Chapitre 6
Les graphes non orientés

Michaël Krajecki

Université de Reims Champagne-Ardenne
michael.krajecki@univ-reims.fr
http://www.univ-reims.fr/crestic

Graphes et algorithmes

Les graphes non orientés

- Bibliographie : *Structures de données et algorithmes*, A. Aho, J. Hopcroft, J. Ullman, InterEditions, 1989
- Les graphes non orientés permettent la représentation de relations *symétriques*
- Objectifs :
  1. Construire un arbre couvrant de poids minimum
  2. Définition des composantes 2-connexes
  3. Couplages maximaux

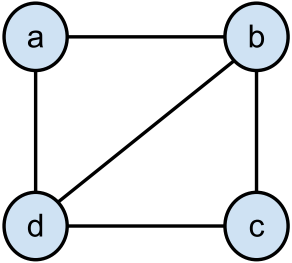
Définitions

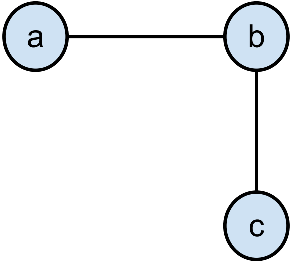
- Terminologie proche de celle employée par les graphes orientés
- On parle d'*arête* à la place d'arc
- Les sommets s et t sont *adjacents* s'il existe une arête (s, t) ou (t, s)
- La suite de sommets $s_1, s_2, \dots, s_n$, où chaque (s_i, s_{i+1}) est une arête, définit une *chaîne* ou chemin non orienté
- Un graphe est connexe si pour tous sommets s et t, tel que $s \neq t$, il existe un chemin non orienté qui relie s et t (à rapprocher de graphe fortement connexe dans les graphes orientés)

Définitions (suite)

- Soit $G = (S, A)$, où S désigne l'ensemble des sommets et A l'ensemble des arêtes
- Si $G' = (S', A')$ est un graphe tel que :
  - $S' \subset S$, on dit alors que G' est un *sous-graphe engendré* par S'
  - $A' \subset A$, on dit alors que G' est un *graphe partiel engendré* par A'
  - $S' \subset S$ et $A' \subset A$, on dit alors que G' est un *sous-graphe partiel* de G
  - Si, de plus A' contient toutes les arêtes reliant des sommets de S', on dit alors que G' est un *sous-graphe induit* de G

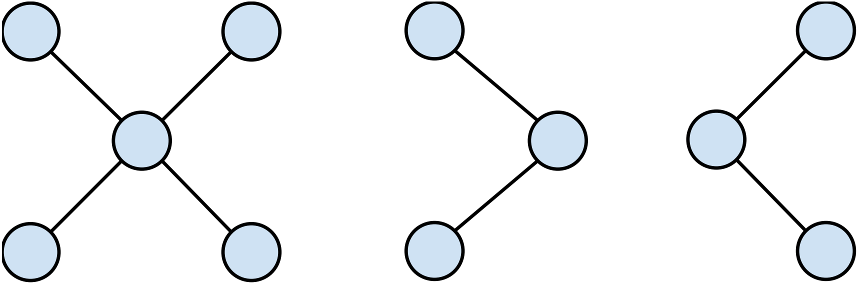
Exemple de sous graphe induit





Composante connexe

- Une *composante connexe* de G est un sous-graphe induit connexe maximal de G



Représenter un graphe non orienté

- Les représentations par matrice ou par liste d'adjacence présentées pour les graphes orientés peuvent être utilisées pour les graphes non orientés
- Il suffit de considèrer chaque arête (s, t) comme 2 arcs entre (s, t) et entre (t, s)

Arbre couvrant de poids minimal

Définition (Arbre couvrant de poids minimal)

- *Soit $G = (S, A)$ un graphe connexe, où chaque arête est pondérée par un poids noté $p(s, t)$*
- *Un arbre couvrant de G un arbre libre joignant tous les sommets de G*
- *Le poids d'un arbre couvrant est défini par la somme des poids des arêtes qu'il contient*

- Exemple : construire un réseau de télécommunication entre les grandes villes d'une région ou d'un pays

Plan

Principe de l'algorithme de Prim

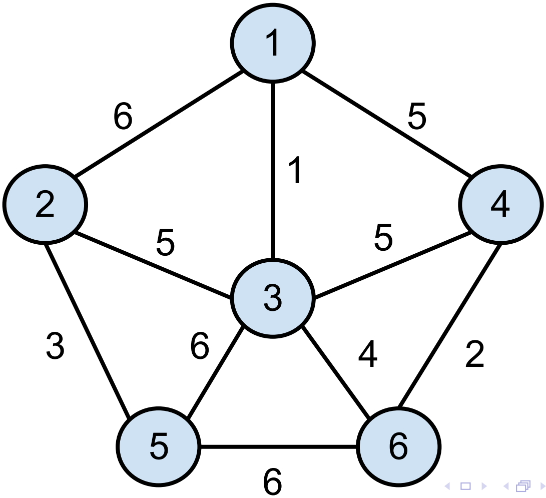
- On suppose que S s'exprime de la façon suivante : $S = \{1, 2, \dots, n\}$
- On construit l'ensemble de sommets T
- Au départ T est réduit à $\{1\}$
- A chaque tour, on choisit le sommet t pris dans $S - T$ le plus proche des sommets déjà présents dans T
- C'est à dire le sommet t tel que $p(s, t)$ est minimal avec s dans T

Algorithme de Prim

Procédure *Prim(G=(S,A) :graphe, R :ensembles d'arêtes).*
variable *: S,T :ensemble de sommets ; s,t : sommet*
Début
  $R \leftarrow \varnothing$
  $T \leftarrow \{1\}$
  Tant que $S \neq T$ ***faire***
    Choisir (s,t) *l'arête de poids minimal telle que* $s \in T$ *et* $t \in S - T$
    $R \leftarrow R \cup \{(s,t)\}$
    $T \leftarrow T \cup \{t\}$
  Ftant
Fin.

Remarque : on suppose que S est de la forme $\{1, 2, \dots, n\}$

Exemple



Trouver efficacement un arête de poids minimal

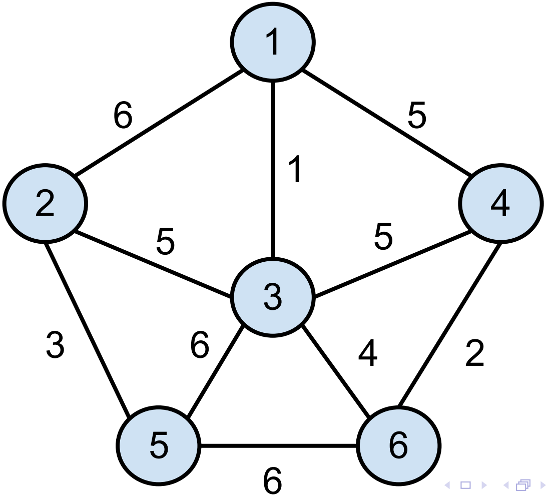
- L'algorithme repose sur la recherche de l'arête de poids minimal entre les ensembles S et $S - T$
- Il est possible de réaliser cette recherche à l'aide de 2 tableaux
- $PlusPres[i]$: désigne le sommet de T le plus proche du sommet $i \in S - T$
- $Poids[i]$: désigne le point de l'arête entre $i \in S - T$ et $PlusPres[i] \in T$
- A chaque itération de la boucle *tant que* :
  1. On choisit le sommet k tel que $Poids[k]$ est minimal
  2. On ajoute l'arête $(k, PlusPres[k])$
  3. On met à jour les 2 tableaux car k est ajouté à T

Plan

Algorithme de Kruskal

- Soit $G = (S, A)$ un graphe connexe, où chaque arête est pondérée par un poids noté $p(s, t)$
- Au départ, Kruskal initialise R à $(S, \emptyset)$
- Chaque sommet défini alors une composante connexe
- A chaque étape, Kruskal relie 2 composantes connexes entre elles
- Pour cela, il choisit l'arête de poids minimum qui relie 2 sommets appartenant à deux composantes connexes différentes
- Ce principe est appliqué jusqu'à obtenir une seule composante connexe, qui définit l'arbre couvrant de poids minimal

Exemple

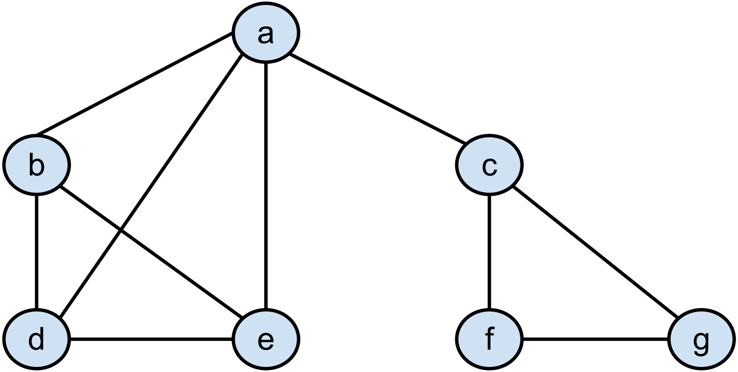


Plan

Parcours en profondeur d'abord

- La procédure de parcours en profondeur définie pour les graphes orientés peut s'appliquer aux graphes non orientés
- On peut remarquer que chaque arbre composant la forêt est une composante connexe du graphe G
- Si le graphe G est connexe, le parcours en profondeur aura alors pour résultat un unique arbre
- Il est possible de définir 2 types d'arêtes : les *arêtes d'arbre* et les *arêtes de retour*

Exemple : parcours en profondeur



Plan

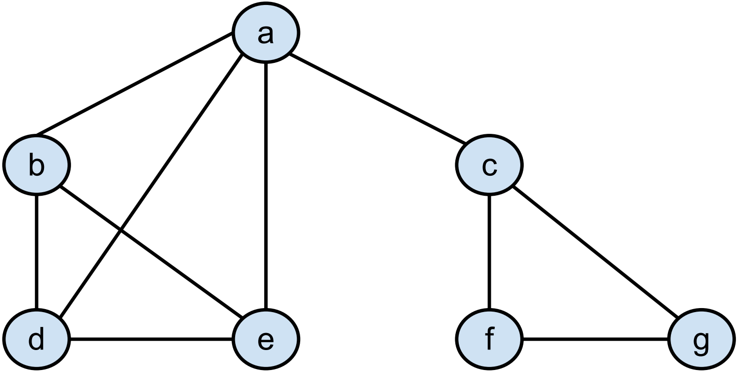
Parcours en largeur d'abord

- La procédure de parcours en *largeur d'abord* explore depuis s tous les sommets qui lui sont adjacents
- Il est possible de construire un *Arbre de Parcours en Largeur : APL*
- Si (x, y) apparaît dans l'arbre *APL*, alors le sommet y a été exploré directement depuis x par le parcours en largeur

Parcours en largeur d'abord

Procédure *Largeur(s : sommet).*
variable *: f : file de sommets ; x,y :sommet*
Début
etat[*s*] ← *exploré ; enfiler(f,s)*
Tant que *non Vide(f)* ***faire***
x ← *tete(f) ; defiler(f)*
Pour *chaque sommet y adjacent à x* ***faire***
Si *etat*[*y*] = *inexploré* ***alors***
etat[*y*] ← *exploré ; enfiler(f,y)*
insérer$((x, y), APL)$
Fsi
Fpour
Ftant
Fin.

Exemple : parcours en largeur



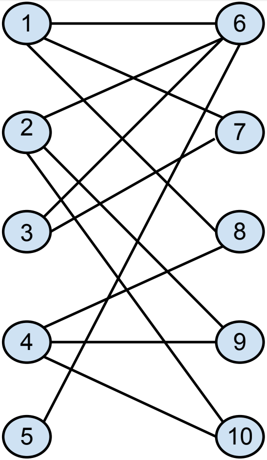
Point d'articulation

- Un *point d'articulation* s est un sommet du graphe qui, s'il est retiré, rend le graphe non connexe
- Sur l'exemple précédent a et c sont des points d'articulation
- Un graphe sans point d'articulation est un graphe *2-connexe*
- la connexité d'un graphe est une mesure de sa robustesse
- Un graphe est dit k-connexe, s'il est possible de retirer $k-1$ sommets sans perdre la connexité du graphe
- Exemple d'utilisation : réseau de communications

Graphe biparti

- Soit $G = (S, A)$
- Un graphe est biparti si :
  1. $S = S_1 \cup S_2$
  2. Toute arête de A est définie par $a = (s_1, s_2)$ où $s_1 \in S_1$ et $s_2 \in S_2$
- Exemple : S_1 désigne les enseignants et S_2 les cours à enseigner
- Une arête (e, c) exprime le fait que l'enseignant e peut enseigner le cours c

Exemple de graphe biparti



Problème de couplage maximal

- Soit $G = (S, A)$
- Un sous-ensemble $A' \subset A$ ne contenant aucun couple d'arêtes aboutissant au même sommet de S est appelé un *couplage*
- Rechercher le plus grand ensemble A' est maximal est appelé *problème de couplage maximal*