Parcours d'arbres

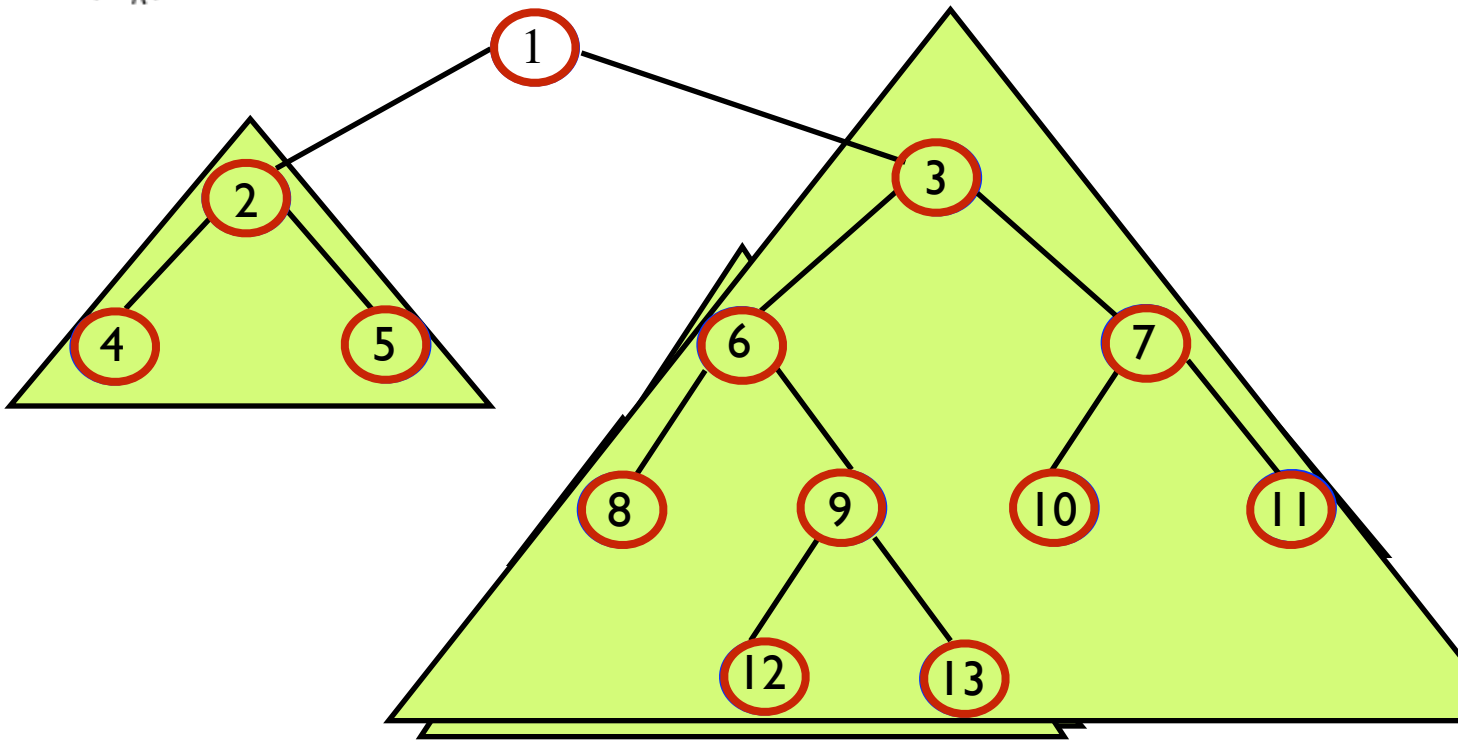
- Une fonction: arbre $\longrightarrow$ liste de ses noeuds
- Pour explorer complètement les arbres
- Deux types
  - parcours en profondeur:
    préfixe, suffixe, symétrique
    parcours branche après branche
  - parcours en largeur:
    hiérarchique
    parcours niveau par niveau

Parcours préfixe

Arbre non vide $A = (r, A_1, A_2, \ldots, A_k)$

Parcours préfixe

$P(A) = (r).P(A_1). \ \ldots \ .P(A_k)$

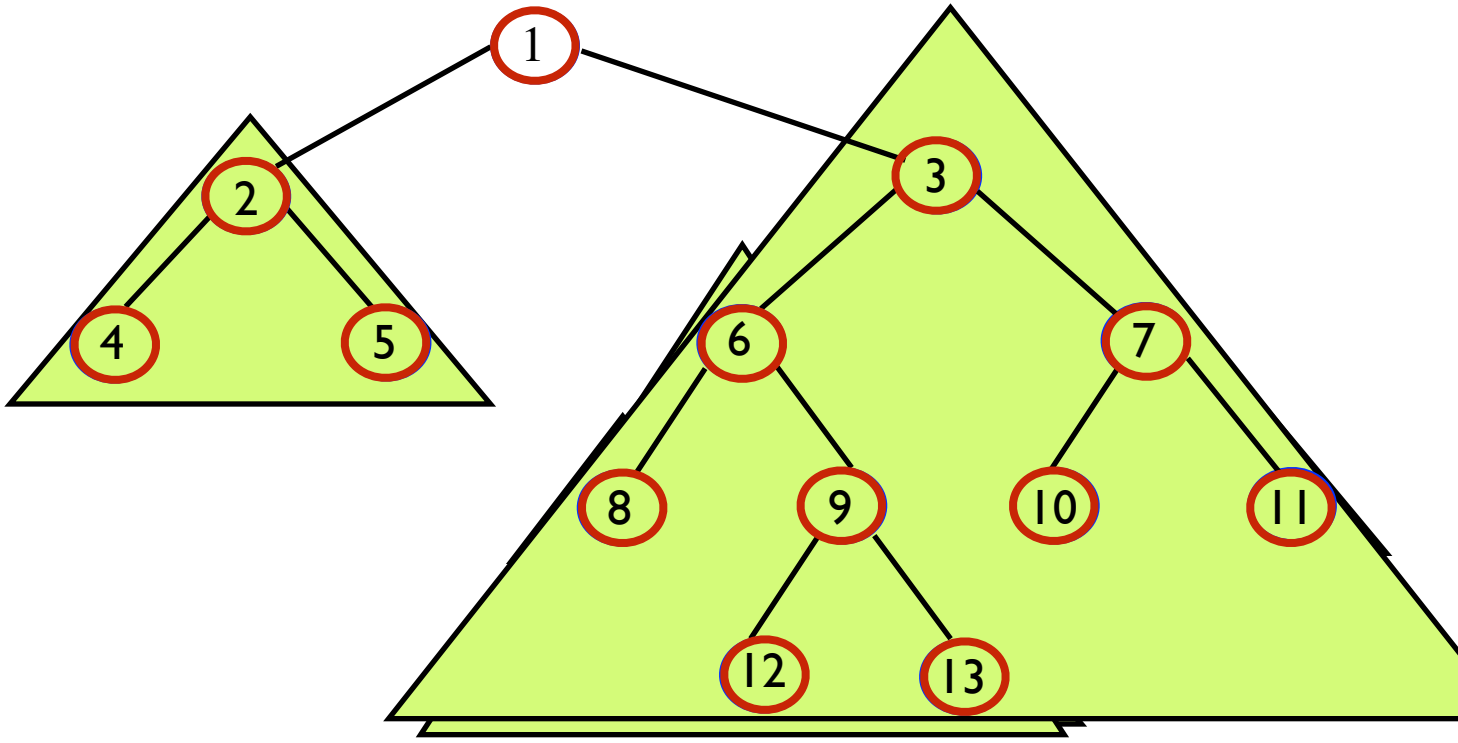


(1, 2, 4, 5, 3, 6, 8, 9, 12, 13, 7, 10, 11)

Parcours suffixe

Parcours suffixe: un noeud est visité après ses descendants

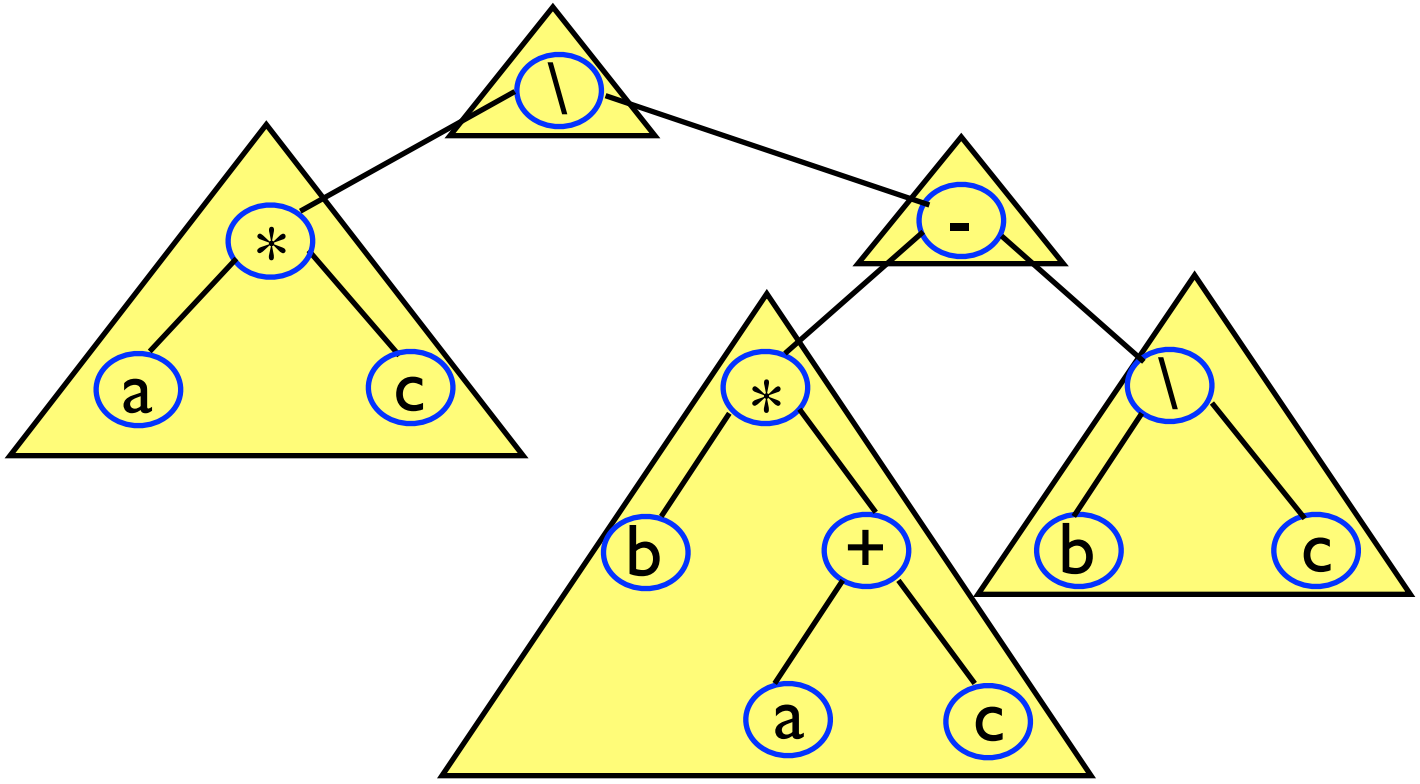
$$S(A)=S(A_1)\cdot S(A_2)\ldots\cdot S(A_k)\cdot r$$



(4, 5, 2, 8, 12, 13, 9, 6, 10, 11, 7, 3, 1)

Parcours symétrique (arbre binaire)

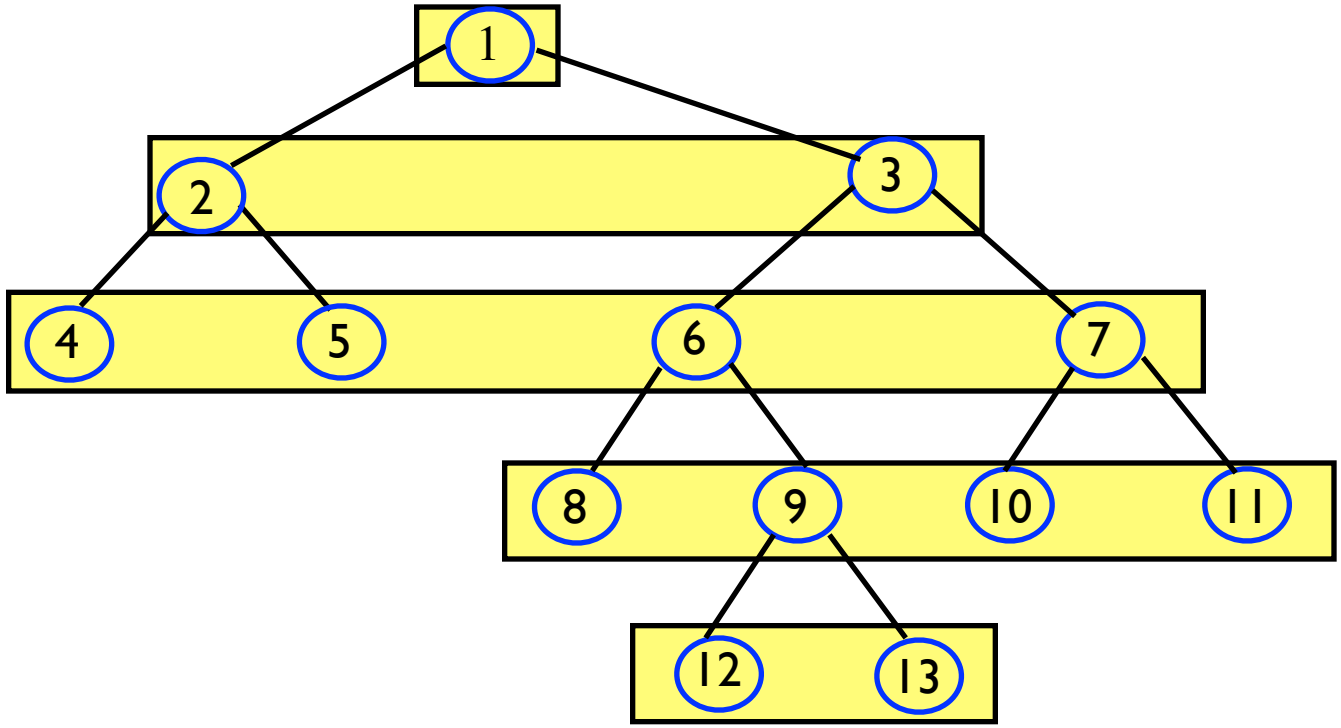
Parcours symétrique: un noeud est visité après son sous-arbre gauche et avant son sous-arbre droit



(a∗c) \ [b∗(a+c) - (b\c)]

Parcours hiérarchique

Parcours hiérarchique: on visite les noeuds niveau par niveau, de gauche à droite, en commençant par la racine



(1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13)

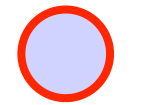
Parcours de graphes

1) Parcours en profondeur (Depth-First Search)

- Un parcours en profondeur (DFS) d'un graphe G
  - Visite tous les sommets et toutes les arêtes de G
  - Détermine si G est connexe ou non
  - Calcule les composantes connexes de G
  - Calcule une forêt couvrante pour G
- L'algorithme de parcours en profondeur (DFS) d'un graphe G prend un temps $O(n+m)$
- L'algorithme de parcours en profondeur peut être étendu pour résoudre d'autres problèmes sur les graphes:
  - Trouver un chemin entre 2 sommets
  - Trouver un cycle dans un graphe

Exemple:

Sommets non visités

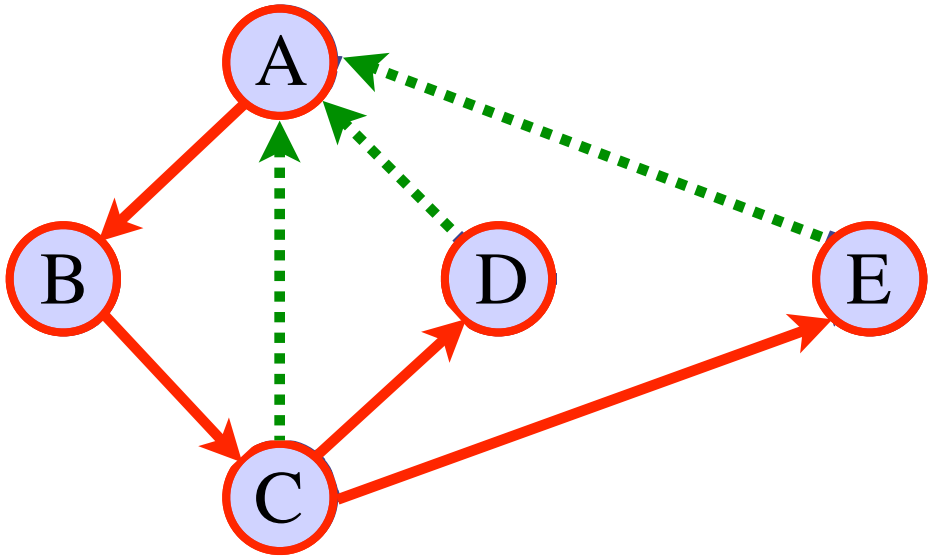


Sommets visités

Arêtes non visitées

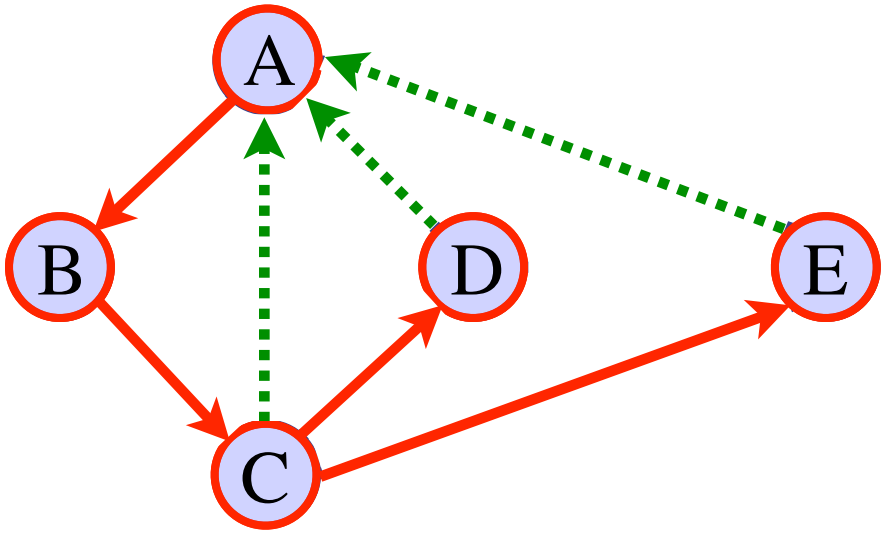
Arêtes sélectionnées

Arêtes de retour



Propriétés du parcours en profondeur:

- Propriété 1: DFS(G,s) visite tous les sommets et les arêtes de la composante connexe de s



- Propriété 2: Les arêtes sélectionnées lors du parcours DFS(G,s) forme un arbre couvrant pour la composant connexe de s

Complexité en temps du parcours en profondeur:

- Étiquetter ou "lire" l'étiquette d'un sommet ou d'une arête $O(1)$
- Chaque sommet est étiquetté deux fois
  - une fois "non visité"
  - une fois "visité"

  } $O(n)$
- L'opération Incident(u) est appelée une fois sur chaque sommet u
- Comme on a que

$$\sum_{v \in N} deg(v) = 2m$$

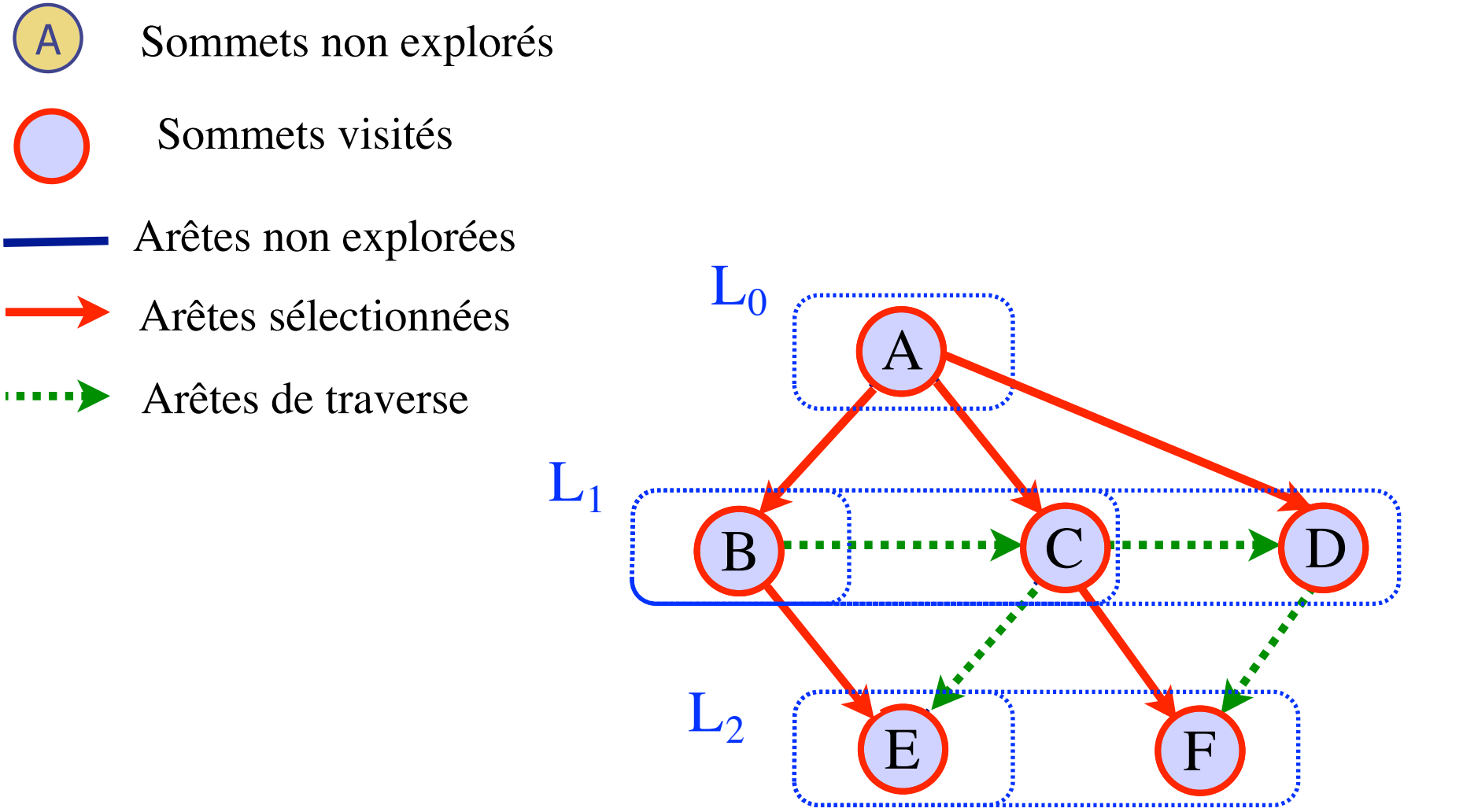
la complexité en temps de l'algorithme DFS est $O(m+n)$

Parcours de graphes

2) Parcours en largeur (Breadth-First Search)

- Un parcours en largeur (BFS) d'un graphe G
  - Visite tous les sommets et toutes les arêtes de G
  - Détermine si G est connexe ou non
  - Calcule les composantes connexes de G
  - Calcule une forêt couvrante pour G
- L'algorithme de parcours en largeur (BFS) d'un graphe G prend un temps $O(n+m)$
- L'algorithme de parcours en largeur peut être étendu pour résoudre d'autres problèmes sur les graphes:
  - Trouver le plus court chemin entre 2 sommets
  - Trouver un cycle simple dans un graphe

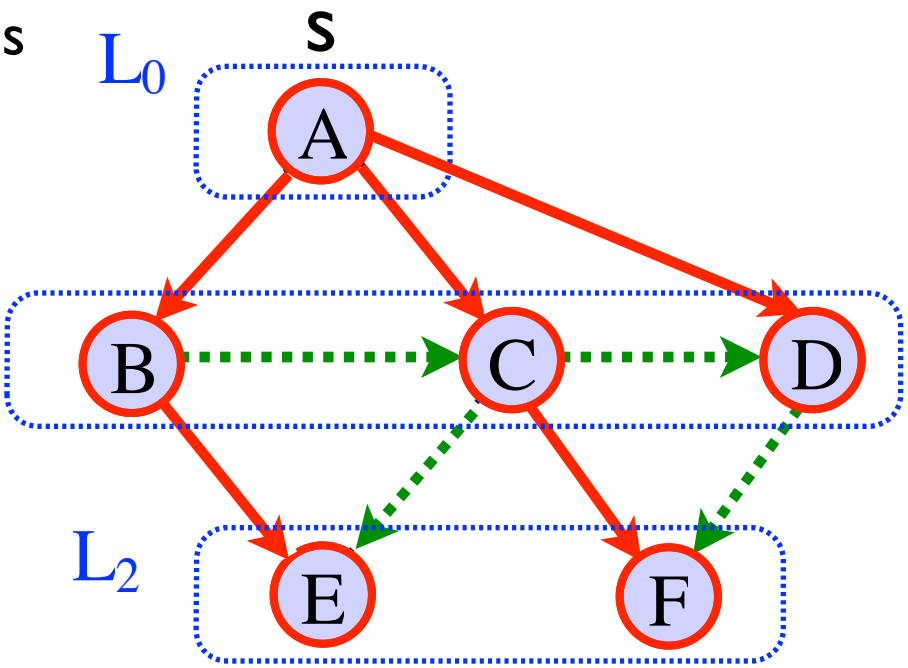
Exemple:



© adapté de Goodrich et Tamassia 2004

Propriétés du parcours en largeur:

- Propriété 1: BFS(G,s) visite tous les sommets et les arêtes de la composante connexe de s
- Propriété 2: Les arêtes sélectionnées lors du parcours DFS(G,s) forme un arbre couvrant pour la composant connexe de s
- Propriété 3: Pour tous sommets u dans L_i, le chemin de s à u suivant les arêtes de l'arbre couvrant contient exactement i arêtes et i+1 sommets. C'est un plus court chemin de s à u.



Complexité en temps du parcours en largeur:

- Étiquetter ou "lire" l'étiquette d'un sommet ou d'une arête $O(1)$
- Chaque sommet est étiquetté deux fois
  - une fois "non exploré"
  - une fois "visité"

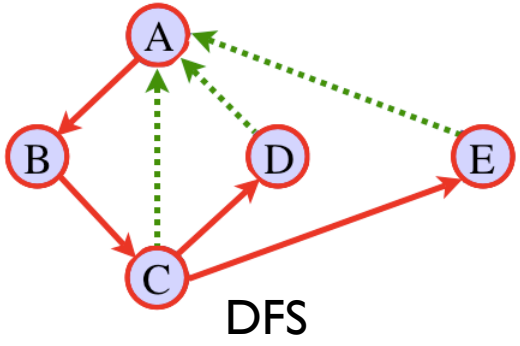
  } $O(n)$
- L'opération incident(u) est appelée une fois sur chaque sommet u
- Comme on a que

$$\sum_{v \in N} deg(v) = 2m$$

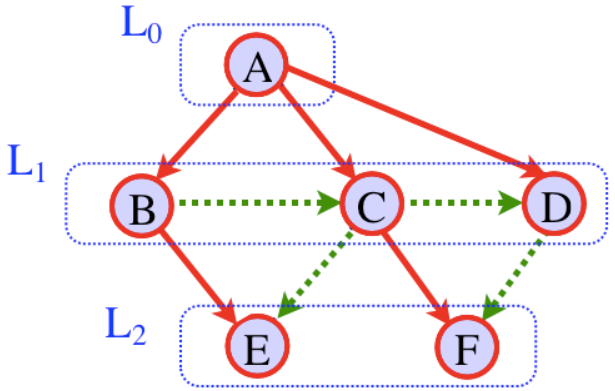
la complexité en temps de l'algorithme BFS est $O(m+n)$

DFS vs BFS

Applications	DFS	BFS
fôret couvrante, composantes connexes	ok	ok
chemins entre deux sommets, cycles	ok	ok
plus court chemin		ok



DFS



BFS

Parcours d'arbres orientés

- On spécialise les parcours (en profondeur et en largeur) de graphes aux graphes orientés en traversant les arêtes seulement selon leur direction.

- Lorsqu'on exécute un algorithme de parcours à partir d'un sommet s d'un graphe orienté, les sommets visités représentent l'ensemble des sommets accessibles du sommet s