Programmation d'une Intelligence Artificielle : arbres ou apprentissage automatique

Problèmes sur les Graphes.

Paul Mangold

L3 MIASHS

12 Février 2021

Introduction

Pour l'instant on a vu plusieurs problèmes sur les graphes :

- parcours de graphes ;
- recherche de chemins ;
- arbres couvrants.

On a aussi évoqué des structures comme l'arbre binaire de recherche, etc.

On peut trouver des solutions à ces problèmes en temps polynomial !

$\rightarrow$ sur une entrée de taille n on a un algorithme qui donne une réponse en $O(n^k)$ pour un entier $k > 0$.

Par exemple, sur un graphe $G = (V, E)$:

- parcours de graphes : $O(|V| + |E|)$;
- recherche de chemins : $O(|V| + |E|)$;
- arbres couvrants : $O(|E| \log |E|)$.

Il existe des problèmes pour lesquels on ne trouve pas de solution polynomiale de façon déterministe...

Par exemple, pour le morpion, on est obligé-es de parcourir une bonne partie de l'arbre du jeu. On peut l'exprimer de deux façons :

- pour **vérifier** si séquence de coups est gagnante, il suffit de jouer les coups et de voir qui gagne ;
- pour **trouver une solution** on doit parcourir tout l'arbre du jeu (on vérifie donc toutes les séquences de coups possibles) ;

Ainsi, on peut classer les problèmes par complexité, par exemple :

- P : on peut **trouver une solution** en temps polynomial ($O(n^k)$) ;
- NP : on peut **vérifier qu'une solution est bonne** en temps polynomial ($O(n^k)$) ;
- $EXPTIME$: on peut **trouver une solution** en temps exponentiel ($O(k^n)$).

On appelle ces ensembles de problèmes des **classes de complexité**.

On sait que

$$P \subseteq NP \subsetneq EXPTIME,$$

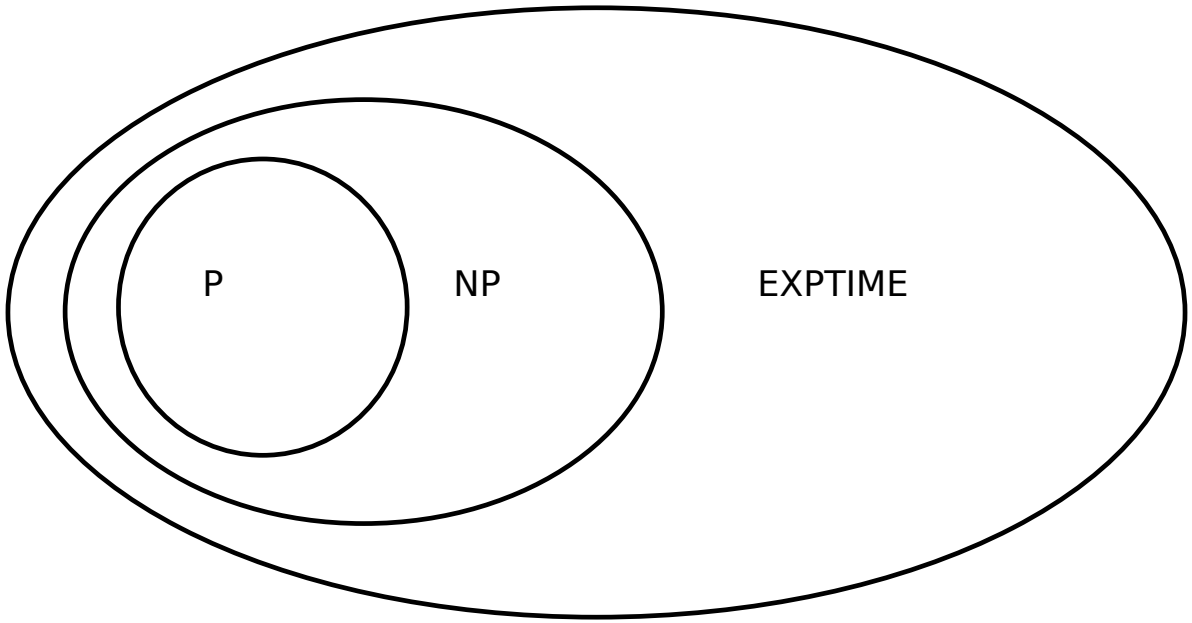
C'est à dire que :

- si on peut trouver une solution en temps polynomial, on peut bien sûr vérifier qu'une solution marche en temps polynomial ;
- si on peut vérifier qu'une solution marche en temps polynomial, on peut en trouver une en temps exponentiel.

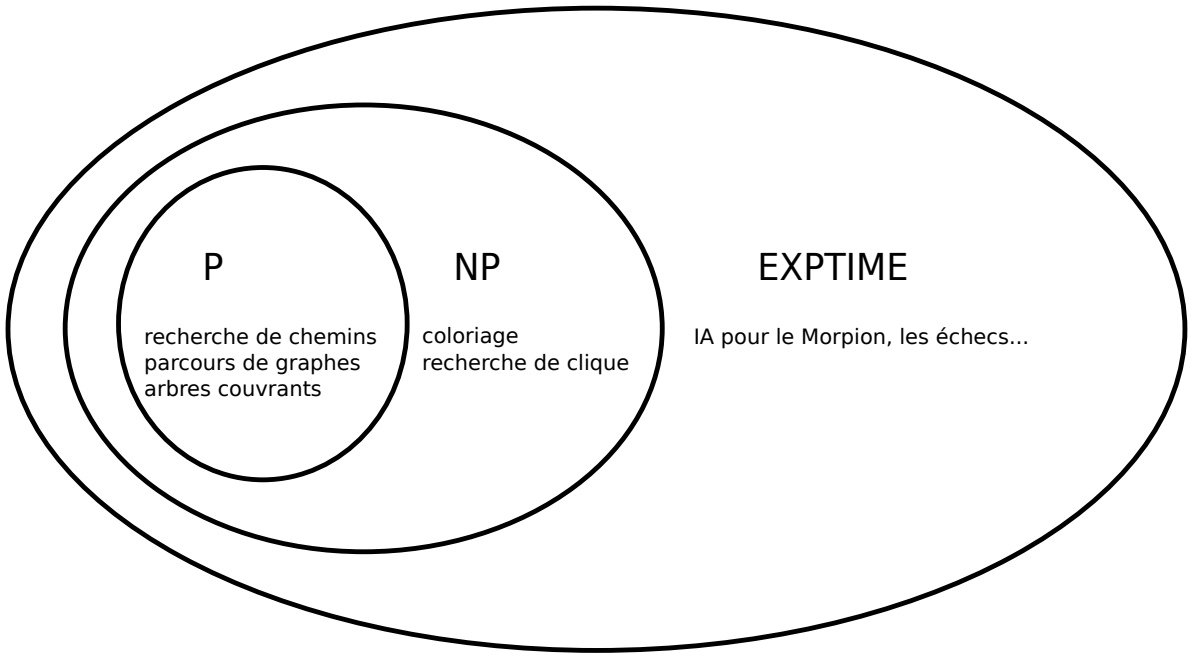
Petite parenthèse : $P = NP$ reste une question ouverte.

En d'autres termes : on ne sait toujours pas s'il est fondamentalement plus difficile de trouver une solution que de vérifier si une solution est bonne.

Pour résumer :



Donc, dans notre cas, on avait :

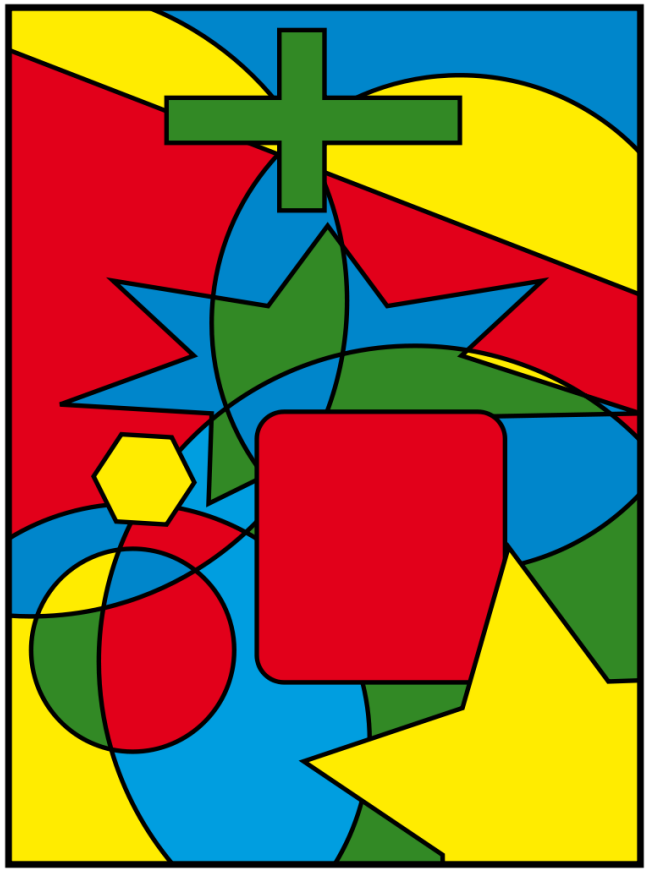


Aujourd'hui, on va s'intéresser à deux problèmes :

- la recherche de coloriage de graphes ;
- la recherche de cliques.

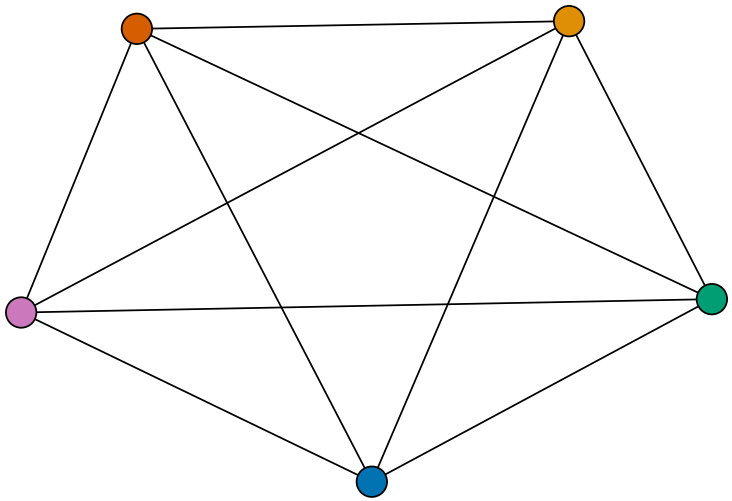
Graphes Planaires et Coloriages

Au premier cours, je vous avais montré ce vitrail :

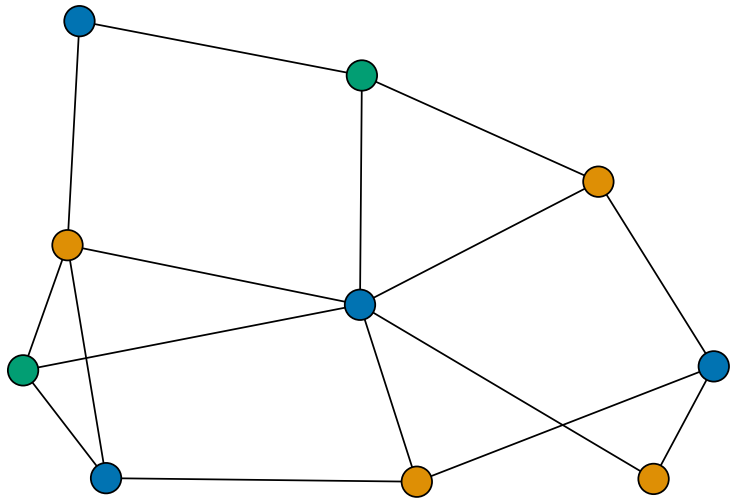


où il y a 4 couleurs, et où deux faces qui sont en contact sont de couleurs différentes.

En fait, on appelle **coloriage d'un graphe** une façon d'attribuer une couleur à chacun des sommets d'un graphe de façon à ce que deux sommets reliés par une arête soient de couleur différente :

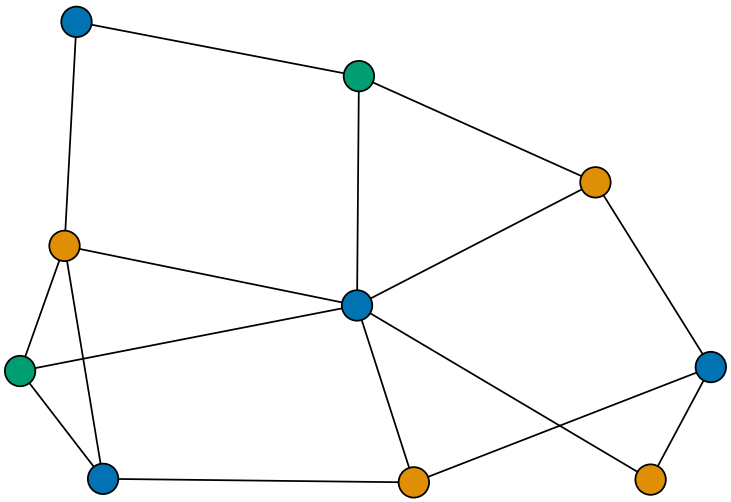


(a) 5 couleurs

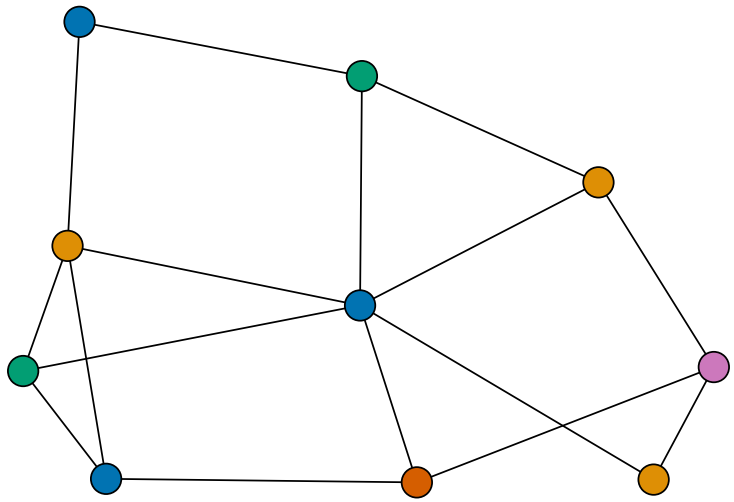


(b) 3 couleurs

Souvent, on cherche à avoir un coloriage avec le moins de couleurs possibles :



(a) 3 couleurs



(b) 5 couleurs

Le nombre minimal de couleurs pour colorier un graphe est appelé **le nombre chromatique** du graphe.

On peut trouver un coloriage facilement, par exemple :

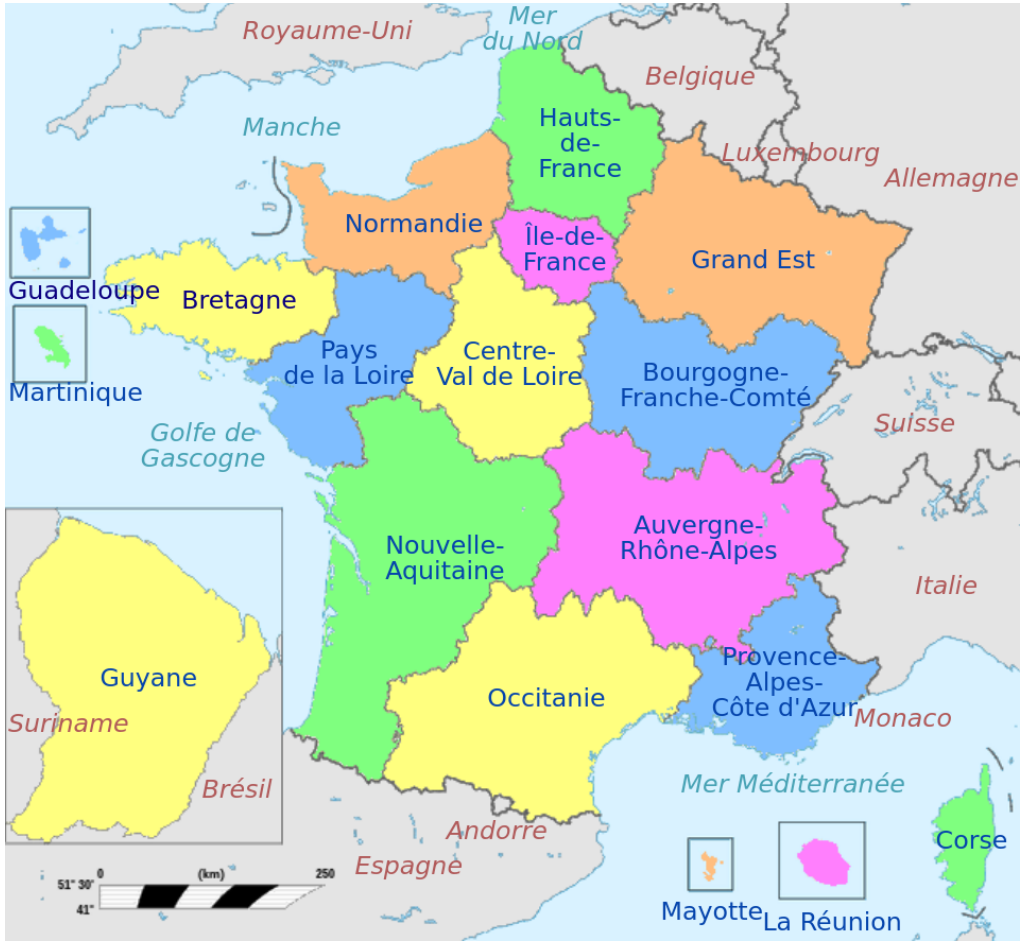
- au début, aucun sommet n'est coloré ;
- pour chaque sommet $s \in V$:
  - regarder les couleurs voisines ;
  - colorier le sommet s avec la première couleur non utilisée par ses voisins.

C'est un algorithme de **coloriage glouton**, il donne un coloriage, mais il n'a pas nécessairement le nombre minimal de couleurs.

En fait, le problème du coloriage avec le moins de couleurs possible est un problème **NP-complet**.

$\rightarrow$ c'est à dire qu'il est au moins aussi difficile que les autres problèmes de la classe de complexité NP.

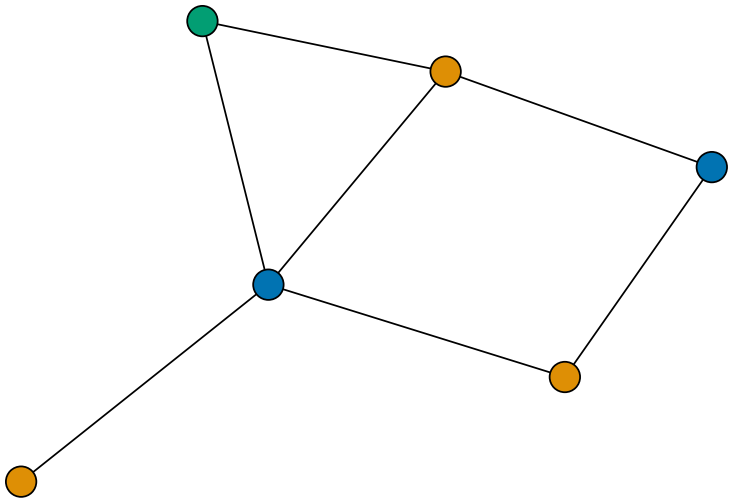
À l'origine, on coloriait plutôt des cartes que des graphes.



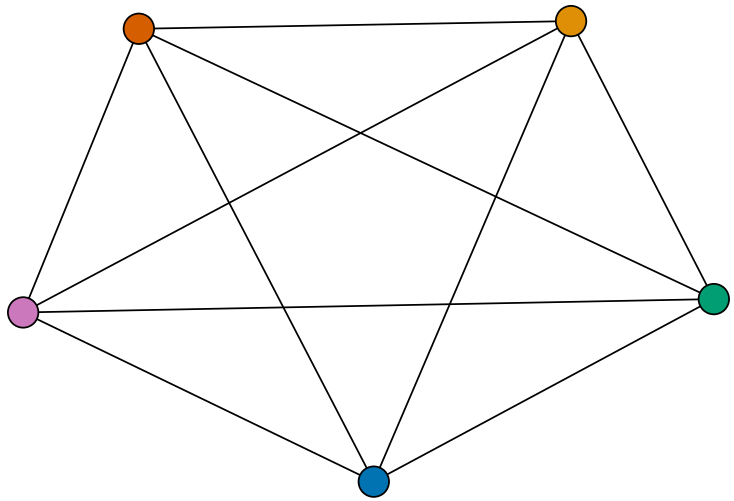
On peut représente une carte par un graphe : chaque région est un sommet, et les arêtes représentent les régions voisines.

En fait ces graphes sont très particuliers : on les appelle des **graphes planaires**.

Un graphe est **planaire** si on peut le représenter dans le plan sans que ses arêtes se croisent :



(a) planaire,



(b) non-planaire.

Theorem (Théorème des Quatre Couleurs)

Tout graphe planaire peut-être colorié avec au plus quatre couleurs.

ie. le nombre chromatique d'un graphe planaire est au plus 4.

ie. on peut colorier une carte avec 4 couleurs.

Cependant trouver un tel coloriage reste difficile...

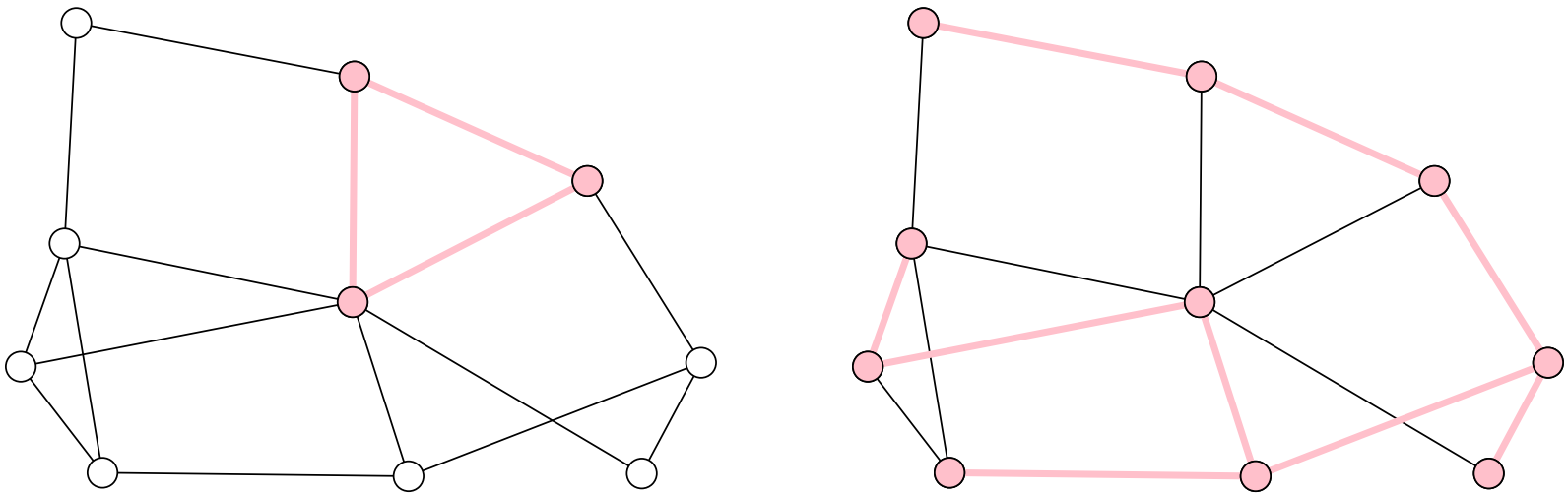
En revanche, on peut colorier un graphe planaire avec 5 couleurs en temps linéaire.

Cliques

Pour deux graphes $G = (V, E)$ et $G' = (V', E')$, on dit que G est un **sous-graphe** de G' si :

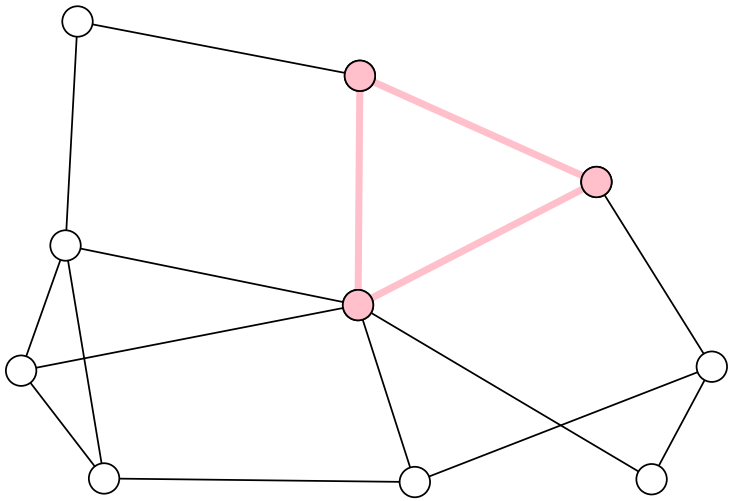
- $V \subseteq V'$;
- $E \subseteq E'$.

Autrement dit, les arêtes et sommets de G le sont aussi dans G' :

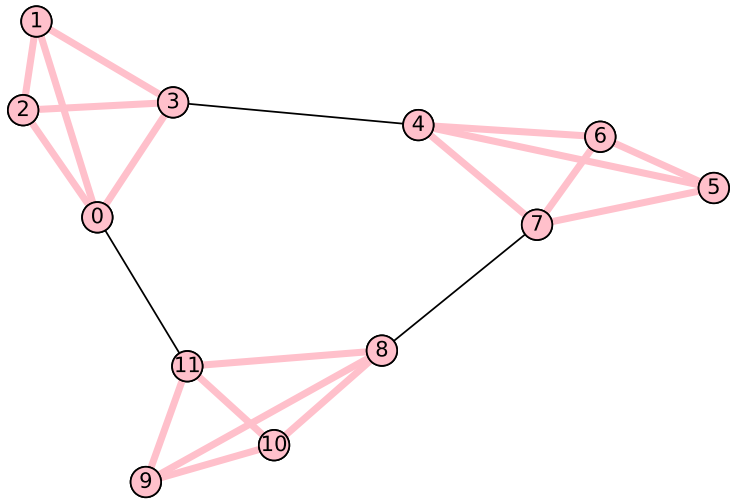


Un arbre couvrant un graphe G est un sous-graphe de G !

Un sous-graphe complet d'un graphe est appelé une **clique**.



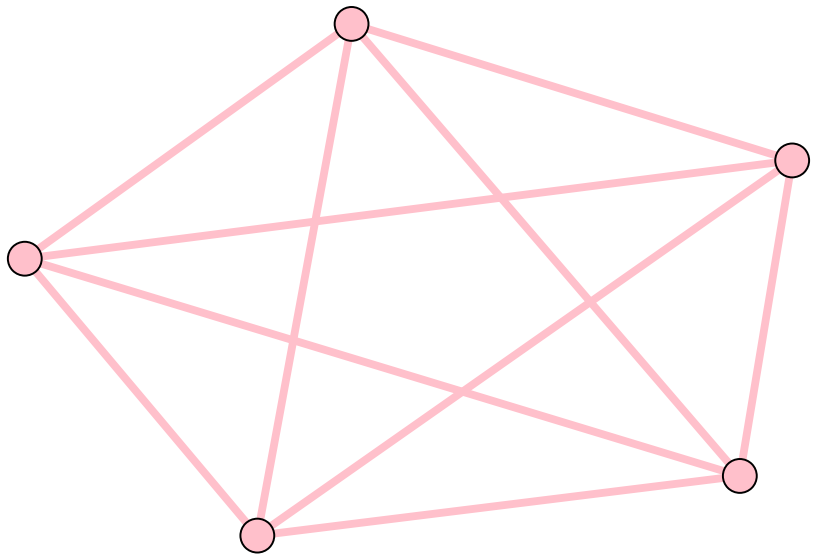
(a) clique de taille 3,



(b) cliques de taille 4.

Dans un graphe planaire, les cliques sont de taille au plus 4.

Sinon ce sous-graphe est dans le graphe :



et il n'est pas planaire.

Dans des réseaux sociaux, une clique peut représenter un ensemble d'ami-es, une communauté particulière, etc., en supposant que toustes les ami-es d'un groupe se connaissent.

On peut se poser plusieurs questions :

- quelle est la taille d'un groupe d'ami-es d'une personne dans un réseau social ?
- quel est le plus grand groupe d'ami-es du réseau ?
- mêmes questions si l'on considère que "les ami-es de mes ami-es sont aussi mes ami-es".

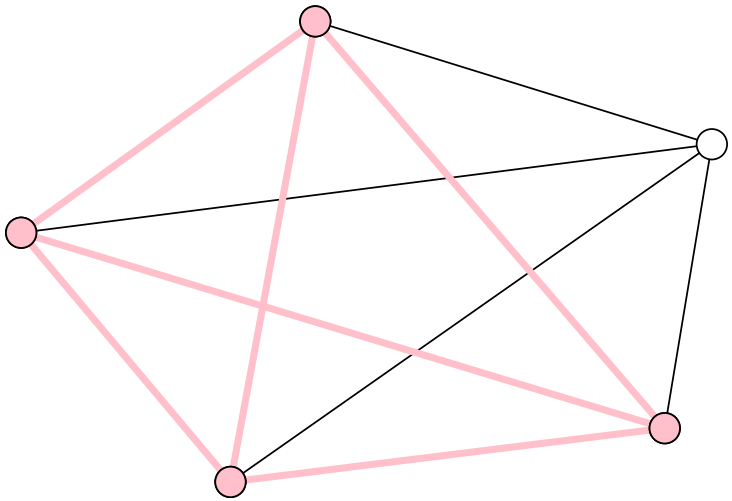
"Quelle est la taille d'un groupe d'ami-es d'une personne ?"

Ici, on entend par "groupe d'ami-es" une clique !

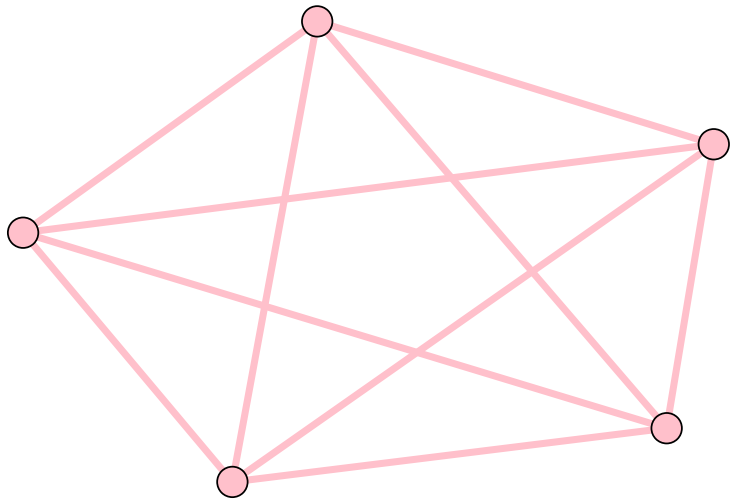
$\rightarrow$ Cela revient à chercher une "grande clique" dont fait partie cette personne !

On appelle **clique maximale** une clique qui n'est pas incluse dans une clique plus grande.

Remarque importante : toute sous-clique est incluse dans une clique maximale :



(a) clique de taille 4,



(b) clique maximale.

$\rightarrow$ cela permet de créer un algorithme efficace de **recherche de clique maximale** (voir TP).

C'est facile de trouver une clique... mais en trouver une de taille k dans un graphe est un problème **NP-complet**.

En fait, on est obligé-es de regarder une bonne partie des sous-graphes de taille k, ce qui demande très vite beaucoup de calculs.

TP

TP

Dans le TP on va :

- chercher des cliques maximales ;
- chercher des cliques où les arêtes sont en fait des chemins de longueur k ;
- utiliser la représentation d'un graphe sous forme matricielle.