

Parcours de graphes I

Le cas non orienté

Michel Habib
Magistère Cachan 2011–2012

23 décembre 2011

Plan

- 1 Parcours Générique
- 2 Parcours en largeur
 - BFS classique
- 3 Parcours en profondeur
 - DFS classique
- 4 Parcours selon le voisinage maximal MNS
- 5 Autres parcours selon un voisinage maximal
 - Maximal Cardinality Search
 - LexBFS
 - DFS lexicographique

- 1 Parcours Générique
- 2 Parcours en largeur
 - BFS classique
- 3 Parcours en profondeur
 - DFS classique
- 4 Parcours selon le voisinage maximal MNS
- 5 Autres parcours selon un voisinage maximal
 - Maximal Cardinality Search
 - LexBFS
 - DFS lexicographique

Le problème

Étant donné un graphe $G = (V, E)$ (non-orienté), explorer l'ensemble des sommets G en "traversant" les arêtes du graphe.

Résultat

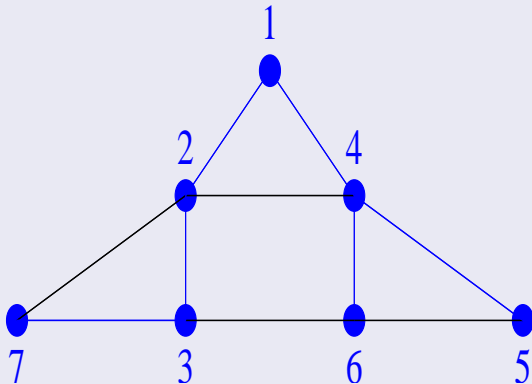
- Un arbre de parcours
- Un ordre total sur les sommets du graphe

Questions

- À quelle condition un ordre σ sur les sommets correspond à un parcours ?
- Quelles sont les propriétés de ces ordres ?

Référence principale :

Ces questions simples n'ont été posées que très récemment :
D.G. Corneil et R. M. Krueger, A unified view of graph searching,
SIAM J. Discrete Math, 22, Num 4 (2008) 1259-1276



Invariant

À chaque étape, on sélectionne une arête entre un sommet visité et un non-visité

Parcours générique

$S \leftarrow \{s\}$

pour $i \leftarrow 1$ à n **faire**

 Extraire un sommet non-numéroté v de S

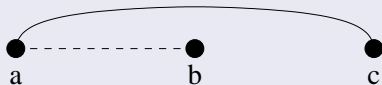
$\sigma(i) \leftarrow v$

pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

$S \leftarrow S \cup \{w\}$

Question ?

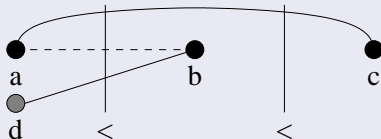
Soient a , b et c trois sommets tels que $ab \notin E$ et $ac \in E$.



À quelle condition est-il possible de visiter a puis b puis c ?

Propriété (Gen)

Étant donné un ordre σ sur V , si $a < b < c$ et $ac \in E$ et $ab \notin E$, alors il existe un sommet d tel que $d < b$ et $db \in E$



Théorème

Pour un graphe $G = (V, E)$, un ordre σ sur V est un parcours de G ssi σ vérifie la propriété (Gen).

- 1 Parcours Générique
- 2 **Parcours en largeur**
 - BFS classique
- 3 Parcours en profondeur
 - DFS classique
- 4 Parcours selon le voisinage maximal MNS
- 5 **Autres parcours selon un voisinage maximal**
 - Maximal Cardinality Search
 - LexBFS
 - DFS lexicographique

Parcours en largeur (BFS)

Données: Un graphe $G = (V, E)$ et un sommet source s

Résultat: Un ordre total σ de V

Initialiser la file S à s

pour $i \leftarrow 1$ à n **faire**

 Extraire le sommet v de la tête de la **file** S

$\sigma(i) \leftarrow v$

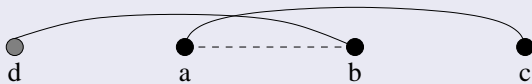
pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

si w *n'est pas dans* S **alors**

 Ajouter w en fin de la file S

Propriété (B)

Étant donné un ordre σ sur V , si $a < b < c$ et $ac \in E$ et $ab \notin E$, alors il existe un sommet d tel que $d < a$ et $db \in E$



Théorème

Pour un graphe $G = (V, E)$, un ordre σ sur V est un parcours BFS de G ssi σ vérifie la propriété (B).

Parcours en profondeur (DFS)

Données: Un graphe $G = (V, E)$ et un sommet source s

Résultat: Un ordre total σ de V

Initialiser la pile S à s

pour $i \leftarrow 1$ à n **faire**

 Extraire le sommet v du haut de la **pile** S

$\sigma(i) \leftarrow v$

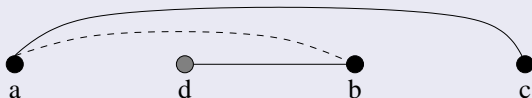
pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

si w *n'est pas dans* S **alors**

 Ajouter w en haut de la pile S

Propriété (D)

Étant donné un ordre σ sur V , si $a < b < c$ et $ac \in E$ et $ab \notin E$, alors il existe un sommet d tel que $a < d < b$ et $db \in E$.



Théorème

Pour un graphe $G = (V, E)$, un ordre σ sur V est un parcours DFS de G ssi σ vérifie la propriété (D).

Quelques exemples d'application

- Test de planarité (utilisation d'un DFS pour placer les arêtes autour de l'arbre associé au parcours).
- Composantes 2-connexes (resp. composantes fortement connexes dans le cas des graphes orientés).
- Tri topologique (extension linéaire) des graphes sans circuits, applications aux mécanismes d'héritage. . . .

- 1 Parcours Générique
- 2 Parcours en largeur
 - BFS classique
- 3 Parcours en profondeur
 - DFS classique
- 4 Parcours selon le voisinage maximal MNS**
- 5 Autres parcours selon un voisinage maximal
 - Maximal Cardinality Search
 - LexBFS
 - DFS lexicographique

Parcours MNS

Parcours par voisinage maximal (MNS)

Données: Un graphe $G = (V, E)$ et un sommet source s

Résultat: Un ordre total σ de V

Affecter l'étiquette \emptyset à chaque sommet

$label(s) \leftarrow \{n\}$

pour $i \leftarrow n$ à 1 **faire**

 Choisir un sommet v d'étiquette **maximal pour l'inclusion.**

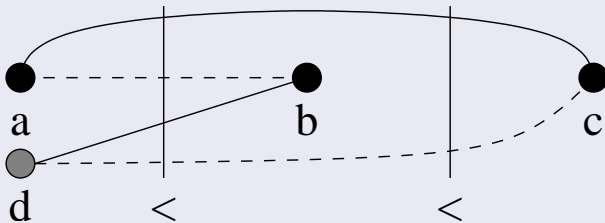
$\sigma(i) \leftarrow v$

pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

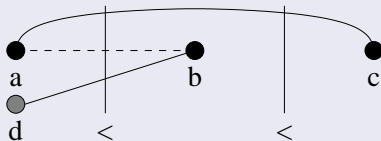
$label(w) \leftarrow \{i\} \cup label(w)$

Propriété (MNS)

Étant donné un ordre σ sur V , si $a < b < c$ et $ac \in E$ et $ab \notin E$, alors il existe un sommet d tel que $d < b$, $db \in E$ et $dc \notin E$.



Parcours générique



Le parcours suivant le voisinage maximal est donc le complété du parcours générique (par analogie à LexBFS (resp. LesDFS) qui sont les complétés de BFS (resp. DFS)). Ainsi MNS fut initialement appelé LexGen.

- 1 Parcours Générique
- 2 Parcours en largeur
 - BFS classique
- 3 Parcours en profondeur
 - DFS classique
- 4 Parcours selon le voisinage maximal MNS
- 5 Autres parcours selon un voisinage maximal
 - Maximal Cardinality Search
 - LexBFS
 - DFS lexicographique

Maximal Cardinality Search

Données: Un graphe $G = (V, E)$ et un sommet source s

Résultat: Un ordre total σ de V

Affecter l'étiquette 0 à chaque sommet

$label(s) \leftarrow \{n\}$

pour $i \leftarrow n$ à 1 **faire**

 Choisir un sommet v d'**étiquette maximum**

$\sigma(i) \leftarrow v$

pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

$label(w) \leftarrow label(w) + 1$

Parcours en largeur lexicographique (LexBFS)

Données: Un graphe $G = (V, E)$ et un sommet source s

Résultat: Un ordre total σ de V

Affecter l'étiquette \emptyset à chaque sommet

$label(s) \leftarrow \{n\}$

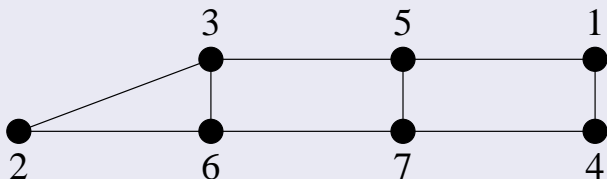
pour $i \leftarrow n$ à 1 **faire**

 Choisir un sommet v d'**étiquette lexicographique max.**

$\sigma(i) \leftarrow v$

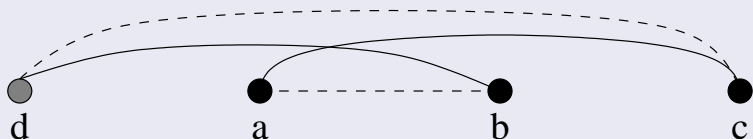
pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

$label(w) \leftarrow label(w). \{i\}$



Propriété (LexB)

Étant donné un ordre σ sur V , si $a < b < c$ et $ac \in E$ et $ab \notin E$, alors il existe un sommet d tel que $d < a$ et $db \in E$ et $dc \notin E$.



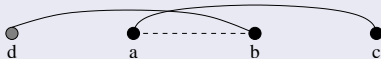
Théorème

Pour un graph $G = (V, E)$, un ordre σ sur V est un parcours LexBFS de G ssi σ vérifie la propriété (LexB).

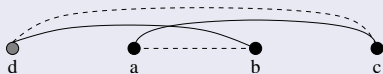
- 1 Parcours Générique
- 2 Parcours en largeur
 - BFS classique
- 3 Parcours en profondeur
 - DFS classique
- 4 Parcours selon le voisinage maximal MNS
- 5 Autres parcours selon un voisinage maximal
 - Maximal Cardinality Search
 - LexBFS
 - DFS lexicographique

BFS vs LexBFS

BFS

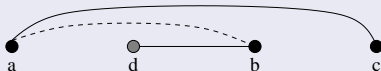


LexBFS

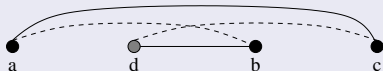


DFS vs LexDFS

DFS

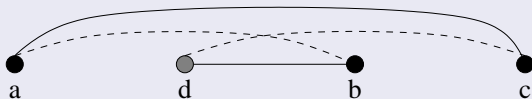


LexDFS



Propriété (LD)

Étant donné un ordre σ sur V , si $a < b < c$ et $ac \in E$ et $ab \notin E$, alors il existe un sommet d tel que $a < d < b$ et $db \in E$ et $dc \notin E$.



Théorème

Pour un graphe $G = (V, E)$, un ordre σ sur V est un parcours LexDFS de G ssi σ vérifie la propriété (LD).

LexDFS

Parcours en profondeur lexicographique (LexDFS)

Données: Un graphe $G = (V, E)$ et un sommet source s

Résultat: Un ordre total σ de V

Affecter l'étiquette \emptyset à chaque sommet

$label(s) \leftarrow \{0\}$

pour $i \leftarrow 1$ à n **faire**

 Choisir un sommet v d'étiquette **lexicographique max.**

$\sigma(i) \leftarrow v$

pour chaque *sommet non-numéroté* $w \in N(v)$ **faire**

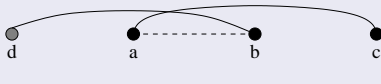
$label(w) \leftarrow \{i\}.label(w)$

Exercices

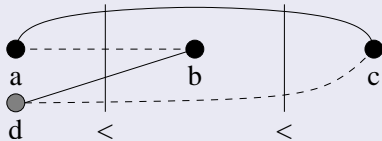
- 1 Vérifier que les inclusions sont bien strictes.
- 2 Que peut-on dire d'un parcours à la fois BFS et MNS ?
A-t-on $\text{BFS} + \text{MNS} = \text{LexBFS}$?
- 3 Idem pour DFS et MNS
A-t-on $\text{DFS} + \text{MNS} = \text{LexDFS}$?

BFS + MNS

BFS

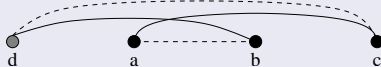


MNS



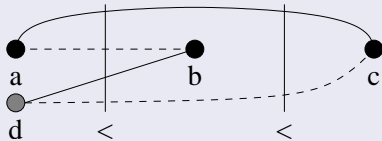
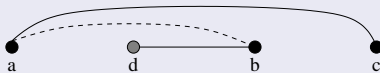
= LexBFS?

LexBFS



DFS + MNS

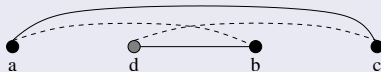
DFS



MNS

= LexDFS ?

LexDFS



Layer ordering or distance level ordering

un ordre préservant les distances au sommet initial
 \neq BFS