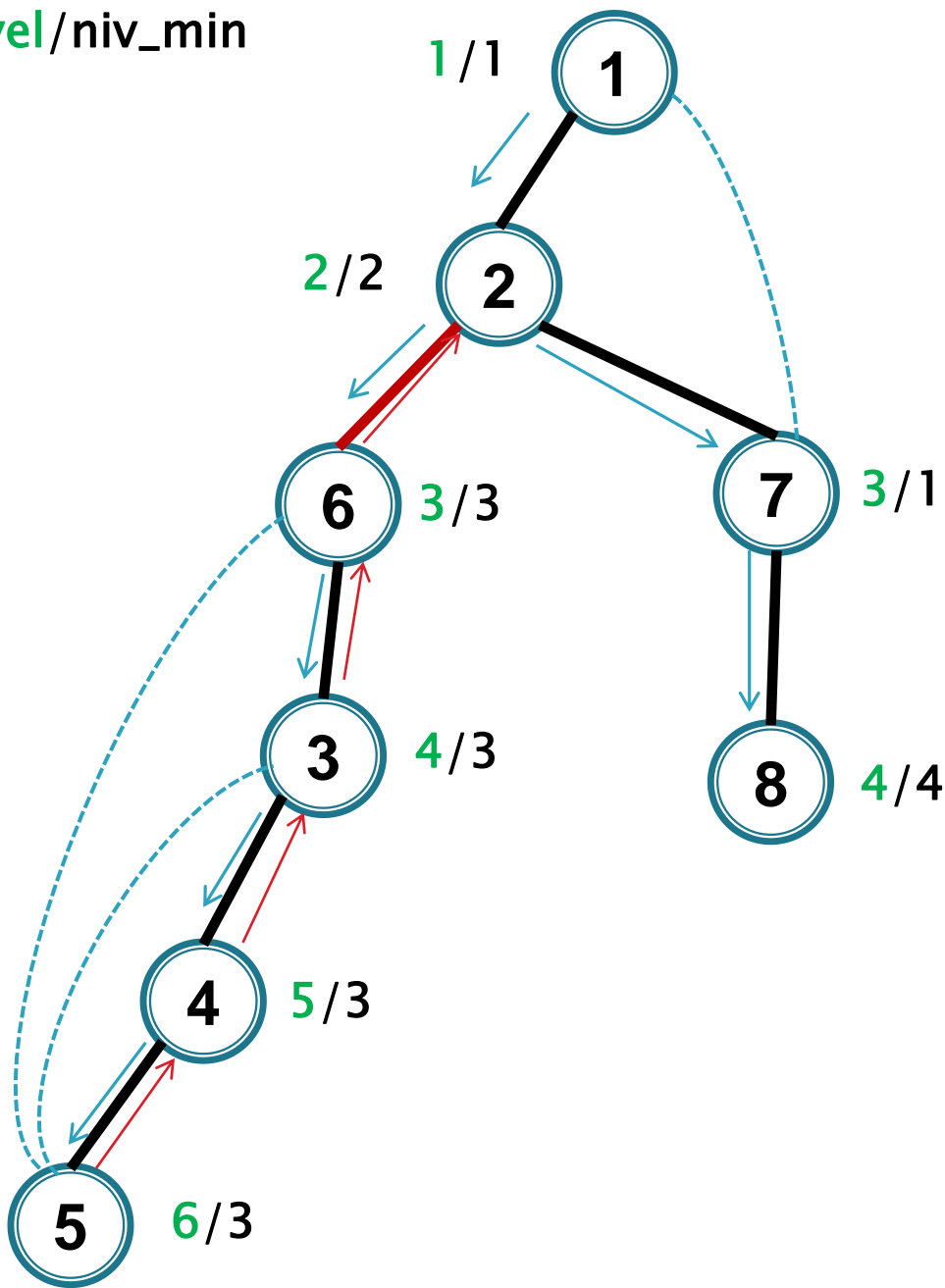
nivel/niv_min

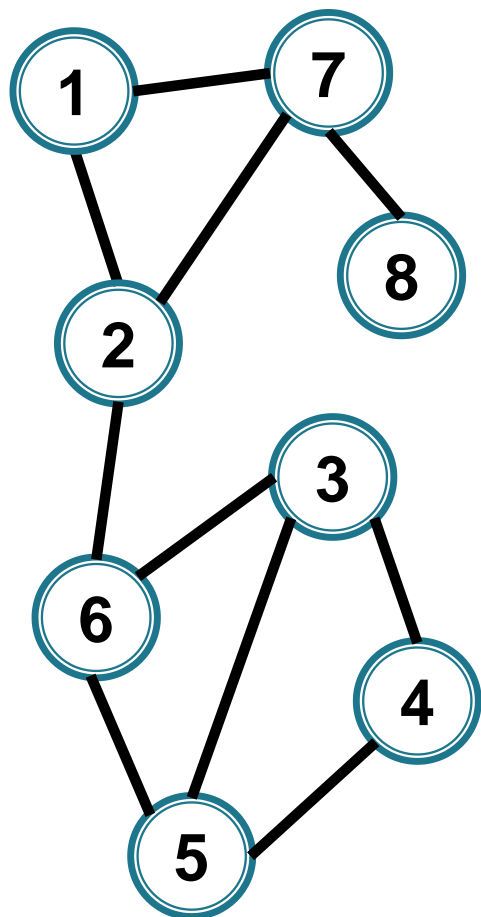




S:
 1 2
 2 7
 1 7
 7 8

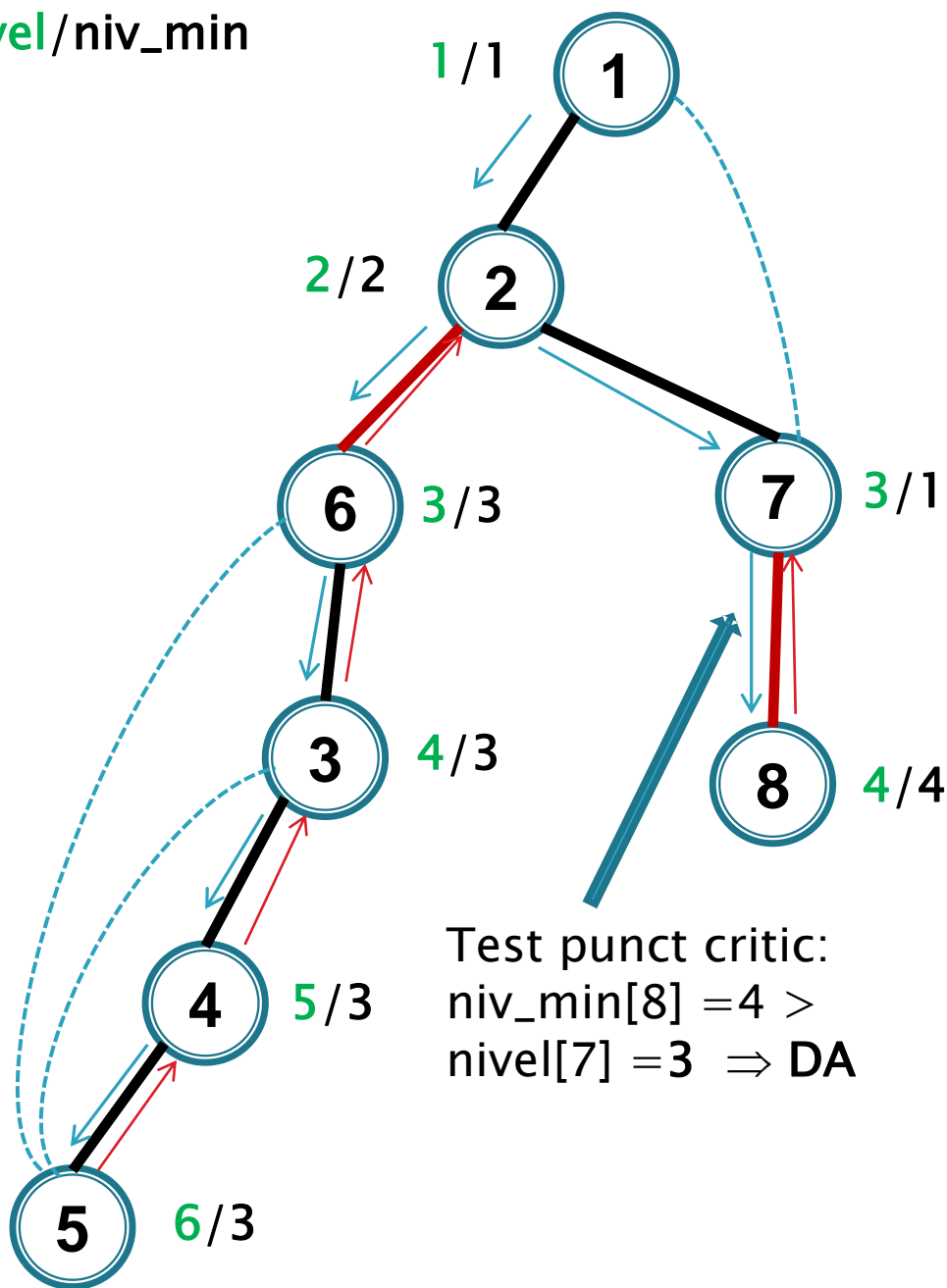
nivel/niv_min



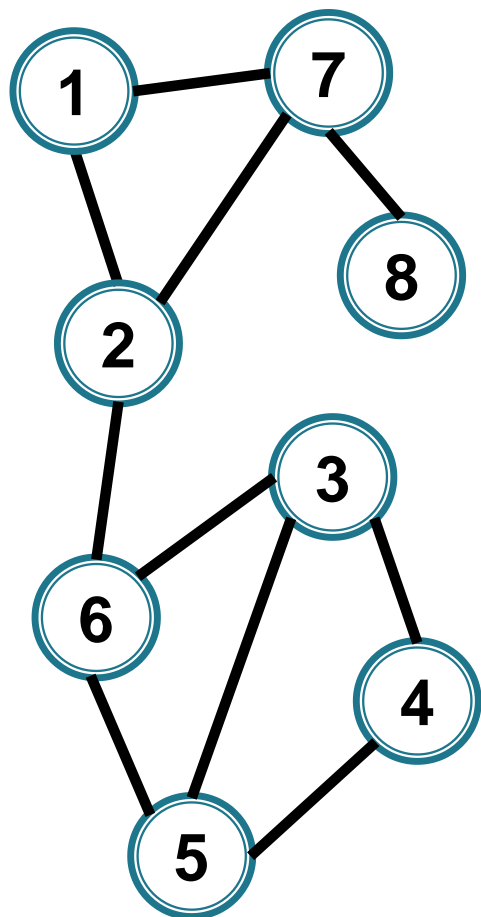


S:
 1 2
 2 7
 1 7
 7 8

nivel/niv_min

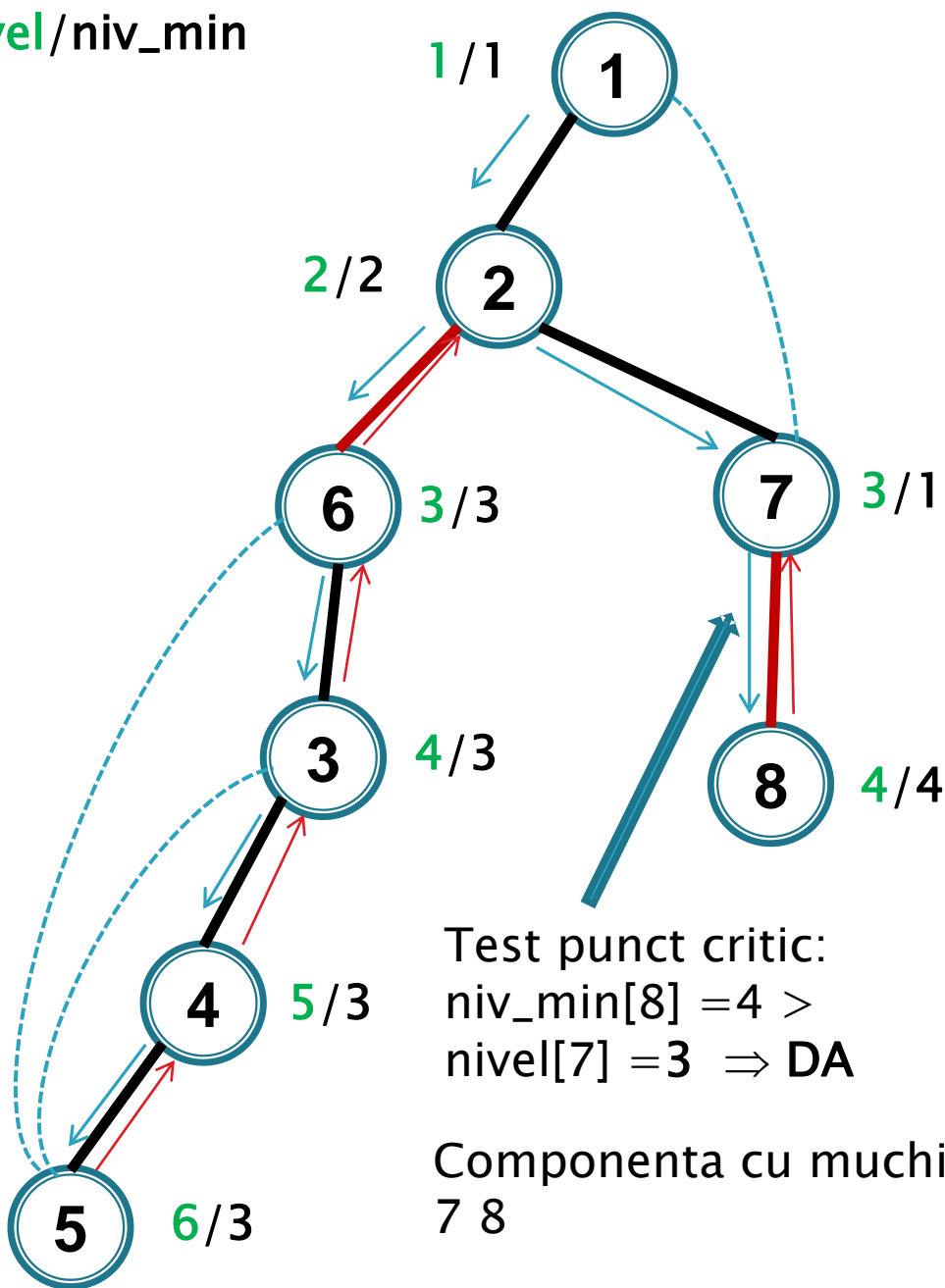


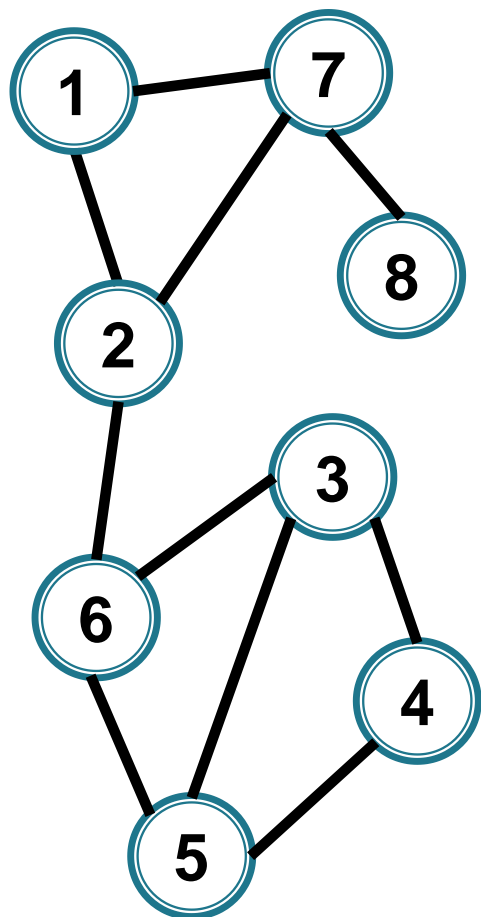
Test punct critic:
 $niv_min[8] = 4 >$
 $nivel[7] = 3 \Rightarrow \text{DA}$



S:
1 2
2 7
1 7

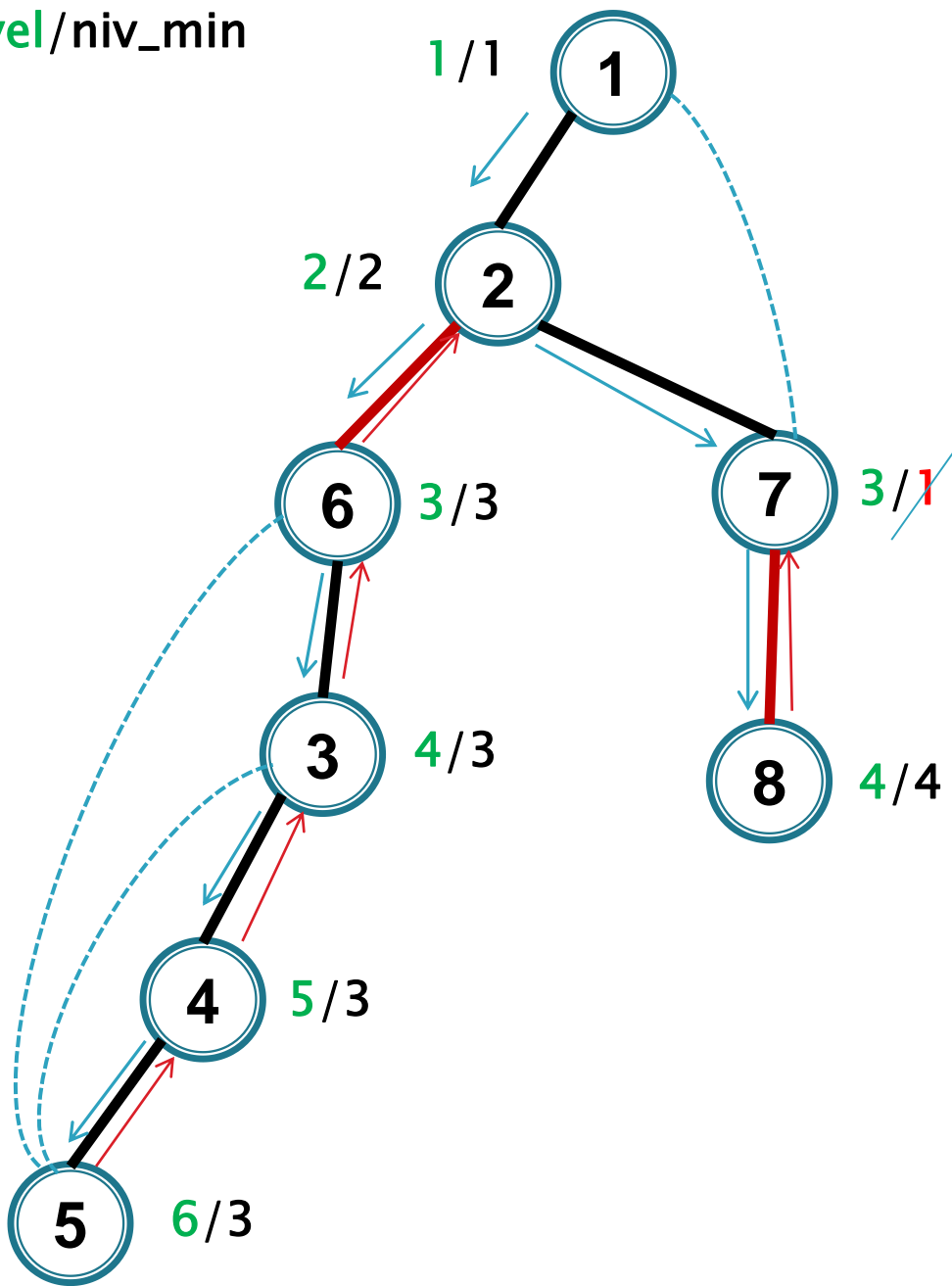
nivel/niv_min

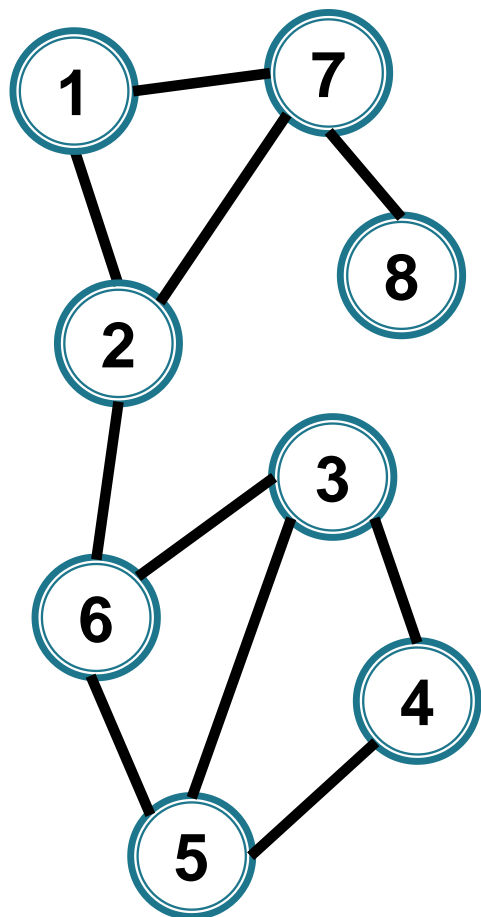




S:
1 2
2 7
1 7

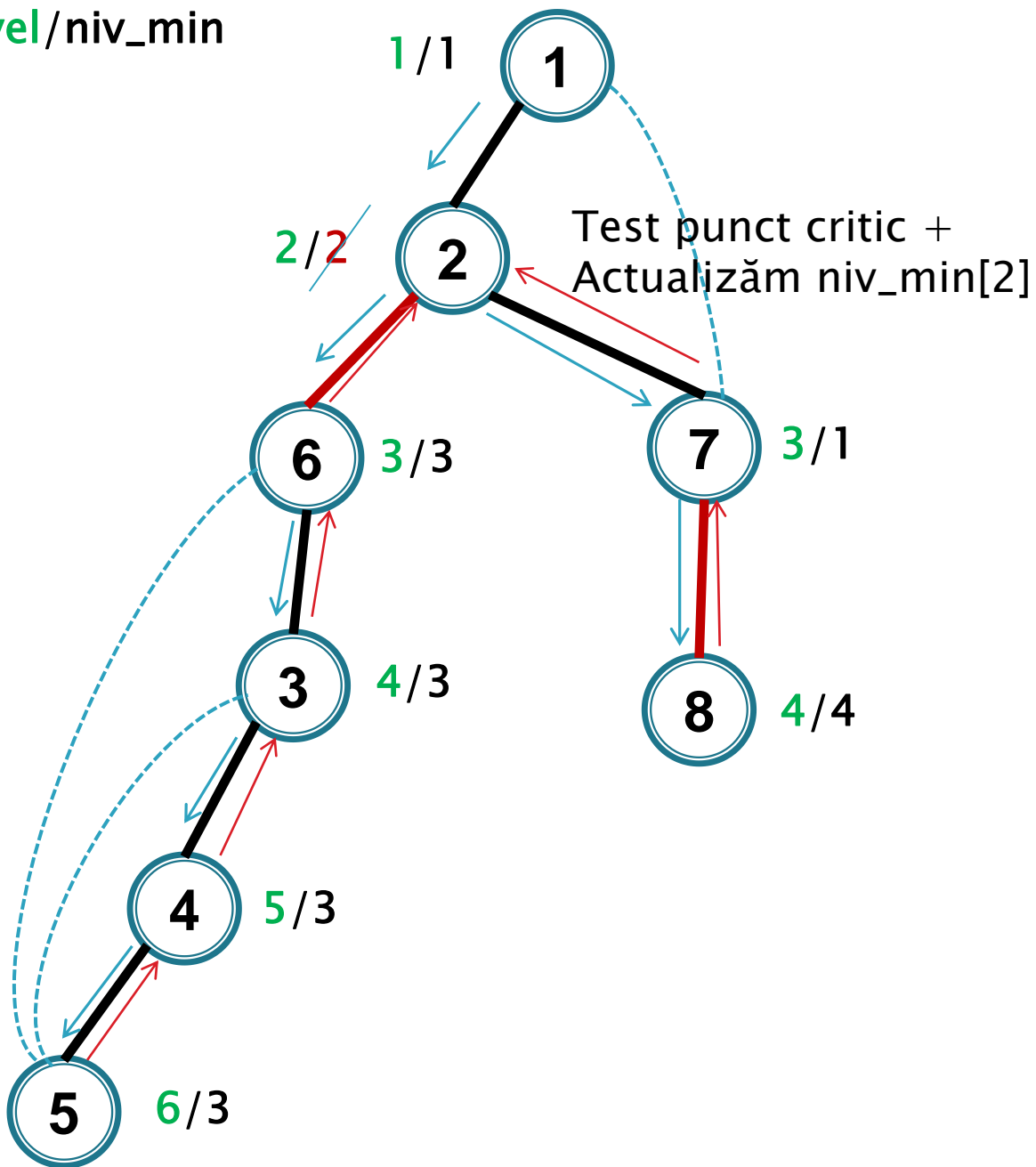
nivel/niv_min

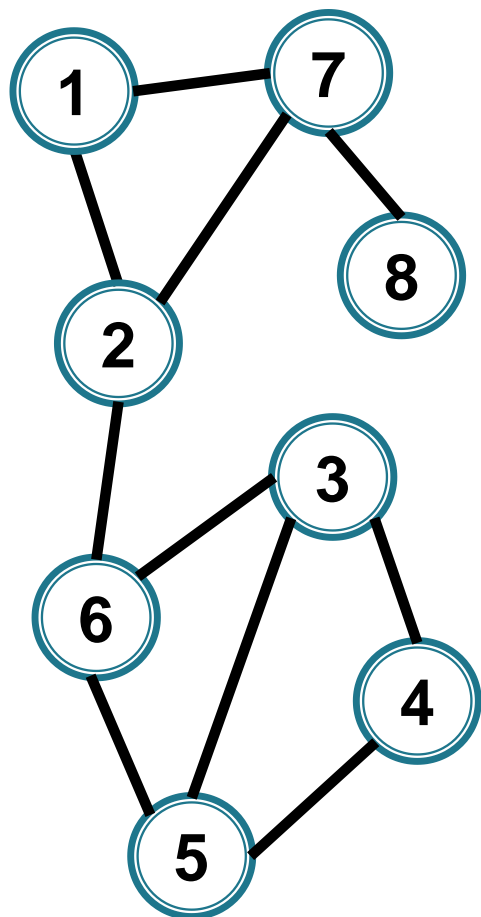




S:
1 2
2 7
1 7

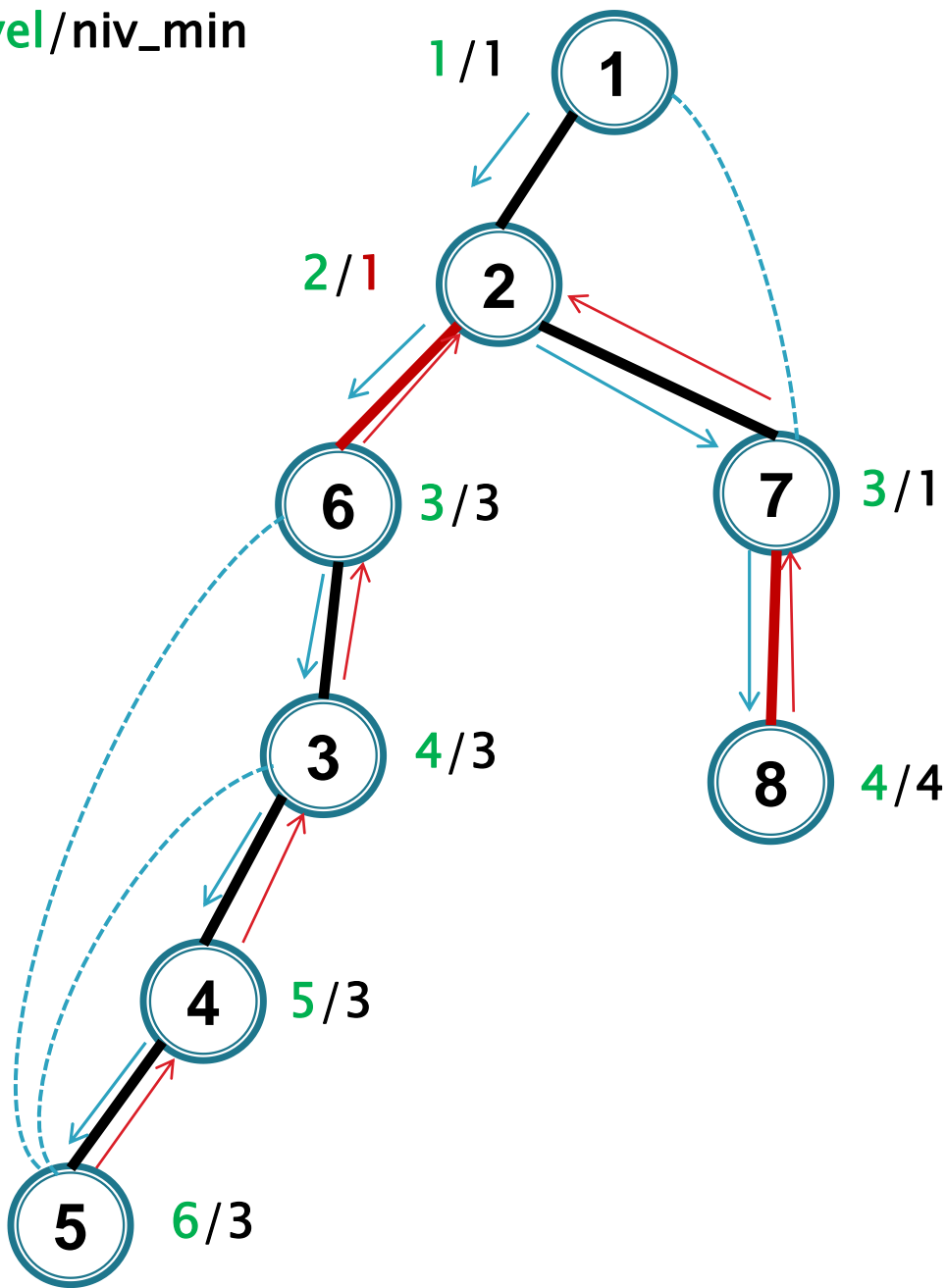
nivel/niv_min

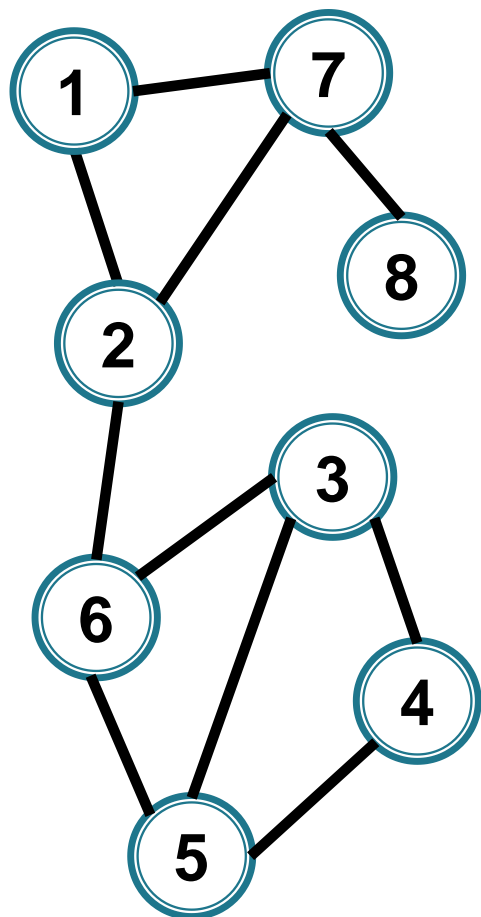




S:
1 2
2 7
1 7

nivel/niv_min

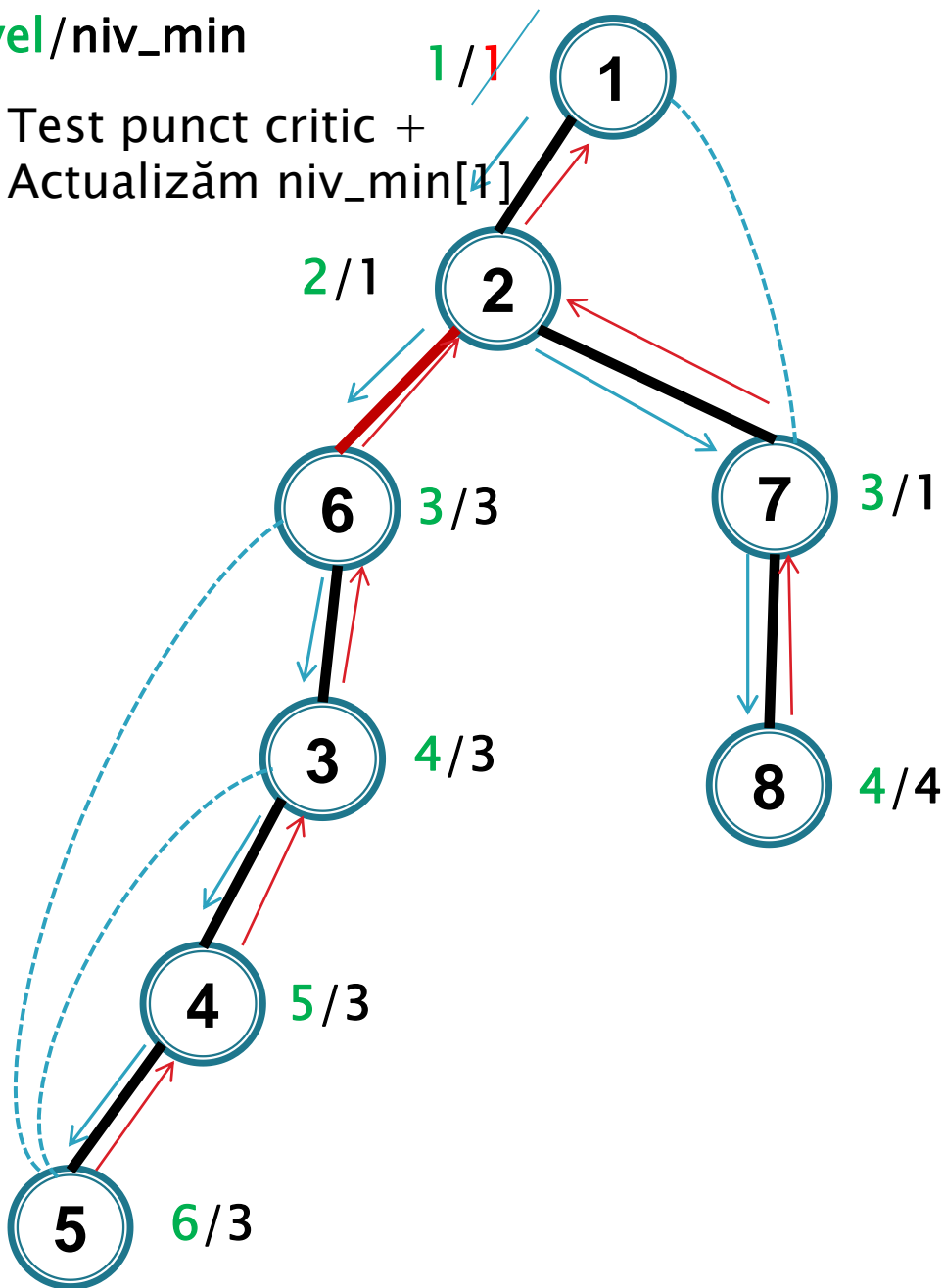


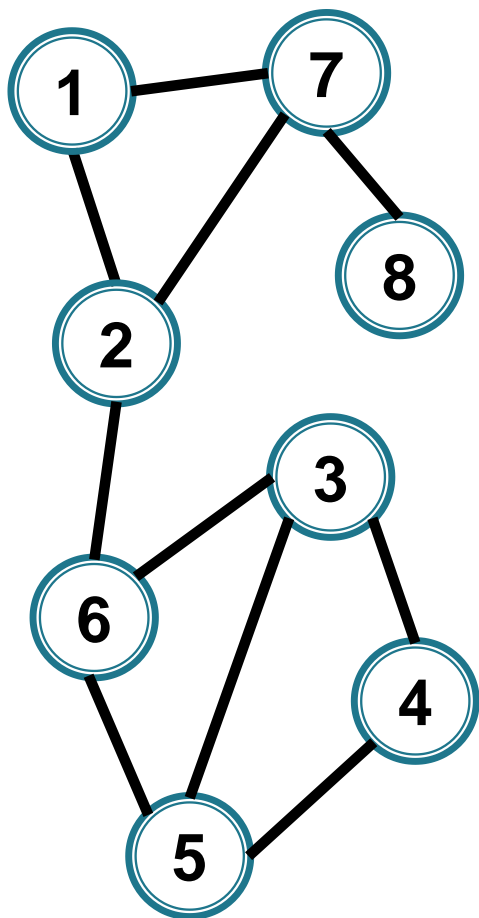


S:
1 2
2 7
1 7

nivel/niv_min

Test punct critic +
Actualizăm niv_min[4]

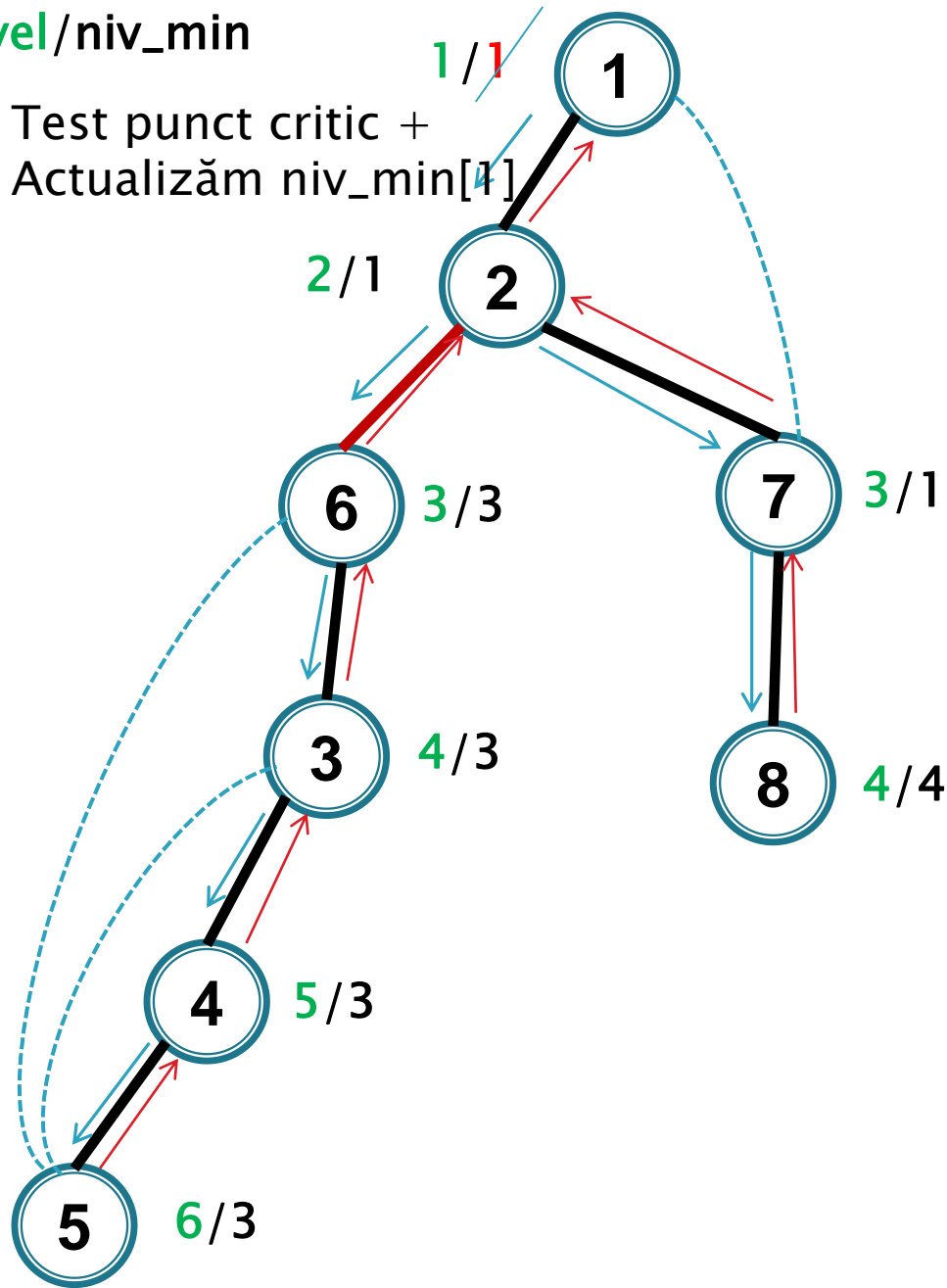




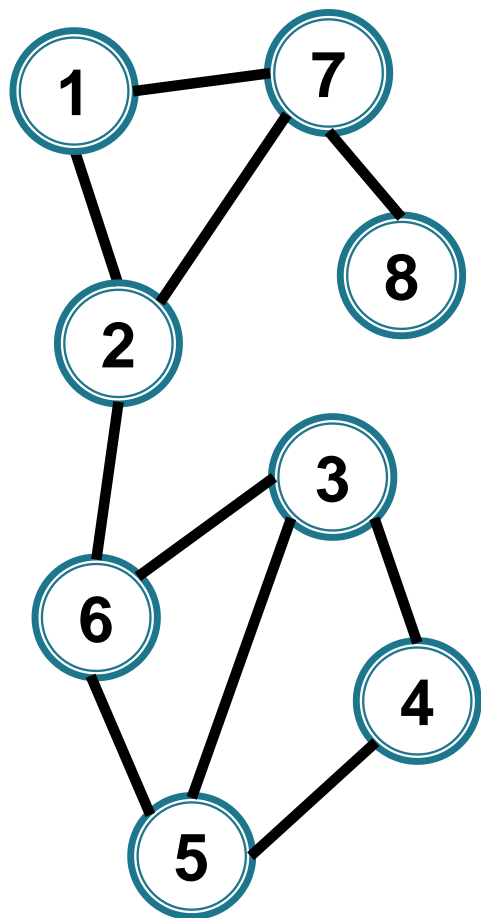
S:
1 2
2 7
1 7

nivel/niv_min

Test punct critic +
Actualizăm niv_min[4]



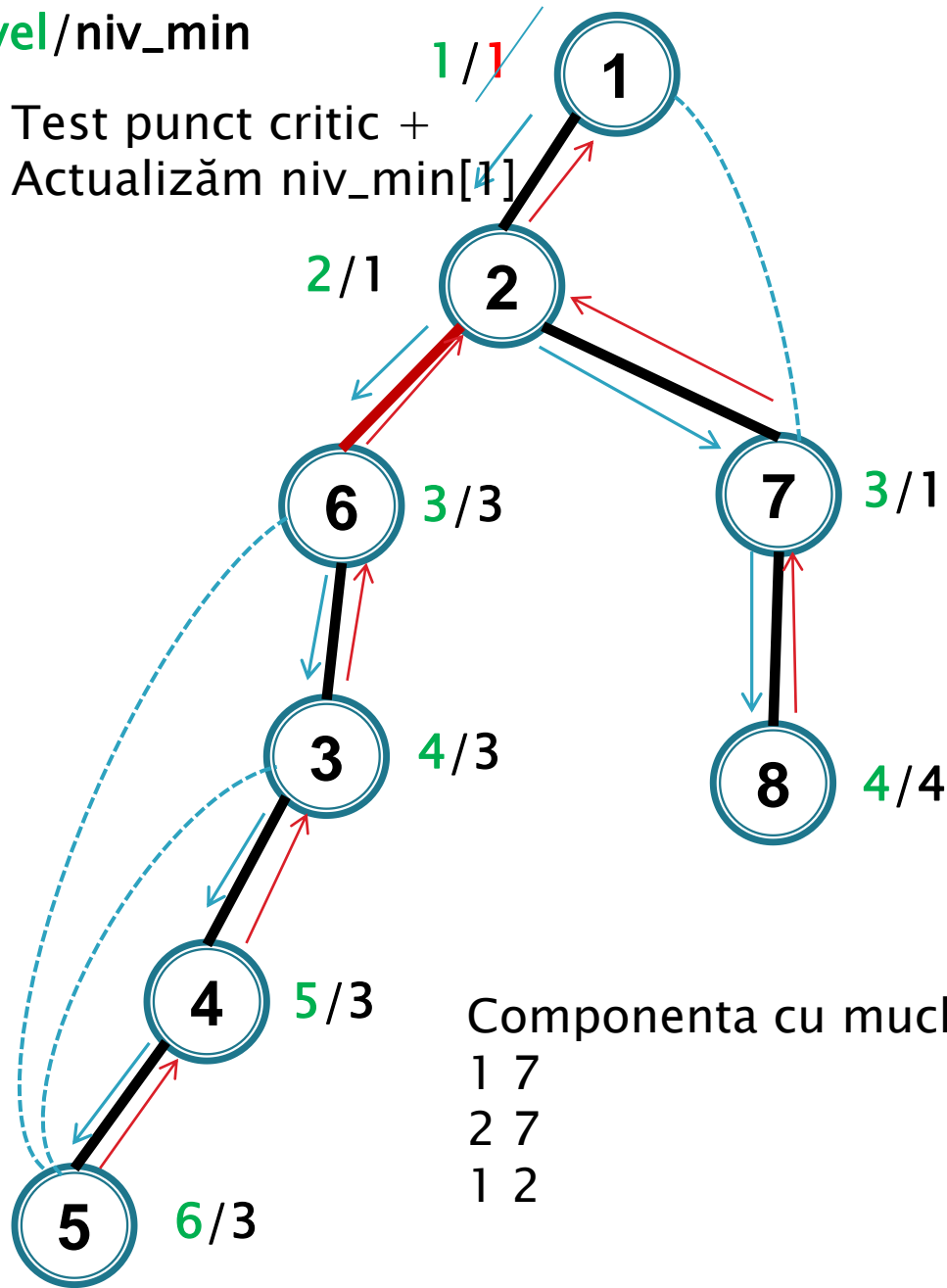
Rădăcina caz particular de
test de punct critic – aici nu
mai este necesar, se “rupe”
o componenta biconexă
chiar dacă are un singur fiu



S:
1 2
2 7
1 7

nivel/niv_min

Test punct critic +
Actualizăm niv_min[4]



Rădăcina caz particular de
test de punct critic – aici nu
mai este necesar, se “rupe”
o componenta biconexă
chiar dacă are un singur fiu

Componenta cu muchiile
1 7
2 7
1 2

Exerciții

Exerciții

1. Se dă un graf neorientat conex $G = (V, E)$. Se consideră parcurgerile DFS și BFS care pornesc din nodul 1, iar vecinii unui nod sunt considerați în ordine crescătoare.

Ce proprietăți trebuie să respecte G pentru ca cele două parcurgeri să obțină același arbore parțial?

Exerciții

2. Fie G_1 și G_2 două grafuri cu proprietatea că ordinea în care sunt parcurse vârfurile în BF este aceeași pentru ambele grafuri, la fel și în DF (vecinii unui vârf sunt parcurși în ordine crescătoare).

Sunt G_1 și G_2 egale?

Exerciții

3. Fie G_1 și G_2 două grafuri cu proprietatea că arborii BF și DF ai celor două grafuri sunt egali (vecinii unui vârf sunt parcurși în ordine crescătoare).

Sunt G_1 și G_2 egale?

Exerciții

4. Se dă un graf neorientat conex $G = (V, E)$ și un vârf s . Care este numărul minim de muchii ale unui graf parțial al lui G care conservă distanțele de la s la celelalte vârfuri

Exerciții

5. Fie T un graf neorientat cu n vârfuri.

Arătați că T este arbore $\Leftrightarrow T$ este conex și are $n-1$ muchii

Exerciții

6. Arătați că în orice grup de n persoane între care există relații de prietenie reciprocă există două persoane cu același număr de prieteni