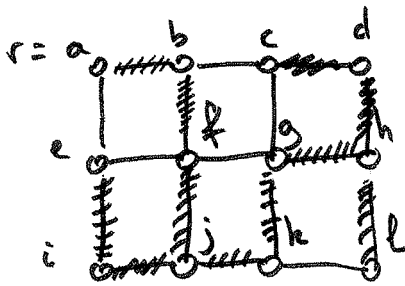


$L_{15}$   $\left[ \begin{array}{l} \text{Si } dv(y) = 0 \text{ alors} \\ \text{Empiler } y \text{ sur AT;} \\ dv(y) \leftarrow 1; \\ \text{d\u00e9but}(y) \leftarrow t; \\ t \leftarrow t + 1; \\ \text{p\u00e8re}(y) \leftarrow x; \end{array} \right.$   
 $L_{16}$   
 $L_{17}$   
 $L_{18}$   
 $L_{19}$   
 $L_{20}$

Exemple :



AT: ~~a~~ ~~b~~ ~~f~~ ~~j~~ ~~k~~ ~~g~~ ~~h~~ ~~d~~ ~~e~~ ~~i~~ ~~j~~ ~~l~~

	a	b	c	d	e	f	g	h	i	j	k	l
père	a	a	d	h	i	b	h	g	j	f	j	h
début	1	2	11	10	18	3	6	7	17	4	5	8
fin	24	23	12	13	19	22	15	14	20	21	16	9

$Vois(a) = \{ \underline{b}, \underline{e} \}$

$Vois(b) = \{ \underline{f}, \underline{c}, \underline{a} \}$

$Vois(c) = \{ \underline{b}, \underline{g}, \underline{d} \}$

$Vois(d) = \{ \underline{c}, \underline{h} \}$

$Vois(h) = \{ \underline{j}, \underline{g}, \underline{l} \}$

$Vois(e) = \{ \underline{a}, \underline{f}, \underline{i} \}$

$Vois(f) = \{ \underline{j}, \underline{g}, \underline{b}, \underline{e} \}$

$Vois(l) = \{ \underline{h}, \underline{h} \}$

$Vois(g) = \{ \underline{h}, \underline{f}, \underline{c} \}$

$Vois(h) = \{ \underline{g}, \underline{l}, \underline{d} \}$

(Ah, mince les piles sont à l'envers...)

On obtient l'ordre :

$a \ b \ f \ j \ k \ g \ h \ l \ l \ d \ c \ c \ d \ h \ g \ h \ i \ e \ e \ i \ j \ f \ b \ a$   
 (On explore les descendants de h (a, c est fini))



Chaque intervalle correspond au temps de présence dans la pile.

Analyse :

Terminaison / complexité : Si le temps d'empil<sup>+</sup>/dépil<sup>+</sup> est  $O(1) \rightarrow O(n)$

Correction:

On se sert des intervalles de présence dans la pile pour voir qu'on a un arbre normal:

fait 1: On n'a pas  $\boxed{x \mid y \mid x \mid y}$ .

Pr: Ça vient du fonctionnement de la pile AT: si on empile  $x$  avant  $y$ , il faudra dépiler  $y$  avant  $x$ .  $\square$ .

Ainsi les seuls cas de figure sont:  $\boxed{\boxed{\phantom{x}}}$  et  $\boxed{\phantom{x}}\boxed{\phantom{x}}$ .  
(on a un système de parenthésage). Pour  $x$  fixé, son intervalle est:  
 $\boxed{x \mid f_1 \dots f_i \mid f_{i+1} \dots f_j \mid \dots \mid f_k \dots f_{k-1} \mid x}$  avec  $k \geq 0$ .

fait 2: les  $f_i$  sont exactement les fils de  $x$  dans l'arbre.

Pr:  $\Rightarrow f_1$  est empilé juste après  $x$ , donc  $\text{père}(f_1) = x$ .

A la fin de l'intervalle de  $f_1$ ,  $f_1$  est dépilé de AT et  $x$  se trouve en haut de AT.

$f_2$  est alors empilé, donc  $\text{père}(f_2) = x \dots$   
de même  $\forall i$ .

$\Leftarrow$  Inversement si  $y$  a pour père  $x$ , il a été traité lorsque  $x$  était en haut de AT, donc  $y$  est l'un des  $f_i$ .  $\square$


On a ainsi la propriété suivante:

$y$  descendant de  $x \iff \boxed{x \mid y \mid y \mid x}$

$(\text{debut}(x) < \text{debut}(y) < \text{fin}(y) < \text{fin}(x))$

fait 3: On a un arbre normal.

Pr. Soit  $xy \in E(G)$ , avec  $\text{débüt}(x) < \text{débüt}(y)$

On ne peut pas avoir  $[x \dots x] \dots [y \dots y]$  puisqu'au moment où on dépile  $x$ , tous ses voisins ont été visités, donc on a:  $[x \dots [y \dots y] \dots x]$  et  $y$  est un descendant de  $x$  

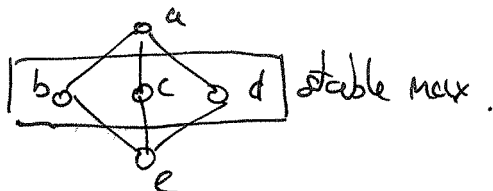
## - Problème du jour n° 5 -

STABLE-MAX :

Entrée :  $G = (V, E)$

Sortie : ens  $X$  de sommet de  $G$  formant un stable de taille maximale

Exemple :



-  $\{a, e\}$  est stable maximal par inclusion  
 -  $\{b, c, d\}$  stable maximum (= de cardinalité maximale).

le problème est NP-difficile et même:

1000-APPROX-STABLE-MAX :

Entrée :  $G = (V, E)$

Sortie : ens  $X$  de sommet de  $G$  formant un stable et de taille  $\geq \frac{1}{1000}$  de celle d'un stable maximum.

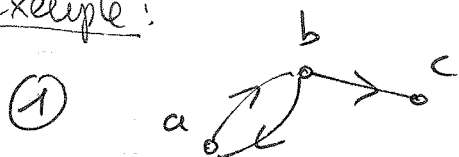
C'est NP-complet et on peut remplacer 1000 par n'importe quoi, ça reste NP-complet: STABLE-MAX est Non-Approximable.

## Chap IV: Graphes orientés.

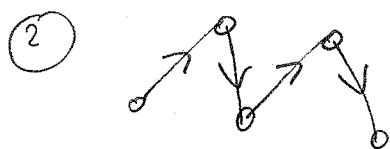
### I. Définitions de base.

• Un graphe orienté  $D = (V, A)$  est formé de sommets :  $V$  et d'arcs :  $A$ , qui sont des couples d'éléments distincts de  $V$  (et plus des paires). Si  $xy$  est un arc de  $D$ ,  $yx$  peut aussi être un arc de  $D$  mais pas forcément :

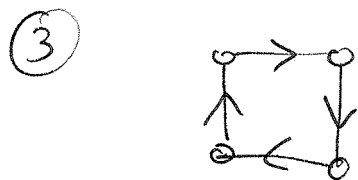
• Exemple :



les arcs sont  $ab$ ,  $ba$  et  $bc$



le chemin orienté  
de longueur 4.



le circuit de longueur 4.

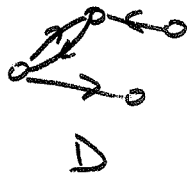
• Lorsque  $xy \in A$ , on dit que  $y$  est voisin sortant de  $x$  et que  $x$  est voisin entrant de  $y$ . le voisinage sortant de  $x$  est :

$$N^+(x) = \{y : xy \in A\} \quad (\text{ou } \text{Vois}^+(x)).$$

Degré sortant est :  $d^+(x) = |N^+(x)|.$

- On définit de même voisinage entrant et degré entrant.
- lorsque  $d^-(x) = 0$ , on dit que  $x$  est une source, si  $d^+(x) = 0$ ,  $x$  est un puits.
- lorsqu'on oublie les orientations, on obtient un graphe non-orienté, dit graphe sous-jacent. Pour un graphe orienté  $D$ , on note  $\cup(D)$  son graphe sous-jacent.

Ex:



$D$

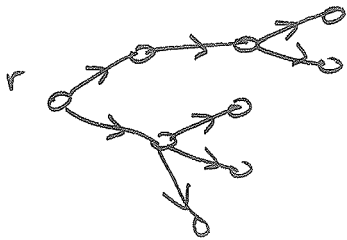


$\cup(D)$ .

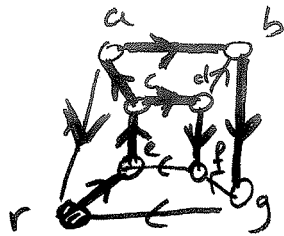
- Codage: on peut coder  $D$  par :
  - 1 - Liste des arcs
  - 2 - Liste des voisins sortants par chaque  $x$ :  $N^+(x)$  ou  $\text{Vois}^+(x)$ .
  - 3 - Matrice d'adjacence:  $A_D = (a_{ij})$  où 
$$\begin{cases} a_{ij} = 0 & \text{si } ij \notin A \\ a_{ij} = 1 & \text{si } ij \in A. \end{cases}$$
 ( $A_D$  n'est plus symétrique!).

## II) Parcours orientés.

- C'est comme dans le cas non-orienté, sauf qu'on remplace  $\text{Vois}(x)$  par  $\text{Vois}^+(x)$ : on suit le sens des arcs.
- On va obtenir des arborescences sortantes: une arborescence sortante est l'orientation d'un arbre  $T$  de racine de telle façon que par tout  $x$ , il existe un chemin orienté de  $r$  à  $x$ .



• Exemple de parcours en largeur:



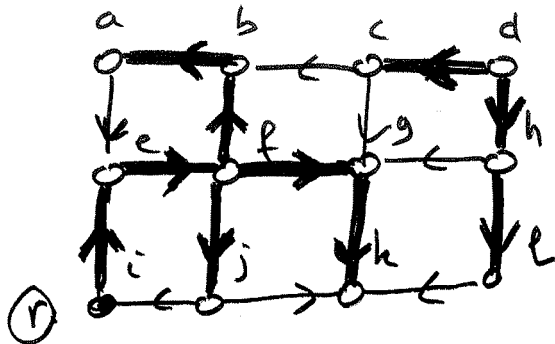
AT: ~~r~~ ~~c~~ ~~a~~ ~~e~~ ~~d~~ ~~b~~ ~~f~~

sommet	a	b	c	d	e	f	g	r
père	c	a	e	c	r	d	a	r
ordre	4	6	3	5	2	7	8	1
niveau	3	4	2	3	1	4	5	0

• Si tous les sommets ne sont pas atteignables par un chemin depuis r, souvent on choisit un sommet non atteint et on continue le parcours à partir de ce sommet.

• Exemple de parcours en profondeur:

On choisit un sommet restant.  
↓



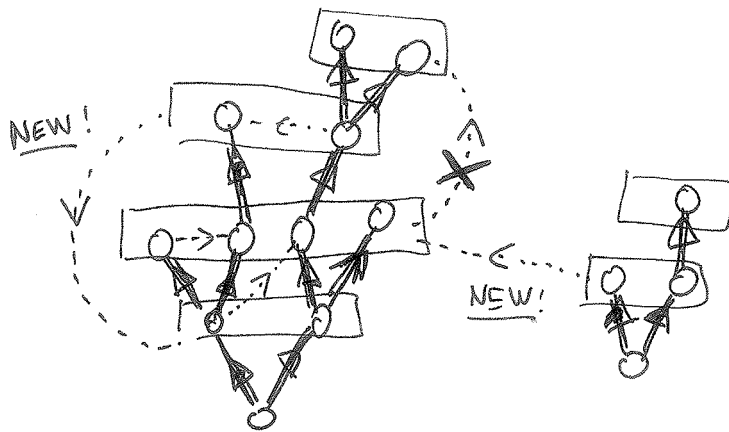
AT: ~~r~~ ~~b~~ ~~f~~ ~~d~~ ~~d~~ ~~i~~ ~~e~~ ~~f~~ ~~d~~ ~~i~~ ~~f~~ ~~g~~ ~~h~~

sommet:	a	b	c	d	e	f	g	h	i	j	k	l	
père:		b	f	d	d	i	e	f	d	i	f	g	h
début:	5	4	18	17	2	3	8	20	1	12	9	21	
fin:	6	7	19	24	15	14	11	23	16	13	10	22	
niveau:	4	3	X	X	1	2	3	X	0	3	4	X	

Il n'y a pas de chemin orienté de i à d, donc pas de niveau.  
(dist(i, d) = ∞)

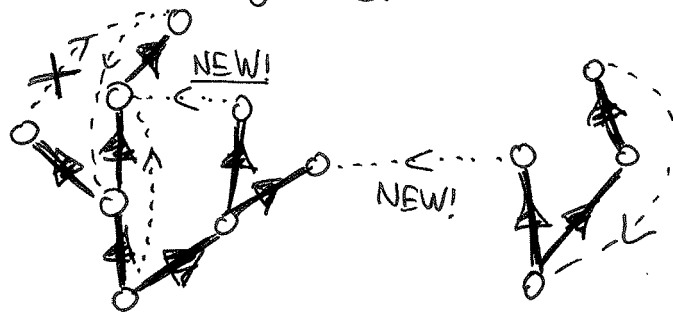
• Sans preuve, on va regarder les arborescences qu'on obtient à l'issue d'un parcours (en profondeur et largeur).

## Parcours en largeur:



- Entre niveau identique: toujours possible
- D'un niveau  $k$  à  $k+1$ : toujours possible.
- D'un niveau  $k$  à  $k'$  avec  
 $k > k'$ : possible; nouveau.  
 $k < k'$ : impossible.

## Parcours en profondeur:



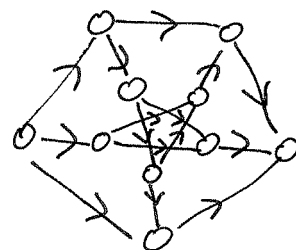
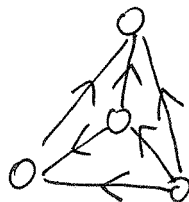
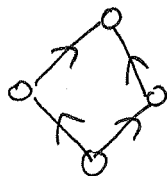
- A l'intérieur d'une même branche: toujours possible
- D'un sommet  $x$  vers un sommet  $y$  n'étant pas dans la même branche:  
 si  $d(x) > d(y)$ : possible; nouveau.  
 si  $d(x) < d(y)$ : impossible.

• Bq: on peut aussi faire des parcours étendus...

## III) DAG.

- Un DAG (directed acyclic graph) est un graphe orienté sans circuit. Même si c'est l'analogue des arbres pour les graphes non-orientés, leur structure est beaucoup plus complexe.

• Exemple:



En fait, tout graphe admet une orientation sans circuit.