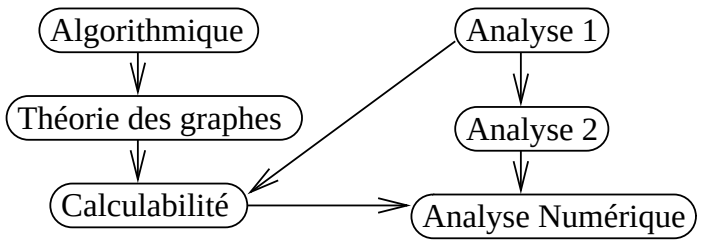
THÉORIE DES GRAPHES (3)

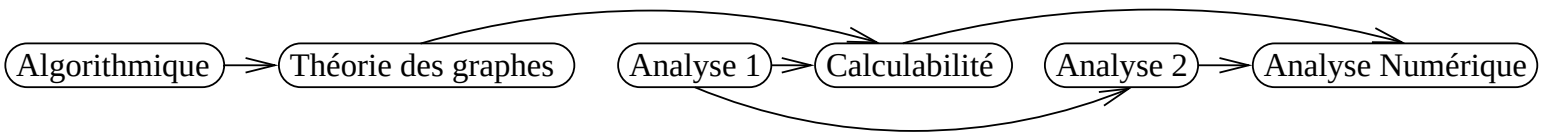
Michel Rigo

http://www.discmath.ulg.ac.be/

Année 2006–2007

TRI TOPOLOGIQUE





déterminer une indexation des sommets d'un graphe orienté sans cycle de manière telle que s'il existe un arc de v_i à v_j, alors $i < j$.

Tri topologique

cas des graphes simples

Lemme

Si un graphe simple orienté $G = (V, E)$ est sans cycle, alors $\exists\, v$ tel que $d^-(v) = 0$ (resp. $d^+(v) = 0$).

Considérons un chemin simple $(x_1, \ldots, x_k)$ de G de longueur maximale déterminé par des sommets de G.

Si $d^-(x_1) > 0$, alors il existe $y \in \mathrm{pred}(x_1)$.

Si y = un des x_j, on aurait un cycle $(y, x_1, \ldots, x_j)$, impossible.

Or par maximalité du chemin $(x_1, \ldots, x_k)$, il n'est pas possible d'avoir un sommet distinct des x_j et tel que $(y, x_1) \in E$.

Tri topologique

Proposition

Un graphe simple orienté $G = (V, E)$ est sans cycle SSI $\exists v \in V$ tel que $d^-(v) = 0$ et $\forall v$ tel que $d^-(v) = 0$, le graphe $G - v$ est sans cycle.

$\Rightarrow$ Si G est sans cycle, on peut appliquer le lemme précédent.
Il existe un sommet v tel que $d^-(v) = 0$.
De plus, tout sous-graphe $G - v$ d'un graphe sans cycle G est sans cycle.

$\Leftarrow$ Soit v un sommet tel que $d^-(v) = 0$.
Par hypothèse, $G - v$ est sans cycle.

Si G possède un cycle, ce dernier doit passer par v.
Si un cycle passe par v, $d^-(v) \geq 1$. Impossible !

TRI TOPOLOGIQUE

Algorithme permettant de décider si un graphe est sans cycle.

Tant qu'il existe $v \in V$ tel que $d^-(v)=0$,
$G := G - v$
Si $G=\emptyset$
alors sortie : "oui, G sans cycle"
sinon sortie : "non, G possède un cycle"

COMPLEXITÉ

Si on implémente cet algorithme à l'aide de listes d'adjacence,détecter v t.q. $d^-(v) = 0$ nécessite de parcourir l'ensemble du graphe.

Un tel parcours est effectué à chaque étape de la boucle.

Complexité est quadratique en $\#E + \#V$.

Tri topologique

Théorème

Le graphe simple orienté $G = (V, E)$ est sans cycle SSI
il est possible d'énumérer les sommets de $V = \{v_1, \ldots, v_n\}$ t.q.
$\forall i = 1, \ldots, n$, le demi-degré entrant de v_i restreint au graphe
$G_i = G - v_1 - \cdots - v_{i-1}$ soit nul : $d^-_{G_i}(v_i) = 0$.

$\Rightarrow$ Supposons G sans cycle.
Par le lemme ..., $\exists v_1$ de $G = G_1$ tel que $d^-(v_1) = d^-_{G_1}(v_1) = 0$.
Par la prop..., $G_1 - v_1 = G_2$ est sans cycle.
Par le lemme ..., $\exists v_2$ tel que $d^-_{G_2}(v_2) = 0$.

$\vdots$

De proche en proche, on obtient l'énumération proposée.

TRI TOPOLOGIQUE

$\Leftarrow$ Supposons disposer d'une énumération des sommets ayant les propriétés indiquées. Procédons par récurrence.

G_n est restreint à l'unique sommet v_n : sans cycle.

G_{n-1} contient les sommets v_n et v_{n-1} et $d^-_{G_{n-1}}(v_{n-1}) = 0$.
G_{n-1} possède au mieux un arc de v_{n-1} à v_n :sans cycle.

Appliquons ce raisonnement pour une étape i quelconque.

Si le graphe G_{i+1} est sans cycle,
alors G_i se compose du graphe G_{i+1} auquel on ajoute v_i
et éventuellement des arcs de v_i vers les sommets de G_{i+1}.
On en conclut que G_i est sans cycle.

TRI TOPOLOGIQUE

ALGORITHME BASÉ SUR LE THM.

La variable d associée à chaque sommet du graphe permet de stocker le demi-degré entrant du sommet (par rapport au graphe envisagé au moment de la construction).

```
Pour tout v ∈ V, initialiser d(v)=0
Pour tout v ∈ V,
  Pour tout w ∈ succ(v), d(w)=d(w)+1
aTraiter := ∅
nbSommet := 0
Pour tout v ∈ V,
  Si d(v) = 0, alors
    aTraiter=aTraiter∪{v}
    nbSommet :=nbSommet+1
```

TRI TOPOLOGIQUE

```
Tant que aTraiter ≠ ∅, faire
  Soit v, le premier élément de aTraiter
  aTraiter := aTraiter \{v}
  Pour tout w ∈ Succ(v), faire
    d(w)=d(w)-1
    si d(w)=0, alors
      aTraiter=aTraiter∪{w}
      nbSommet := nbSommet+1
Si nbSommet=#V
  alors sortie : "oui, G sans cycle"
  sinon sortie : "non, G possède un cycle"
```

IDÉE

Si v est enlevé de `aTraiter`, v est énuméré.
Lorsqu'un sommet est traité, on le supprime du graphe et on modifie en conséquence les demi-degrés entrants.
Si tous les sommets ont été traités, le graphe est sans cycle.

Tri topologique

Définition

Soit $G = (V, E)$ un graphe simple orienté. Un tri topologique de G est une énumération $v_1, \ldots, v_n$ des sommets de G t.q. si (v_i, v_j) est un arc de G, alors $i < j$.

Théorème

Un graphe simple orienté admet un tri topologique SSI il est sans cycle.

Si un graphe possède un cycle, alors quelle que soit l'énumération de ses sommets, il ne peut s'agir d'un tri topologique.

Si un graphe est sans cycle, alors une énumération de ses sommets donnant lieu à un tri topologique estdonnée par le thm précédent.

Tri topologique

Remarque

Il n'y a pas qu'un seul tri topologique pour un graphe donné $G = (V, E)$. En effet, si on dénote par

$$S(G) = \{v \in V \mid d^-(v) = 0\}$$

l'ensemble des *sources* de G, alors l'ensemble des tris topologiques de G est donné par la formule récursive suivante

$$\Pi(G) = \bigcup_{v \in S(G)} \{v.\sigma \mid \sigma \in \Pi(G - v)\}$$

où σ est un tri topologique de $G - v$ et où $v.\sigma$ désigne l'énumération des sommets de G en débutant par v puis en suivant l'énumération prescrite par σ.

Arbres

Définition

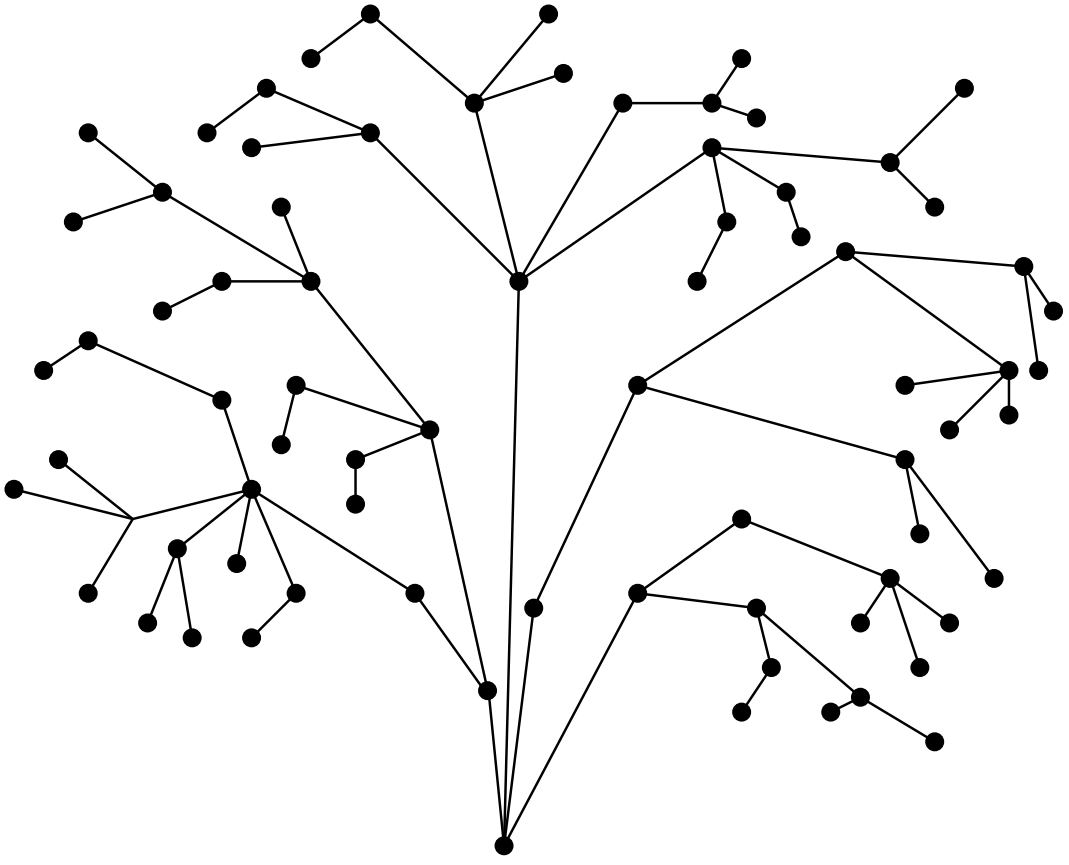
Un graphe simple non orienté $A = (V, E)$ est un arbre s'il est connexe et sans cycle

(sous-entendu, un "véritable" cycle : une piste fermée, pas un cycle "artificiel" comme un trivial $(\{a, b\}, \{b, a\})$; imposer l'absence d'une piste fermée évite de telles arêtes doublées et on pourrait de manière équivalente imposer l'absence de circuit simple).

Une forêt est un graphe simple non orienté dont chaque composante connexe est un arbre.

Un arbre $A = (V, E)$ est qualifié de n-aire si pour tout sommet $v \in V$, $\deg(v) \leq n$.

Arbres



PROPOSITION

Soit $G = (V, E)$ un arbre ayant n sommets.

- Toute paire de sommets distincts de G est connectée par exactement un chemin simple.
- Soit $e \in (V \times V) \setminus E$ qui n'est pas une boucle. Le graphe $G + e$ contient un cycle (i.e., une piste fermée), c'est-à-dire, $G + e$ n'est plus un arbre.
- Le graphe G a exactement $n - 1$ arêtes.

PROPOSITION

Un graphe $G = (V, E)$ simple connexe est un arbre SSI chacune de ses arêtes est une arête de coupure.

Soient G un arbre et e une de ses arêtes. Puisque G est sans cycle, il ne possède aucune piste fermée.

Vu prop..., e est une arête de coupure.

Inversement, si G est un graphe connexe possédant une piste fermée alors, par la prop..., les arêtes de cette piste ne peuvent être des arêtes de coupure.

COROLLAIRE

Tout graphe connexe possède un sous-arbre couvrant.

Soient $G = (V, E)$ un graphe connexe et $C = (V, E')$ un sous-graphe couvrant connexe minimal (i.e., on ne peut pas remplacer E' par un sous-ensemble strict et garder connexité).

Vu la minimalité de C, chacune de ses arêtes est une arête de coupure de C. Par la prop. précédente : C est un arbre.

COROLLAIRE

Si $G = (V, E)$ est un graphe (simple non orienté) connexe, alors $\#E \geq \#V - 1$.

G possède un sous-arbre couvrant $C = (V, E')$. De là, il vient

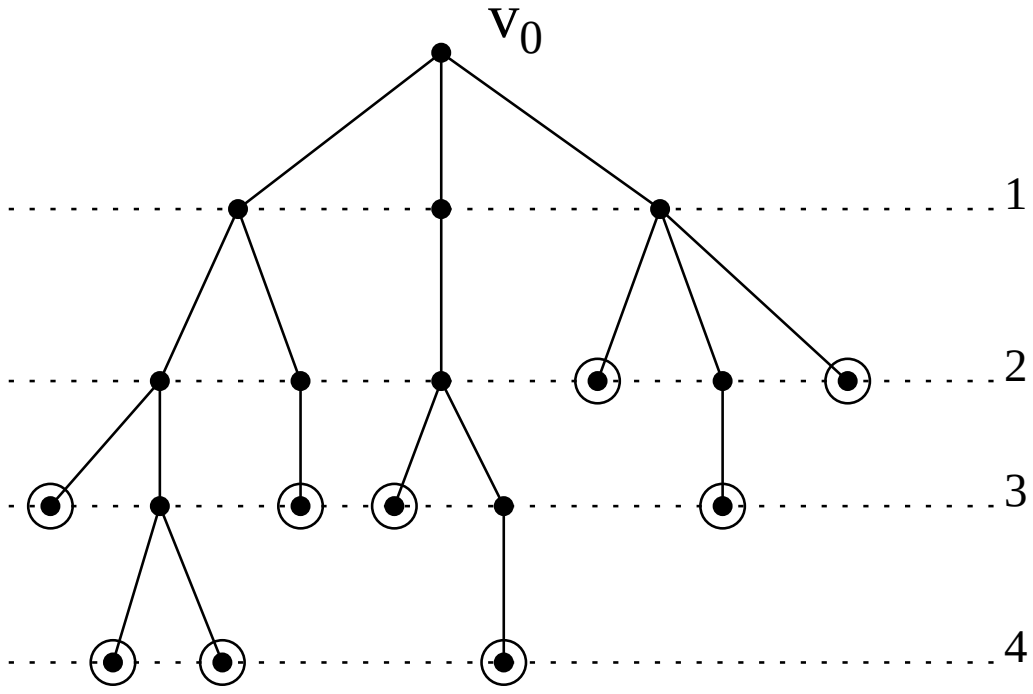
$$\#E \geq \#E' = \#V - 1$$

Arbres

Définition

Un arbre $A = (V, E)$ avec un sommet privilégié v_0 est un arbre pointé : (A, v_0). v_0 est la racine de l'arbre.

sommets de A ordonnés suivant leur distance à v_0.
Si $d(v_0, v) = i$ v est un sommet de niveau i.



Arbres

Définition

Si v est un sommet de niveau i et si tous ses voisins sont de niveau $i-1$, on dit alors que v est une feuille de l'arbre.

La hauteur d'un arbre est le niveau maximal de ses feuilles.

Définition

Pointer un arbre définit naturellement une orientation des arêtes de l'arbre : orienter les arcs de façon à ce qu'ils joignent les sommets de niveau i aux sommets de niveau $i+1$.

fils (resp. père) d'un noeud v : ses successeurs (resp. son unique prédécesseur).

descendants (resp. ancêtres de v : les éléments de $\mathrm{succ}^*(v)$ (resp. $\mathrm{pred}^*(v)$).

DÉFINITION

Un arbre pointé est k-aire si tout sommet a au plus k fils.

Si $k = 2$, on parle d'arbre binaire.

Un arbre k-aire de hauteur n possède au plus

$$1 + k + \cdots + k^n = \frac{k^n - 1}{k - 1}$$

sommets. S'il en possède exactement ce nombre, on parle d'arbre k-aire complet.

Parcours d'arbres

ordonner les noeuds : on suppose que les fils d'un noeud v_i sont ordonnés $v_{i,1}, \ldots, v_{i,k_i}$. Cet ordre est connu et fixé une fois pour toutes.

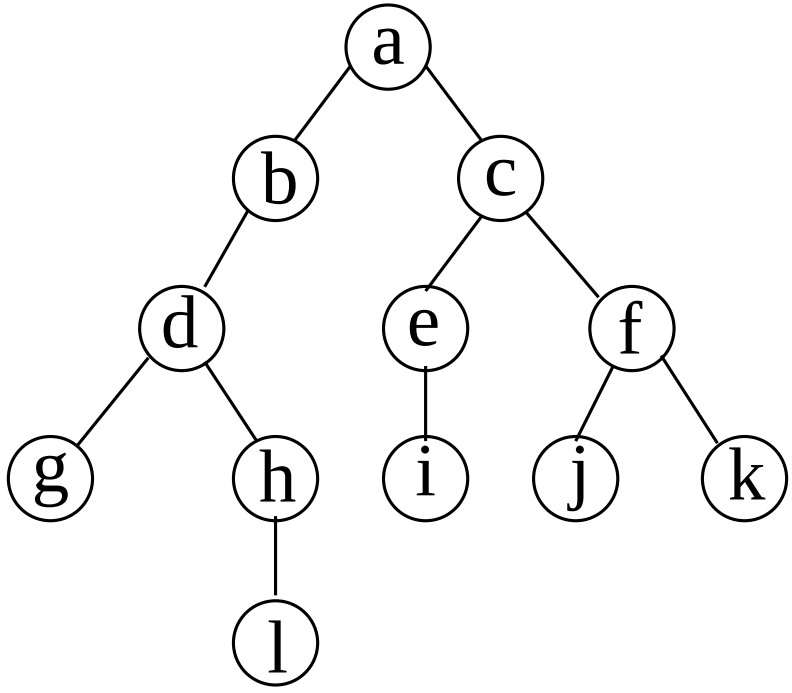
Parcours en profondeur

- parcours préfixe : d'abord la racine puis, de manière récursive, les sous-arbres pointés de racine respective $v_{0,1}, \ldots, v_{0,k_0}$.
- parcours suffixe : d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.
- parcours infixe (arbre binaire) : d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

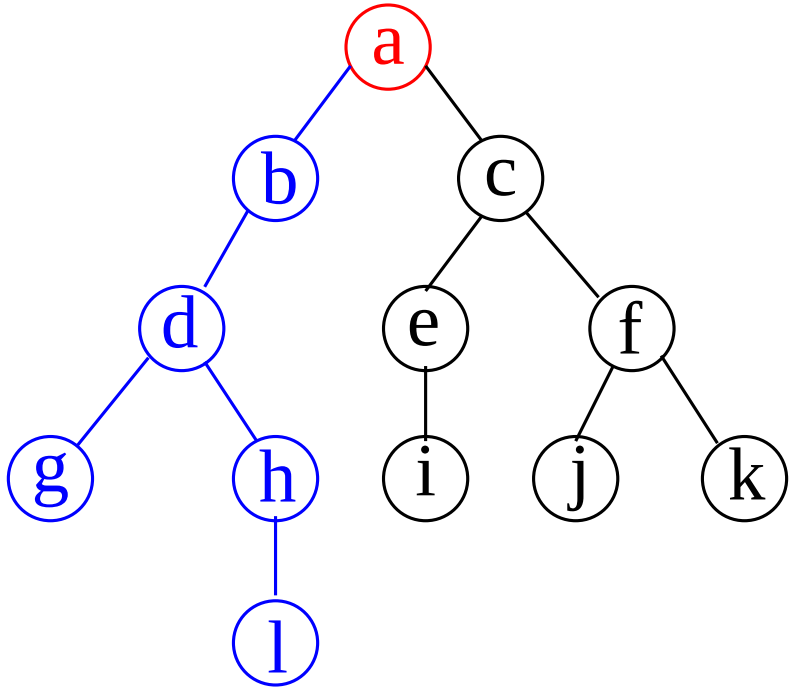


.

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

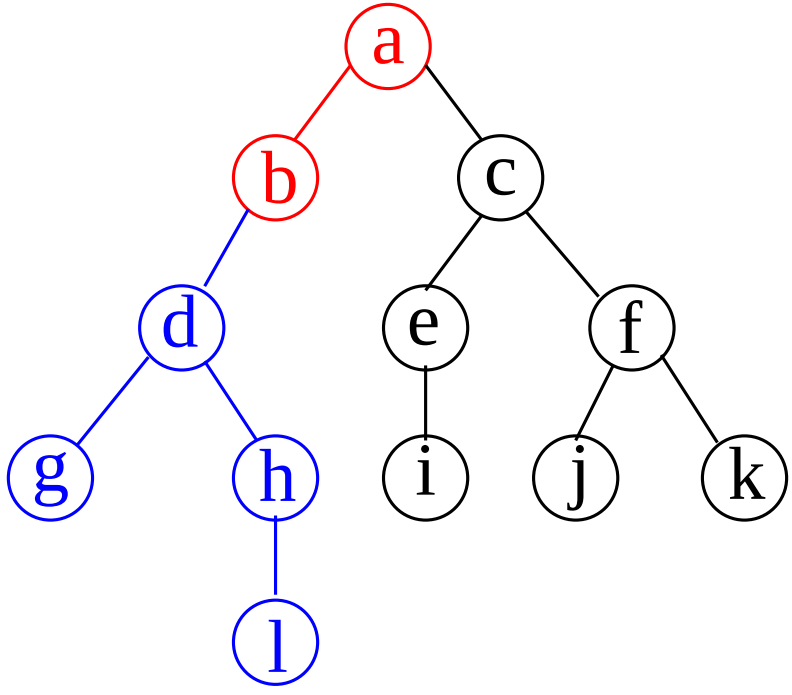


a

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

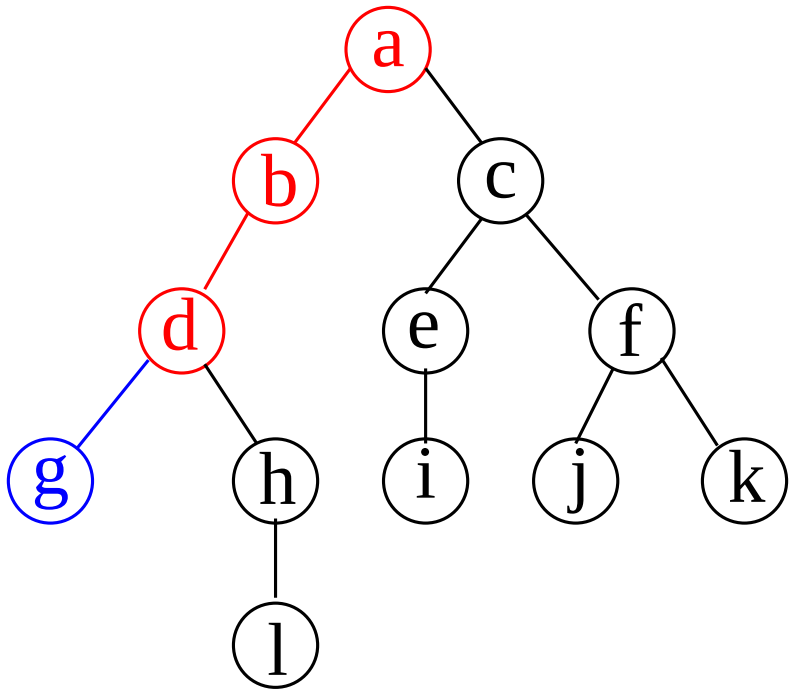


a, b

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

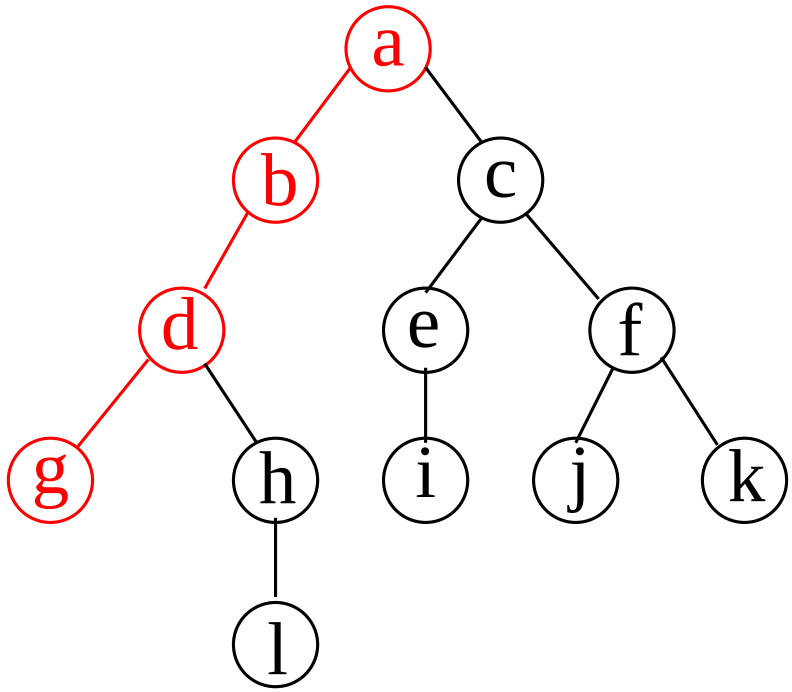


a, b, d

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

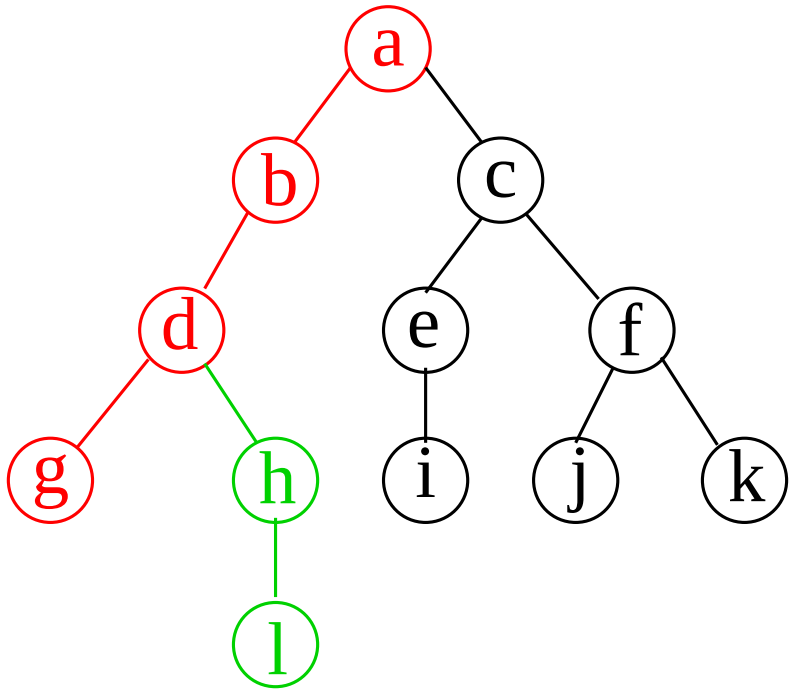


a, b, d, g

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

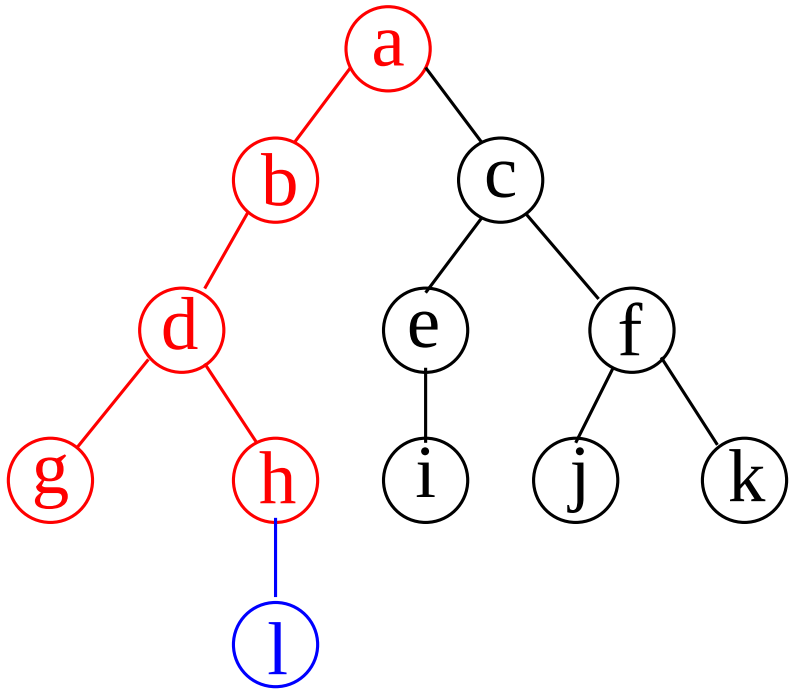


a, b, d, g

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

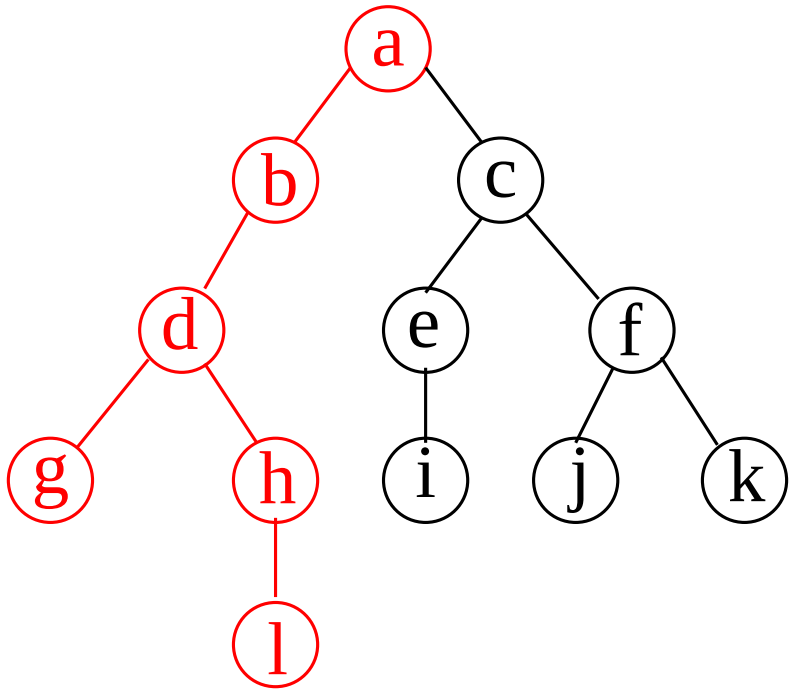


a, b, d, g, h

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

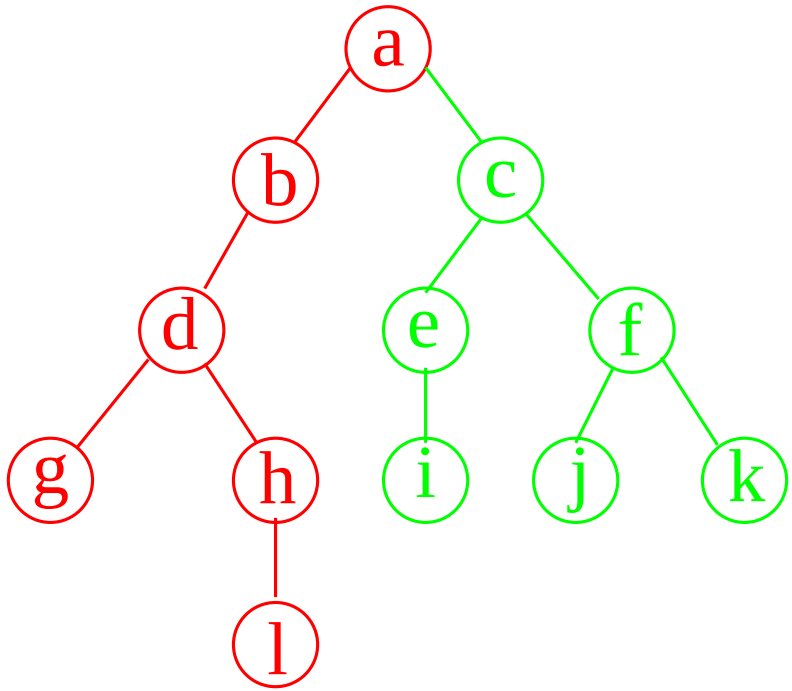


a, b, d, g, h, l

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

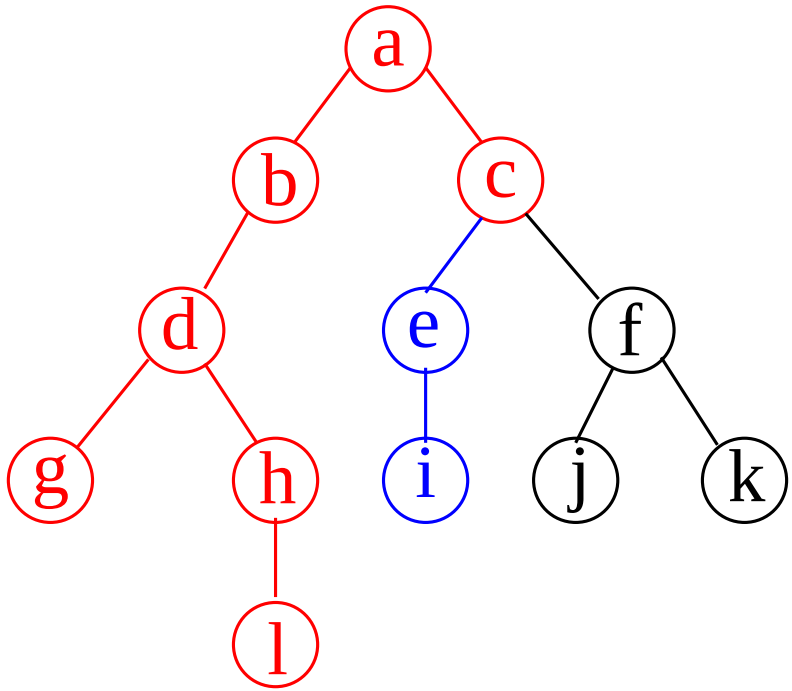


a, b, d, g, h, l

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

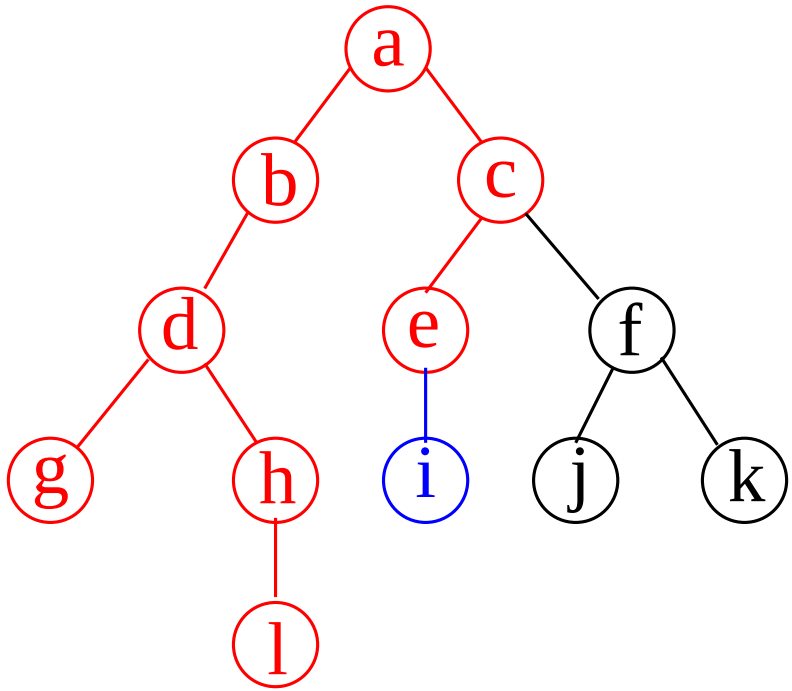


a, b, d, g, h, l, c

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

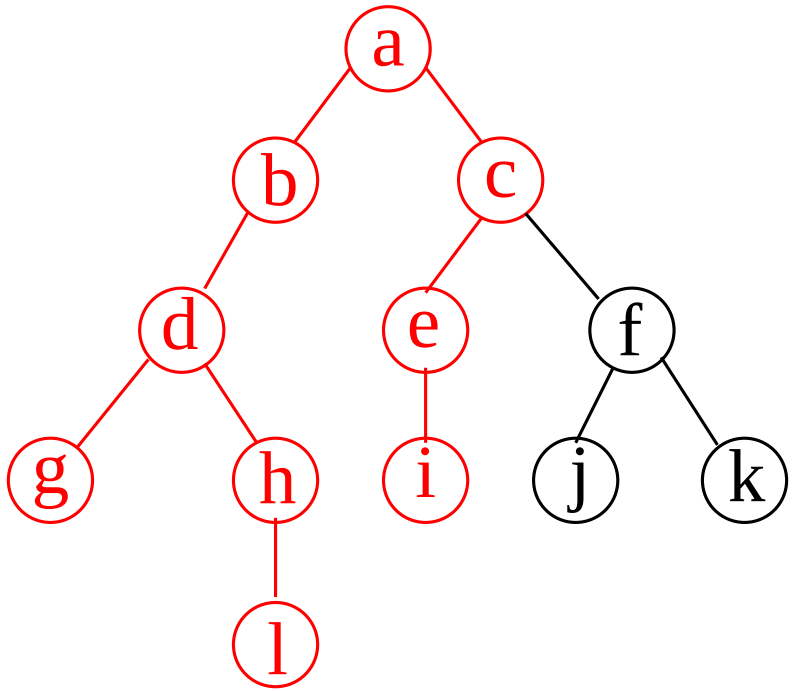


a, b, d, g, h, l, c, e

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

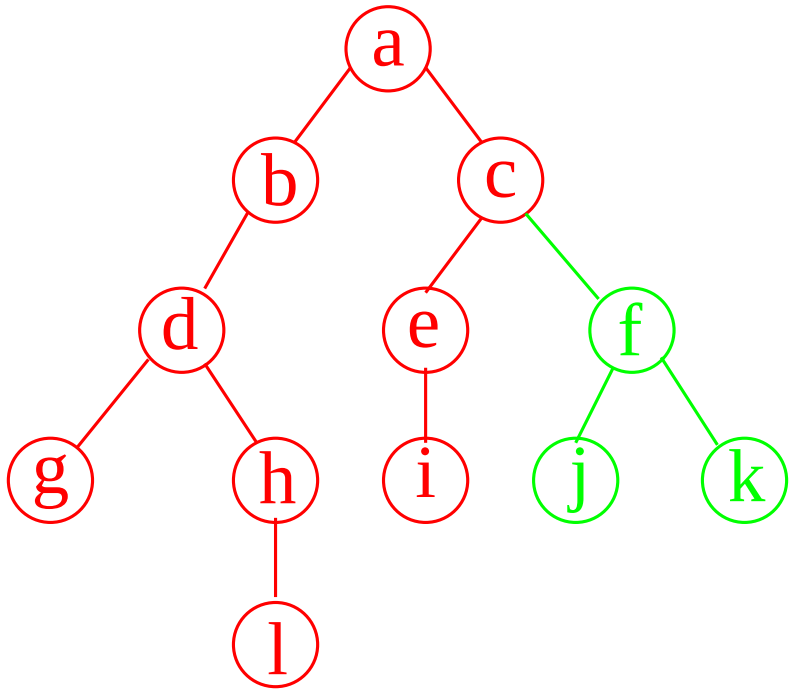


$a, b, d, g, h, l, c, e, i$

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

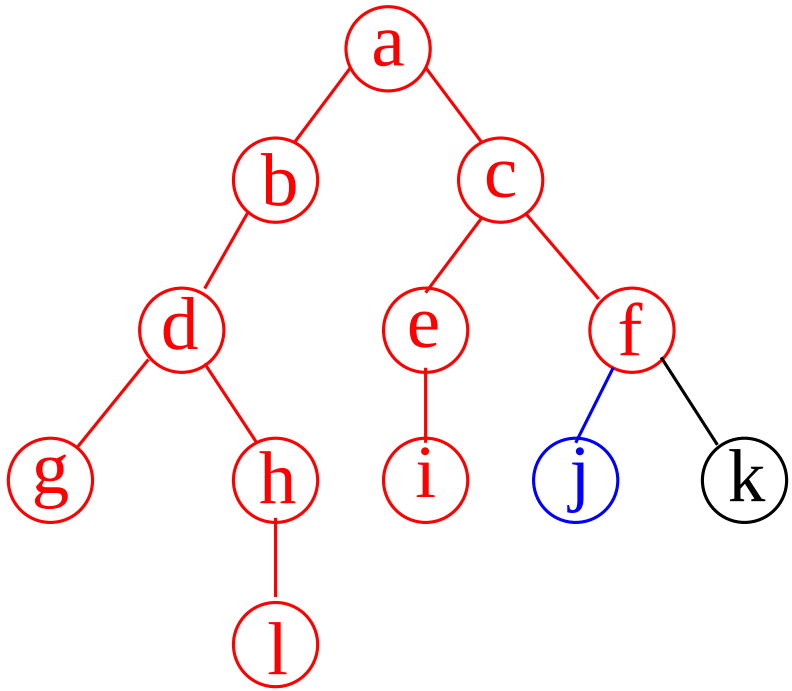


$a, b, d, g, h, l, c, e, i$

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

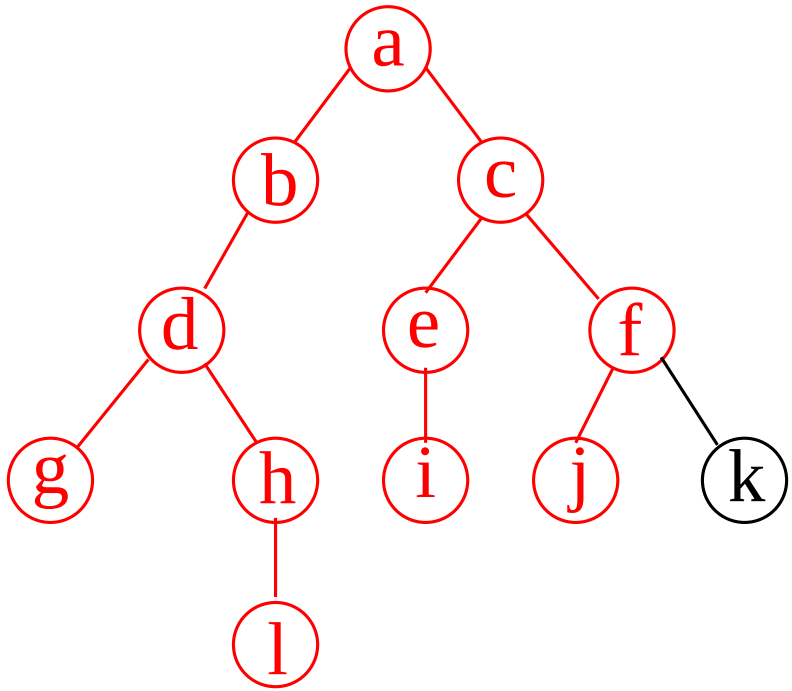


$a, b, d, g, h, l, c, e, i, f$

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

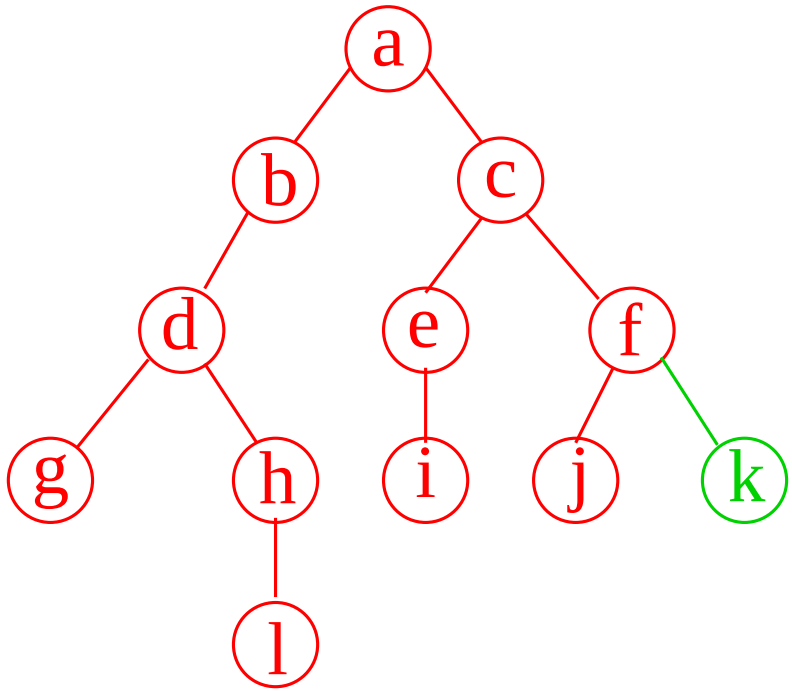


$a, b, d, g, h, l, c, e, i, f, j$

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

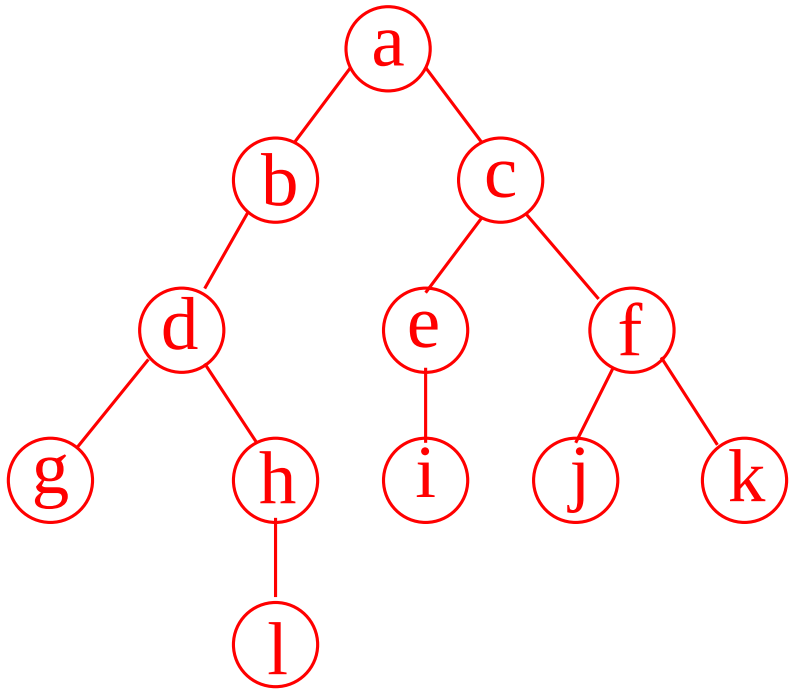


$a, b, d, g, h, l, c, e, i, f, j$

PARCOURS D'ARBRES

PARCOURS PRÉFIXE

d'abord la racine puis, de manière récursive, les sous-arbres pointés

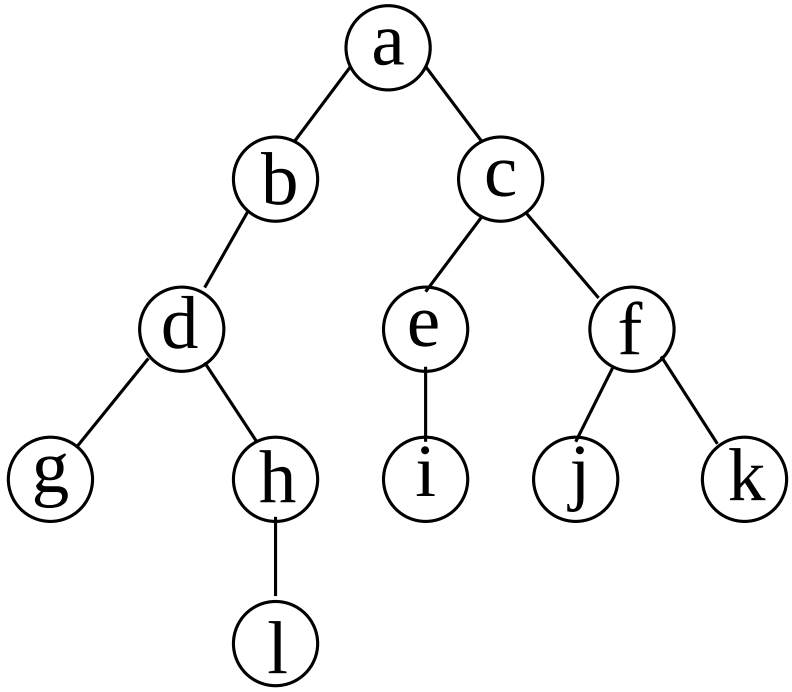


$a, b, d, g, h, l, c, e, i, f, j, k$

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

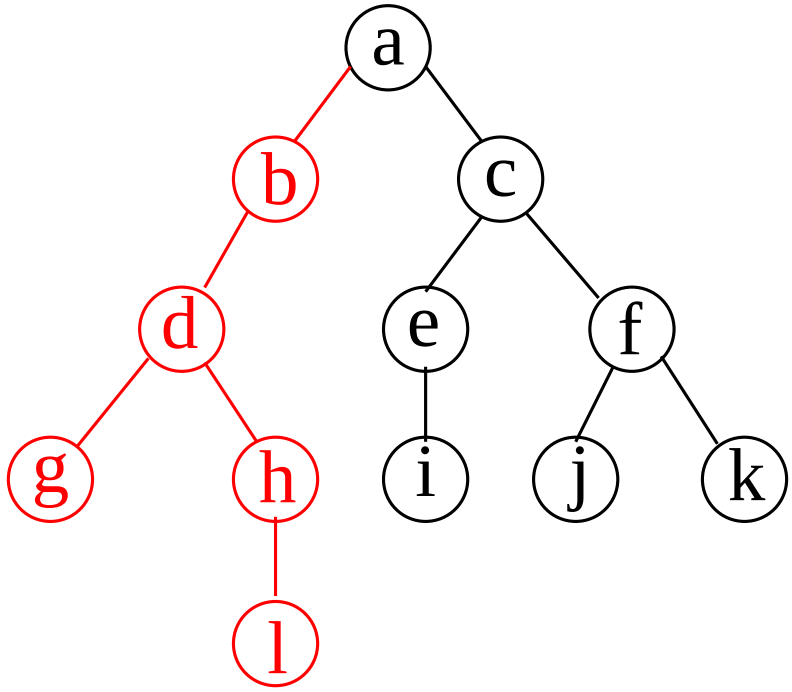


.

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

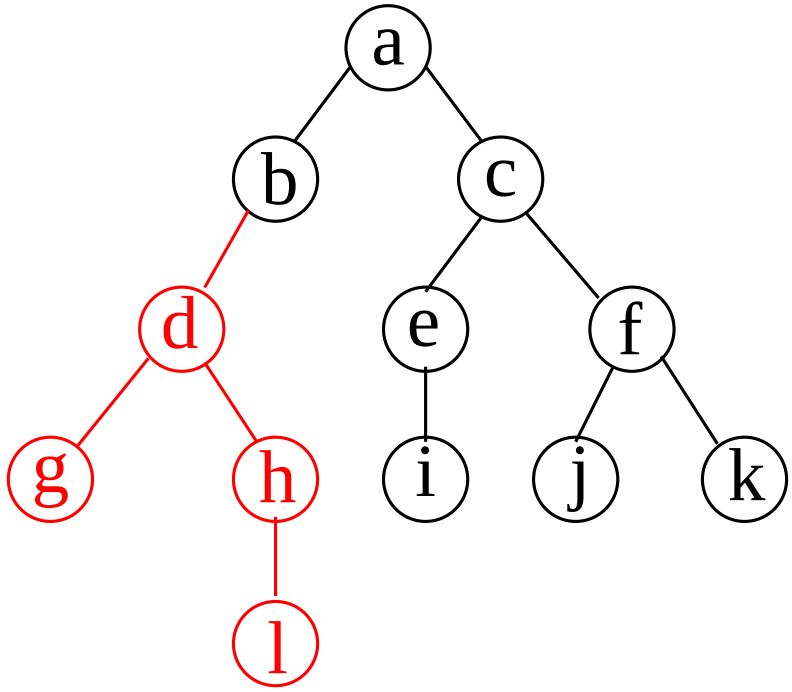


.

PARCOURS D'ARBRES

PARCOURS SUFFIXE

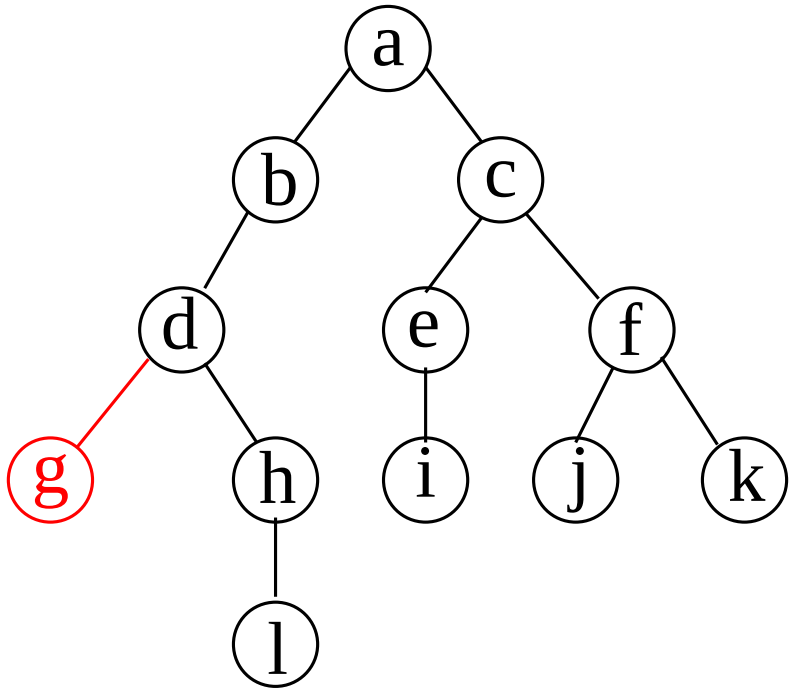
d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.



.

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

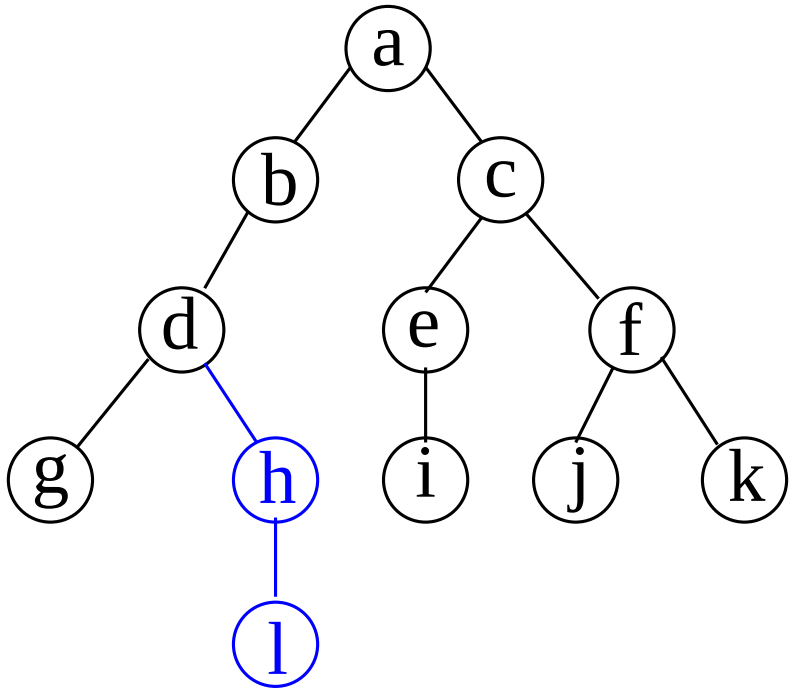


g

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

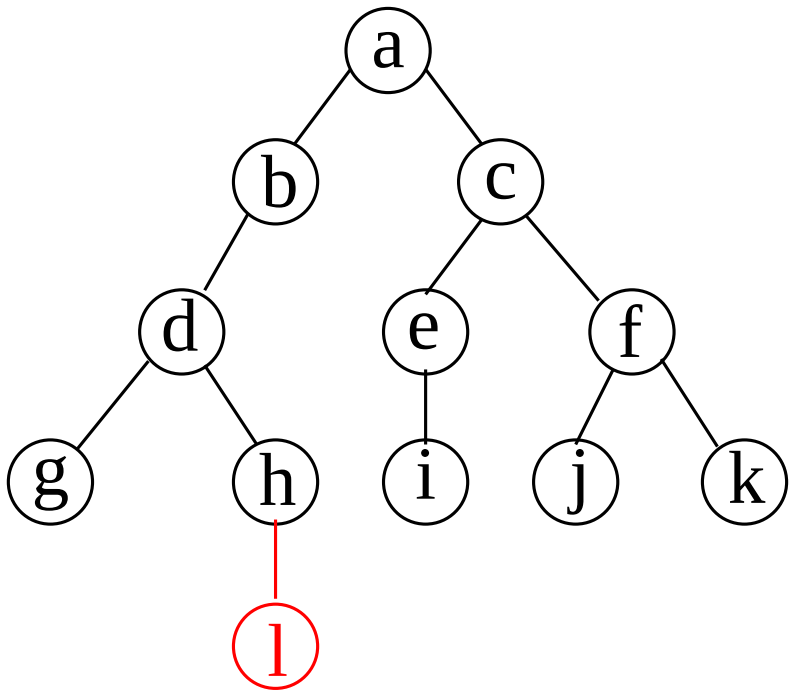


g

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

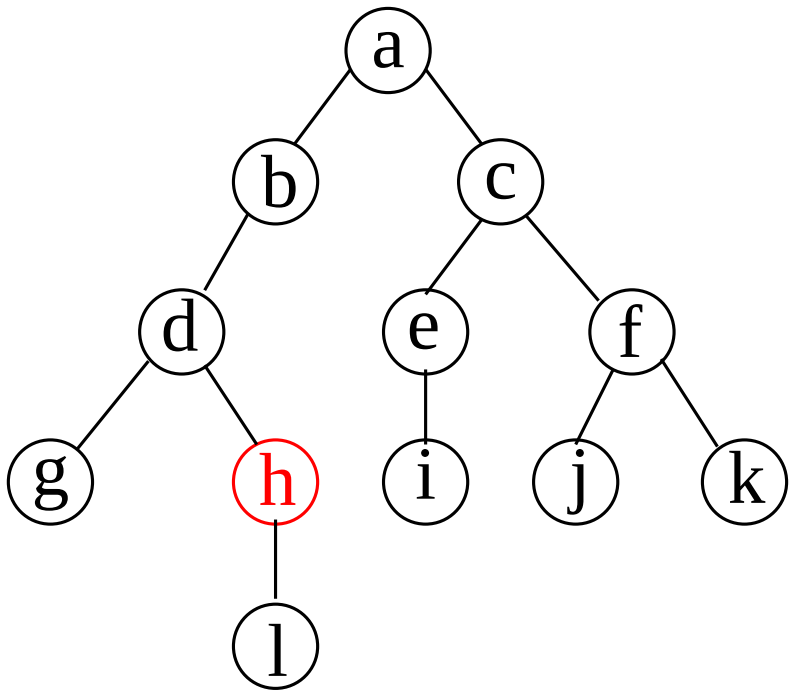


g, l

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

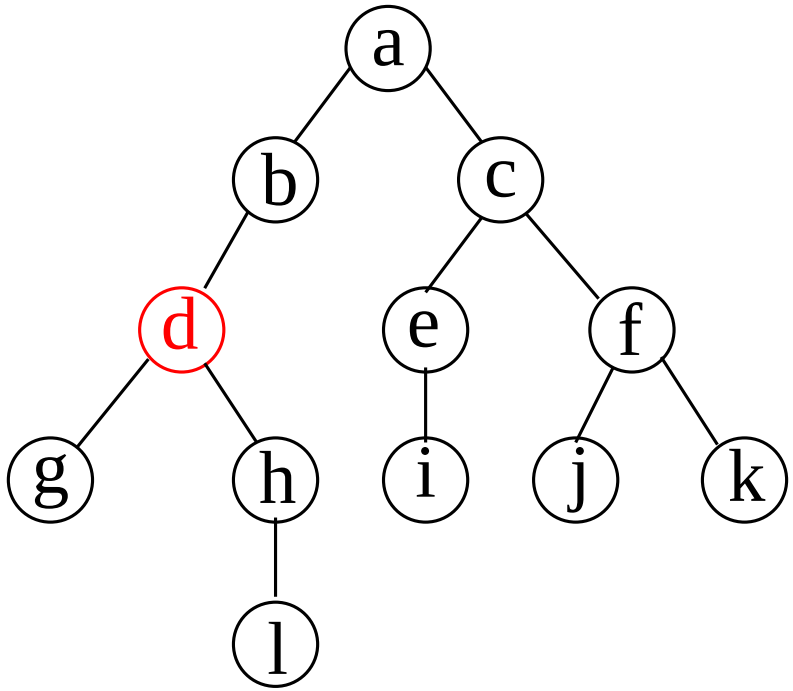


g, l, h

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

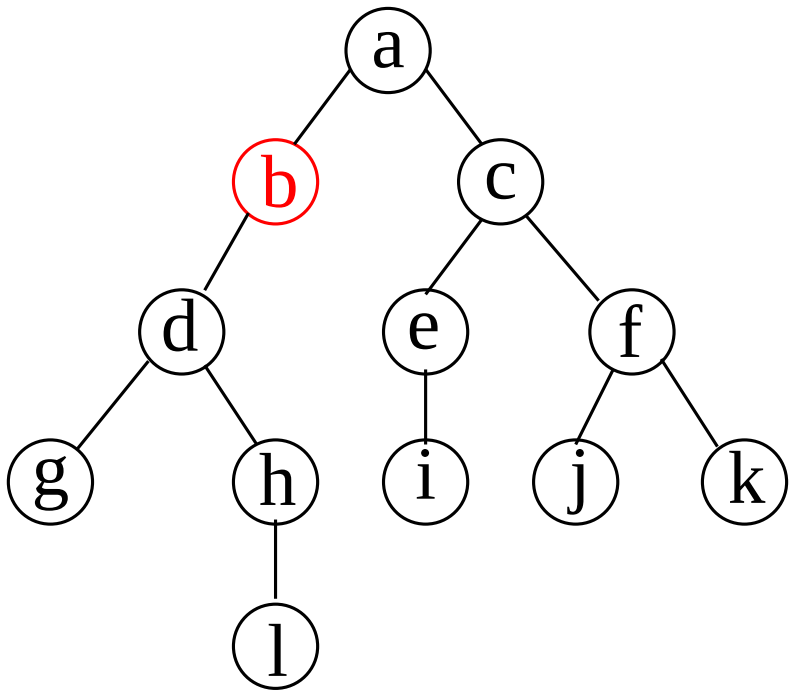


g, l, h, d

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

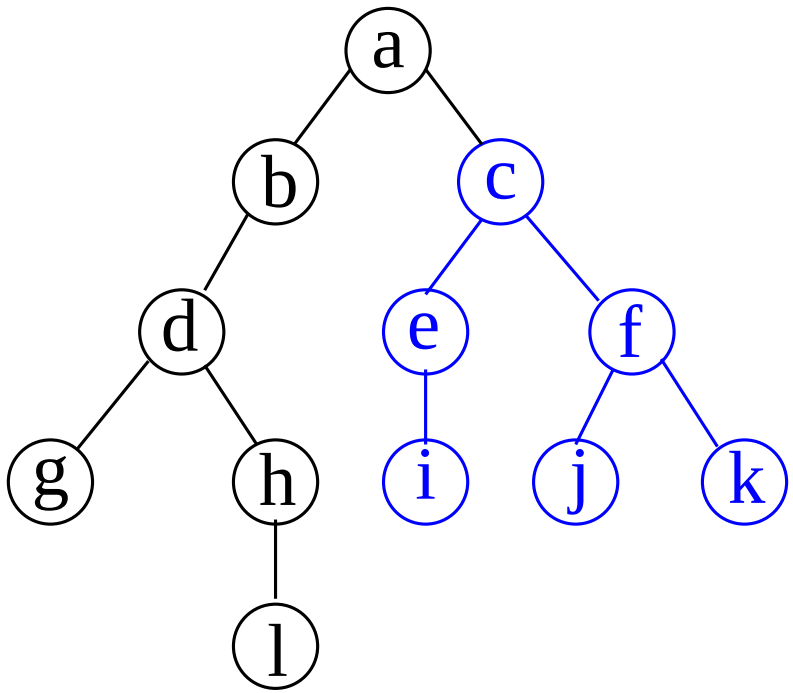


g, l, h, d, b

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

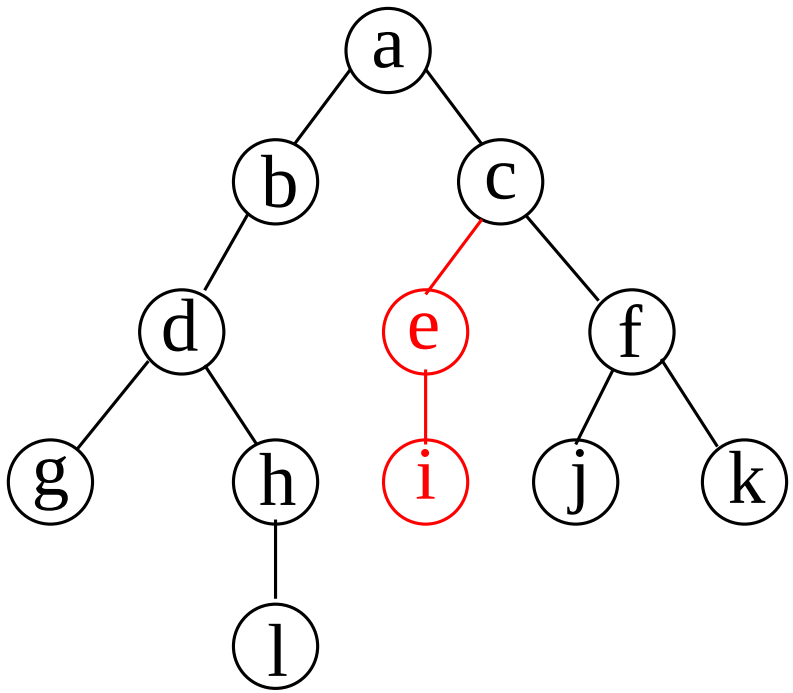


g, l, h, d, b

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

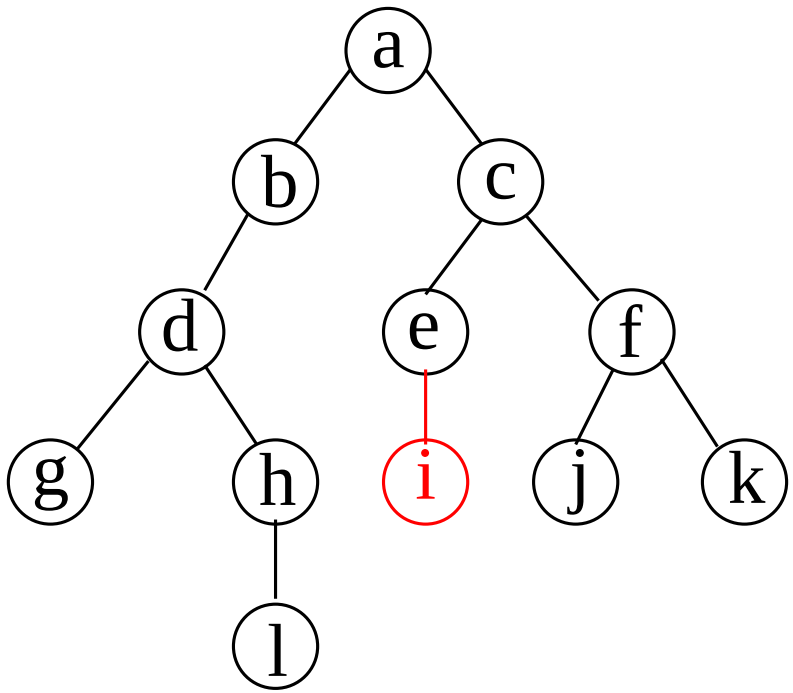


g, l, h, d, b

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

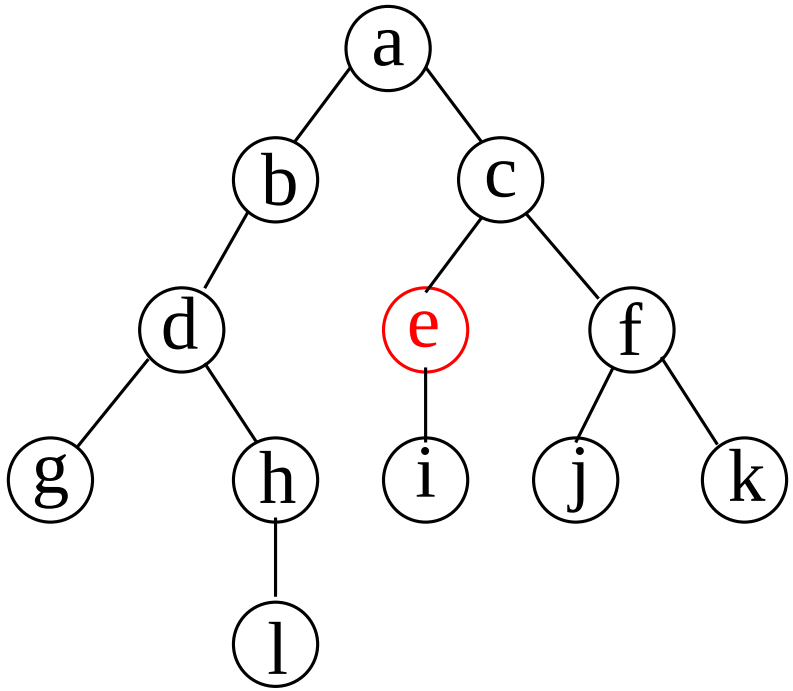


g, l, h, d, b, i

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

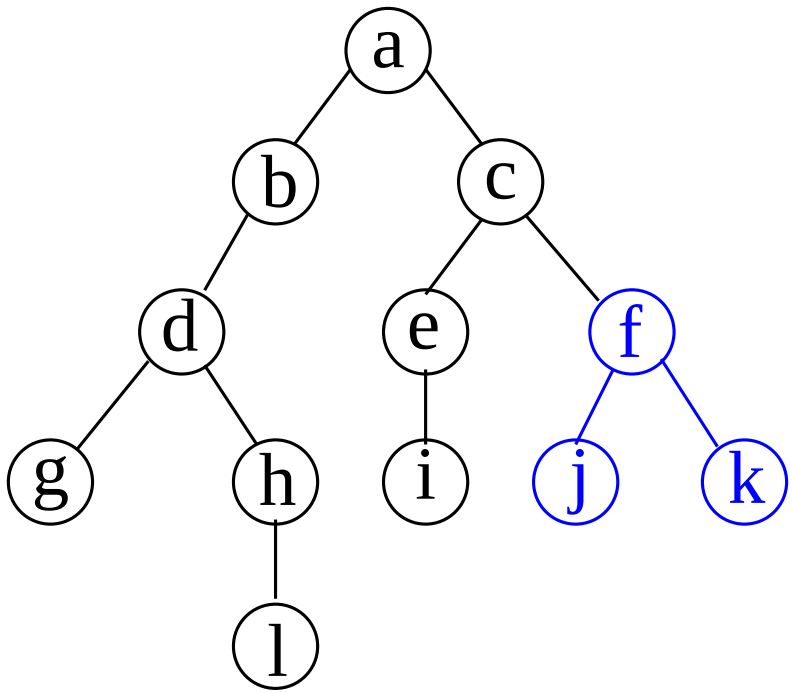


g, l, h, d, b, i, e

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

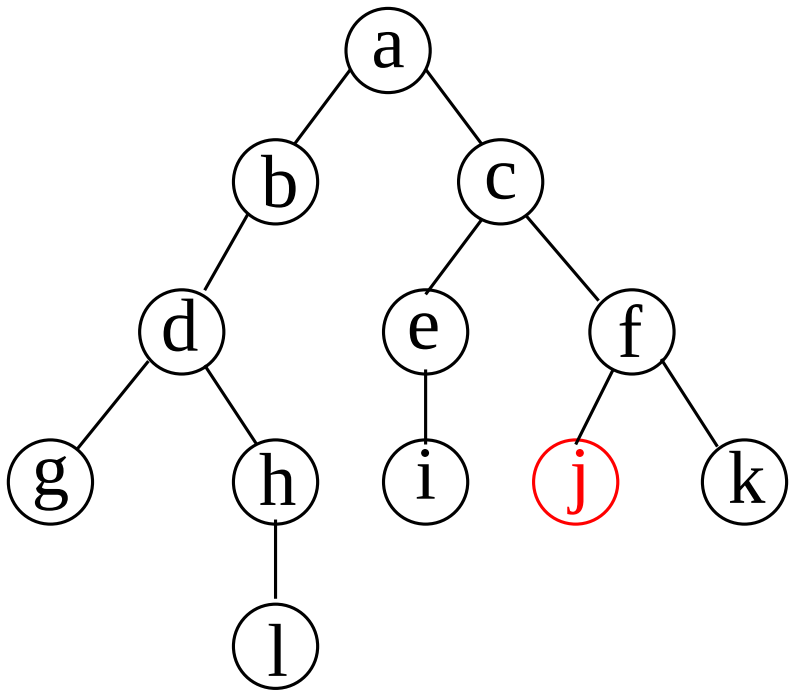


g, l, h, d, b, i, e

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

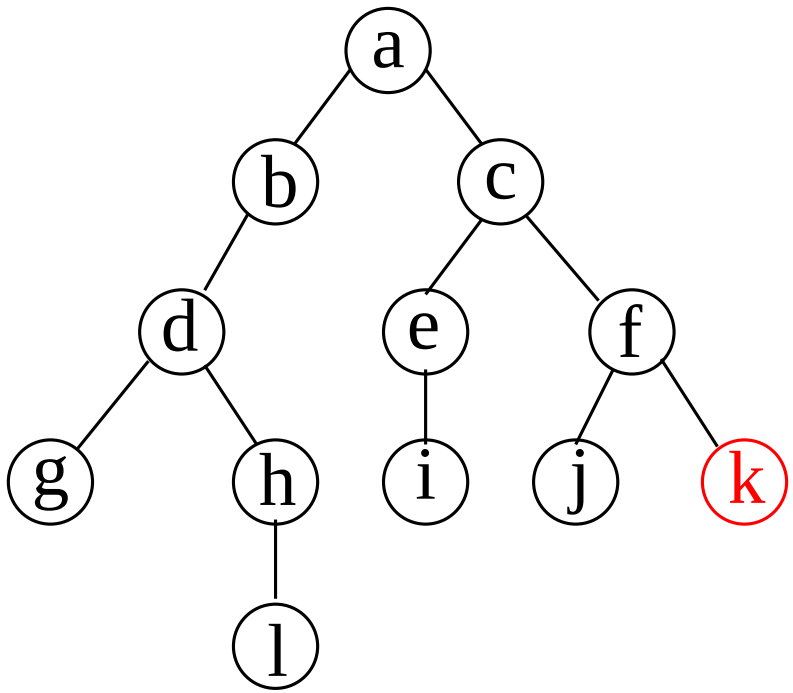


g, l, h, d, b, i, e, j

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

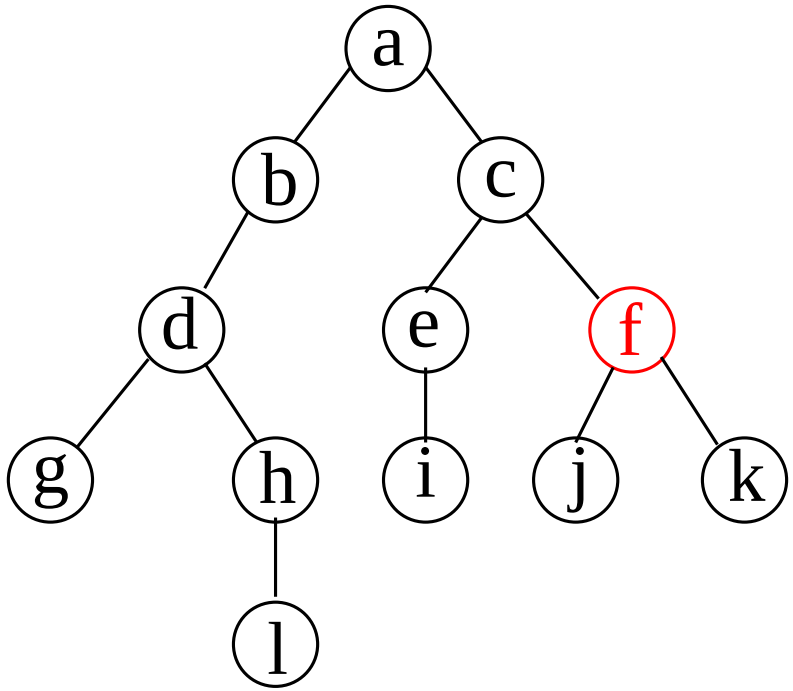


$g, l, h, d, b, i, e, j, k$

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

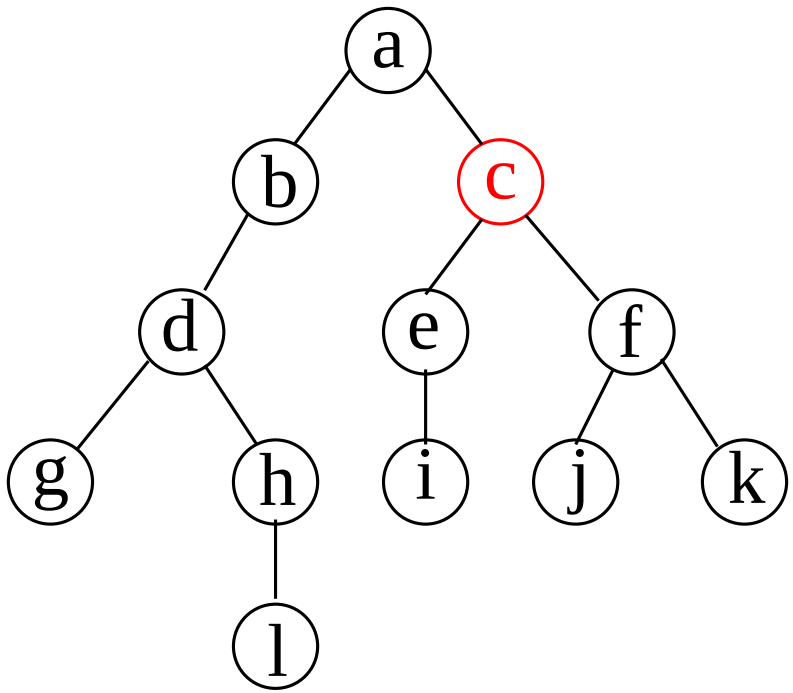


$g, l, h, d, b, i, e, j, k, f$

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

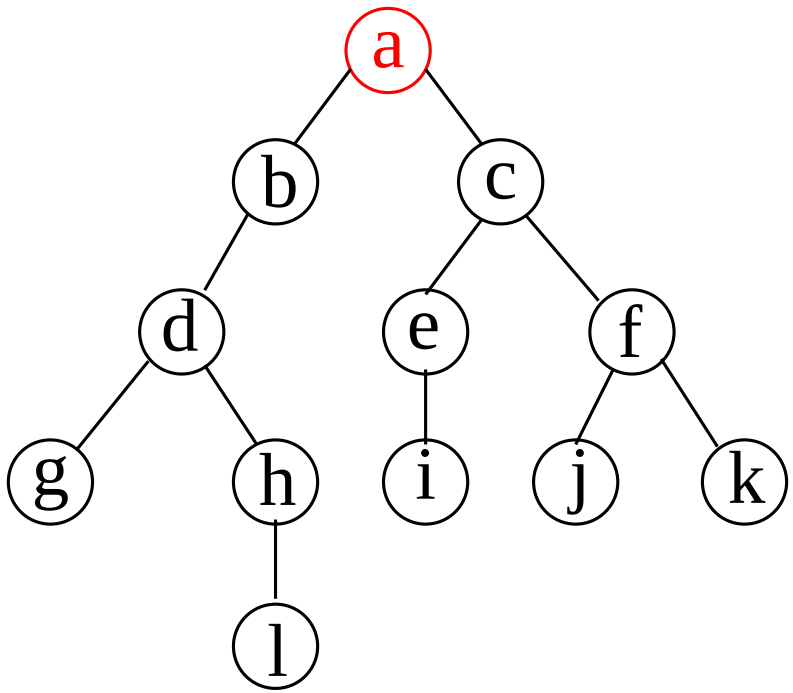


$g, l, h, d, b, i, e, j, k, f, c$

PARCOURS D'ARBRES

PARCOURS SUFFIXE

d'abord, de manière récursive, les sous-arbres pointés de racine $v_{0,1}, \ldots, v_{0,k_0}$, puis la racine v_0.

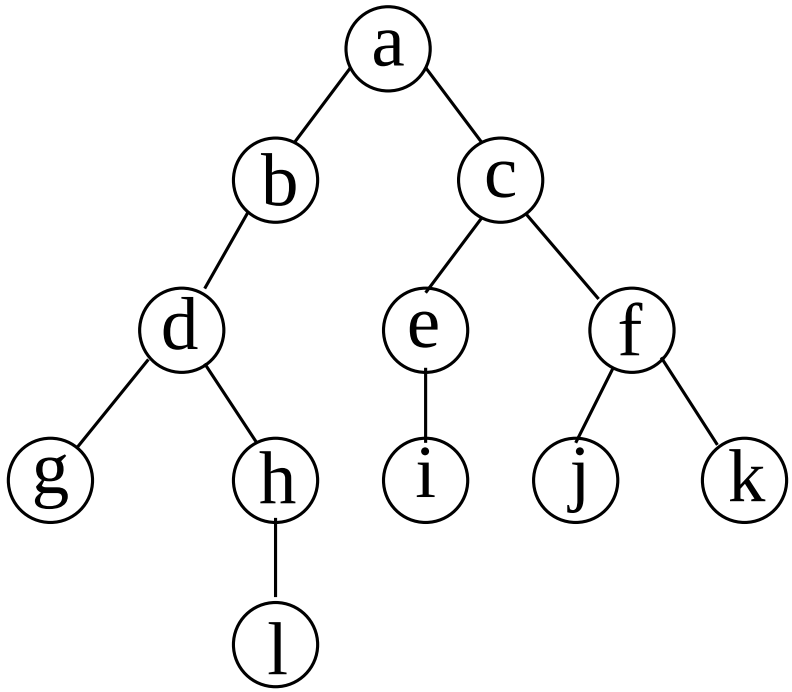


$g, l, h, d, b, i, e, j, k, f, c, a$

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

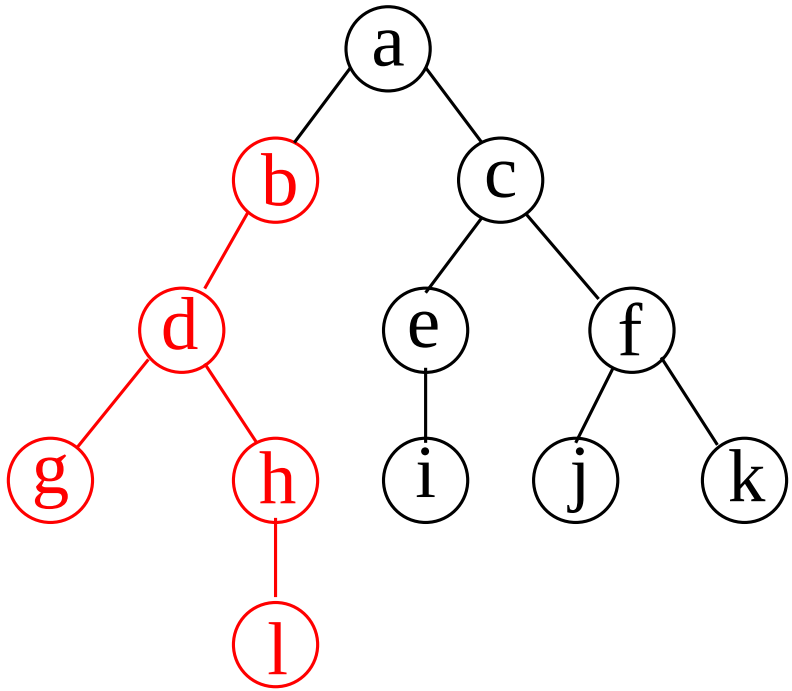


.

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

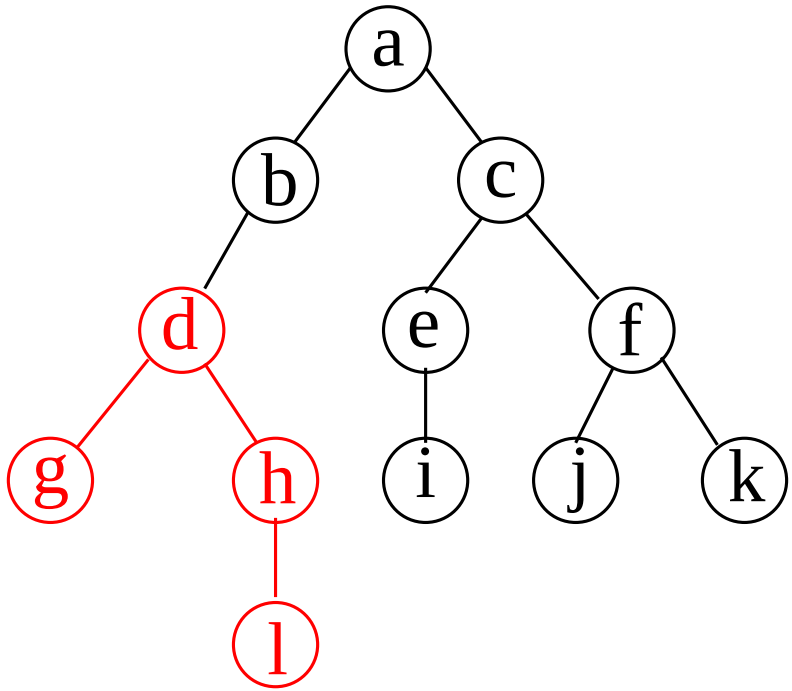


.

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

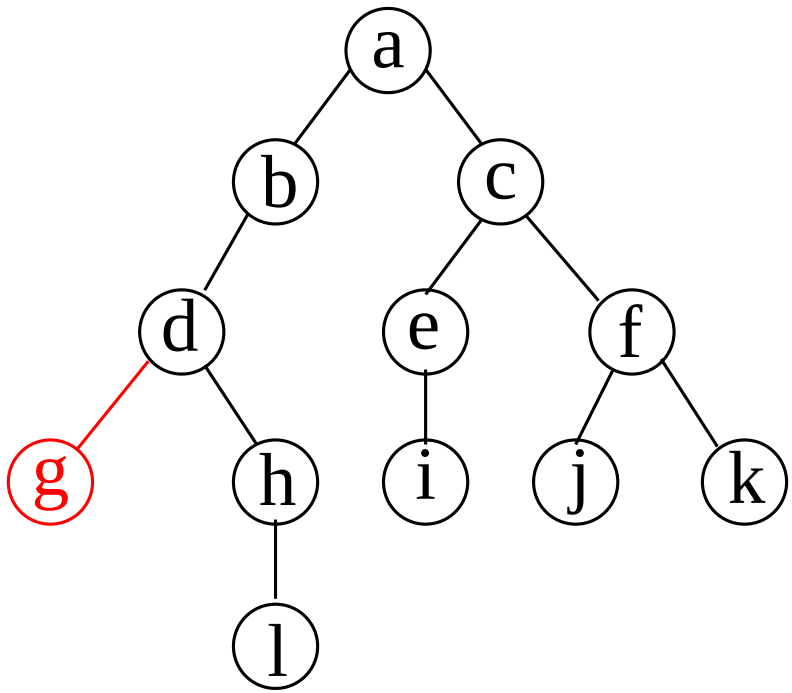


.

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

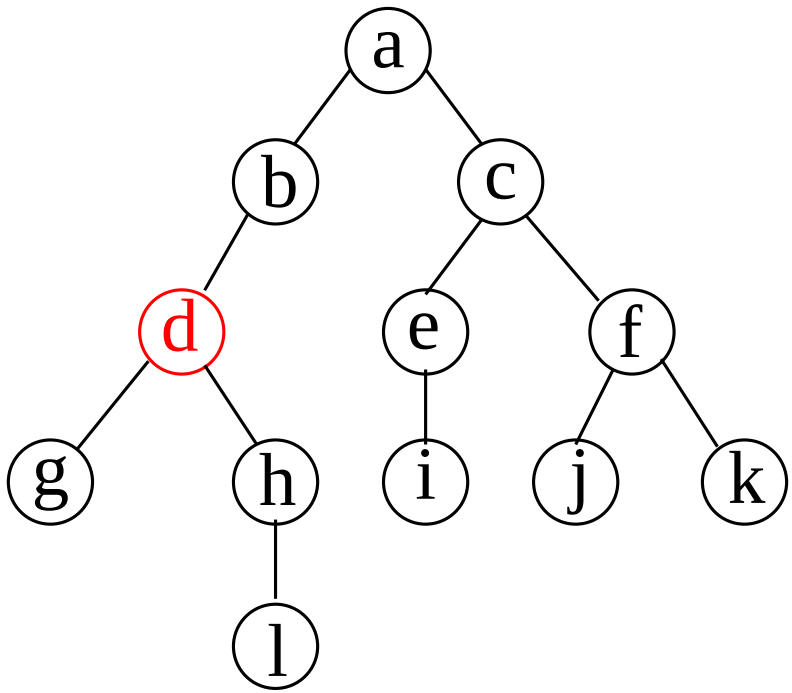


g

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

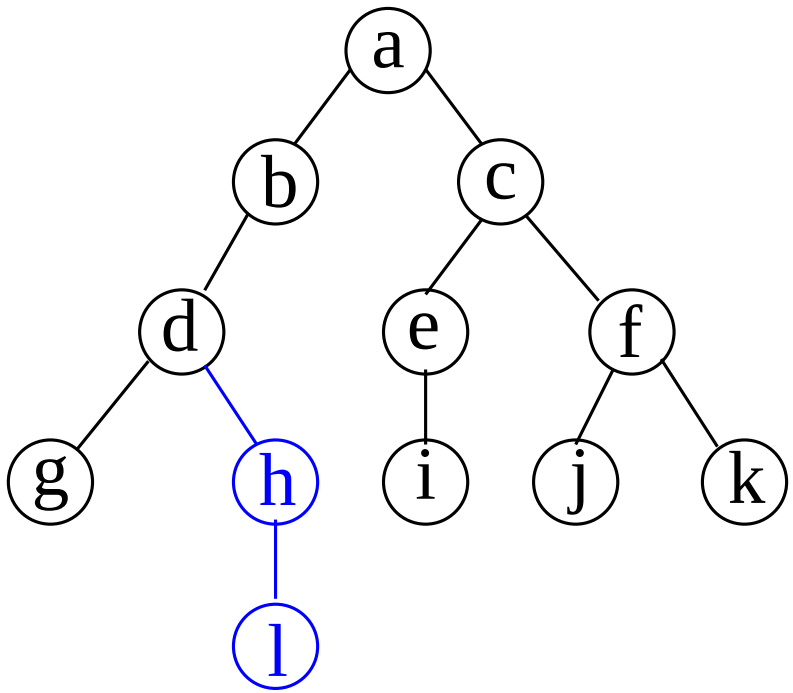


g, d

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

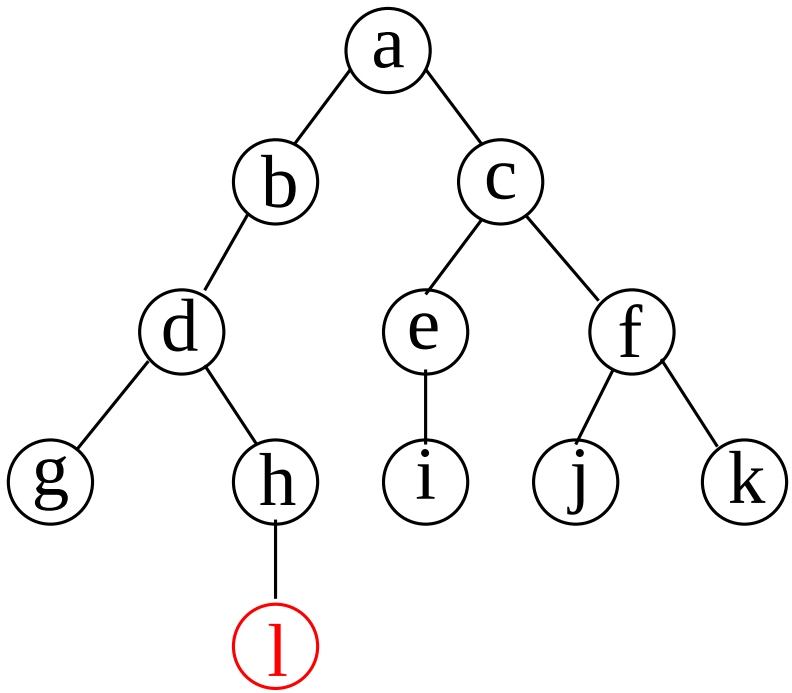


g, d

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

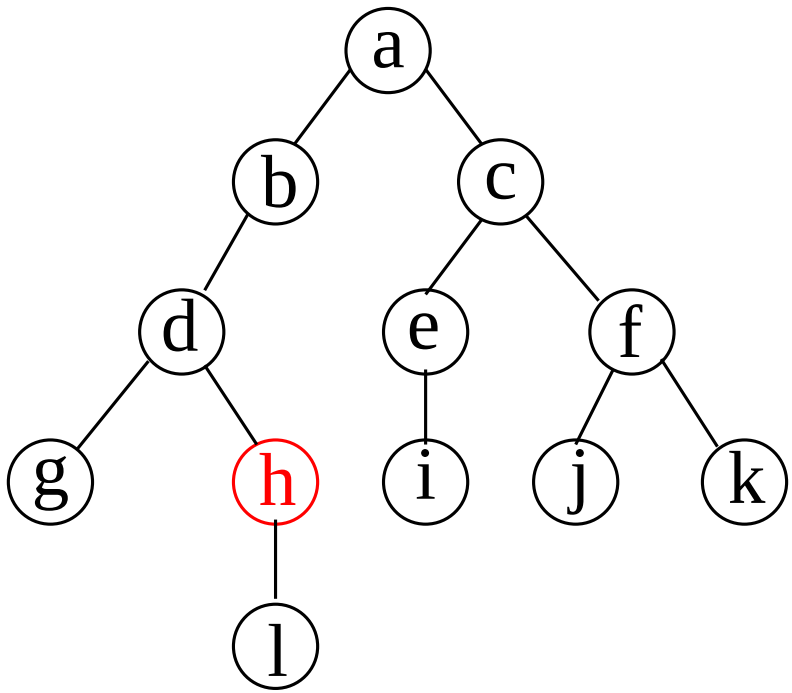


g, d, l

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

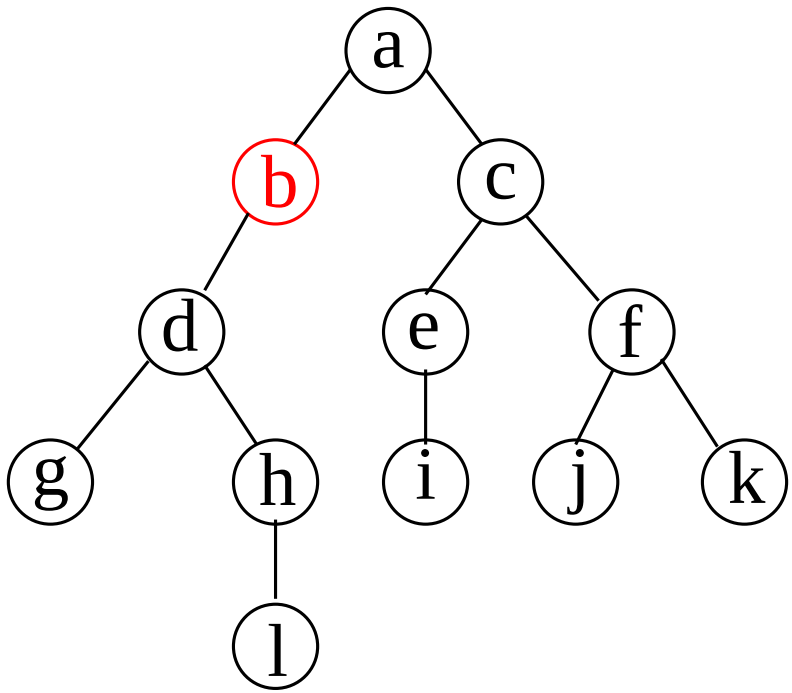


g, d, l, h

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

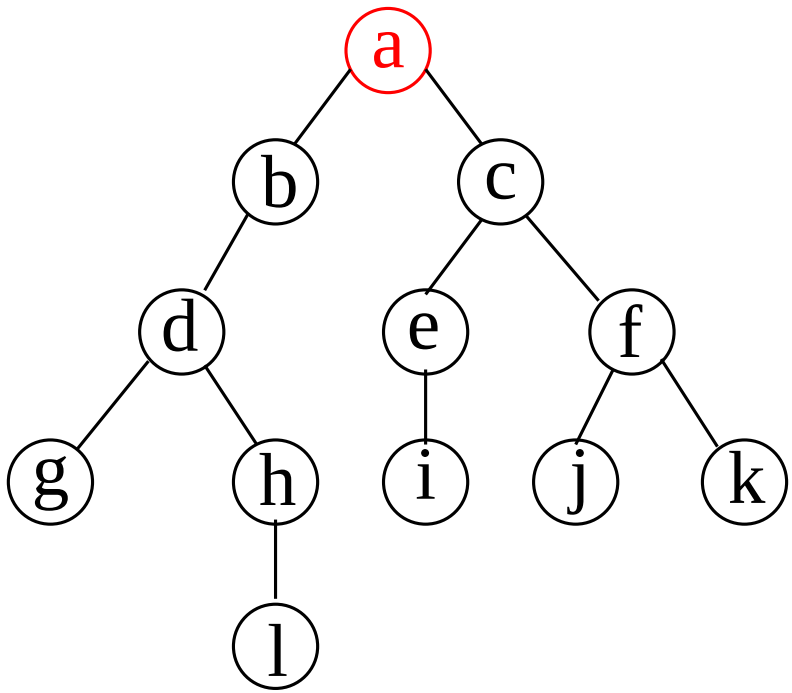


g, d, l, h, b

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

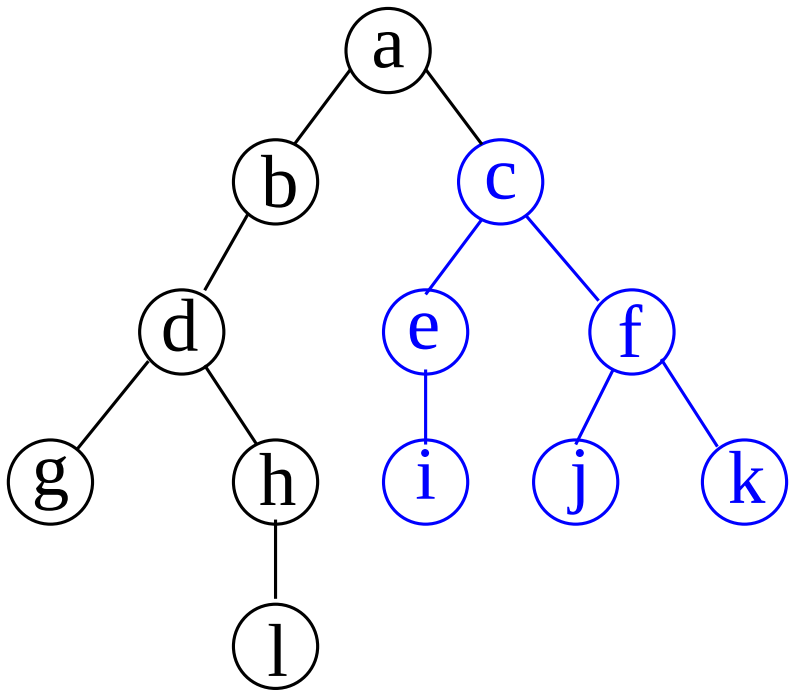


g, d, l, h, b, a

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

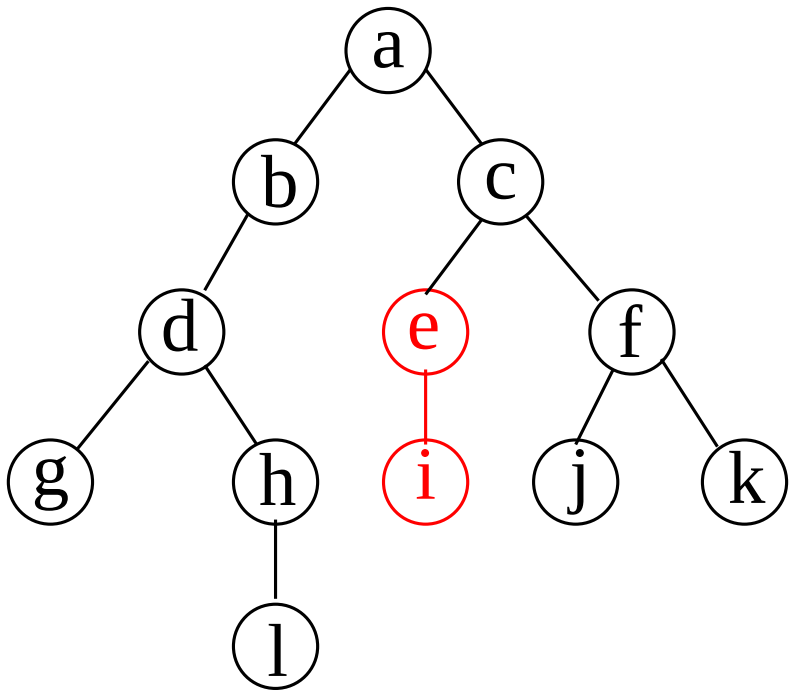


g, d, l, h, b, a

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

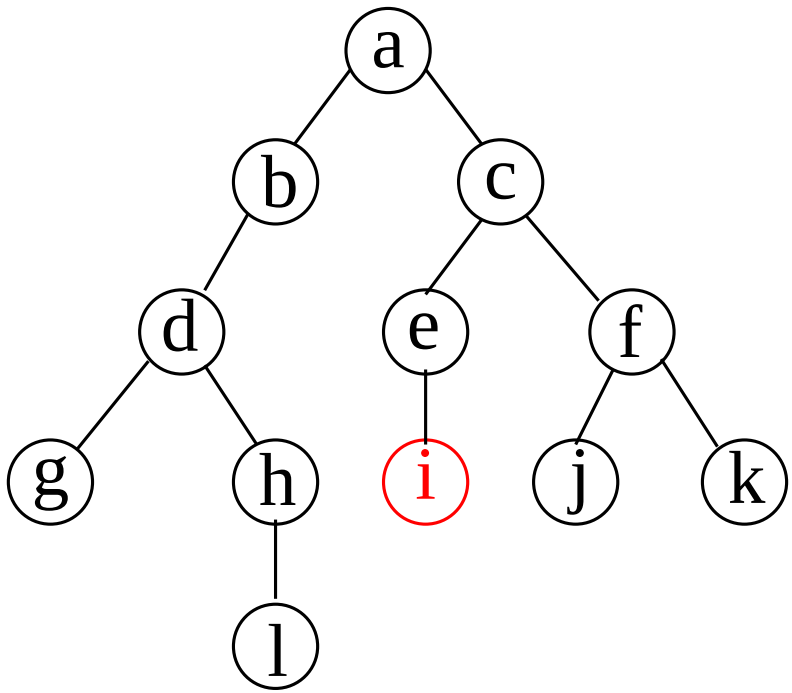


g, d, l, h, b, a

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

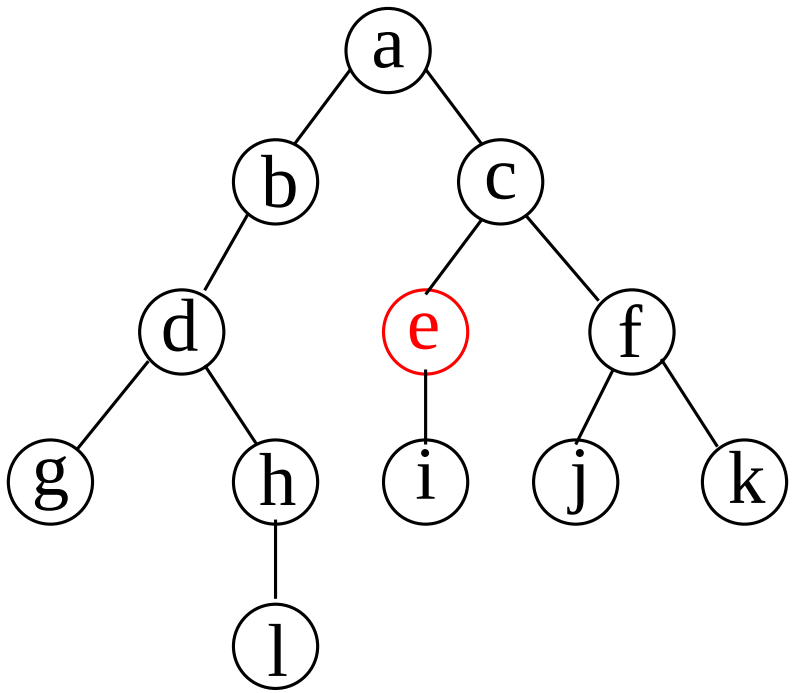


g, d, l, h, b, a, i

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

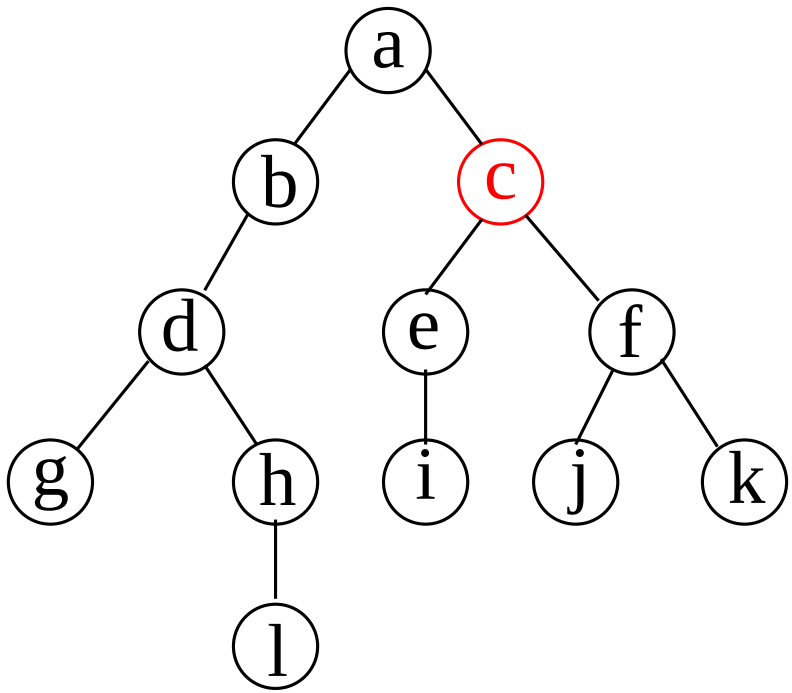


g, d, l, h, b, a, i, e

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

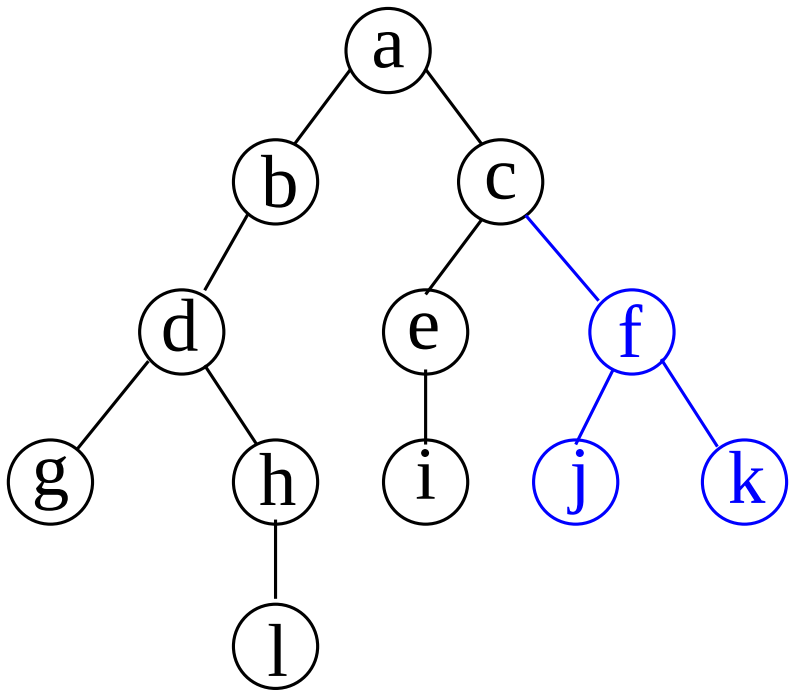


$g, d, l, h, b, a, i, e, c$

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

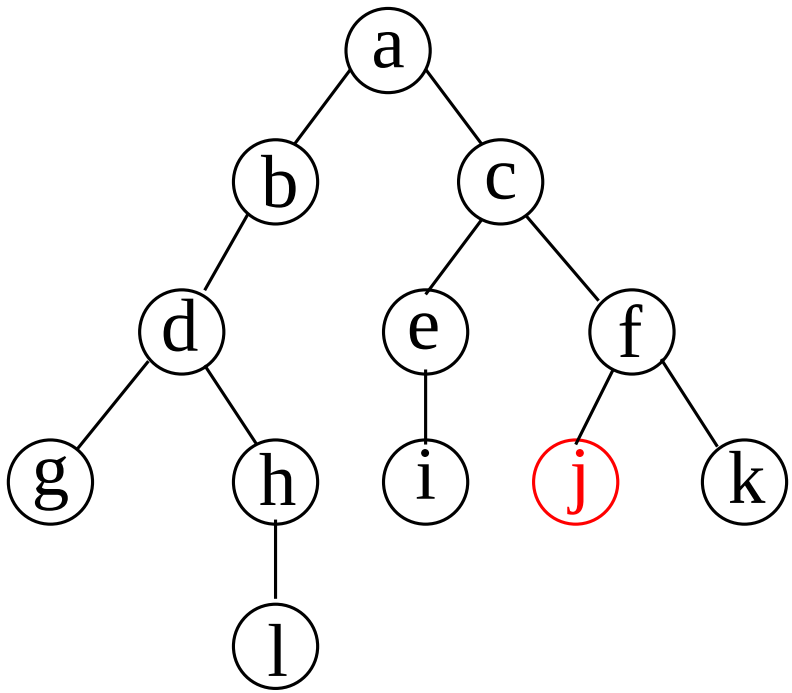


$g, d, l, h, b, a, i, e, c$

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

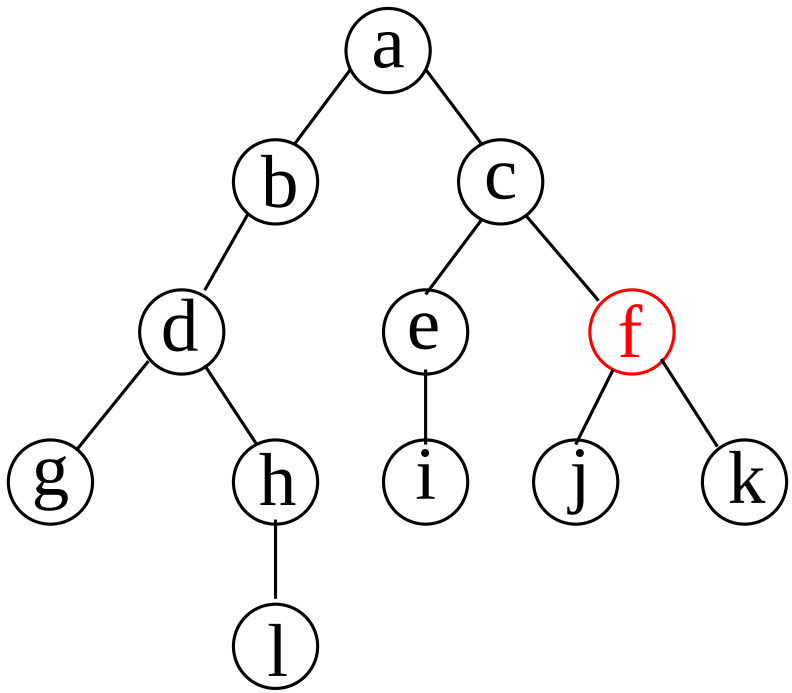


$g, d, l, h, b, a, i, e, c, j$

PARCOURS D'ARBRES

PARCOURS INFIXE

d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).

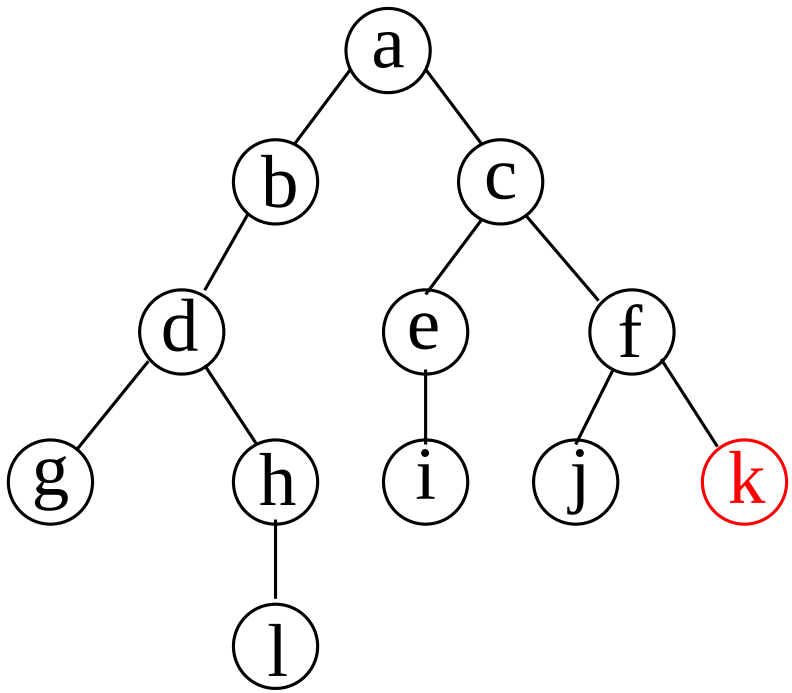


$g, d, l, h, b, a, i, e, c, j, f$

PARCOURS D'ARBRES

PARCOURS INFIXE

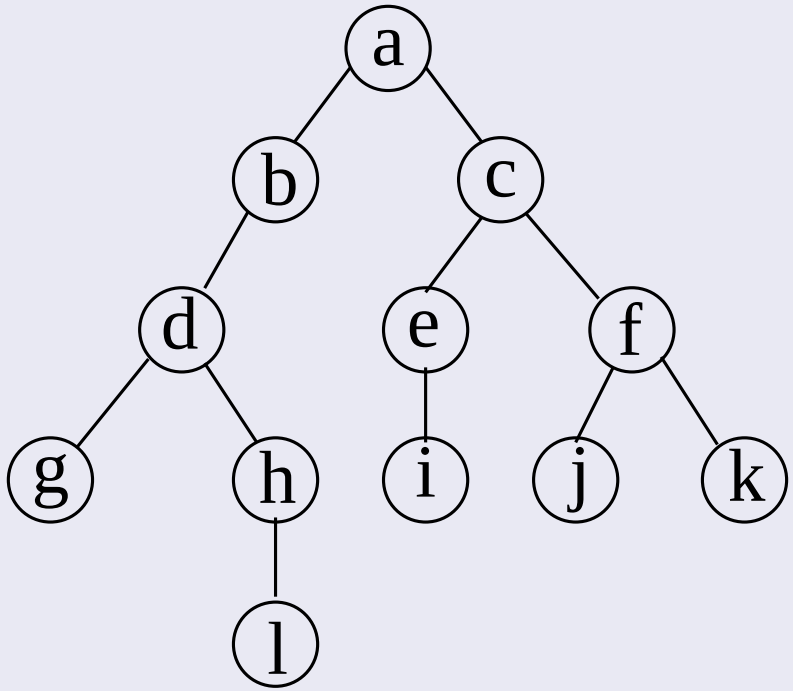
d'abord, de manière récursive, le sous-arbre de gauche, puis la racine, et enfin le sous-arbre de droit (Si un sommet n'a qu'un seul fils : sous-arbre de droite vide).



$g, d, l, h, b, a, i, e, c, j, f, k$

PARCOURS EN LARGEUR

parcours en largeur : parcours des noeuds de l'arbre pointé par niveau croissant.



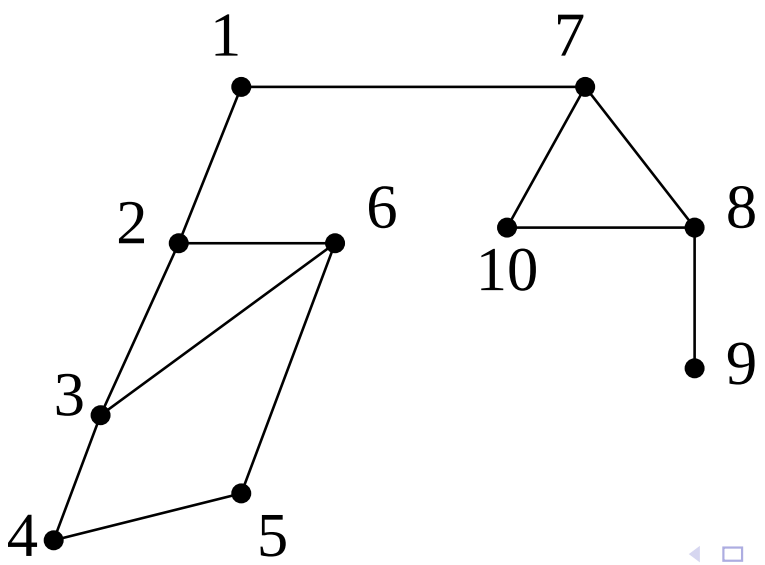
$a, b, c, \ldots, k, l$.

PARCOURS D'ARBRES

REMARQUE

Soit $G = (V, E)$ un graphe (orienté ou non) simple et connexe.
Un parcours en profondeur de G est défini récursivement.

Sélectionner un sommet v_0.
A l'étape $k \geq 1$, choisir un voisin de v_{k-1} qui n'a pas encore été sélectionné.
Si un tel voisin n'existe pas, on cherche dans l'ordre, un voisin non sélectionné de $v_{k-2}, \ldots, v_0$.



Isomorphismes de graphes

Définition

Soient $G_i = (V_i, E_i)$, $i = 1, 2$, deux digraphes.

$f : V_1 \to V_2$ est un homomorphisme de G_1 dans G_2 si

$$(x, y) \in E_1 \Rightarrow (f(x), f(y)) \in E_2$$

Définition

Soient $G_i = (V_i, E_i)$, $i = 1, 2$, deux graphes non orientés.

$f : V_1 \to V_2$ est un homomorphisme de G_1 dans G_2 si

$$\{x, y\} \in E_1 \Rightarrow \{f(x), f(y)\} \in E_2).$$

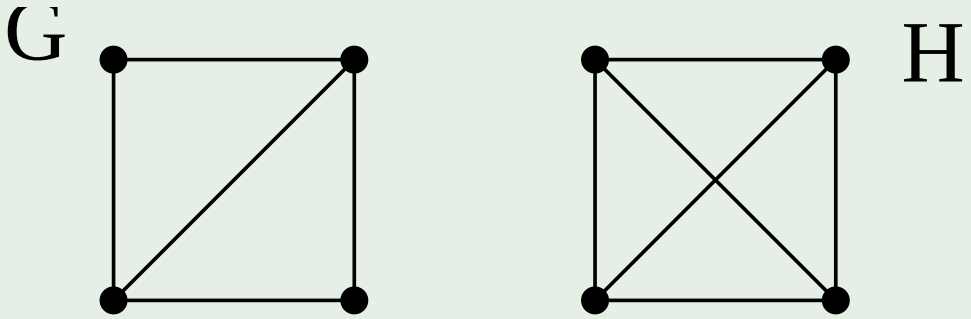
Isomorphismes de graphes

Remarque

La composée d'homomorphismes est encore un homomorphisme.

Exemple

Homomorphisme de G dans H



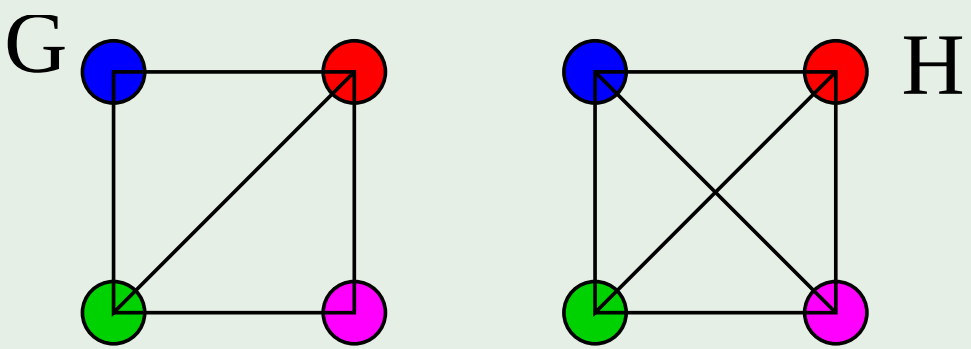
Isomorphismes de graphes

Remarque

La composée d'homomorphismes est encore un homomorphisme.

Exemple

Homomorphisme de G dans H



Homomorphisme de G ds $H \not\Rightarrow$ homomorphisme de H ds G.

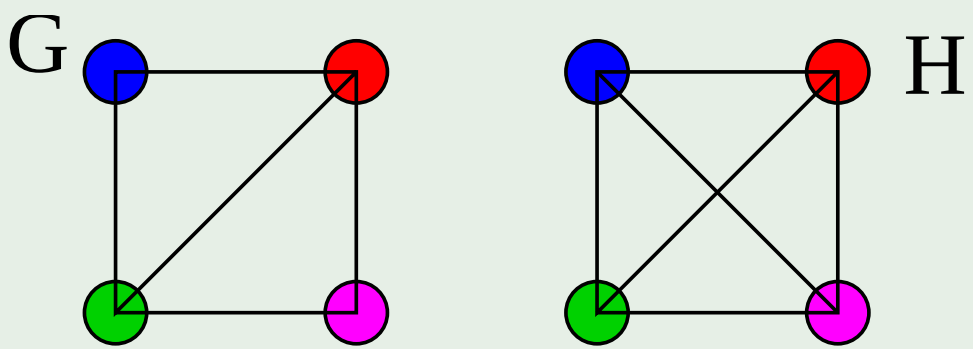
Isomorphismes de graphes

Remarque

La composée d'homomorphismes est encore un homomorphisme.

Exemple

Homomorphisme de G dans H

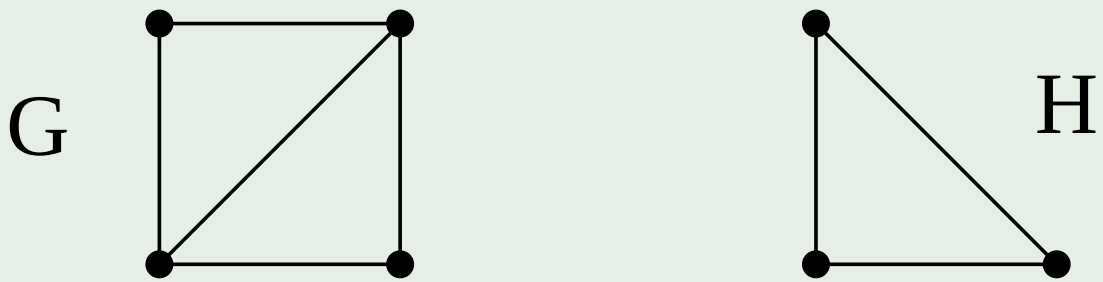


Homomorphisme de G ds $H \not\Rightarrow$ homomorphisme de H ds G.

Isomorphismes de graphes

Exemple

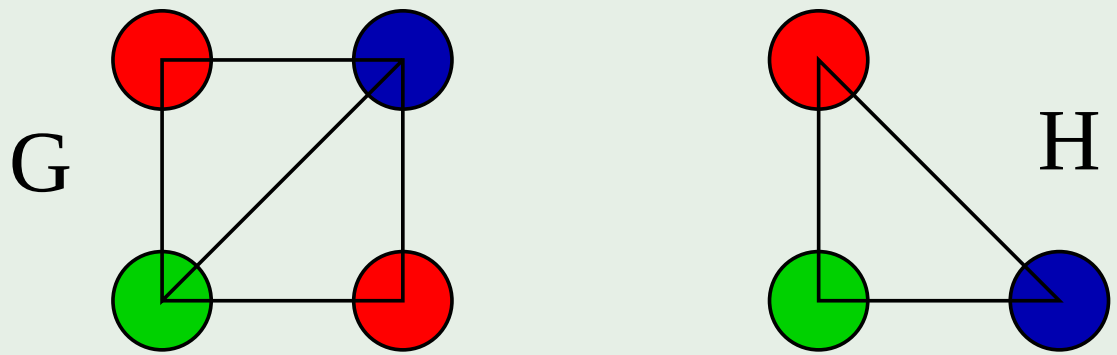
Homomorphisme non injectif de G dans H



Isomorphismes de graphes

Exemple

Homomorphisme non injectif de G dans H



Isomorphismes de graphes

Définition

2 digraphes (resp. 2 graphes non orientés) $G_i = (V_i, E_i)$, $i = 1, 2$, sont isomorphes si $\exists$ bijection $f : V_1 \to V_2$ t.q.

$$(x, y) \in E_1 \Leftrightarrow (f(x), f(y)) \in E_2$$

(resp. telle que $\{x, y\} \in E_1 \Leftrightarrow \{f(x), f(y)\} \in E_2$).

Définition

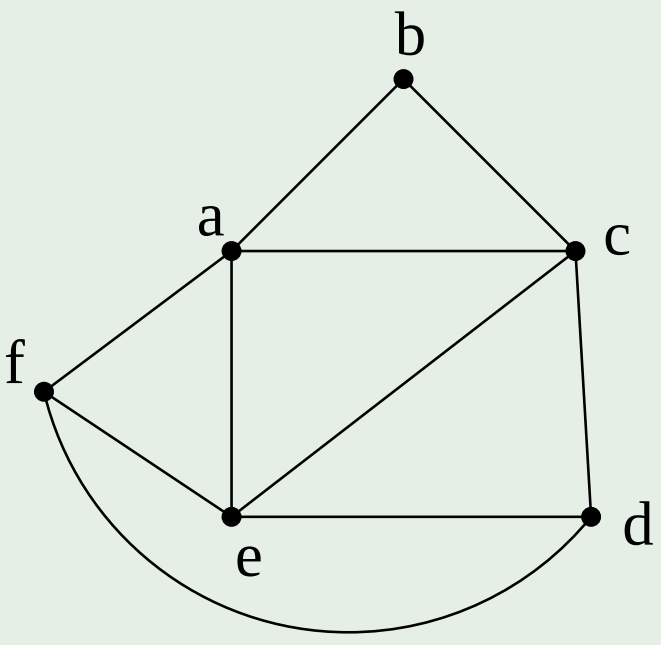
2 multi-graphes $G_i = (V_i, E_i)$, $i = 1, 2$, sont isomorphes si $\exists$ bijection $f : V_1 \to V_2$ t.q. (x, y) arc de multiplicité k de G_1 SSI $(f(x), f(y))$ arc de multiplicité k de G_2.

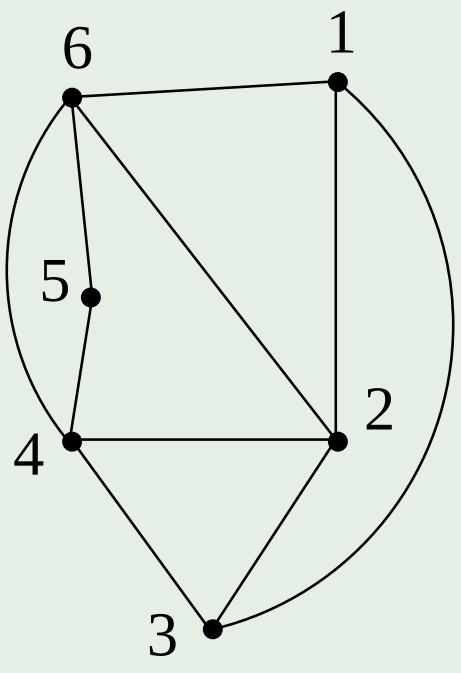
Remarque

Si f est un isomorphisme, f^{-1} aussi.

Isomorphismes de graphes

Exemple

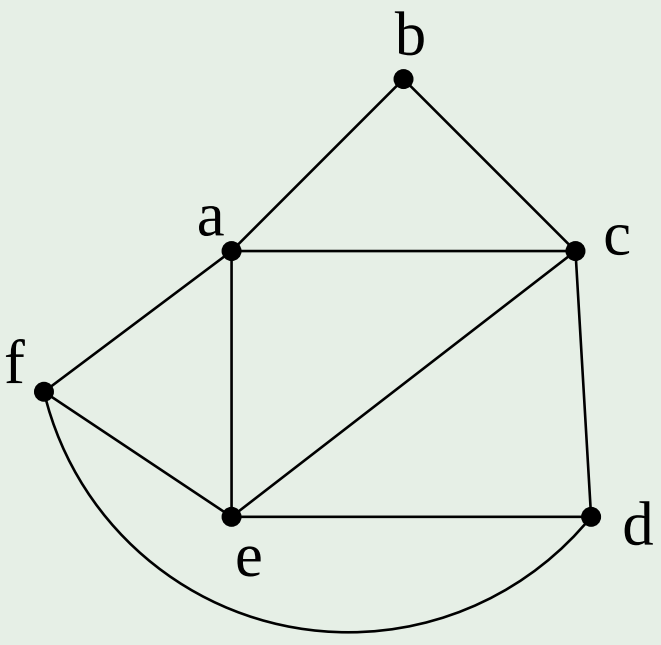


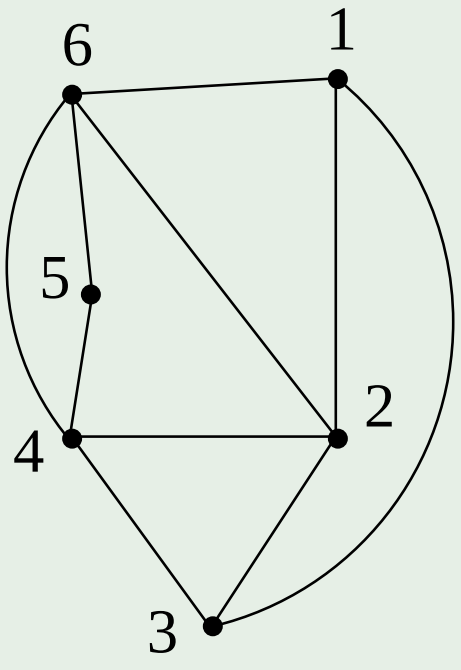


$\varphi : a \mapsto 4,\ b \mapsto 5, c \mapsto 6, d \mapsto 1,\ e \mapsto 2,\ f \mapsto 1.$

Isomorphismes de graphes

Exemple





$\varphi : a \mapsto 4,\ b \mapsto 5, c \mapsto 6, d \mapsto 1,\ e \mapsto 2,\ f \mapsto 1.$

Isomorphismes de graphes

Définition

Soit $G = (V, E)$ un graphe (orienté ou non). Un automorphisme de G est un isomorphisme de G dans G.

$Aut(G)$: groupe des automorphismes de G muni de la loi de composition d'applications

sous-groupe du groupe symétrique $\mathcal{S}_n$ des permutations de $n = \#V$ éléments.

Un graphe pour lequel $Aut(G)$ est réduit à l'identité id_V est asymétrique

Exemple

$Aut(K_n) = \mathcal{S}_n$.

Proposition

Soient G un graphe (simple non orienté) et φ un automorphisme de G. Pour tous sommets u, v, on a

- $\deg(u) = \deg(\varphi(u))$,
- $\mathrm{d}(u, v) = \mathrm{d}(\varphi(u), \varphi(v))$.

Définition

Si deux graphes $G_i = (V_i, E_i)$, $i = 1, 2$ ont leurs sommets pondérés par $p_i : V_i \to \Sigma$, la définition d'un isomorphisme $f : V_1 \to V_2$ doit naturellement s'étendre en respectant

$$p_1(v) = p_2(f(v)),\ \forall v \in V_1.$$

DÉFINITIONS

Alphabet $\Sigma = \{a, b\}$
Mots : aa, bba, b, $abbbaabaa$ (suites finies de symboles).
Arbre lexicographique : arbre binaire infini,
ses noeuds en bijection avec les mots sur $\{a, b\}$.

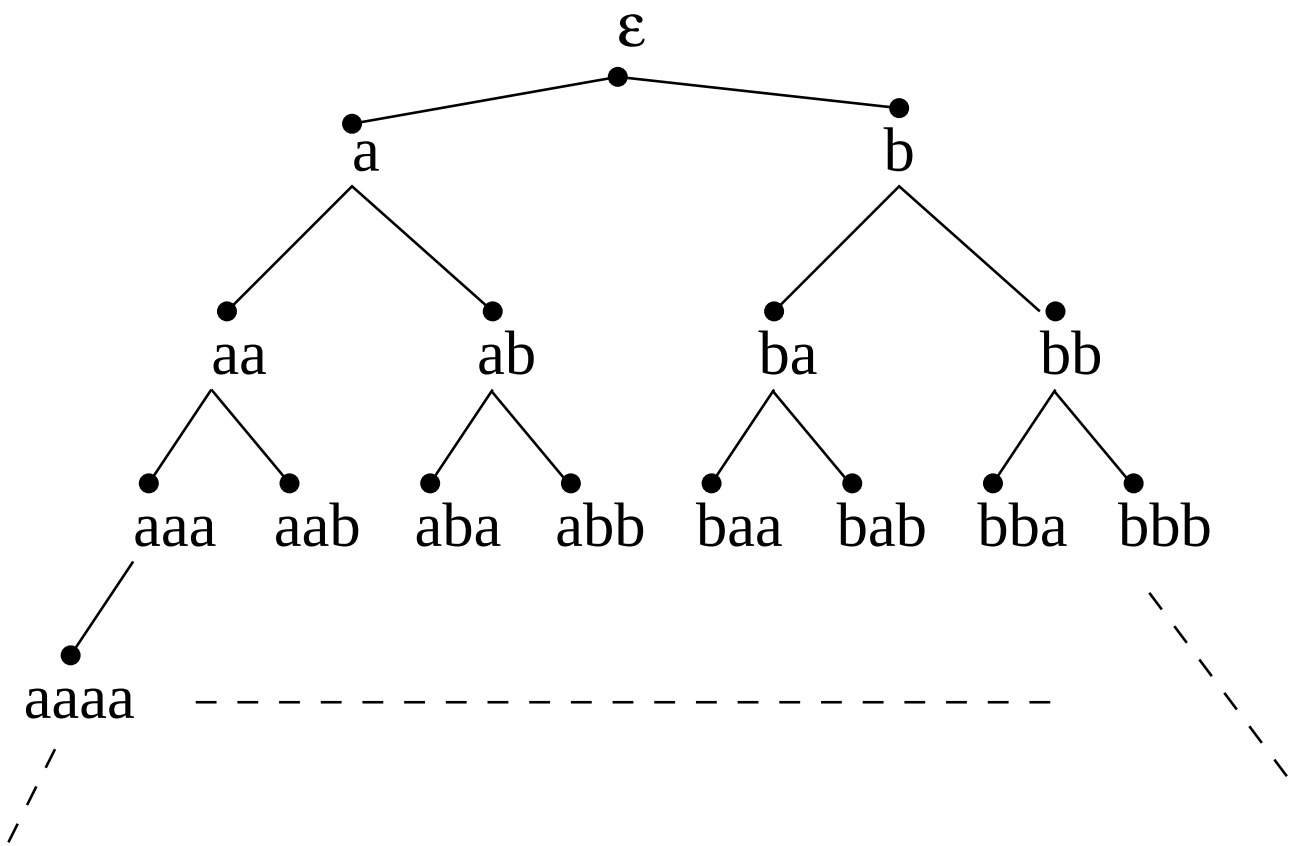
Si un noeud est en bijection avec le mot m :

- son fils de gauche est en bijection avec ma
- son fils de droite est en bijection avec mb

La racine de l'arbre correspond au mot vide : ε.

Cet arbre possède exactement 2^i noeuds de niveau i, les mots de longueur i : $\underbrace{a \cdots aa}_{i\times}, a \cdots ab, \ldots, b \cdots ba, \underbrace{b \cdots bb}_{i\times}$.

ARBRES INFINIS ET ISOMORPHISME

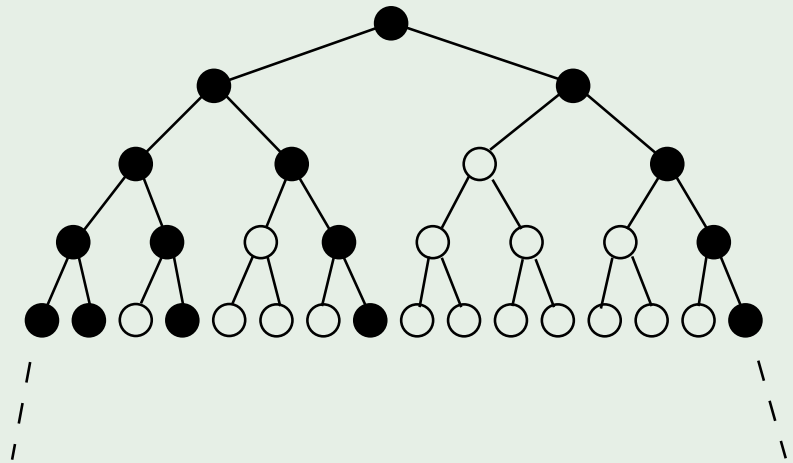


Arbres infinis et isomorphisme

Soit un ensemble L de mots écrits sur $\{a, b\}$.
p_L : à un mot m associe 1 (resp. 0) si $m \in L$ (resp. $m \notin L$).
La pondération est un codage définissant le dictionnaire des mots de L.

UN LANGAGE ET L'ARBRE PONDÉRÉ

L formé des mots commençant par un nombre arbitraire de a (éventuellement aucun) et suivi par un nombre arbitraire de b (éventuellement aucun), l'arbre pondéré A_L

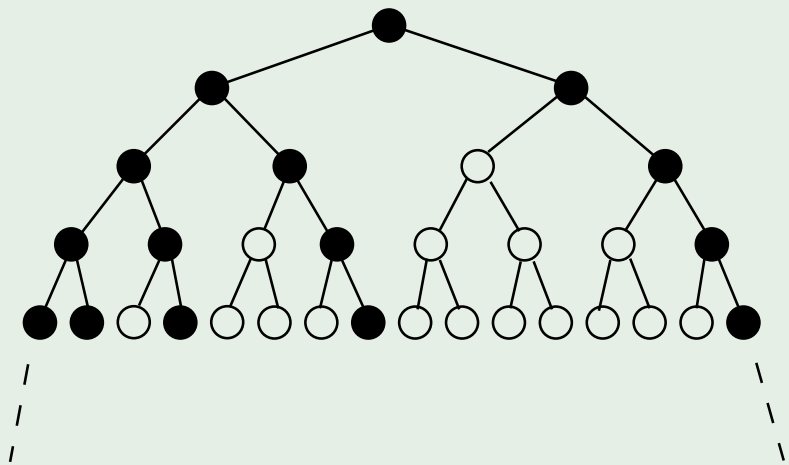


$$\varepsilon, a, b, aa, ab, bb, aaa, aab, abb, bbb, \ldots$$

Arbres infinis et isomorphisme

A_m : sous-arbre obtenu en considérant comme nouvelle racine le noeud m et en ne conservant dans A_m que les descendants de m

Arbre régulier

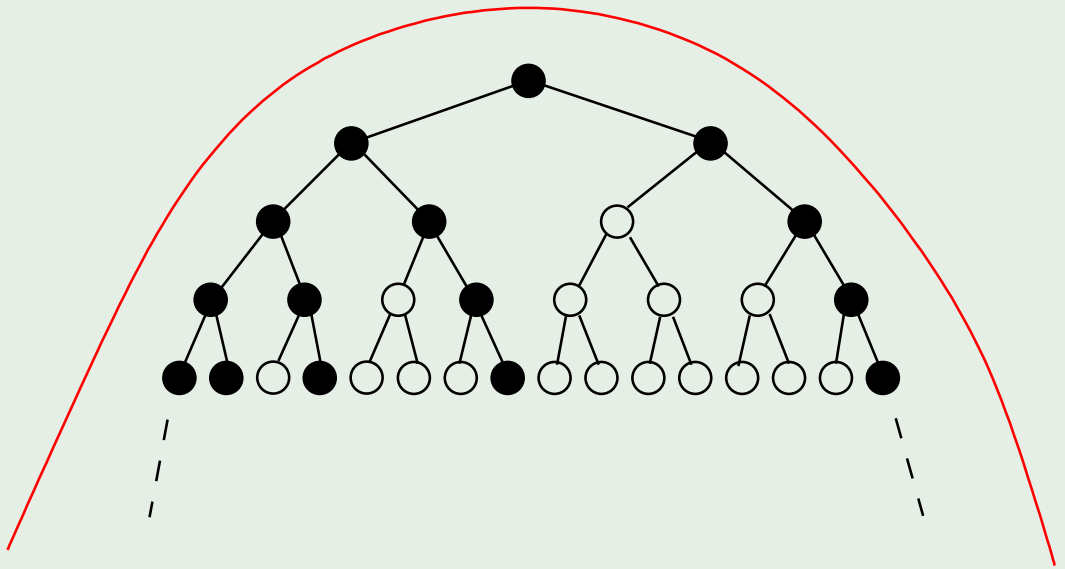


l'arbre A_L ne possède, à isomorphisme près, que 3 sous-arbres non isomorphes (par exemple, A_L lui-même, A_b et A_{ba})

nombre fini de sous-arbres non isomorphes : arbre régulier.

Arbres infinis et isomorphisme

Arbre régulier

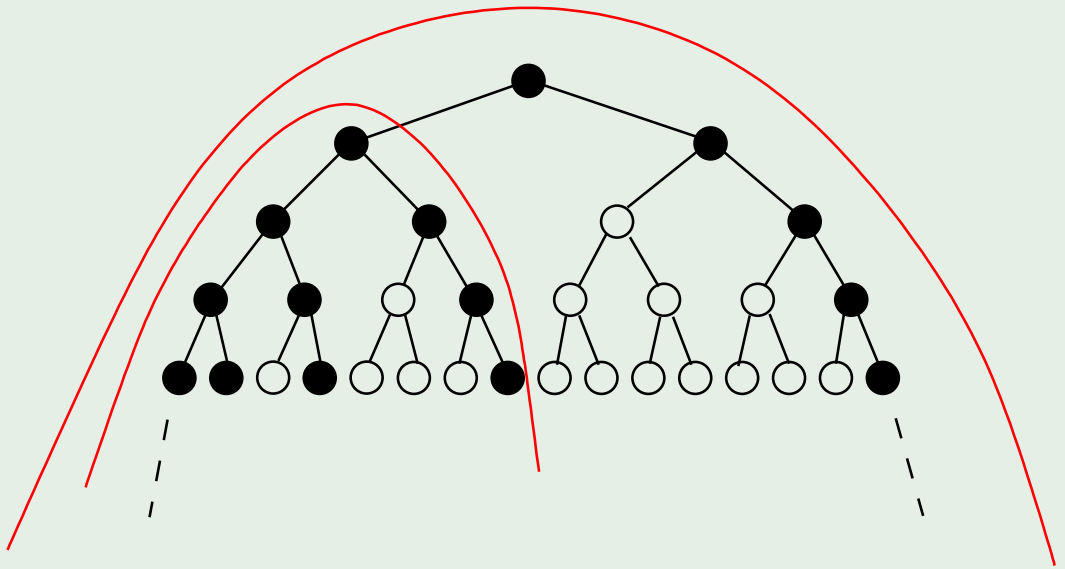


l'arbre A_L ne possède, à isomorphisme près, que 3 sous-arbres non isomorphes (par exemple, A_L lui-même, A_b et A_{ba})

nombre fini de sous-arbres non isomorphes : arbre régulier.

Arbres infinis et isomorphisme

Arbre régulier

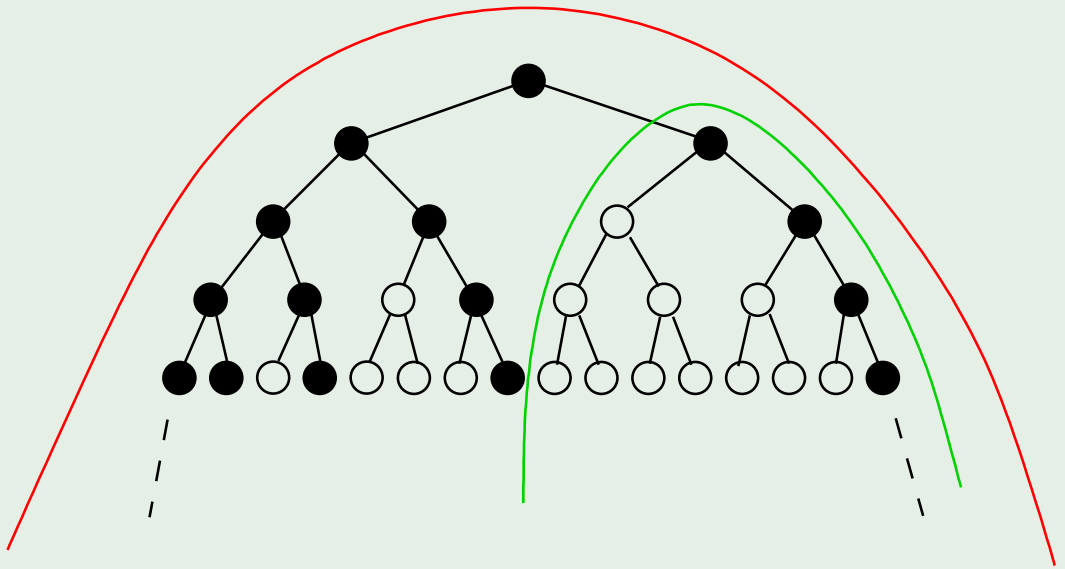


l'arbre A_L ne possède, à isomorphisme près, que 3 sous-arbres non isomorphes (par exemple, A_L lui-même, A_b et A_{ba})

nombre fini de sous-arbres non isomorphes : arbre régulier.

Arbres infinis et isomorphisme

Arbre régulier

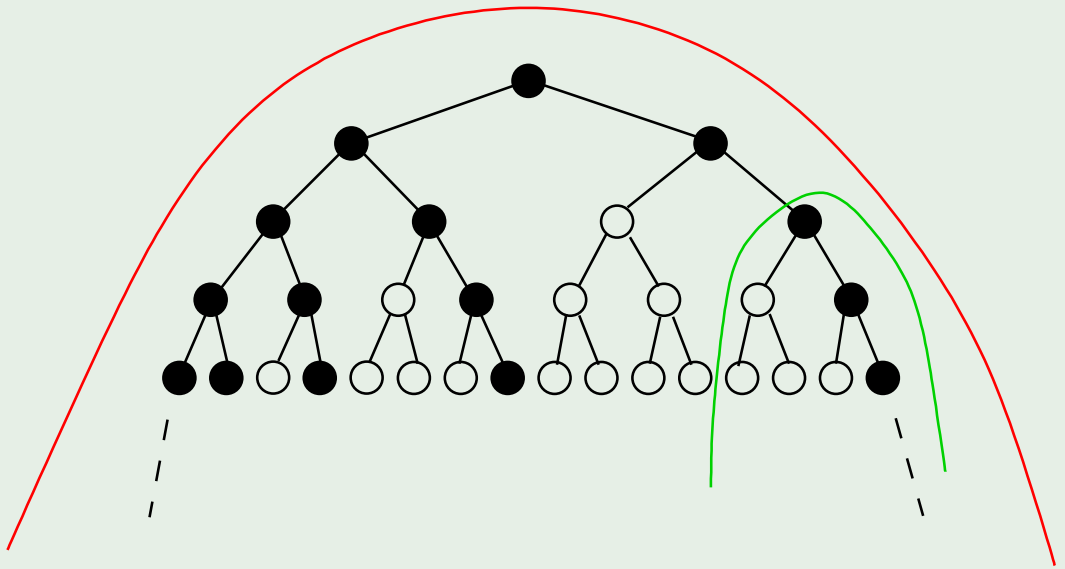


l'arbre A_L ne possède, à isomorphisme près, que 3 sous-arbres non isomorphes (par exemple, A_L lui-même, A_b et A_{ba})

nombre fini de sous-arbres non isomorphes : arbre régulier.

Arbres infinis et isomorphisme

Arbre régulier

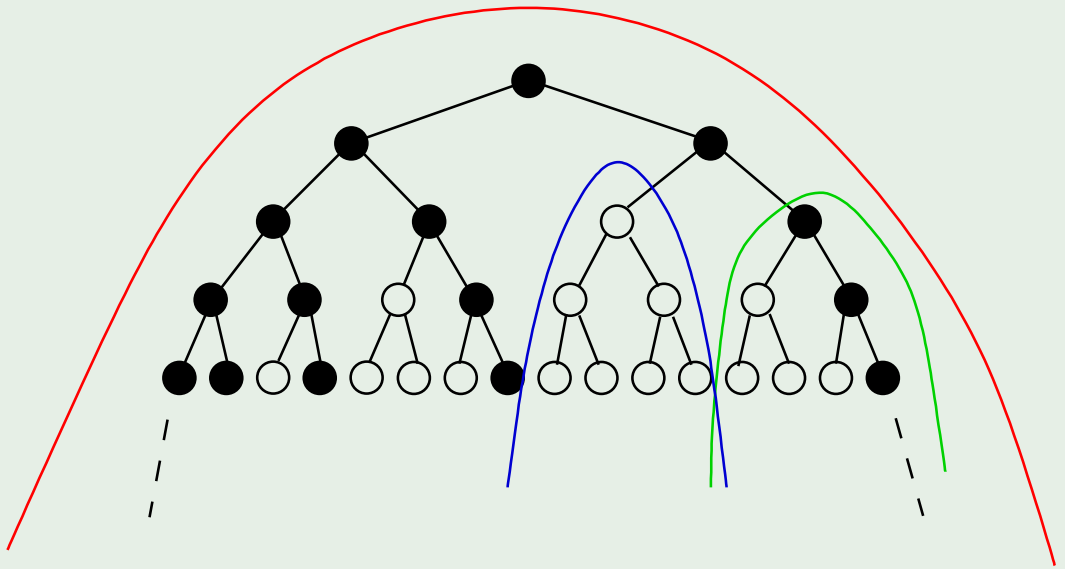


l'arbre A_L ne possède, à isomorphisme près, que 3 sous-arbres non isomorphes (par exemple, A_L lui-même, A_b et A_{ba})

nombre fini de sous-arbres non isomorphes : arbre régulier.

Arbres infinis et isomorphisme

Arbre régulier



l'arbre A_L ne possède, à isomorphisme près, que 3 sous-arbres non isomorphes (par exemple, A_L lui-même, A_b et A_{ba})

nombre fini de sous-arbres non isomorphes : arbre régulier.

Graphes hamiltoniens

Rappel

Graphe eulérien : circuit passant 1 ! fois par chaque arête.

Question de Sir W. R. Hamilton

Existence d'un Circuit passant 1 ! fois par chaque sommet.
A ce jour, pas de méthode efficace (pb. NP-complet) :
passer en revue toutes les permutations des n sommets...

Définition

Un chemin (resp. circuit) hamiltonien de G : passe une et une seule fois par chaque sommet de G.

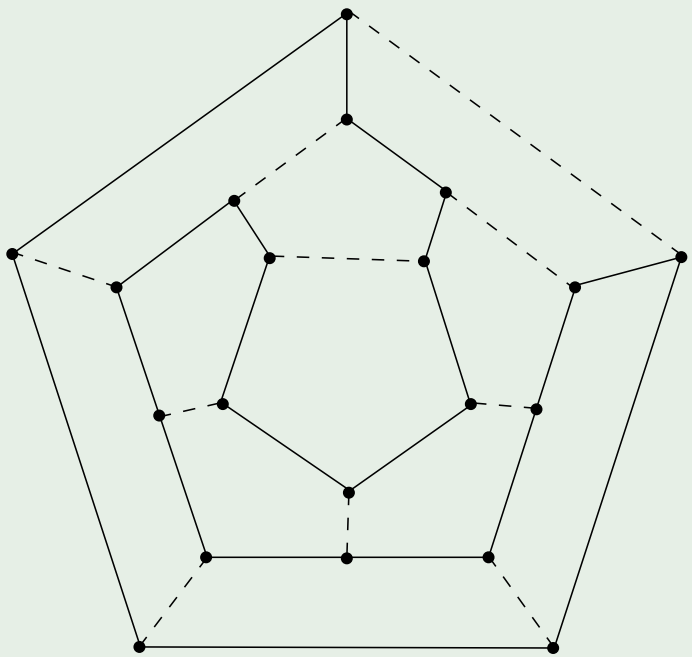
Un graphe hamiltonien : graphe possédant un circuit hamiltonien

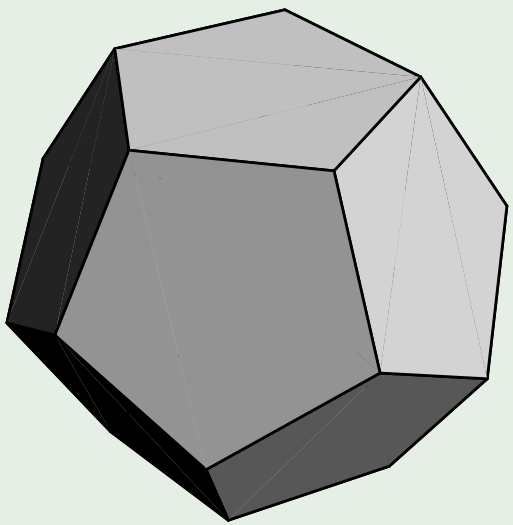
CAS PATHOLOGIQUES

un graphe restreint à une unique arête $\{a, b\}$ (resp. restreint à un unique sommet) est hamiltonien : circuit $\{a, b\}$, $\{b, a\}$.
Par contre, un arbre contenant 3 sommets au moins n'est jamais hamiltonien.

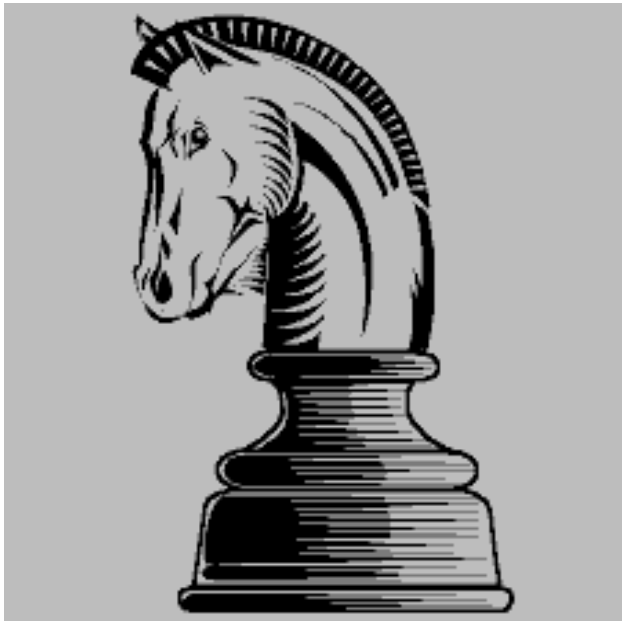
Graphes hamiltoniens

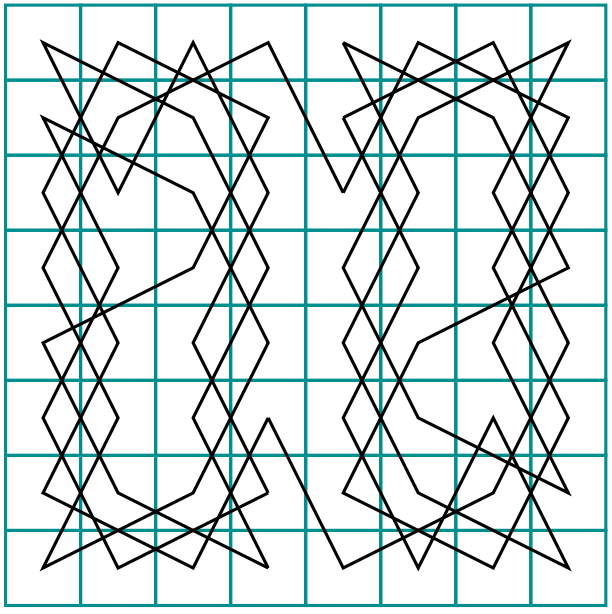
Exemple





GRAPHES HAMILTONIENS





Graphes hamiltoniens

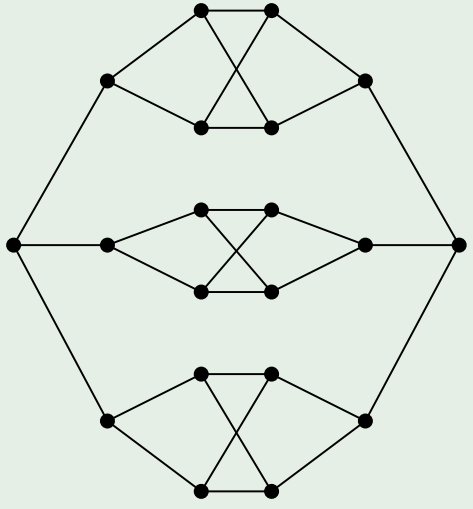
Condition nécessaire pour qu'un graphe soit hamiltonien.

Proposition

Si $G = (V, E)$ est un graphe (simple et non orienté) hamiltonien, alors pour tout ensemble non vide $S \subseteq V$, le nombre de composantes connexes de $G - S$ est $\leq \#S$.

Exemple

Ce graphe est-il hamiltonien ?



Graphes hamiltoniens

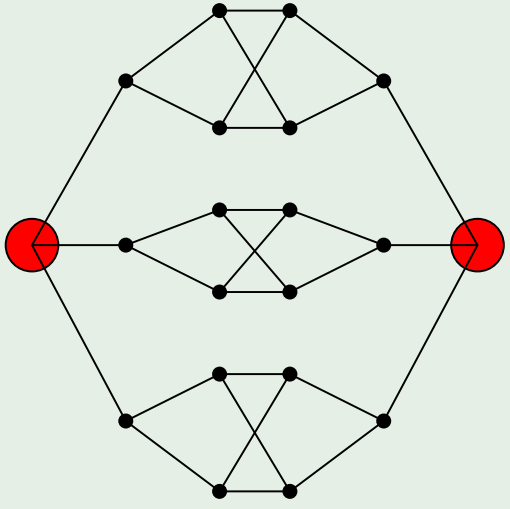
Condition nécessaire pour qu'un graphe soit hamiltonien.

Proposition

Si $G = (V, E)$ est un graphe (simple et non orienté) hamiltonien, alors pour tout ensemble non vide $S \subseteq V$, le nombre de composantes connexes de $G - S$ est $\leq \#S$.

Exemple

Ce graphe est-il hamiltonien ?



Graphes hamiltoniens

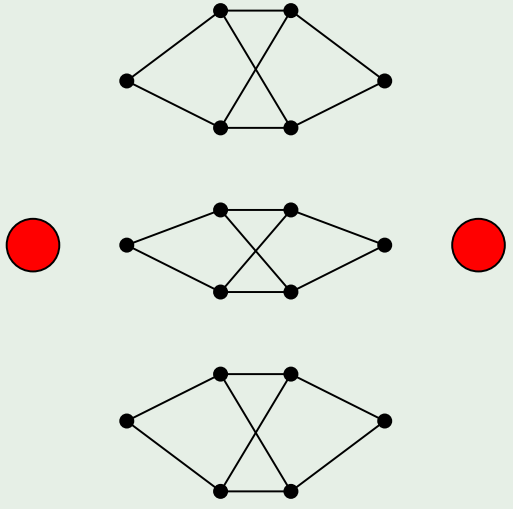
Condition nécessaire pour qu'un graphe soit hamiltonien.

Proposition

Si $G = (V, E)$ est un graphe (simple et non orienté) hamiltonien, alors pour tout ensemble non vide $S \subseteq V$, le nombre de composantes connexes de $G - S$ est $\leq \#S$.

Exemple

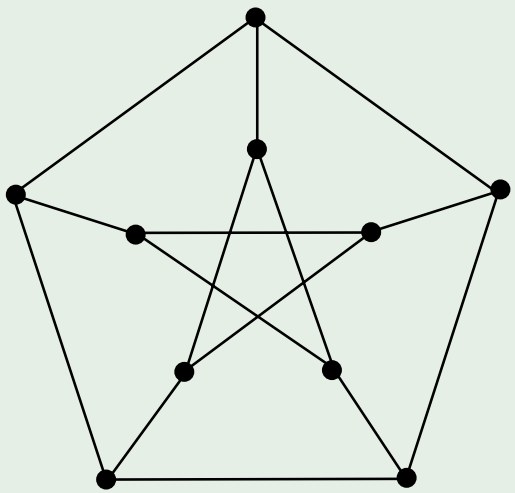
Ce graphe est-il hamiltonien ?



Graphes hamiltoniens

La condition n'est PAS suffisante

Le graphe de Petersen



Graphes hamiltoniens

Un aperçu de conditions suffisantes, graphe G (simple et non orienté) ayant $n \geq 3$ sommets.

Théorème de Dirac (1952)

Si le degré de chaque sommet est $\geq n/2$, G est hamiltonien.

"premier" Théorème d'Ore (1960)

Si $\exists$ 2 sommets x et y t.q. $\deg(x) + \deg(y) \geq n$.
G est hamiltonien SSI le graphe $G + \{x, y\}$ l'est.

"deuxième" Théorème d'Ore

Si pour tout couple de sommets non adjacents (x, y), on a $\deg(x) + \deg(y) \geq n$, alors G est hamiltonien. En particulier, si $\min_{v \in V} \deg(v) \geq n/2$, alors G est hamiltonien.

THÉORÈME DE CHVÁTAL (1971)

Soit G un graphe (simple et non orienté) ayant $n \geq 3$ sommets ordonnés par degré croissant, i.e.,

$$\deg(v_1) \leq \deg(v_2) \leq \cdots \leq \deg(v_n).$$

Si, pour tout $k < n/2$, le graphe satisfait

$$\deg(v_k) \leq k \Rightarrow \deg(v_{n-k}) \geq n - k, \tag{1}$$

alors G possède un circuit hamiltonien.

REMARQUE

La condition du théorème de Chvátal peut facilement être testée pour un graphe donné. Il suffit d'ordonner la suite des degrés et de vérifier une condition combinatoire élémentaire.

Rappelons que d'un point de vue strictement logique, si pour $k < n/2$, on a $\deg(v_k) > k$, alors l'implication $\deg(v_k) \leq k \Rightarrow \deg(v_{n-k}) \geq n - k$ est toujours vraie.

A ce jour, on ne connaît pas de condition suffisante plus générale pour qu'un graphe soit hamiltonien. Dans certains cas, on peut obtenir des conditions plus fortes mais en se restreignant à des classes particulières de graphes.

THÉORÈME DE CHVÁTAL-ERDÖS

Soient G un graphe simple et non orienté ayant au moins trois sommets, $\alpha(G)$ le nombre maximal de sommets indépendants et $\kappa(G)$ la taille minimale d'un ensemble d'articulation. Si $\kappa(G) \geq \alpha(G)$, alors G est hamiltonien.

Fermeture d'un graphe

Idée...

"premier" Théorème d'Ore (1960)

Si $\exists$ 2 sommets x et y t.q. $\deg(x) + \deg(y) \geq n$.
G est hamiltonien SSI le graphe $G + \{x, y\}$ l'est.

Fermeture d'un graphe

Définition

Fermeture d'un graphe simple et non orienté $G_0 = (V_0, E_0)$.
On définit une suite finie de graphes (simples)

$$G_0, G_1, \ldots, G_i = (V_i, E_i), \ldots, G_k$$

Pour tout i, on ajoute à G_i une arête comme suit :

$$G_{i+1} = G_i + \{u, v\}$$

où u et v sont t.q. $\{u, v\} \notin E_i$ et

$$\deg_{G_i}(u) + \deg_{G_i}(v) \geq \#V$$

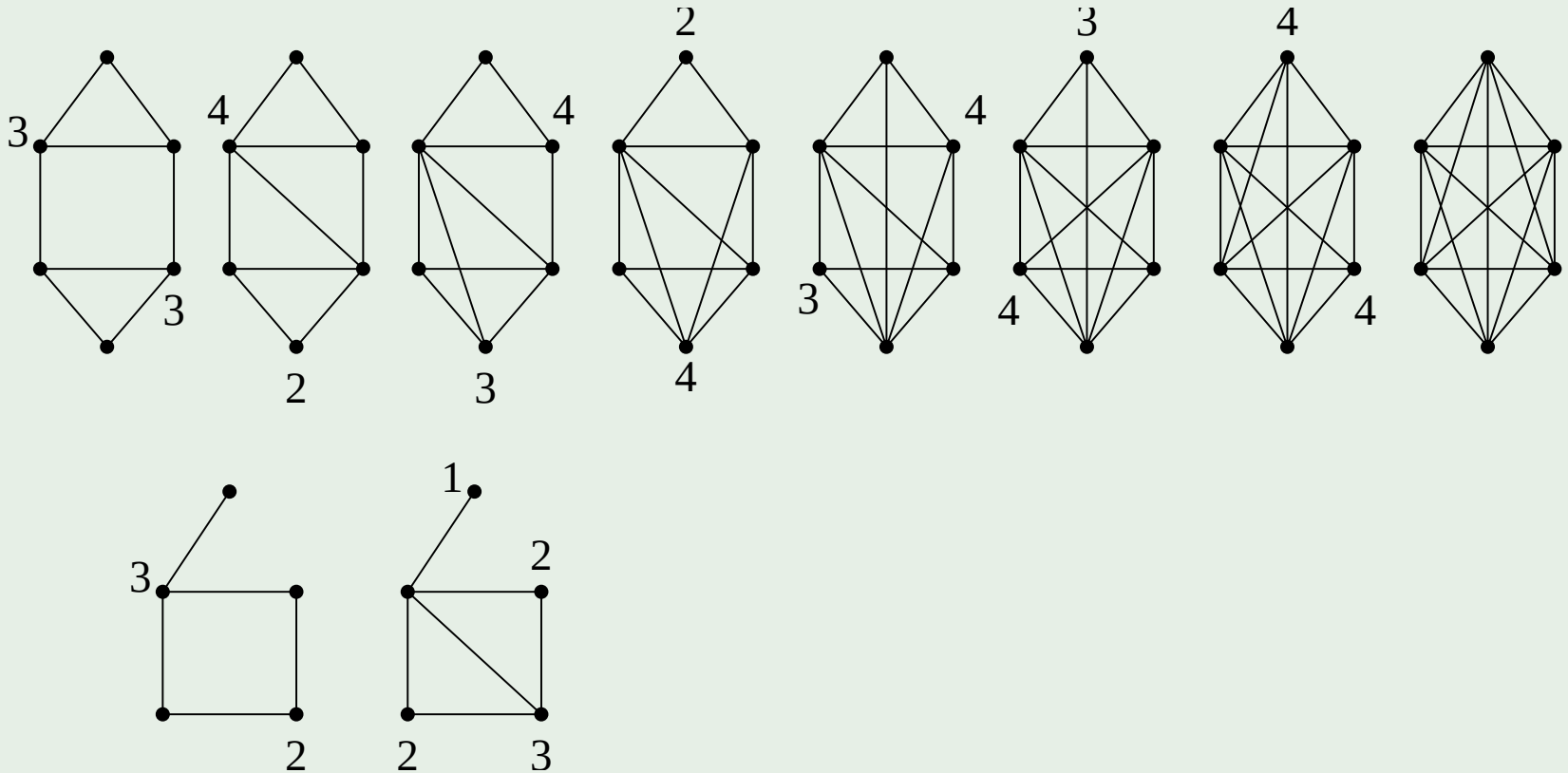
où $\deg_{G_i}$ désigne le degré d'un sommet dans le graphe G_i.

La procédure s'arrête à G_k si pour tous sommets u, v, soit $\{u, v\} \in E_k$, soit $\deg_{G_k}(u) + \deg_{G_k}(v) < \#V$.

FERMETURE D'UN GRAPHE

...Quels que soient les choix d'arêtes réalisés dans les étapes intermédiaires, on aboutit toujours au même graphe : $\mathcal{F}(G_0)$

EXEMPLE



Fermeture d'un graphe

Lemme

Pour tout graphe ayant au moins trois sommets, la fermeture d'un graphe est unique.

Supposons avoir 2 fermetures des $G \neq$:

$$H = G + \{e_1, \ldots, e_r\} \quad \text{et} \quad H' = G + \{f_1, \ldots, f_s\}$$

Thèse : $H = H'$.

$H_i := G + \{e_1, \ldots, e_i\}$ et $H_{i'} := G + \{f_1, \ldots, f_i\}$, $G = H_0 = H'_0$.
Si $H \neq H'$, supposons que $e_k = \{u, v\}$ est la première arête de H qui n'appartient pas à H'. (Ainsi, $e_1, \ldots, e_{k-1}$ sont des arêtes de H' et e_k diffère de tous les f_i.)

Fermeture d'un graphe

Puisque e_k est l'arête ajoutée à H_{k-1} pour construire H_k,

$$\deg_{H_{k-1}}(u) + \deg_{H_{k-1}}(v) \geq n.$$

H_{k-1} est un sous-graphe de H' :

$$\deg_{H'}(u) + \deg_{H'}(v) \geq n$$

donc, e_k devra aussi être ajoutée à H', contradiction !

H est un sous-graphe de H'. Par symétrie, on en tire que $H = H'$.

Fermeture d'un graphe

Théorème

Soit G un graphe (simple et non orienté) ayant au moins trois sommets.

- G est hamiltonien SSI sa fermeture $\mathcal{F}(G)$ l'est.
- Si $\mathcal{F}(G)$ est un graphe complet, alors G hamiltonien.

$\Rightarrow$ G est un sous-graphe de $\mathcal{F}(G)$.
Si G est hamiltonien, alors $\mathcal{F}(G)$ l'est aussi.

$\Leftarrow$ soit une suite de graphes $G_0 = G, \ldots, G_{k-1}, G_k = \mathcal{F}(G)$ donnant la fermeture de G.
Thm. d'Ore : si G_k est hamiltonien, alors G_{k-1} l'est aussi. De proche en proche, $G_0 = G$ est hamiltonien.

La deuxième partie : Tout graphe complet est hamiltonien, appliquer la première partie.

Fermeture d'un graphe

Remarque

La condition suffisante donnée dans le théorème précédent n'est PAS nécessaire.

graphe avec $n > 4$ sommets et de n arêtes et formant un unique circuit hamiltonien.

Chaque sommet est de degré 2, le graphe est égal à sa fermeture et bien que le graphe soit hamiltonien, la fermeture n'est pas le graphe complet K_n.

Fermeture d'un graphe

"deuxième" Théorème d'Ore

Si pour tout couple de sommets non adjacents (x, y), on a $\deg(x) + \deg(y) \geq n$, alors G est hamiltonien. En particulier, si $\min_{v \in V} \deg(v) \geq n/2$, alors G est hamiltonien.

La preuve est immédiate : $\mathcal{F}(G) = K_n$!

FERMETURE D'UN GRAPHE

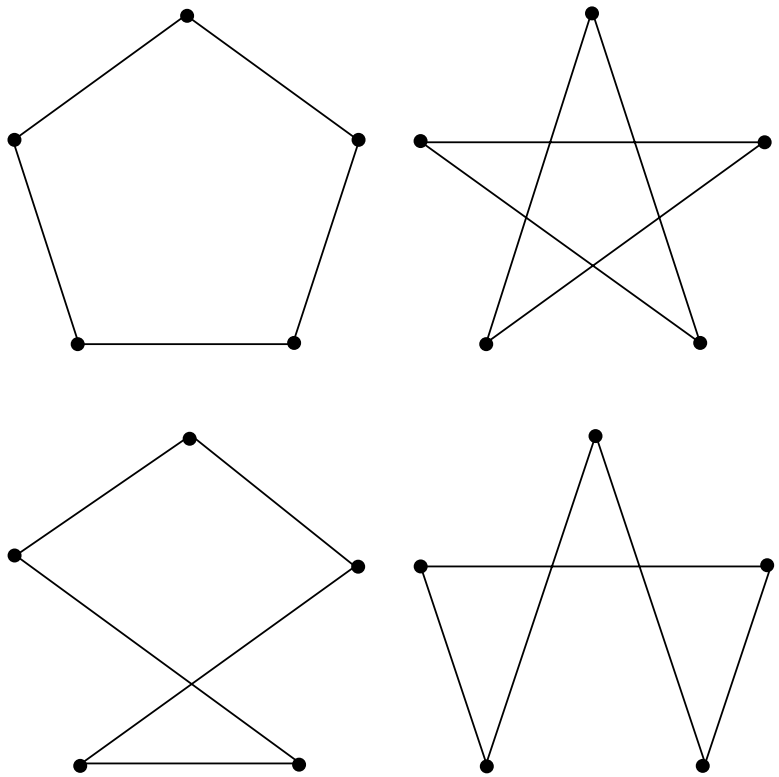
A DEMONTRER ...

- Thm. Dirac
- Premier Thm. d'Ore
- Thm de Chvátal

Partition de K_n en circuits hamiltoniens

Proposition

Pour $n \geq 3$, K_n peut être partitionné en circuits hamiltoniens disjoints SSI n impair. Le nombre de tels circuits partitionnant K_n vaut $(n-1)/2$.



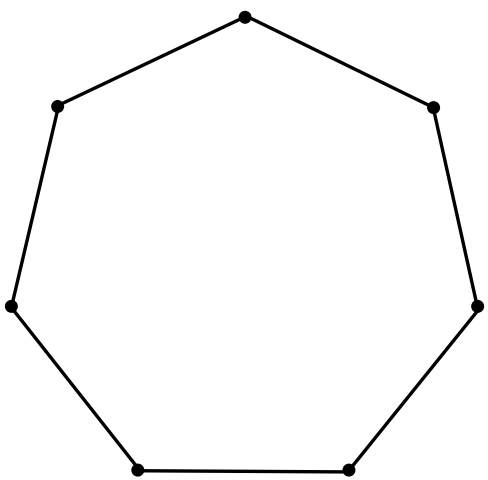
Partition de K_n en circuits hamiltoniens

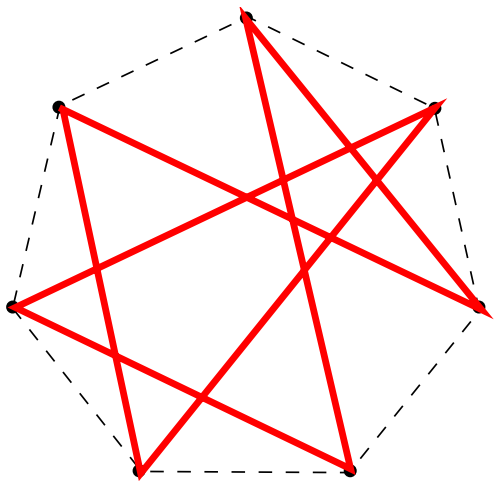
K_n est $(n-1)$-régulier. Un circuit hamiltonien est 2-régulier. Il est nécessaire que $n-1$ soit pair pour décomposer K_n en cycles hamiltoniens disjoints !

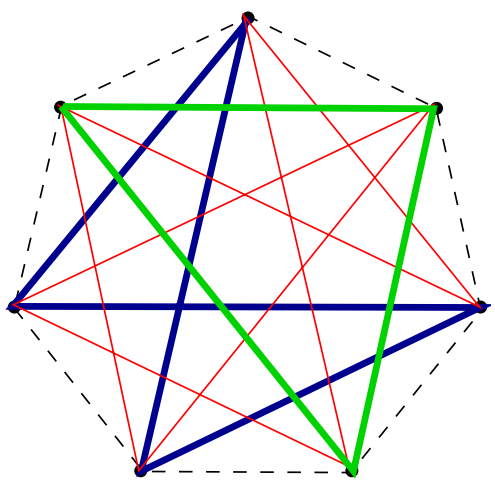
Supposons n impair. K_n peut être décomposé en au plus $(n-1)/2$ circuits hamiltoniens disjoints :

- choix d'un premier cycle hamiltonien formé de n arêtes
- second cycle de K_n disjoint du précédent, le choix se restreint, etc...

peu convaincant !





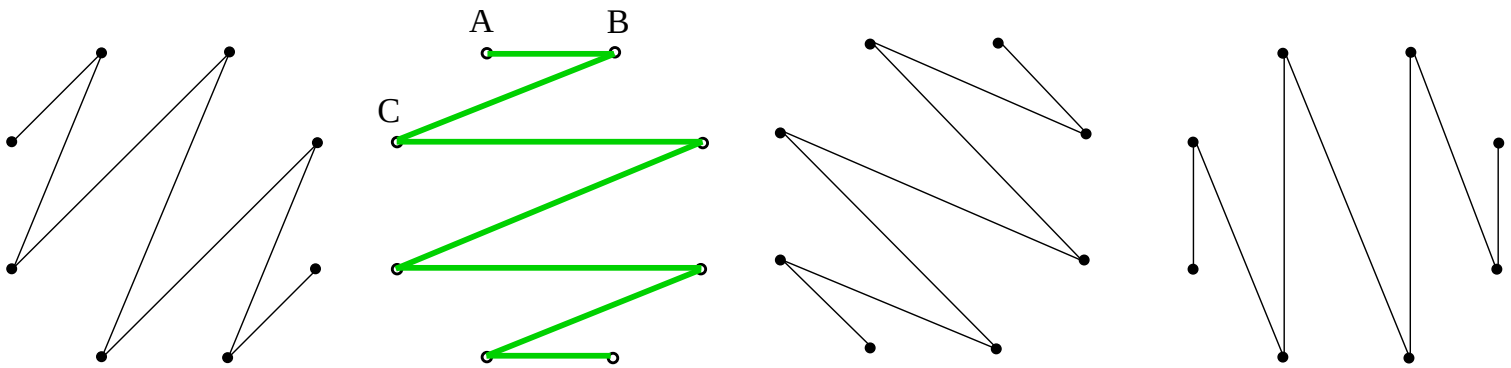


Partition de K_n en circuits hamiltoniens

Lemme

Si n pair, K_n peut être partitionné en $n/2$ chemins hamiltoniens disjoints.

Identifier K_n avec un polygone régulier ayant une symétrie orthogonale par rapport à une horizontale .



Chemin hamiltonien $\mathcal{C}$ en reliant les sommets se trouvant sur une même horizontale et en reliant le sommet gauche du niveau horizontal i avec le sommet de droite de niveau $i+1$. Toutes ces arêtes “obliques” ont une même pente π/n

Partition de K_n en circuits hamiltoniens

En effectuant une rotation de la figure $\mathcal{C}$ de $2k\pi/n$, $k = 1, \ldots, n/2 - 1$, on obtient $n/2$ chemins hamiltoniens distincts.

Tous distincts ? raisonner sur la pente respective des arêtes les constituant : π/n, $3\pi/n$, $5\pi/n$, $\ldots$, $(1 - 1/n)\pi$.

Partition de K_n en circuits hamiltoniens

Lemme

Soit $n \geq 3$ impair. K_n partitionné en $(n-1)/2$ circuits hamiltoniens disjoints SSI K_{n-1} partitionné en $(n-1)/2$ chemins hamiltoniens disjoints.

Si K_n peut être partitionné et qu'on lui supprime un sommet, on passe à K_{n-1} et chaque circuit hamiltonien C donne naissance à un chemin hamiltonien (les extrémités du chemin étant les sommets voisins dans C du sommet supprimé).

Réciproquement, si K_{n-1} est partitionné, l'adjonction d'un sommet permet de passer à K_n en "fermant" chaque chemin hamiltonien pour obtenir des circuits disjoints.