

Théorie des graphes

TD/TP 1

Construire des graphes – Modéliser avec des graphes

Graphes d'états – Automates

Coloration de graphes

Matrice d'adjacence – Listes d'adjacence

Implémentation (C++)

I. Modéliser avec des graphes

Exercice 1 :

Montrer que dans un groupe de personnes, il y a au moins deux personnes ayant le même nombre d'amis présents (éventuellement 0).

Exercice 2 :

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'exactly deux groupes de travail. Combien de personnes doit comporter chaque groupe de travail ? Combien faut-il de personnes au total ?

Exercice 3 : 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.

							1	2
								3
	2	3				4		
	1	8						5
	6			7		8		
					9			
	8	5						
9				4		5		
4	7				6			

Formaliser le problème sous-forme de graphe. A quelle problématique correspond la résolution d'une grille de sudoku ?

II. Automates

Un automate fini déterministe est un quintuplé $(Q, \Sigma, \delta, q_0, F)$ constitué des éléments suivants :



- un alphabet fini (Σ)
- un ensemble fini d'états (Q)
- une fonction de transition ($\delta : Q * \Sigma \rightarrow Q$)
- un état initial ($q_0 \in Q$)
- un ensemble d'états finaux (ou acceptant) $F \subseteq Q$

L'automate prend en entrée un mot et l'accepte (le reconnaît) ou le rejette. Le langage associé à un automate est constitué de l'ensemble des mots qu'il reconnaît.

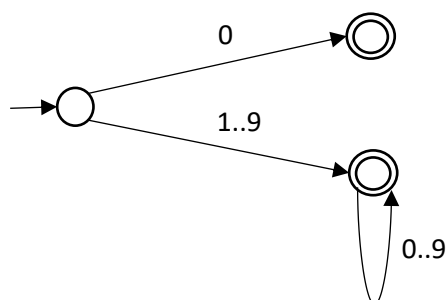
Fonctionnement de l'automate :

- Le processus commence à l'état de départ q_0 .
- Les symboles du mot sont lus les uns après les autres.
- À la lecture de chaque symbole, on utilise la fonction de transition δ pour se déplacer vers le prochain état.
- Le mot est reconnu si et seulement si le dernier état atteint est un état final.

On peut schématiser un automate par un graphe dont les sommets sont les états et dont les arcs représentent les transitions. Sur chaque arc est indiqué le symbole lu provoquant le changement d'état.

- Etat initial 
- Etats finaux 

Exemple : automate permettant de valider un nombre entier normalisé



Exercice 4 : Faire un automate qui reconnaît les nombres écrits en notation scientifique. En notation scientifique, les nombres sont écrits sous la forme d'une mantisse (nombre décimal a tel que $1 \leq |a| < 10$) et d'un exposant entier précédé d'un E. (exemples : 2, 7.3, 2E15, -3.8E-5)

III. Coloration de graphes

Implémentation en C++ de l'algorithme de Welsh et Powell

Exercice 5 : 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 vu en cours pour trouver une coloration propre.
 - L'algorithme fournit-il le nombre chromatique ?

Exercice 6 : Allocation de fréquences¹ - Implémentation C++

On désire attribuer des fréquences à n stations émettrices. A cause des interférences, deux stations distantes de moins de d_{min} km ne peuvent émettre avec la même fréquence. Connaissant la position des stations, combien faut-il de fréquences distinctes et comment les attribuer aux différentes stations ?

On peut représenter ce problème sous forme d'un graphe dans lequel les sommets sont les stations et deux stations sont adjacentes si elles sont trop rapprochées pour émettre à la même fréquence. On appellera ce graphe « graphe d'interférence ».

Il sera représenté par listes d'adjacence : chaque sommet possède la liste (un vecteur) de ses sommets adjacents (chaque station possède la liste des stations avec lesquelles elle interfère).

Un code de base ([TP1_MrFercoq_frequencies.rar](#)) vous est fourni :

- Etudiez-le.
- Complétez-le en implémentant les algorithmes de coloration vus en cours (algorithme naïf et algorithme de Welsh et Powell).

Pour trier les sommets par ordre de degré décroissant (algo de Welsh et Powell), vous pouvez utiliser la méthode de tri de la bibliothèque standard. L'instruction suivante trie le vecteur *m_stations* par ordre de degré décroissant :

```
std::sort(m_stations.begin(), m_stations.end(), [](Station* s1, Station* s2)
{
    return s1->getDegre() > s2->getDegre();
});
```

- Testez les algorithmes et comparez les résultats obtenus avec les exemples de réseaux donnés sous-forme de fichiers textes. (Dans chaque fichier sont indiqués, la distance minimale d_{min} , le nombre de stations, puis pour chaque station son numéro et ses coordonnées. Les codes de chargement du réseau et de la construction du graphe d'adjacence sont fournis (→ constructeurs).)

¹ Les fréquences disponibles sont une ressource rare. Il s'agit de minimiser le nombre de fréquences utilisées ou pour un nombre de fréquences donné, de minimiser les interférences. Les problèmes d'attribution de fréquences dans les réseaux hertziens (ou de longueurs d'onde dans les réseaux optiques) sont en réalité plus complexes. On rejoint des problèmes de multicoloration de graphes ou encore de coloration impropre.

- Un algorithme de recherche exhaustive (brute force) vous est également donné. Il permet d'obtenir une coloration optimale mais il ne peut être utilisé que sur des réseaux de petite taille car il est de complexité exponentielle.

Exercice supplémentaire (facultatif)

IV. Graphes d'états

Exercice 7 : Le jeu des allumettes.

Ce jeu se joue à deux, avec n tas de l allumettes. A tour de rôle chaque joueur retire un certain nombre d'allumettes de l'un des tas. Celui qui retire la dernière allumette perd la partie.

1. Modéliser ce jeu à l'aide d'un graphe pour $n=2$ et $l=3$, dans le cas où un joueur peut retirer 1 ou 2 allumettes à chaque fois.
2. Que doit jouer le premier joueur pour être sûr de gagner la partie ?