PARCOURS EN PROFONDEUR, CYCLES, TRI TOPOLOGIQUE ET COMPOSANTES FORTEMENT CONNEXES

Soit $G = (S,A)$ un graphe orienté. On note $n = \#S$ et $m = \#A$.

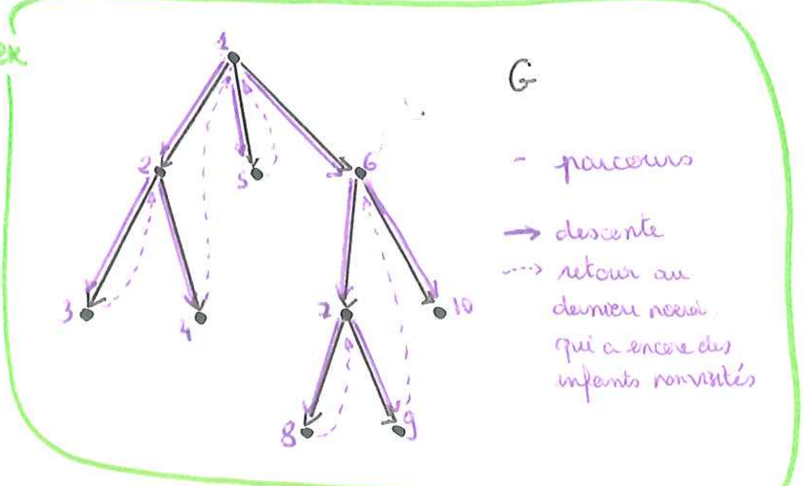
Intro

On parle de parcours car la procédure doit visiter une et une seule fois chaque sommet du graphe.

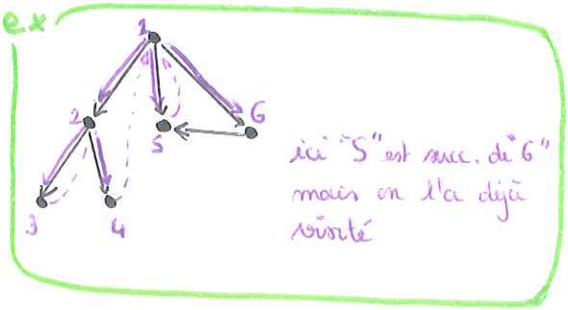
La notion de profondeur est plus visible dans le cas où G est un arbre, où l'on identifie les successeurs d'un sommet à des enfants du noeud.
L'idée est alors que tant que c'est possible on descend dans l'arbre, c-à-d que tant que le noeud a des enfants, on visite l'un de ces enfants, et lorsqu'il n'en n'a plus on revient au noeud le plus profond (qu'on a déjà visité et) dont on n'a pas encore visité tous les enfants, c-à-d qu'on peut descendre par une autre branche.

Pour un graphe, qui a priori peut avoir des cycles, on visite les noeuds de successeurs en successeurs tant que les sommets ont des successeurs et qu'on n'a pas encore visité ceux-ci, précaution qu'on n'avait pas besoin de prendre en l'absence de cycle.
On voit là apparaître la détection de cycle.

Mais l'exploration en profondeur permet aussi de fournir une décomposition du graphe en composantes fortem^t connexes, et dans le cas acyclique un tri topologique des sommets (cf déf ci-après).



⚠ Même dans un graphe acyclique un sommet peut avoir pour successeur un sommet déjà visité



Déf L est un <u>parcours en profondeur</u> de G

$\Leftrightarrow$
- chaque sommet de G apparaît une et une seule fois dans L
- $\forall i \in [1..n]$ parmi les sommets apparaissant dans $L[i+1..n]$ ceux accessibles depuis le sommet $L[i]$ sont tous avant ceux non accessibles.

Déf L est un <u>tri topologique</u> de G

$\Leftrightarrow$
- chaque sommet de G apparaît une et une seule fois dans L
- $\forall (i,j) \in [1..n]^2,\ (L[i], L[j]) \in A \Rightarrow i < j$

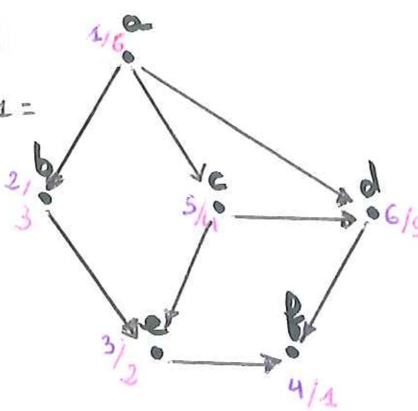
Déf Les <u>composantes fortement connexes</u> (CFC) de G sont les classes d'équivalence pour la relation $\sim$ définie sur S par

$s \sim t \Leftrightarrow s = t$ ou $\begin{cases} \text{il existe un chemin de } s \text{ à } t \text{ dans } G \\ \text{et } t \text{ à } s \end{cases}$

Pté Les CFC forment une partition de S.

ex

$G_1 =$



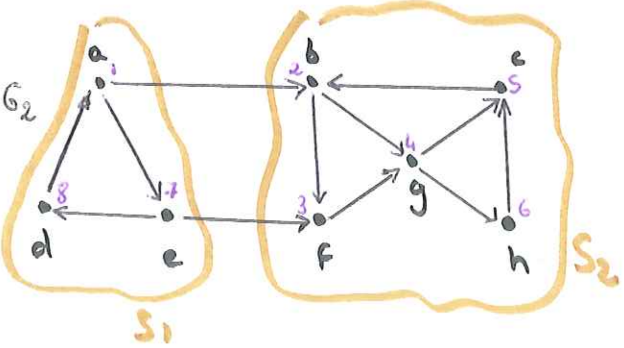
- a b e f c d est un parcours en profondeur de G_1
- a d c b e f est un tri topologique des sommets de G_1.

arcs de A, tous vont vers "l'avant".

- G_1 est acyclique

ex

G_2



- G_2 admet des cycles (ex aed ; bgc ...)
- S_1 et S_2 sont les 2 CFC de G_2
- abfgched est un parcours en profondeur de G_2

algorithme

PARCOURS_PROF (G)

```
n ← G. nombre_de_sommets

O ← tableau indexé par [1..n], initialisé à 0
F ← ____________________
V ← ____________________

cpt_O ← 1 ;
cpt_F ← 1 ;
num ← 1 ;

cycle ← FAUX ;

Pour r de 1 à n
    Si V[r] ≠ 0
    alors AUX_1 (G, r, num, cpt0, cpt F, V, O, F, cycle)
          num ++ ;

[ T ← tableau indexé par [1..n]
  Pour j de 1 à n
      T[O[j]] ← j
  retourner T ]

R ← tableau indexé par [1..n]
Pour j de 1 à n
    R[n+1-F[j]] ← j ;
    V[j] ← j ;
    O[j] ← j ;
    F[j] ← j ;

cpt0 ← 0 ;
cpt_F ← 0 ;
num ← 0 ;

Pour k de 1 à n
    Si V[R[k]] ≠ 0
    alors AUX_2 (G, R[k], num, cpt0, cpt_F, V, O, F, cycle)
          num ++ ;

[ retourner cycle ; ]

[ Si cycle
  alors erreur "graphe avec cycle ;
                pas de tri topo possible"
  sinon retourner R ; ]

[ retourner V ; ]
```

Entrée G un graphe orienté
codé si possible par liste de successeurs
et liste de prédécesseurs.

Sorties

- un parcours en profondeur des sommets de G
- VRAI s'il y a des circuits dans G
  FAUX sinon
- un tri topologique des sommets de G s'il en existe un, ie si G acyclique
- la donnée des CFC de G sous la forme d'un tableau de n° de CFC pour chaque sommet

```
AUX_2
AUX_1 (G, r, num, cpt0, cptF, O, F, V, cycle)

P ← PILE_VIDE() ;
P. AJOUTER (r) ;
V[r] ← num ;

Tant que P. EST_NON_VIDE().
    s ← P. SOMMET_DE_PILE()
    Si O[s] = 0
    alors O[s] ← cpt_O ;
          cpt_O ++ ;
          pour t ∈ G. succ (s)
                   G. pred (s)
              si V[t] = 0
              alors V[t] ← num
                    P. AJOUTER (t)
              sinon si O[t] ≠ 0 et F[t] = 0
                    alors cycle ← VRAI
    sinon F[s] ← cpt_F ;
          cpt_F ++ ;
          P. DÉPILER() ;
```

ex

cpt_O ← 1 ; cpt_F ← 1 , nabr ← 1 , cycle ← FAUX

r ← 1 = a

V[a] = 0

AUX_1 (G, a , 1,1,1, 0, F, V)

[ ] pile vide.

[a] , V[a] ← 1

s ← a , O[a] = 0 , O[A] ← 1 cpt_O ← 2 , [a b c d] car "a" a pour successeurs "b, c, d"

s ← d , O[d] = 0 , O[d] ← 2 cpt_O ← 3 , [a b c d f] car "d" a pour seul successeur "f"

s ← f , O[f] = 0 , O[f] ← 3 cpt_O ← 4 , [a b c d f] car "f" n'a pas de successeurs

s ← f , O[f] ≠ 0 , F[f] ← 1 cpt_F ← 2 , [a b c d]

s ← d , O[d] ≠ 0 , F[d] ← 2 cpt_F ← 3 , [a b c]

s ← c , O[c] = 0 , O[c] ← 4 cpt_O ← 5 , [a b c e] car "c" a pour successeur "d" déjà visité et "e"

s ← e , O[e] = 0 O[e] ← 5 cpt_O ← 6 , [a b c e] car "e" n'a que des successeurs déjà visités

s ← e , O[e] ≠ 0 , F[e] ← 3 cpt_F ← 4 , [a b c]

s ← c , O[c] ≠ 0 , F[c] ← 4 cpt_F ← 5 , [a b]

s ← b , O[b] = 0 , O[b] ← 6 cpt_O ← 7 , [a b] car b n'a qu'un successeur déjà visité

s ← b , O[b] ≠ 0 , F[b] ← 5 cpt_F ← 6 [a]

s ← a , O[a] ≠ 0 , F[a] ← 6 cpt_F ← 7 , [ ]

nabr ← 2

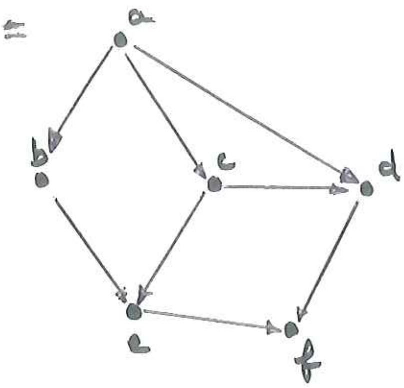
r ← 2 = "b"
V[b] ≠ 0

r ← 3 = "c"
V[c] ≠ 0

....

r ← 6 = "f"
V[f] ≠ 0.

G1 =



	a = 1	b = 2	c = 3	d = 4	e = 5	f = 6
O	~~0~~ 1	~~0~~ 6	~~0~~ 4	~~0~~ 2	~~0~~ 5	~~0~~ 3
F	~~0~~ 6	~~0~~ 5	~~0~~ 4	~~0~~ 2	~~0~~ 3	~~0~~ 1
V	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1

[cycle est resté à FAUX → pas de cycle

[T[O[a]] ← 1 ; T[O[b]] ← 2 ...
  (O[a] = 1, O[b] = 6)

T = | 1 | 4 | 6 | 3 | 5 | 2 |
L = [a, d, f, c, e, b]

[Pas de cycle T[F[a]] ← 1 ; T[F[b]] ← 2 ; ...

T = | 6 | 4 | 5 | 3 | 2 | 1 |
L = a, b, c, e, d, f.

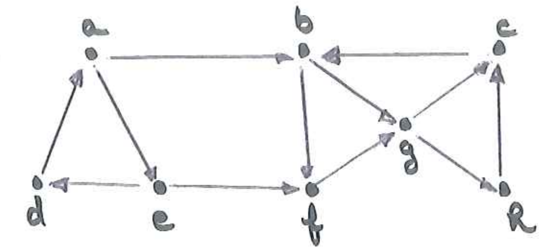
[Pas de cycle donc les CFC sont

a	b	c	d	e	f
1	2	3	4	5	6

esc On ne détaille plus ici l'évolution de cpt_O et cpt_F.

n abr ← 1
cycle ← FAUX

G2 :



	1 = a	2 = b	3 = c	4 = d	5 = e	6 = f	7 = g	8 = R
O	~~0~~ 1	~~0~~ 8	~~0~~ 6	~~0~~ 7	~~0~~ 2	~~0~~ 3	~~0~~ 4	~~0~~ 5
F	~~0~~ 8	~~0~~ 7	~~0~~ 2	~~0~~ 5	~~0~~ 6	~~0~~ 4	~~0~~ 3	~~0~~ 1
V	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1	~~0~~ 1

x ← 1 = "a"

V[a] = 0

AUX_1 (G, a, 1, cpt_O, cpt_F, V, O, F, cycle)

⊔ pile vide

[a], V[a] ← 1

s ← a, O[a] = 0, O[a] ← 1, [e, b, a]

s ← e, O[e] = 0, O[e] ← 2, [f, d, e, b, a]

s ← f, O[f] = 0, O[f] ← 3 [g, f, d, e, b, a]

s ← g, O[g] = 0, O[g] ← 4, [R, c, g, f, d, e, b, a]

s ← R, O[R] = 0, O[R] ← 5 idem

s ← R, O[R] ≠ 0, F[R] ← 1 [c, g, f, d, e, b, a]

s ← c, O[c] = 0, O[c] ← 6, idem,

s ← c, O[c] ≠ 0, F[c] ← 2 [g, f, d, e, b, a]

s ← g, O[g] ≠ 0, F[g] ← 3 [f, d, e, b, a]

s ← f, O[f] ≠ 0, F[f] ← 4 [d, e, b, a]

s ← d, O[d] = 0, O[d] ← 7 [d, e, b, a]

s ← d, O[d] ≠ 0, F[d] ← 5 [e, b, a]

s ← e, O[e] ≠ 0, F[e] ← 6, [b, a]

s ← b, O[b] = 0, O[b] ← 8, [b, a]

s ← b, O[b] ≠ 0, F[b] ← 7, [a]

s ← a, O[a] ≠ 0, F[a] ← 8 ⊔.

nabr ← 2

x ← 2 = "b"
V[b] ≠ 0

...

x ← 8 = "R"
V[R] ≠ 0.

T[O[a]] ← 1, T[O[b]] ← 2 ...
$\;\;\;\;\;\;\underbrace{}_{=1}\;\;\;\;\;\;\;\;\underbrace{}_{=8}$

T = [1 | 5 | 6 | 7 | 8 | 3 | 4 | 2]
L = [a, e, f, g, R, c, d, b]

$R[\underbrace{n+1-F[a]}_{1}] \leftarrow a$, V[a] ← 0, O[a] ← 0, F[a] ← 0;

$R[\underbrace{n+1-F[b]}_{2}] \leftarrow b$, V[b] ← 0, O[b] ← 0, F[b] ← 0;

$R[\underbrace{n+1-F[c]}_{7}] \leftarrow c$, V[c] ← 0, O[c] ← 0, F[c] ← 0;

...

$R[\underbrace{n+1-F[R]}_{8}] \leftarrow R$, V[R] ← 0, O[c] ← 0, F[c] ← 0;

cpt O ← 1; cpt F ← 1; ← 1;

	1	2	3	4	5	6	7	8
R	a	b	e	d	f	g	c	h
	a=1	b=2	c=3	d=4	e=5	f=6	g=7	h=8
V	$\cancel{0}$ 1	$\cancel{0}$ 2	$\cancel{0}$ 2	$\cancel{0}$ 1	$\cancel{0}$ 1	$\cancel{0}$ 2	$\cancel{0}$ 2	$\cancel{0}$ 2
O	$\cancel{0}$ 1	$\cancel{0}$ 4	$\cancel{0}$ 5	$\cancel{0}$ 2	$\cancel{0}$ 3	$\cancel{0}$ 7	$\cancel{0}$ 6	$\cancel{0}$ 8
F	$\cancel{0}$ 3	$\cancel{0}$ 8	$\cancel{0}$ 7	$\cancel{0}$ 2	$\cancel{0}$ 1	$\cancel{0}$ 4	$\cancel{0}$ 5	$\cancel{0}$ 6

⚠ dans R la case i se trouve le n° du sommet qu'on doit considérer en i-ème pour choisir une racine, mais pas quelquechose qui concerne le sommet de n° i.

A l'inverse V[i], resp O[i], F[i], indiquent quand le sommet n° i a été visité (ie dans quel arbre), resp quand il a été ouvert et fermé (ou plutôt en combientième).

k = 1 (R[k] = a)

V[a] = 0

AUX_2(G, a, 1, cptO, cptF, V, O, F, cycle)

[ ] pile vide.

[a] ; V[a] ← 1.

s ← a, O[a] = 0, O[a] ← 1, [d, a]

s ← d, O[d] = 0, O[d] ← 2 ; [e, d, a]

s ← e, O[e] = 0, O[e] ← 3, [e, d, a], cycle ← VRAI

en effet e a pour prédecesseur a qui est ouvert (O[a] ≠ 0) mais pas encore fermé (F[a] = 0)

s ← e, O[e] ≠ 0, F[e] ← 1 [d, a]

s ← d, O[d] ≠ 0, F[d] ← 2 [a]

s ← a, O[a] ≠ 0, F[a] ← 3 [ ]

nclm ← 2 ;

k = 2 (R[k] = b)

V[b] = 0

AUX_2(G, b, 2, cptO, cptF, V, O, F, cycle)

[ ] pile vide

[b], V[b] ← 2.

s ← b, O[b] = 0, O[b] ← 4, [c, b]

s ← c, O[c] = 0, O[c] ← 5, [g, h, c, b]

s ← g, O[g] = 0, O[g] ← 6 [f, g, h, c, b], cycle ← VRAI

car b préd de g déjà ouvert et non encore fermé

s ← f, O[f] = 0, O[f] ← 7, idem, cycle ← VRAI

car b préd de f.

s ← f, O[f] ≠ 0, F[f] ← 4, [g, h, c, b]

s ← g, O[g] ≠ 0, F[g] ← 5, [h, c, b]

⋮

s ← h, O[h] = 0, O[h] ← 8, [h, c, b]

s ← h, O[h] ≠ 0, F[h] ← 6 [c, b]

s ← c, O[c] ≠ 0, F[c] ← 7 [b]

s ← b, O[b] ≠ 0, F[b] ← 8, [ ]

nclm ← 3

k = 3 (R[k] = e)

V[e] ≠ 0

k = 4

...

k = 8 (R[k] = h)

V[h] ≠ 0

[renvoyer VRAI]

[erreur " pas de tri topo possible"]

[CFC n° 1 → a, d, e (car V[·] = 1)
CFC n° 2 → b, c, f, g, h (car V[·] = 2)]