MA351 : Introduction à la théorie des graphes

Graphes non orientés

Yann Kieffer

ESISAR

Motivation pour les graphes non orientés

Certaines structures finies relationnelles font intervenir uniquement des relations symétriques.

En chimie...

Une molécule peut être représentée comme un graphe dont les sommets sont des atomes ; la relation entre deux atomes ("liaison covalente") est symétrique.

Réseaux sociaux

Si on s'intéresse à une population d'individus ("les sommets"), ainsi qu'à la relation "x et y se connaissent", nous sommes face à une relation symétrique, et donc un graphe non-orienté.

Définition formelle de graphe non-orienté

A partir de la notion de graphe orienté

Un graphe orienté $\overrightarrow{G} = (V, A)$ est dit *symétrique* si : $\forall \overrightarrow{xy} \in A, (y, x) \in A$.
Un *graphe non-orienté* est un graphe orienté symétrique sans boucle.

Exemples :

- $\overrightarrow{G}_1 = (\{a, b, c\}, \{\overrightarrow{ca}, \overrightarrow{bc}, \overrightarrow{ac}, \overrightarrow{cb}\})$
- $\overrightarrow{G}_2 = (\{a, b, c\}, \{\overrightarrow{ba}, \overrightarrow{ab}\})$
- $\overrightarrow{G}_3 = (\{a, b, c\}, \emptyset)$

S'agit-il de graphes non-orientés ?

- $\overrightarrow{G}_4 = (\{a, b, c\}, \{\overrightarrow{ca}, \overrightarrow{bc}, \overrightarrow{ab}\})$
- $\overrightarrow{G}_5 = (\{a, b, c\}, \{\overrightarrow{ba}, \overrightarrow{ab}, \overrightarrow{bc}, \overrightarrow{cc}\})$
- $\overrightarrow{G}_6 = (\{a, b, c, d\}, \{\overrightarrow{ba}, \overrightarrow{cb}, \overrightarrow{bd}, \overrightarrow{ab}, \overrightarrow{db}, \overrightarrow{bc}\})$

Définition équivalente

Définition de graphe non-orienté

Un graphe non-orienté est un couple $G = (V, E)$ où :

- V est un ensemble fini quelconque, appelé *ensemble de sommets* ;
- E est un ensemble de paires d'éléments de V, appelé *ensemble d'arêtes*.

Une *paire* est un ensemble de deux éléments. On note ab l'arête $\{a, b\}$.

Exemples :

- $G_1 = (\{a, b, c\}, \{ca, bc\})$
- $G_2 = (\{a, b, c\}, \{ab\})$
- $G_3 = (\{a, b, c\}, \emptyset)$

Quiz

Si $G' = (V, E)$ représente via cette définition le même graphe que $\overrightarrow{G} = (V, A)$ via la première définition, quelle relation numérique y a-t-il entre $|E|$ et $|A|$?

Relations de voisinage, degrés

Si $xy = \{x, y\}$ est une arête d'un graphe, on dit que :

- x et y sont *adjacents* ;
- x et y sont *voisins* ;
- y est un *voisin* de x ;
- l'arête xy et le sommet x sont *incidents* ;
- x et y sont les *extrémités* de l'arête xy.

Si x est un sommet d'un graphe, on note :

- $N(x)$ l'ensemble de ses voisins ;
- $d(x)$ le *degré* de x, qui est défini comme le nombre de ses voisins.

. . . et on appelle :

- *sommet isolé* tout sommet de degré 0 ;
- *feuille* ou *sommet pendant* tout sommet de degré 1.

Chaînes et cycles

Chaîne dans un graphe non-orienté :

Une chaîne C de a à b dans le graphe non-orienté $G = (V, E)$ est une succession de sommets de G : $C = (x_0, \ldots, x_k)$ telle que :

- $x_0 = a$;
- $x_k = b$;
- $\forall i \in \{1, \ldots, k\} : x_{i-1}x_i \in E$.

La *longueur* d'une telle chaîne est k (nombre d'arêtes traversées).

Propriétés des chaînes :

Une chaîne $C = (x_0, \ldots, x_k)$ est dite :

- *simple* si $\forall i < j, x_{i-1}x_i \neq x_{j-1}x_j$;
- *close* si $x_0 = x_k$.

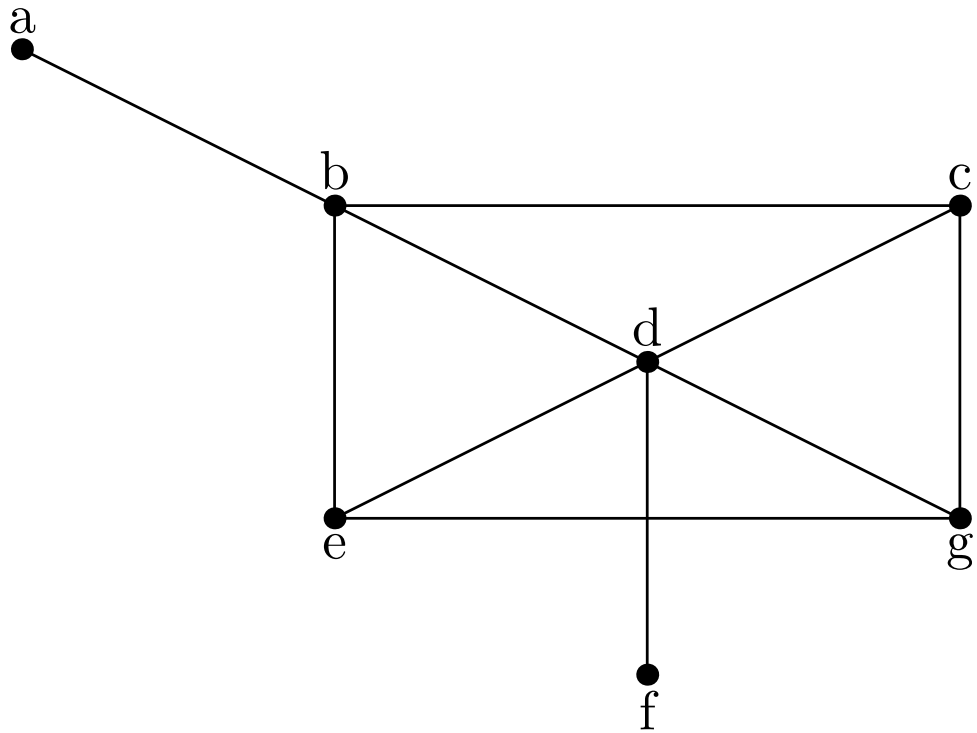
Cycle dans un graphe non-orienté :

Un cycle C est une chaîne simple, close, de longueur > 0.

Chaînes et cycles : exemples

Voici quelques chaînes :

- $C1 = (e, d, c, g)$;
- $C2 = (b, a, b, d, e)$;
- $C3 = (d, c, g, d, b, e)$;
- $C4 = (e, b, d, e)$;
- $C5 = (b, c, d, g, c, d)$;
- $C6 = (f, d)$;
- $C7 = (f)$;
- $C8 = (c, g, d, e, b, d, c)$;



Lesquelles sont simples ?
Lesquelles sont closes ?
Lesquelles sont des cycles ?

Connexité

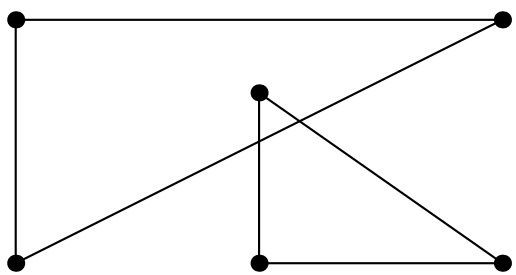
Composantes connexes d'un graphe non-orienté

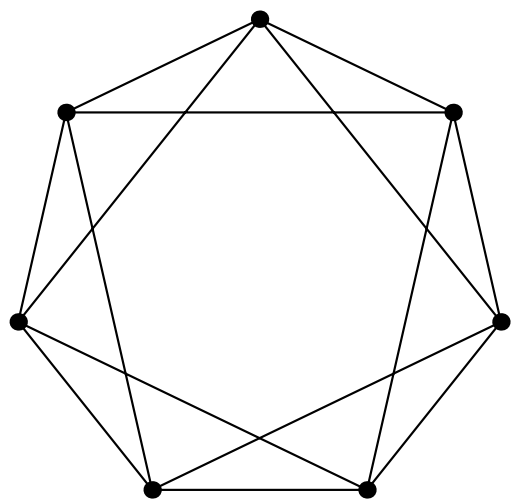
Les *composantes connexes* du graphe $G = (V, E)$ sont les composantes fortement connexes du graphe orienté $\overrightarrow{G} = (V, A)$ correspondant.

Connexité d'un graphe

Un graphe non-orienté est dit *connexe* si et seulement s'il a exactement une composante connexe.

Exemples.





Détection de cycles ou feuilles

```
1  Cycle_ou_feuille{G = (V, E), x ∈ V, xy ∈ E)}:
2  – M ← {x, y}
3  – o ← {(y, x)}
4  – cycle ← Faux
5  – TantQue d(y) ≠ 1 et cycle == Faux , Faire :
6      * soit z un sommet de N(y) \ {x}
7      * x ← y
8      * y ← z
9      * Si y ∈ M ,
10       Alors : cycle ← Vrai ;
11       Sinon :
12           • M ← M ∪ {y}
13           • o ← o ∪ {(y, x)}
14 – Si d(y) == 1 ,
15     Alors : Retourner (" Feuille ", y );
16     Sinon : Retourner (" Cycle ",
17                 Ajout_en_tete(x , Reconstruire_chemin(o, y, x ))
```

Théorème : correction de l'algorithme Cycle_ou_feuille()

Théorème

A l'issue de l'appel :
$(s, r) =$ Cycle_ou_feuille($G = (V, E)$, $v \in V$, $xy \in E$),
s reçoit la valeur "Cycle" ou "Feuille", et selon la valeur de s,
r contient soit un cycle, soit une feuille de G.

Début de la preuve de correction Cycle_ou_feuille()

Exécutabilité

En parcourant ligne à ligne l'algorithme, on trouve deux difficultés :
L.6 : soit z un sommet de $N(y) \setminus \{x\}$:
puisque nous sommes à l'intérieur de la boucle, nous savons que $d(y) \neq 1$; et comme x et y sont voisins (qu'on entre pour la première fois dans la boucle, ou qu'on y soit déjà passé), on sait que $d(y) \neq 0$. Donc $d(y) \geq 2$, ce qui suffit à garantir que $N(y) \setminus \{x\}$ n'est pas vide.
L.17 : appel Reconstruire_chemin(o, y, x) :
Il faut s'assurer que cet appel s'exécute correctement. Notez qu'ici, "$s == y$ et $t == x$". Il est recommandé d'avoir une illustration sous les yeux pour suivre la preuve !
Rappelons-nous les conditions d'une bonne exécution de Reconstruire_chemin() : il faut que pour les valeurs successives prises par la variable de cette routine, on trouve une valeur dans la fonction $o()$, sauf peut-être pour la valeur s. Et il faut pouvoir garantir l'arrêt de cette routine en exhibant un tri topologique des arcs représentés par $o()$.

Exécutabilité : exécutabilité de Reconstruire_Chemin()

Ces deux parties se traîtent d'une manière similaire à notre précédente preuve concernant Reconstruire_Chemin() au chapitre Accessibilité ; commençons par la bonne exécutabilité de Reconstruire_Chemin().
Si on note $x_0, \ldots, x_k$ les valeurs successives de la variable x, et $y_0, \ldots, y_k$ les valeurs successives de y dans l'algorithme Cycle_ou_Feuille(), alors :

- une telle exécution fait k passages dans la boucle ;
- $x = x_k$ et $y = y_k$;
- (y_i, x_i) est ajouté à $o()$ à l'étape i ;
- les valeurs retournées par $o()$ sont toutes dans M ;
- le seul élément de M n'ayant pas de valeur pour la fonction $o()$ est x_0 ;
- si Reconstuire_Chemin() atteint x_0, c'est que $y = y_k = x_0$; il s'arrêtera donc sur ce sommet ;
- pour tous les autres sommets rencontrés par Reconstruire_Chemin(), il s'agit de sommets dans $M \setminus \{x_O\}$, qui ont donc bien une valeur pour $o()$.

Exécutabilité : arrêt de Reconstruire_Chemin()

Pour prouver l'arrêt de Reconstruire_Chemin(), il suffit d'exhiber un tri topologique du graphe $\overrightarrow{H}$ défini comme le miroir du graphe orienté (M, o).
Il suffit d'affecter -1 à x_0, puis i à y_i pour obtenir un tel tri topologique.
La vérification que c'est bien un tri topologique vous est laissée à titre d'exercice.

Arrêt de Cycle_ou_Feuille()

Arrêt

La seule problématique d'arrêt est la boucle TantQue, lignes 5 à 13. Observez qu'à chaque fois qu'on choisit une arête xy, y n'est pas encore dans M ; puis y est placé dans M. Comme ce sont les seules modifications de M, et que $M \subseteq V$, on en déduit que le nombre de passages dans la boucle est borné par le nombre d'éléments dans V.

Correction des valeurs retournées

Si l'algorithme retourne ("Feuille", y) :
On n'atteint ce cas que si $d(y) = 1$: y est donc bien une feuille de G.

Si l'algorithme retourne ("Cycle", C) :
Dans ce cas, l'algorithme a quitté la boucle avec $d(y) \neq 1$; et donc nécessairement, $cycle = Vrai$.
Or Reconstuire_Chemin(o, y, x) renvoie une chaîne C' de y à x de G. Or xy est une arête de G, en raison du choix de z. Donc l'ajout en tête de x à C' produit bien une chaîne C de G. Cette chaîne est close (elle va de x à x) ; et elle est de longueur strictement positive (puisqu'elle emprunte l'arête xy).
Il nous reste à vérifier qu'elle est simple.

Correction des valeurs retournées : suite et fin

Cela découle directement de l'analyse de l'arrêt de Reconstuire_Chemin() : de même que l'existence d'un tri topologique pour le graphe $\overrightarrow{H}$ nous garantit l'arrêt de cette routine, il nous garantit également qu'elle ne passera pas deux fois par le même sommet. Donc la chaîne produite est nécessairement simple. Il faut compléter ceci par l'observation que l'arête xy qui est rajoutée à C' n'est pas une arête de C'.
Donc C est une chaîne simple, close, et de longueur non-nulle : c'est donc un cycle.

Conclusion et moralité

L'algorithme Cycle_ou_Feuille() trouve dans tout graphe non orienté ayant au moins une arête, soit un cycle, soit une feuille. Attention : un graphe non orienté peut avoir les deux simultanément, comme vous pourrez facilement le vérifier de quelques coups de crayon.
Il n'est donc pas vrai qu'un graphe qui a une feuille n'a pas de cycle, ou inversement. Malgré tout, la réponse fournie par l'algorithme est argumentée par un certificat : une feuille en cas de réponse "Feuille", un cycle en cas de réponse "Cycle".
Mais que peut-on alors tirer de ce théorème ? Ce théorème-ci :

Théorème

Un graphe non orienté qui a une arête, mais pas de feuille, a forcément un cycle.

. . .ou encore :

Théorème

Un graphe non orienté ayant une arête, mais pas de cycle, a forcément une feuille.

Conclusion et moralité

Les voici à nouveau :

Théorème

Un graphe non orienté qui a une arête, mais pas de feuille, a forcément un cycle.

Théorème

Un graphe non orienté ayant une arête, mais pas de cycle, a forcément une feuille.

Le premier de ces théorèmes est le point de départ de la théorie sur les graphes dits *eulériens* : les graphes dont les arêtes sont celles d'un unique cycle passant une fois et une seule par chaque arête.
Le second est le point de départ de la théorie des arbres.