RECHERCHE OPERATIONNELLE

Projet

4ème année

Ingénierie Informatique et Réseaux

REALISER PAR : CHAKOUR Imad

TUTEUR DE L'ECOLE : BENOUAHMANE Brahim

CAMPUS : EMSI AGDAL 2

GROUPE : G4

PERIODE DU PROJET : DU 07/11/2024 AU 07/12/2024

# Résumé

Ce projet vise à résoudre le problème de planification des examens dans une institution académique en minimisant les conflits d'horaires entre les matières ayant des étudiants communs. Pour cela, nous représentons le problème sous forme d'un graphe, où chaque matière correspond à un sommet et chaque conflit entre deux matières est représenté par une arête. L'objectif est de déterminer le nombre minimum de créneaux horaires nécessaires, équivalent au nombre chromatique du graphe. L'algorithme de Welsh-Powell, basé sur une approche gloutonne, a été choisi pour élaborer un planning efficace en respectant les contraintes. Enfin, la solution développée est testée sur des exemples concrets pour démontrer sa pertinence et son efficacité.

Table des matières

Résumé 1

Table des matières 2

Liste des Figures 3

Introduction générale 4

CHAPITRE I : Analyse et modélisation du problème 5

1. Analyse du problème 6

2. Représentation par un graphe 6

3. Exemple avec n=7 m=16 6

4. Conclusion 7

CHAPITRE II : Étude des algorithmes de coloration de graphes 8

1. INTRODUCTION 9

Coloration gloutonne 9

Welsh-Powell 9

2. Sélection de l'algorithme : Welsh-Powell 9

3. Description de l'algorithme Welsh-Powell 9

a. Principe 9

b. Procédure 10

c. Exemple de déroulement de L'Algorithme de Welsh-Powell 10

d. Limites 12

4. Conclusion 12

CHAPITRE III : Implémentation de l'algorithme Welsh-Powell 13

1. INTRODUCTION 14

a.Implémentation en Python 14

b.Déroulement de L'Algorithme de Welsh-Powell 15

2. Cas plus complexes avec n = 12 m = 30 16

Génération du graphe aléatoire : **(Explication du code)** 16

Vérifier la validité des résultats 17

3. Conclusion 17

CONCLUSION et PERSPECTIVES 18

BIBLIOGRAPHIE 19

Liste des Figures

Figure 1: Graphe des matières avec n=7 et m=16 7
Figure 2: Graphe Planaire 10
Figure 3: Graphe Planaire après l'application de l'algorithme 11
Figure 4: Graphe avant l'application de l'algorithme 14
Figure 5: Graphe après l'application de l'algorithme 15
Figure 6: Graphe colorier avec n=12 et m=30...16

Introduction générale

La planification des examens est une tâche essentielle et complexe dans le cadre de la gestion académique, en particulier dans les institutions où plusieurs matières partagent des étudiants communs. L'objectif principal est de garantir que les examens soient organisés de manière optimale, minimisant les conflits d'horaires tout en respectant les contraintes académiques et logistiques. Ce type de problème peut être modélisé mathématiquement à l'aide de graphes, où chaque sommet représente une matière, et chaque arête indique un conflit potentiel entre deux matières. Trouver le nombre minimum de créneaux horaires nécessaires revient alors à résoudre un problème de coloration de graphes. Ce rapport présente une approche algorithmique, notamment basée sur l'algorithme de Welsh-Powell, pour développer une solution efficace à ce problème, avec une application pratique à des cas concrets pour valider les résultats obtenus.

CHAPITRE I : Analyse et modélisation du problème

1. ANALYSE DU PROBLEME

Le problème consiste à planifier les examens d'une institution académique tout en évitant les conflits d'horaires pour les étudiants inscrits à plusieurs matières. Les conflits surviennent lorsque deux matières ayant des étudiants communs sont programmées en même temps.

Pour minimiser la durée totale des examens, il est nécessaire d'attribuer le nombre minimum de créneaux horaires possibles. Cela revient à trouver le **nombre chromatique** d'un graphe où chaque sommet représente une matière, et chaque arête indique un conflit entre deux matières.

2. REPRESENTATION PAR UN GRAPHE

Un graphe G= (V, E) est utilisé pour modéliser ce problème :

- **Sommets (V)** : Chaque sommet représente une matière.
- **Arêtes (E)** : Une arête (u, v) existe entre deux sommets u et v si les deux matières correspondantes ont des étudiants en commun, ce qui crée un conflit potentiel.

L'objectif est de colorier le graphe de manière à ce que deux sommets n'aient jamais la même couleur. Chaque couleur représente un créneau horaire, et le nombre minimal de couleurs utilisées correspond au nombre chromatique y(G).

3. EXEMPLE AVEC N=7 M=16

Chaque matière est représentée comme un sommet d'un graphe G, et les conflits entre matières sont modélisés par des arêtes reliant ces sommets.

Les données fournies définissent les relations entre les matières :

- n=7 : Les matières sont numérotées de 1 à 7.
- m=16 : Les conflits (arêtes) sont définis comme suit :

{(1,2),(1,3),(1,4),(1,7),(2,3),(2,4),(2,5),(2,7),(3,4),(3,6),(3,7),(4,5),(4,6),(5,6),(5,7),(6,7)}

Implémentation en Python avant l'algorithme

Le graphe est construit à l'aide de **NetworkX** pour simplifier les manipulations

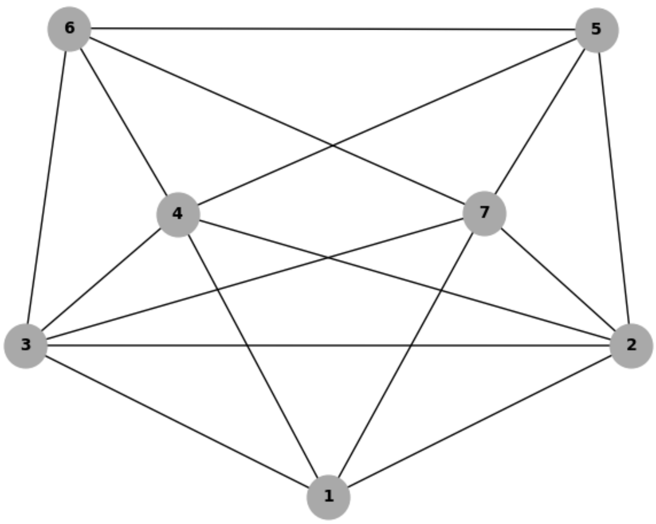

Figure 1: Graphe des matières avec n=7 et m=16

4. CONCLUSION

Ce graphe représente parfaitement le problème de planification des examens. La prochaine étape consiste à utiliser un algorithme de coloration, comme Welsh-Powell, pour attribuer un créneau horaire minimal à chaque matière sans conflit.

CHAPITRE II : Étude des algorithmes de coloration de graphes

1. INTRODUCTION

La coloration de graphes consiste à attribuer une couleur à chaque sommet d'un graphe de manière à ce que deux sommets adjacents n’aient pas la même couleur. Cela modélise des problèmes où des entités liées par des arêtes doivent être séparées en différentes couleurs. Plusieurs algorithmes permettent de résoudre ce problème :

Etude de l’existant :

Coloration gloutonne

- Attribue des couleurs séquentiellement aux sommets dans l'ordre où ils sont visités.
- À chaque sommet, on choisit la première couleur disponible (non utilisée par ses voisins).
- Simple mais sensible à l'ordre des sommets.

Welsh-Powell

- Ordonne les sommets par degré décroissant (les sommets les plus connectés sont traités en premier).
- Utilise une approche gloutonne après cet ordonnancement.
- Plus efficace que la coloration gloutonne dans les cas denses.

2. SELECTION DE L’ALGORITHME : WELSH-POWELL

Nous choisissons l'algorithme **Welsh-Powell** pour sa simplicité et son efficacité dans des graphes modérément denses comme celui du problème de planification des examens. Cet algorithme minimise souvent le nombre de couleurs utilisées tout en ayant une complexité acceptable.

3. DESCRIPTION DE L'ALGORITHME WELSH-POWELL

a. Principe

L’algorithme de Welsh & Powell consiste ainsi à colorer séquentiellement le graphe en visitant les sommets par ordre de degré décroissant. L’idée est que les sommets ayant beaucoup de voisins seront plus difficiles à colorer, et donc il faut les colorer en premier. L’heuristique DSATUR propose une amélioration du principe de l’algorithme de Welsh & Powell afin d’éviter ce problème.

Il y a une certaine procédure à respecter afin de mettre en œuvre cet algorithme.

b. Procédure

1. Repérer le degré de chaque sommet.
2. Ranger les sommets par ordre de degrés décroissants
3. Attribuer au premier sommet (A) de la liste une couleur.
4. Suivre la liste en attribuant la même couleur au premier sommet (B) qui ne soit pas adjacent à (A).
5. Suivre la liste jusqu'au prochain sommet (C) qui ne soit adjacent ni à A ni à B.
6. Continuer jusqu'à ce que la liste soit finie.
7. Prendre une deuxième couleur pour le premier sommet (D) non encore coloré de la liste.
8. Répéter les opérations 4 à 7.
9. Continuer jusqu'à avoir coloré tous les sommets.

c. Exemple de déroulement de L'Algorithme de Welsh-Powell

Prenons ce graphe G :

Le graphe est construit à l'aide du logiciel **draw.io**

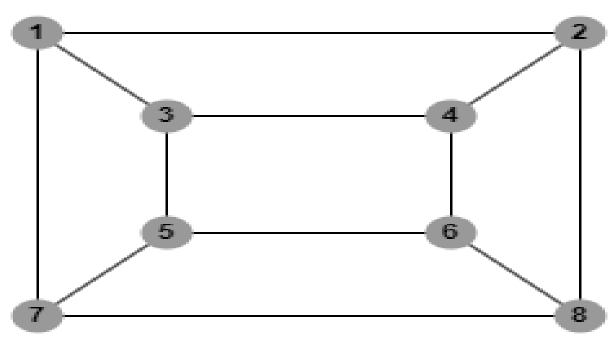

Figure 2: Graphe Planaire

Encadrement du nombre chromatique de ce graphe planaire :

$$\mathbf{n \leq \gamma(G) \leq d + 1}$$

$$\mathbf{2 \leq \gamma(G) \leq 4}$$

- n = 2 (l'ordre du plus grand sous-graphe complet de G)
- d (le plus grand degré du graphe G)
- Le nombre chromatique de tout graphe planaire est au plus 4.

Déroulement

Nous allons dérouler l'algorithme de Welsh Powell :

Sommet	Degré	Successeur
1	3	{2,3,7}
2	3	{1,4,8}
3	3	{1,4,5}
4	3	{2,3,6}
5	3	{3,7,6}
6	3	{4,5,8}
7	3	{1,5,8}
8	3	{6,7,2}

Coloration : Parcours des sommets dans l'ordre trié. Chaque sommet reçoit la première couleur disponible qui n'est pas utilisée par ses voisins.

Couleur/Sommet	1	2	3	4	5	6	7	8
C1 : rouge	**C1**			**C1**	**C1**			**C1**
C2 : vert	**X**	**C2**	**C2**	**X**	**X**	**C2**	**C2**	**X**

Fin de l'algorithme

Avec l'application de l'algorithme, on obtient alors cette coloration : **γ(G) = 2**

Le graphe est construit à l'aide du logiciel **draw.io**

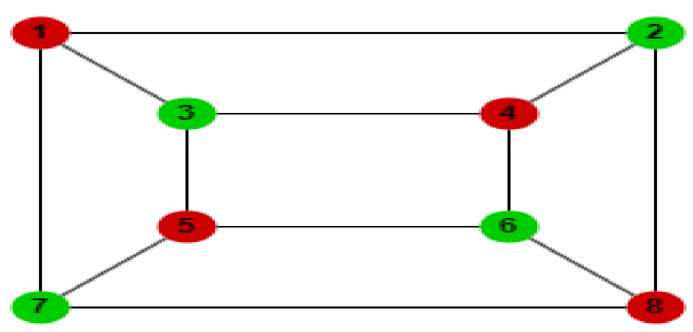

Figure 3: Graphe Planaire après l'application de l'algorithme

d. Limites

L’algorithme de Welsh Powell permet d’avoir une coloration des sommets cohérente.

Néanmoins, la coloration proposée n’est pas toujours optimale, car dans certains cas, un nombre plus restreint de couleurs peuvent être utilisées.

Il est donc parfois plus simple d’étudier le graphe sans ce dernier et de trouver soi-même une coloration optimale, et ensuite vérifier sa coloration avec l’algorithme pour voir si elle est optimale ou non.

4. CONCLUSION

Cette compréhension détaillée de Welsh-Powell permet de justifier son utilisation dans le cadre de la planification d'examens.

CHAPITRE III : Implémentation de l'algorithme Welsh-Powell

1. INTRODUCTION

Dans cette section, nous utilisons un graphe de **n=7** sommets et **m=16** arêtes pour modéliser les données fournies. Les conflits entre matières sont spécifiés par les paires suivantes :

{(1,2), (1,3), (1,4), (1,7), (2,3), (2,4), (2,5), (2,7), (3,4), (3,6), (3,7), (4,5), (4,6), (5,6), (5,7), (6,7)}.

L'objectif est d'appliquer un algorithme efficace, tel que celui de **Welsh-Powell**, pour trouver une solution optimale, c'est-à-dire minimiser le nombre total de créneaux nécessaires.

Cette étude mettra en évidence les étapes de l'algorithme, la modélisation des données en graphe, et les résultats obtenus, accompagnés d'une visualisation des créneaux attribués à chaque matière.

a.Implémentation en Python

Le graphe est construit à l'aide de **NetworkX** pour simplifier les manipulations.

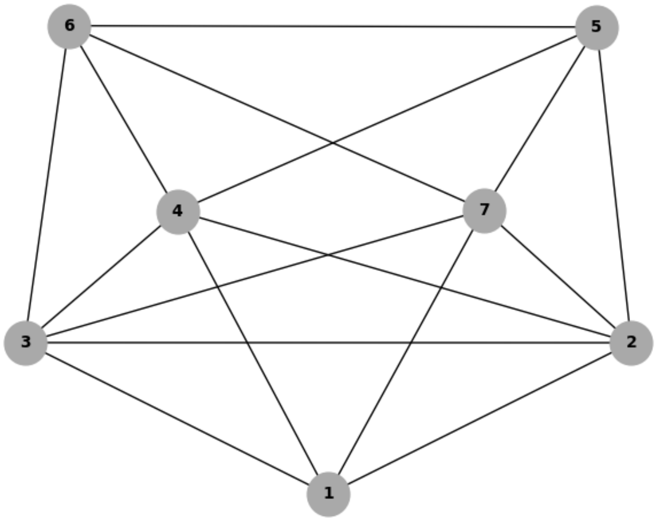

Figure 4: Graphe avant l'application de l'algorithme

Encadrement du nombre chromatique de ce graphe :

$$\mathbf{n \leq \gamma(G) \leq d + 1}$$

$$\mathbf{3 \leq \gamma(G) \leq 6}$$

- n = 3 (l'ordre du plus grand sous-graphe complet de G)
- d (le plus grand degré du graphe G)

b.Déroulement de L'Algorithme de Welsh-Powell

Les sommets sont triés par degré décroissant pour minimiser les conflits possibles dès le début.

Sommet	Degré	Successeur
2	5	{1,3,4,5,7}
3	5	{1,2,4,6,7}
4	5	{1,3,2,5,6}
7	5	{1,2,3,5,6}
1	4	{2,3,4,7}
5	4	{2,4,6,7}
6	4	{3,4,5,7}

Coloration : Parcours des sommets dans l'ordre trié. Chaque sommet reçoit la première couleur disponible qui n'est pas utilisée par ses voisins.

Couleur/Sommet	2	3	4	7	1	5	6
C1 : mauve	**C1**						**C1**
C2 : bleu	**X**	**C2**				**C2**	**X**
C3 : jaune	**X**	**X**	**C3**	**C3**		**X**	**X**
C4 : rouge	**X**	**X**	**X**	**X**	**C4**	**X**	**X**

Fin de l'algorithme

Avec l'application de l'algorithme, on obtient alors cette coloration : **γ(G) = 4**

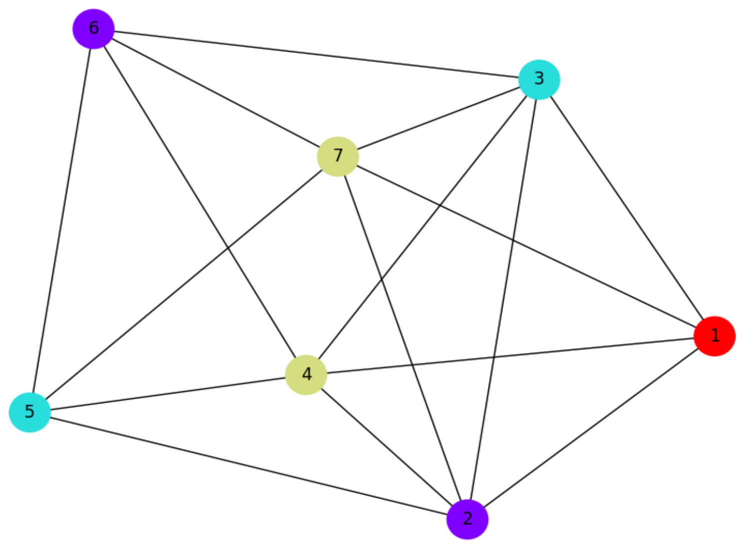

Figure 5: Graphe après l'application de l'algorithme

2. CAS PLUS COMPLEXES AVEC N = 12 M = 30

Pour étendre l’application à des cas plus complexes avec **n=12 m=30**, j’applique l'algorithme Welsh-Powell, et j’affiche les résultats ainsi que le graphe coloré.

Implémentation en Python

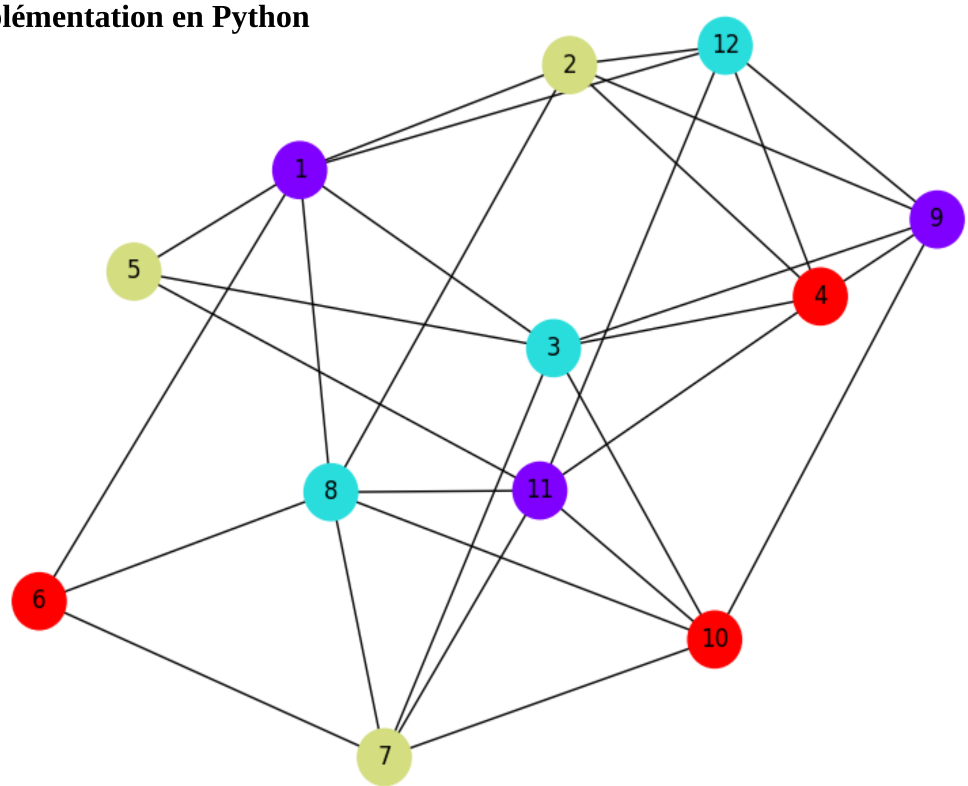

Figure 6: Graphe colorier avec n=12 et m=30

Génération du graphe aléatoire : **(Explication du code)**

- La fonction random_graph crée un graphe avec n sommets et m arêtes aléatoires.
- Elle vérifie que les sommets connectés ne sont pas identiques et évite les doublons d'arêtes.

Coloration avec Welsh-Powell :

- L'algorithme trie les sommets par ordre décroissant de degré.
- Les sommets sont colorés avec la première couleur disponible.

Visualisation du graphe :

- Chaque créneau horaire correspond à une couleur unique.
- Le graphe est affiché avec les nœuds colorés.

Vérifier la validité des résultats

Les sommets sont triés par degré décroissant pour minimiser les conflits possibles dès le début.

Sommet	Degré	Successeur
1	6	{2,3,5,6,8,12}
3	6	{1,4,5,7,9,10}
8	6	{1,2,6,7,10,11}
11	6	{4,5,7,8,10,12}
2	5	{1,4,8,9,12}
4	5	{2,3,9,11,12}
7	5	{3,6,8,10,11}
9	5	{2,3,4,10,12}
10	5	{3,7,8,9,11}
12	5	{1,2,4,9,11}
5	3	{1,3,11}
6	3	{1,7,8}

Coloration : Parcours des sommets dans l'ordre trié. Chaque sommet reçoit la première couleur disponible qui n'est pas utilisée par ses voisins.

Couleur/Sommet	1	3	8	11	2	4	7	9	10	12	5	6
C1 : mauve	**C1**			**C1**				**C1**				
C2 : bleu	**X**	**C2**	**C2**	**X**				**X**		**C2**		
C3 : jaune	**X**	**X**	**X**	**X**	**C3**		**C3**	**X**		**X**	**C3**	
C4 : rouge	**X**	**X**	**X**	**X**	**X**	**C4**	**X**	**X**	**C4**	**X**	**X**	**C4**

Fin de l'algorithme

Avec l'application de l'algorithme, on obtient alors cette coloration
Donc les résultats du graphe obtenus avec le code python sont valide : γ(G) = 4

3. CONCLUSION

La résolution du problème de planification des examens à l'aide de l'algorithme de coloration de graphe, notamment Welsh-Powell, a permis d'optimiser les créneaux horaires en évitant les conflits.

CONCLUSION et PERSPECTIVES

Cette étude a mis en évidence l'importance des algorithmes de coloration de graphes, en particulier dans la résolution de problèmes pratiques comme la planification des examens. Les différences d'efficacité constatées dépendent autant de l'implémentation que des heuristiques utilisées. La coloration de graphe reste un domaine de recherche très actif, avec de nombreuses applications dans divers secteurs.

Il est possible d'envisager des développements futurs, notamment en exploitant la structure des graphes pour concevoir des algorithmes polynomiaux dans des cas particuliers. De plus, le problème de coloration simple peut être étendu à des approches plus complexes, telles que la coloration avec contraintes (nombre limité de couleurs, couleurs imposées à certains sommets) ou la multi-coloration (attribution de plusieurs couleurs à un sommet). Ces perspectives offrent des opportunités pour des recherches et des implémentations encore plus performantes et adaptées à des problèmes spécifiques.

BIBLIOGRAPHIE

- **WordPress.com:**
  https://m6colorationgraphes.wordpress.com/2015/11/30/partie-iii-la-coloration-par-welsh-powel/

- **Loria.fr:**
  https://members.loria.fr/JDumas/files/tipe/rapport.pdf

- **Datacamp.com:**
  Python NetworkX for Graph Optimization Tutorial

- **W3schools.com:**
  https://www.w3schools.com/

Le code source de l’algorithme est disponible sur un fichier python : *graphe.py*

Vous pouvez le consulter pour voir l'ensemble du travail réalisé durant ce projet