Algorithmie Avancée
TP/TD Préliminaire

A. Premières réflexions sur les représentation de problèmes par graphes

Exercice 1 : On souhaite constituer 5 groupes de travail A, B, C, D, E. Deux groupes distincts devront forcément avoir un et un seul membre en commun. Une même personne fera partie d'exactement deux groupes de travail. Combien de personnes doit comporter chaque groupe de travail ? Combien faut-il de personnes au total ?

Exercice 2 : Résolution d'un sudoku

Le jeu du Sudoku consiste à compléter une grille de 9 carrés de 3 * 3 cases chacun, en attribuant à chaque case un chiffre (de 1 à 9) de telle sorte qu'un même chiffre n'apparaisse qu'une et une seule fois sur chaque ligne, sur chaque colonne et dans chaque carré de la grille. Formaliser le problème sous forme de graphe. A quelle problématique correspond la résolution d'une grille de sudoku ? Quelle est l'ordre du graphe et sa taille ?

Exercice 3 : Coloration de cartes

On souhaite colorier une carte de telle sorte que deux pays frontaliers n'aient jamais la même couleur. C'est un problème de coloration de graphe.

1. Modéliser le problème par un graphe : que représentent les sommets ? Quelle est la relation d'adjacence entre deux sommets ?

2. Voici une liste de 11 pays européens et pour chacun d'eux les pays de cette liste qui leur sont frontaliers :

France (Fr) : Espagne, Andorre, Italie, Suisse, Allemagne, Luxembourg, Belgique
Espagne (Es) : France, Andorre, Portugal
Portugal (Po) : Espagne
Andorre (An) : Espagne, France
Italie (It) : France, Suisse, Autriche
Autriche (Au) : Italie, Suisse, Allemagne
Suisse (Su) : France, Italie, Autriche, Allemagne
Allemagne (Al) : France, Suisse, Autriche, Luxembourg, Belgique, Pays-Bas
Luxembourg (Lu) : France, Belgique, Allemagne
Belgique (Be) : France, Luxembourg, Allemagne, Pays-Bas
Pays-Bas (PB) : Belgique, Allemagne

- Dessiner le graphe correspondant au problème de coloration.
- Construire la matrice d'adjacence correspondante.
- Appliquer l'algorithme de Welsh et Powell pour trouver une coloration propre.
- L'algorithme fournit-il le nombre chromatique ?

De façon générale, que sait-on résoudre en matière de coloration de graphe en un temps raisonnable et que ne sait-on pas résoudre ?

B. Structure de données

Vous reprendrez les exercices des deux derniers TP de Programmation Impérative : programmer en C des listes doublement chaînées et des arbres binaires. Vous restructurez votre code si ce n'est déjà fait avec des fichiers *maListe.c* pour l'exécutable final (*fonction main()*), *liste.c* pour les fonctions génériques liées aux listes, et un fichier de prototypage des structures et fonctions associées *liste.h* (principe de compilation séparée voir cours de Génie Logiciel modularité)

Comment étendre la structure de données choisies pour un arbre générique ? Un graphe quelconque ?

S'inspirer de http://nicolasj.ftp-developpez.com/listedouble.pdf

https://nicolasj.developpez.com/articles/listedouble/ **C. Activités supplémentaires**

Exercice 1 : Soit l'algorithme de construction de graphe planaire suivant (voir animation sur Planarity.net) :

```
fonction G = planaire(G)
tant que G n'est pas planaire faire
        pour tout x sommet de G faire
                V = liste des voisins de x
                M = barycentre des sommets de V
                si on diminue le nombre d'intersections d'arcs
                        alors placer x en M
                fin si
        fin faire
fin faire
```

Quelle est la complexité de cet algorithme ? Finit-il toujours ?

Exercice 2 : Soit T un graphe avec n sommets. Démontrer que les propriétés suivantes sont équivalentes.

(1) T est un arbre.

(2) T est un graphe connexe et acyclique.

(3) T est un graphe connexe avec n−1 arêtes.

(4) T est un graphe acyclique avec n−1 arêtes.

Exercice 3 : Considérons trois définitions d'un arbre.

Définition 1 :
Un arbre T est un graphe :
■ qui est simple.
■ et tel que pour tout couple (u, v) de sommets de T (avec u différent de v), il existe un chemin simple (et un seul) de u à v.
On rappelle que cette définition repose sur les deux propositions suivantes :
Proposition 1.
S'il y a un chemin de v à w dans le graphe G, alors il y a aussi chemin simple de v à w.
Proposition 2.
S'il y a un cycle dans G alors il y a un cycle simple dans G.

Définition 2 :
Un graphe non-orienté connexe et acyclique est appelé un arbre.

Définition 3 :
Un arbre est un graphe connexe minimal. C'est-à-dire que toute suppression d'arêtes le rend non connexe.

En vous basant sur les définitions précédentes, démontrer le théorème suivant (en démontrant que les propriétés sont équivalentes).

Théorème - Les propriétés suivantes sont équivalentes.

(a) A est un arbre.
(b) A ne contient aucun circuit et possède n -1 arêtes.
(c) T est un arbre. A est connexe et possède n -1 arêtes.
(d) T est un arbre. A est connexe, et chaque arête est un pont.
(e) T est un arbre. Deux sommets quelconques de A sont reliés par un unique chemin.
(f) T est un arbre. A ne contient aucun circuit mais l'addition d'une quelconque nouvelle arête crée exactement un circuit.