INFO0501

ALGORITHMIQUE AVANCÉE

COURS 4

GRAPHES
REPRÉSENTATION ET ALGORITHMES ÉLÉMENTAIRES

UNIVERSITÉ DE REIMS CHAMPAGNE-ARDENNE

Pierre Delisle
Département de Mathématiques, Mécanique et Informatique
Octobre 2018

Plan de la séance

- Généralités
  - Notions de base et représentation
- Parcours en largeur
- Parcours en profondeur

- Bibliographie
  - T. H. Cormen, C. E. Leiserson, R. L. Rivest, C. Stein, "Algorithmique", 3e édition, Dunod, 2010

GÉNÉRALITÉS SUR LES GRAPHES

Graphes ?

- Les graphes permettent de modéliser une multitude de problèmes
- Omniprésents en informatique
  - Réseaux
  - Systèmes distribués
  - Optimisation combinatoire
  - ……
- Les algorithmes pour les manipuler sont fondamentaux

Graphe non orienté

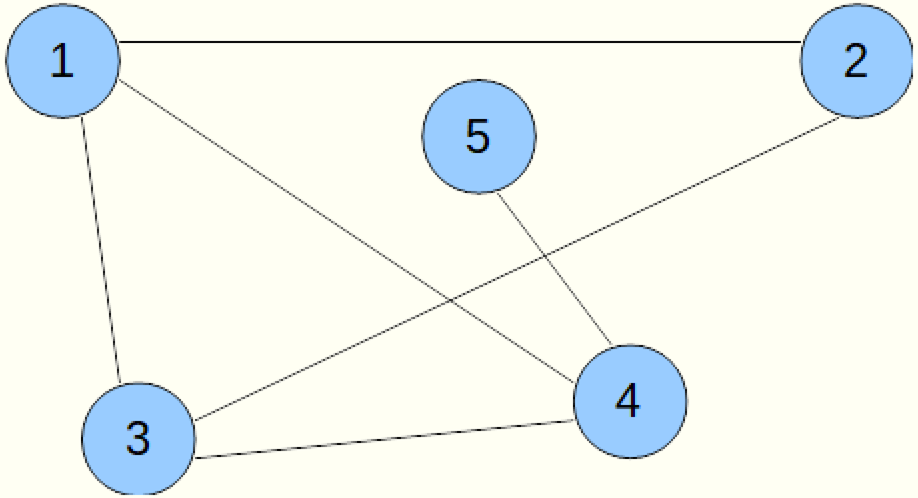
- Un graphe non orienté …

$G = (S, A)$

- … est défini par deux ensembles
  - Ensemble S des *sommets*
  - Ensemble A des *arêtes*
- Une arête, un élément a de A

$a = \{x, y\}$

  - Est défini par une paire de sommets distincts x et y de S
  - N'apparaît pas plusieurs fois dans A
- On dit que
  - x et y sont *incidents* à a
  - x et y sont les *extrémités* de a
  - x et y sont *adjacents*



- $S = \{1, 2, 3, 4, 5\}$
- $A = \{(1, 2), (1, 3), (1, 4), (2, 3), (3, 4), (4, 5)\}$
- 1 et 2 sont adjacents
- 2 et 5 ne sont pas adjacents

Graphe orienté

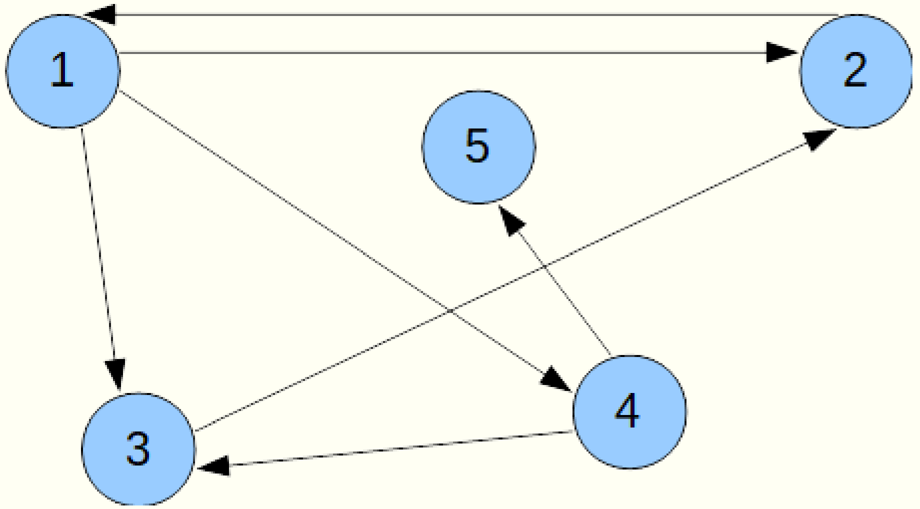
- Un graphe orienté …

$G = (S, A)$

- … est défini par deux ensembles
  - Ensemble S des *sommets*
  - Ensemble A des *arcs*

$a = \{x, y\}$

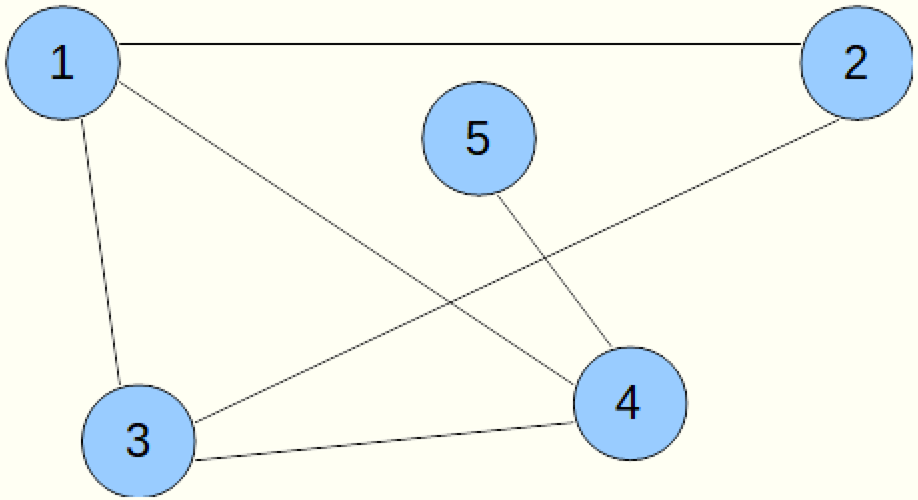
- Une arc, un élément a de A
  - Est défini par un couple de sommets distincts x et y de S
  - N'apparaît pas plusieurs fois dans A
  - Mais on peut avoir (x,y) et (y, x), qui sont deux arcs distincts
- On dit que
  - a admet x comme *origine*, ou *extrémité initiale*
  - a admet y comme *extrémité finale* ou *terminale*



- $S = \{1, 2, 3, 4, 5\}$
- $A = \{(1, 2),(1, 3),(1, 4),(2, 1),(3, 2),(4, 3),(4, 5)\}$

Graphe fini

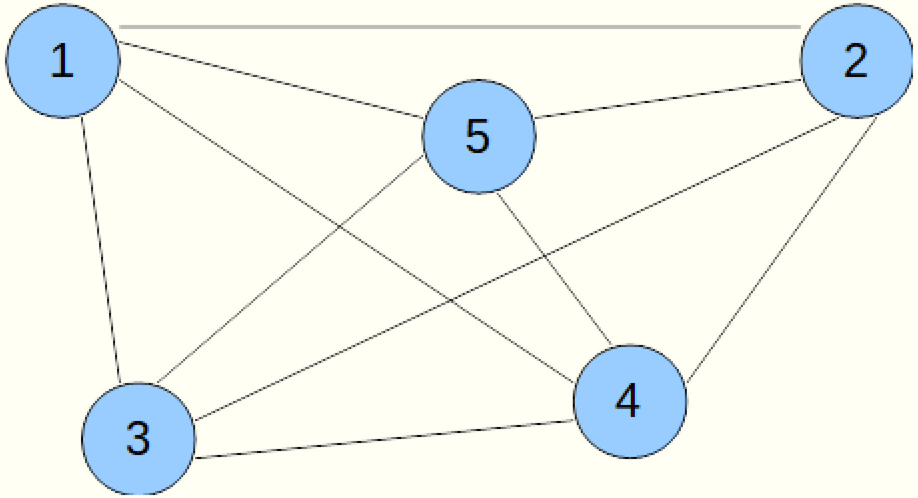
- Le cardinal de S
  - Est appelé *ordre* du graphe (nombre de sommets)
  - $n = |S|$
- Le cardinal de A
  - Est appelé *taille* du graphe (nombre d'arêtes ou arcs)
  - $m = |A|$
- Les cardinaux de S et de A sont finis



- L'ordre du graphe n est 5
- La taille du graphe m est 6

Graphe complet (ou clique)

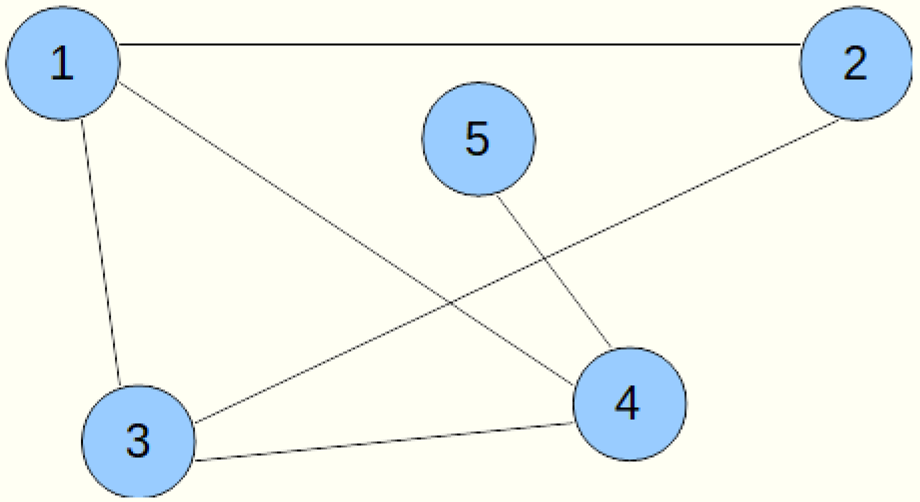
- Un graphe complet à n sommets ...
  - Noté K_n
- ... est un graphe non orienté d'ordre n dont deux sommets quelconques sont adjacents
  - Il est donc de taille $n(n-1) / 2$



- Graphe complet d'ordre 5
- La taille du graphe est de 5 * 4 / 2 = 10

Graphe partiel et sous-graphe

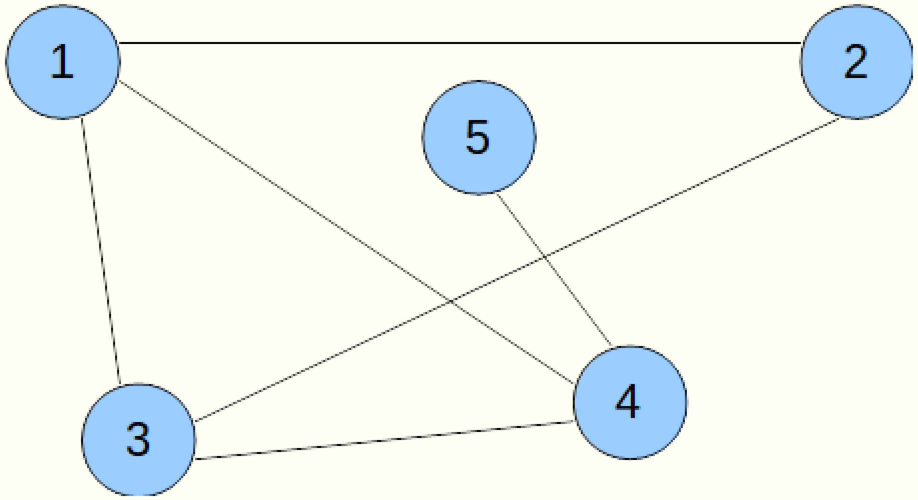
- Un graphe partiel de $G = (S, A)$ est un graphe
  - Ayant le même ensemble de sommets S que G
  - Ayant pour ensemble d'arêtes une partie de A
- Étant donnée une partie Y de S…
- … un sous-graphe F de G engendré par Y
  - Est un graphe ayant pour ensemble de sommets Y
  - Une arête (arc) de G donnant naissance à une arête (arc) de F si et seulement si les deux extrémités de cette arête (arc) sont dans Y
- Autrement dit, un sous graphe F d'un graphe G est un graphe composé de certains sommets de G et de toutes les arêtes qui relient ces sommets dans G



- $G' = (\{1, 2, 3, 4, 5\}, \{(1, 2),(2, 3),(3, 4),(4, 5)\})$ est un graphe partiel
- $G = (\{1, 2, 3\}, \{(1, 2), (1, 3), (2, 3)\})$ est un sous-graphe complet d'ordre 3

Degré d'un sommet

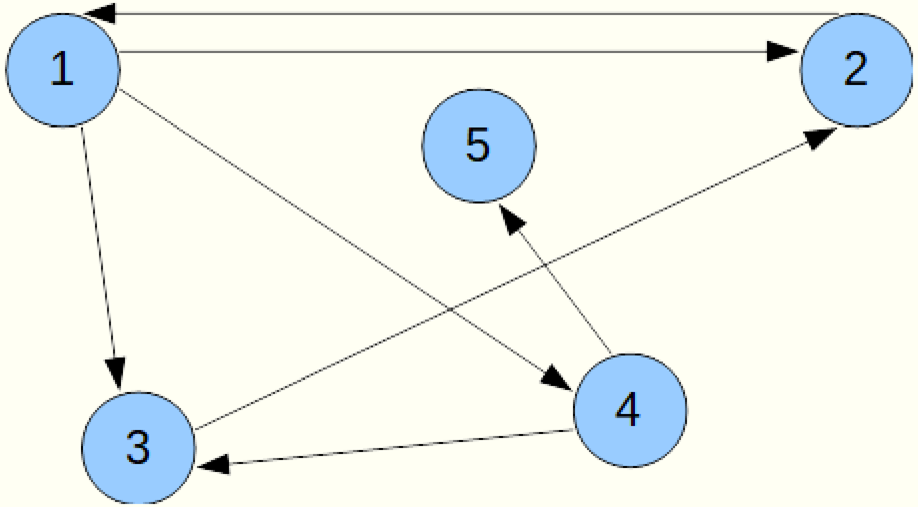
- Étant donné un sommet x d'un graphe G non orienté
  - Le degré de x est le nombre d'arêtes incidentes à x
  - Les autres extrémités de ces arêtes constituent l'ensemble des voisins de x
- Autrement dit, le degré d'un sommet est le nombre d'arêtes dont ce sommet est une extrémité
- Dans un graphe orienté, on parle de degré entrant et sortant selon l'orientation des arcs



- Degré de 1 : 3, Degré de 5 : 1
- 3, 4 et 2 sont les voisins de 1

Prédécesseur et Successeur

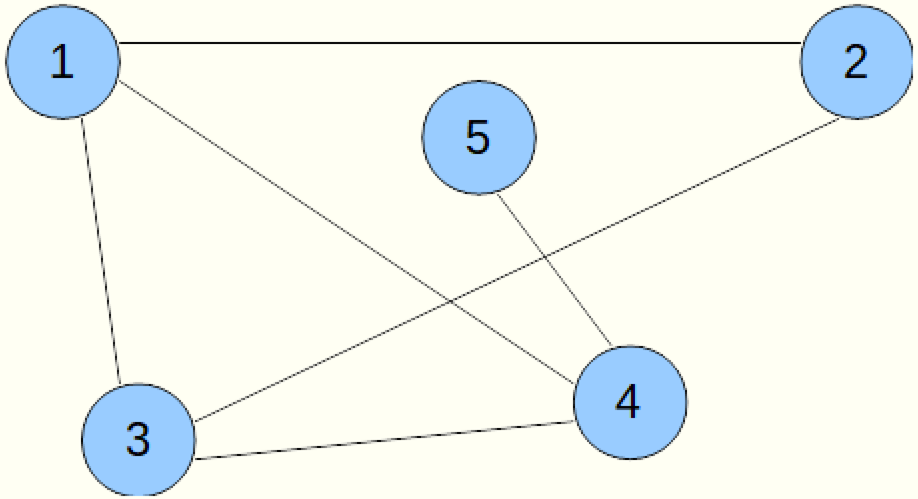
- *y* est un *prédécesseur* de *x*
  - Si l'arc (*y*,*x*) existe
- *y* est un *successeur* de *x*
  - Si l'arc (*x*,*y*) existe
- Quand le prédécesseur de *x* est unique
  - Ce sommet est le *père* de *x*
  - *x* est un *fils* de ce sommet



- 1 est un prédécesseur de 3
- 5 est un successeur de 4
- 1 est le père de 4
- 4 est le fils de 1

Chaîne et Cycle

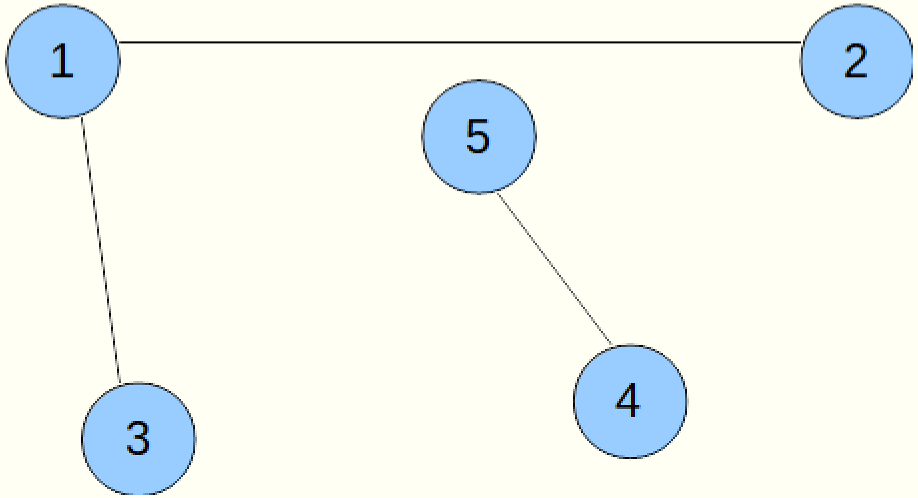
- Soit $G = (S, A)$ un graphe non orienté
- Une chaîne est une suite
  - $s_1 a_1 s_2 a_2 \ldots s_{k-1} a_{k-1} s_k$
  - Avec $s_i \in S$, $a_j \in A$ et $a_j = \{s_j, s_{j+1}\}$
- Autrement dit, une chaîne est une liste ordonnée de sommets telle que chaque sommet de la liste soit adjacent au sommet suivant
- Un cycle est une chaîne dont les deux extrémités coïncident et composée d'arêtes distinctes



- (2, 1, 3, 4, 5) est une chaîne de longueur 4
- (1, 2, 3, 4, 1) est un cycle de longueur 4

Graphe connexe

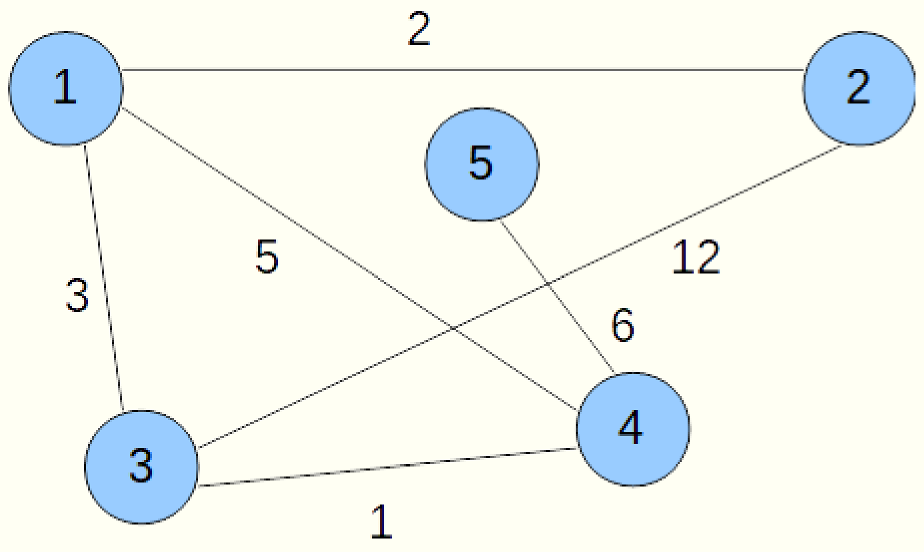
- Un graphe est connexe si, pour toute paire de sommets, il existe une chaîne les joignant
- Une composante connexe d'un graphe est un sous-graphe connexe maximal
  - On ne peut y ajouter d'autres sommets en conservant la connexité du sous-graphe
- À noter d'un arbre est un graphe connexe et sans cycle



- Ce graphe n'est pas connexe
- Ce graphe possède 2 composantes connexes
- Les 2 sous-graphes sont des arbres

Graphe pondéré

- Graphe dont les arêtes ou les arcs sont munis d'une valuation
  - Coût ou poids ou longueur
- Le coût (ou poids, ou longueur) d'un graphe partiel ou d'une chaîne est alors la somme des coûts des arêtes ou des arcs le constituant
- Une plus courte chaîne entre deux sommets est, parmi les chaînes qui la relient, une chaîne de poids minimum



- Le poids de la chaîne (2, 1, 3, 4, 5) est 12
- C'est la plus courte chaîne reliant 2 et 5

Représentation informatique

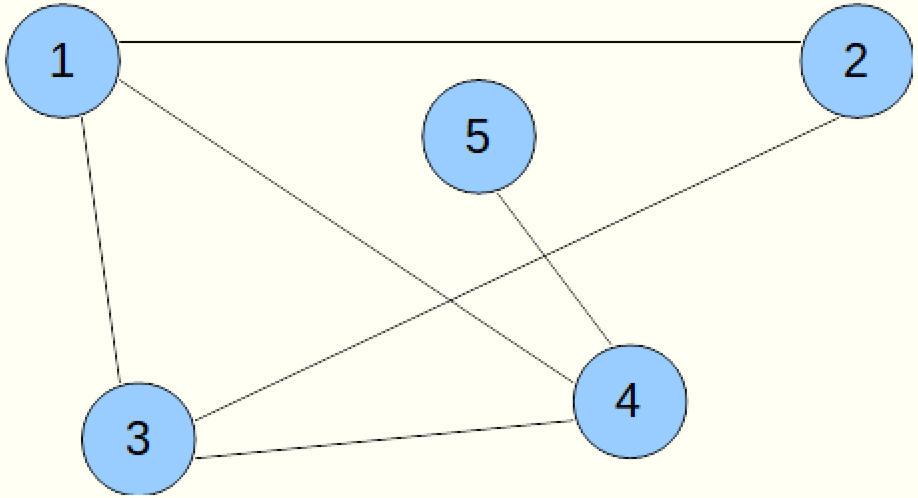
- On suppose les sommets de G numérotés par les entiers de 1 à $|S|$, chaque sommet étant repéré par son numéro
- Deux structures de données sont souvent utilisées pour représenter un graphe
  - Listes d'adjances
  - Matrice d'adjacences

Listes d'adjacences

- On suppose donnés (ex. lus dans un fichier)
  - L'ordre du graphe
  - L'ensemble des arêtes $\{x, y\}$ du graphe
- On définit un tableau *adj* de $|S|$ listes
  - Les $|S|$ cases du tableau correspondent aux sommets 1, 2, … $|S|$ du graphe
- Pour chaque $u \in S$
  - La liste $adj[u]$ est une liste de sommets v
  - …tels qu'il existe un arc $(u, v) \in A$
- Autrement dit
  - Le pointeur en position u est la tête d'une liste chaînée qui contient les sommets adjacents au sommet u
  - Un voisin v de u doit appartenir à la liste correspondant au sommet u
- *adj* → attribut du graphe

Listes d'adjacences

- Somme des longueurs de toutes les listes d'adjacences du graphe
  - Graphe orienté → $|A|$
  - Graphe non orienté → $2|A|$
- Quantité de mémoire requise
  - $O\ (S + A)$
- Graphes pondérés
  - On stocke le poids $p(u,v)$ de l'arc (u,v) avec le sommet v dans la liste de u

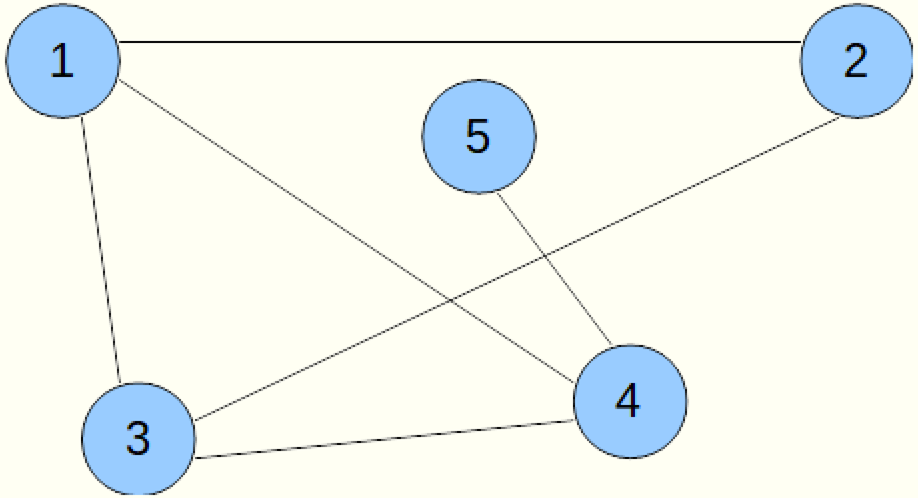


G.adj

1 → 2 → 3 → 4
2 → 1 → 3
3 → 1 → 2 → 4
4 → 1 → 3 → 5
5 → 4

Matrice d'adjacences

- On suppose donnés (ex. lus dans un fichier)
  - L'ordre du graphe
  - L'ensemble des arêtes $\{x, y\}$ du graphe
- On définit une matrice carrée *adj* à $|S|$ lignes et $|S|$ colonnes, telle que
  - $adj[i, j] = 1$ si $(i, j) \in A$ (i et j sont adjacents)
  - $adj[i, j] = 0$ sinon
  - $adj[k, k]$ sera pris égal à 0 ou 1 selon le problème
- Quantité de mémoire requise
  - $O(S^2)$ (peu importe la densité du graphe)
- Graphe non orienté
  - $adj[i,j] = adj[j,i] = 1$
- Graphe pondéré
  - On remplace le 1 par le poids $p(u,v)$ de l'arc (u,v)



adj	1	2	3	4	5
1	0	1	1	1	0
2	1	0	1	0	0
3	1	1	0	1	0
4	1	0	1	0	1
5	0	0	0	1	0

Listes d'adjacences ou matrice d'adjacences ?

- Taille du graphe
  - On peut sauver de l'espace mémoire avec les listes d'adjacences si le graphe est peu dense
- Recherche d'un arc
  - Listes d'adjacences → parcours d'une liste → O(n)
  - Matrice d'adjacences → accès direct → $O(1)$
- Dépend de l'algorithme que l'on veut implémenter

PARCOURS DE GRAPHES

Parcours de graphe ?

- Sert de base à plusieurs algorithmes
- Permet d'étudier les propriétés du graphe
  - Le graphe est-il connexe ?
  - Le graphe est-il biparti ?
- On pourra aussi faire des traitements sur les sommets et les arcs/arêtes durant le parcours
- 2 types de parcours
  - Largeur
  - Profondeur

Parcours en largeur

- Soit un graphe $G = (S,A)$ et un sommet origine s
- Emprunte les arcs de G pour découvrir tous les sommets accessibles depuis s
- Calcule la distance (plus petit nombre d'arcs)
  - Entre s et chaque sommet accessible
- Construit un **arbre de parcours en largeur**
  - De racine s
  - Qui découvre tous les sommets situés à une distance k avant les sommets de distance $k + 1$
- Durant le parcours, chaque sommet devient successivement
  - Non Découvert
  - Découvert
  - Découvert et tous ses sommets adjacents ont été découverts
- Utilise une file pour gérer la découverte des sommets
- Exemple 1 : Parcours en largeur

Temps d’exécution du parcours en largeur

- Initialisation de S sommets
  - $O(S)$
- Chaque sommet est enfilé une fois et défilé une fois
  - $O(S)$
- Chaque liste d'adjacence est parcourue une fois (quand le sommet est défilé)
  - Somme des longueurs de toutes les listes $\rightarrow$ O(A)
- Total
  - $O(S + A)$

Parcours en profondeur

- On descend plus profondément dans le graphe chaque fois que c'est possible
  - On explore les arcs du sommet découvert le plus récemment
  - Si on trouve un sommet non découvert, on l'explore tout de suite même si on n'a pas exploré tous les autres arcs du sommet en cours
  - On revient en arrière plus tard pour explorer les arcs restants
- Construit une forêt de parcours en profondeur
- Le parcours en profondeur date chaque sommet
  - Date de début → découverte du sommet
  - Date de fin → toute la liste d'adjacence du sommet a été examinée
- Exemple 2
  - Parcours en profondeur

Temps d'exécution du parcours en profondeur

- Initialisation de S sommets
  - $O(S)$
- Chaque liste d'adjacence est parcourue une fois
  - Somme des longueurs de toutes les listes → O(A)
- Total
  - $O(S + A)$

PROCHAIN COURS

ARBRES COUVRANT DE POIDS MINIMUM