République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université MOULOUD MAMMERI de TIZI-OUZOU
Faculté des Sciences
Département des Mathématiques

MÉMOIRE DE MASTER

En vue de l'obtention d'un diplôme de Master
Option

Méthodes et Modèles de décisions

Thème

Reconnaissance d'une classe de graphes Split graphe

Présenté par : CHOUALI Hamid **BOUTEMEUR Fatiha**

Soutenu le 09 octobre 2016

Promoteur	**M. B. SADI**	**Maître de conf. A**	**U.M.T.O.**
Président	**M. M. AOUANE**	**Maître assistant A**	**U.M.T.O.**
Examinateur	**M. A. AMIROU**	**Maître assistant A**	**U.M.T.O.**

TIZI-OUZOU, Octobre 2016.

⁂ Remerciements ⁂

Avant tout, nous remercions le bon Dieu de nous avoir donné le courage et la

foi pour mener à bien ce travail, malgré tous les obstacles.

Nos sincères remerciements vont tout particulièrement au Dr B. Sadi.

Nous vous en sommes extrêmement reconnaissants d'avoir accepté de diriger ce modeste travail et nous ferons de notre mieux pour mériter une telle confiance,Nous tenons également à témoigner notre profonde gratitude

à Mr Dj. Talem, pour son aide, son soutien mais par dessus tout, ses précieux encouragements.

Nos sincères remerciements s'adressent également aux membres de jury

pour avoir accepté d'examiner notre travail.

Nous tenons également à remercier tous ceux qui ont contribué de prés ou

de loin à la réalisation de ce modeste travail.

TANMIRTHNWEN.
Merci beaucoup.

⁕ *Dédicaces* ***FATIHA*** ⁕

Je dédie ce modeste travail

Aux deux personnes les plus nobles et les plus chères au monde :

mon pere et ma mère, qui m'ont soutenue jusqu'à

la fin.

À mon FRÈRE NASSIM et à ma petite SOEUR MERIEM, qui n'ont jamais cessé de m'encourager.

À toutes ma famille qui a été d'un grand soutien.

À nos très chers amis et camarades, spécialement CHOUALI HAMID H. OUAFA, R. LOUNIS, S. HAKIM,D. HAKIMA, A. KAHINA et amies SARA, HEDJILA, SOMIA, KAMILIA.

À tous ceux qui nous ont soutenu de prés ou de loin.

⁂ *Dédicaces* ***HAMID*** ⁂

Je dédie ce modeste travail

Aux deux personnes les plus nobles et les plus chères au monde :

Mon chère PÈRE et Ma très chère MÈRE qui ont sacrifié les plus belles années de

leurs vies pour me voir un jour réussir, et qui m'ont soutenus jusqu'à

la fin.

À Mes deux très chère SOEURS, AMEL et SABRINA, qui n'ont jamais cessé de m'encourager et aider.

Ainsi qu'a leurs maris et surtout a mes deux adorables neveux MASTEN et JUBA
et à Mon FRÈRE FARID et sa femme FAZIA, et leurs très chère et adorable fille THIZIRI.

A ma très chère GRAND mère.

À toute la promotion surtout BOUTEMEUR Fatiha S. HAKIM, R. LOUNIS, H. OUAFA et D. HAKIMA, S. NADIA, B. MASSIVA.

À tous les membres de l'association TAFRARA, les amis de la greve des bus d'AZAZGA 2015/2016, ainsi qu'à tous mes très chers amis et camarades.

À tous ceux qui m'ont soutenu de prés ou de loin.

Table des matières

Table des figures

Notations et symboles

G,	Un graphe simple.
V,	Ensemble de sommets de G.
E,	Ensemble d'arrêtes de G.
$n(G)$,	Ordre du graphe G ou le cardinal de V.
$m(G)$,	Taille du graphe ou nombre d'arêtes du graphe G.
$d(G)$,	degré du graphe G.
uv,et vu,	La même arête.
S,	Un ensemble stable.
K,	Un ensemble Clique.
$d\ (u, v)$,	La distance entre deux sommets.
$\bar{G} = (V, \bar{E})$,	Le graphe complémentaire de $G = (V, E)$.
Δ,	Séquence de degré.
δ,	Degré minimum dans un graphe.
xy,	Arête d'un graphe.
(x, y),	Arc d'un graphe.
$\chi(H)$,	Cardinal minimum d'une coloration de H.
$\omega(H)$,	Taille maximale d'une clique de H.
$\alpha(G)$,	Cardinal du stable maximum, ou nombre de stabilité.
$\omega(G)$,	Cardinal de la clique maximum.

Introduction générale

Sans que l'on en soit toujours conscients, la théorie des graphes est aujourd'hui très présente dans notre société moderne. Cette branche des mathématiques, dont on fait remonter l'origine à **EULER**, a connu un essor spectaculaire au cours des cinquante dernières années, notamment grâce aux travaux de **CLAUDE BERGE** qui a grandement participé à sa diffusion. Parce qu'elle permet de modéliser aussi bien des problématiques de réseaux informatiques que de réseaux routiers, de transport de marchandises que d'emplois du temps, d'électronique que de mécanique du solide, la théorie des graphes a bénéficié et bénéficie encore d'un engouement considérable non seulement de la part des mathématiciens, mais également de la communauté scientifique toute entière : on observe ainsi depuis quelques années un grand nombre de publications ayant trait à des problèmes en biochimie, en génétique ou encore en sociologie, en lien direct avec la théorie des graphes. Intuitivement, un graphe est un ensemble de points, dont certaines paires sont reliées.

Plus formellement, un graphe est défini par deux ensembles : son ensemble de sommets et son ensemble d'arêtes, une arête étant une paire de sommets reliés. La théorie des graphes introduit ensuite de nombreuses classes de graphes, des familles de graphes qui vérifient certaines propriétés. Les graphes permettent de modéliser de nombreux problèmes dans le domaine des réseaux, par exemple un réseau de transport (un plan de métro est un graphe, où les sommets représentent les stations), un réseau social (chaque sommet représente une personne et une arête relie deux sommets si les personnes correspondantes se connaissent) ou plus simplement un réseau informatique ; mais également des problèmes de chimie ou de génétique, ou encore affecter un nombre minimal de salles permettant de faire cours sans que deux cours n'aient lieu dans la même salle au même moment est un problème de coloration du graphe d'intervalles représentant les cours. Ces nombreuses applications font de la théorie des graphes un sujet de recherche toujours prolifique. Certains de ces problèmes sont **faciles**, c'est-à-dire qu'il existe un algorithme en temps d'exécution polynomial pour les résoudre, d'autres sont **difficiles** c'est-à-dire qu'on ne connaît pas d'algorithmes polynômiaux pour les résoudre. Cependant même si un problème est difficile sur les graphes

généraux, il est possible que sur certaines classes de graphes il soit facile. Le problème du stable maximum est en général difficile, mais dans le cas des graphes bipartis (les graphes sans cycle de longueur impaire ou de manière équivalente, les graphes pouvant se partitioner en deux ensembles de sommets V1 et V2, tels que les seules arêtes soient entre V1 et V2, c'est-à-dire qu'il n'y ait aucune arête dans V1 ni dans V2) il devient facile. Notons bien que l'adjectif facile veut seulement dire ici qu'il existe un algorithme polynomial (donc dans un certain sens efficace) pour le résoudre et pas que cet algorithme est **facile** à trouver. Par exemple, pour résoudre le problème du stable maximum dans les graphes bipartis, il faut utiliser le théorème de Konig assurant l'égalité entre deux paramètres du graphe et un algorithme de couplage qui est maintenant classique, mais revenons d'abord aux origines de la théorie des graphes.

Le problème des ponts de **Koenigsberg** a été introduit en 1735 par **LEONARD EULER**, considéré comme le fondateur de la théorie des graphes. La ville de Koenigsberg possède sept ponts enjambant la rivière Pregel (Fig. 1) et Euler s'interroge sur l'existence d'une promenade lui permettant de passer par tous les ponts de la ville une et une seule fois, et de revenir à son point de départ. **Euler** modélise ce problème par un graphe : un sommet est associé à chaque parcelle de terre délimitée par la rivière et une arête est associée à chaque pont les reliant (voir Fig. 2). Ainsi, une promenade passant par chacun des ponts une et une seule fois est alors un cycle eulérien dans ce graphe. **Euler** affirme que décider de l'existence d'un cycle eulérien dans un graphe est un problème **facile** : il faut et il suffit que tous les sommets soient de degré pair.

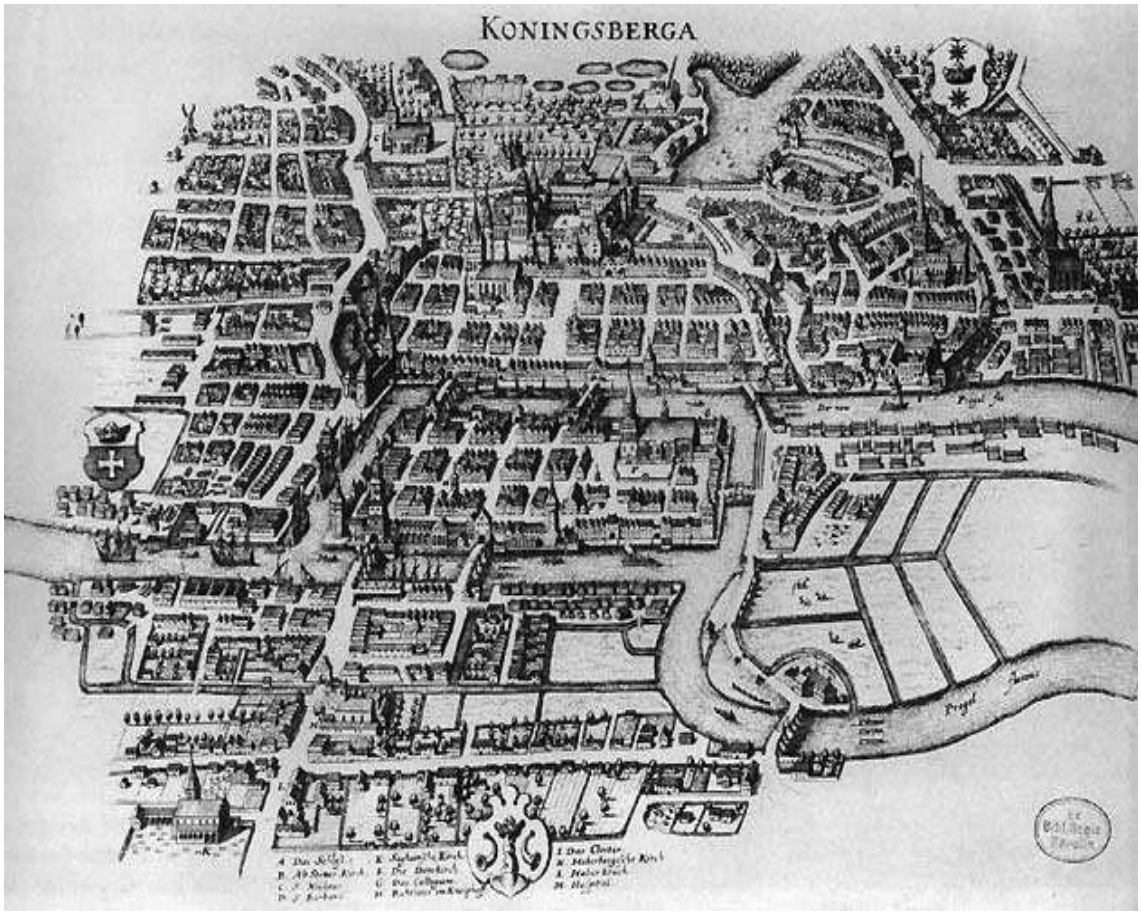


FIGURE 1 – Le plan de la ville de Koenigsberg

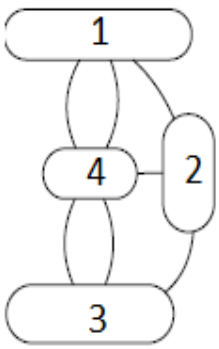


FIGURE 2 – Le Plan et la Modélisation de la ville de Koenigsberg[24]

Dans ce mémoire, nous définissons une nouvelle classe de graphes : les split graphes sachant qu'ils font partie de la famille des graphes de chaîne(ils sont inclus dans les graphes dit de **chaîne (difference)**).Les split graphes vérifient que pour tout graphe qui pouvant être scindé en deux ensembles : un ensemble stable et un ensemble clique, il y ait beaucoup d'intéressantes propriétés telles que la reconnaissance en temps linéaire induite par la caractérisation des sous-graphes, ainsi que des algorithmes polynomiaux pour des problèmes NP-difficiles qui jouissent de la plus haute importance dans la théorie des graphes. Le remplacement du mot **clique** dans la définition des graphiques split avec **bi-clique** apporte des changements tout en conservant beaucoup, en conservant certaines propriétés. Les split graphes vérifient que pour tout graphe qui peuvent être scindé en deux ensemble un ensemble stable et un ensemble clique, ils ont beaucoup de propriétés intéressantes simples telles que la reconnaissance en temps linéaire induite par la caractérisation des sous-graphe, ainsi que des algorithmes polynômiaux pour des problèmes NP-difficiles et sont donc de première importance dans la théorie des graphes et algorithmes de graphes. Remplacement de **clique** dans la définition des graphes split avec la notion de **bi-clique**, cependant, change beaucoup, en conservant une certaine attrayante propriétés. Pour cette classe de graphes, nous étudions et démontrons plusieurs propriétés.

Dans ce mémoire, nous étudions, définissons et exposons les propriétés mathématiques des split graphes en effectuant un état de l'art sur cette classe de graphe tout en donnant certaines de leurs extensions et généralisations ainsi que leurs applications en pratique. Le travail est structuré comme suit :

- Le Chapitre un rappelle rapidement sur la théorie des graphes, un état de l'art.
- Le chapitre deux est dédié a la complexité algorithmique.
- Les split graphes avec leurs propriétés sont présentées dans le chapitre trois.
- Dans le chapitre quatre on expose les bisplit graphes et quelqu'une de leurs propriétés.
- Des applications des split en pratique font l'objet du cinquième chapitre.
- Le document se termine par une conclusion et des perspectives.

Chapitre I

Etat de l'art

I.1 Introduction

Dans ce premier chapitre, nous définirons les notions qui serviront dans la suite de ce document. Nous commencerons par une introduction des notions et notations de la théorie des graphes et en particulier certaines classes de graphes classiques. Puis, nous introduirons quelques problèmes de reconnaissance.
Le chapitre qui suit sera dédié à une introduction et quelques rappels sur la théorie de la complexité algorithmique.

I.2 Définition d'un Graphe

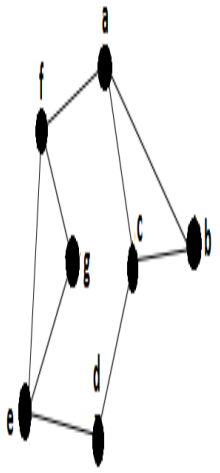
Un **graphe** est un schéma constitué de sommets, dont certains sont reliés par des arêtes. Un **graphe simple** est un graphe sans boucle dont chaque couple de sommets est relié par au plus une arête. Un **graphe orienté** est un graphe dont les arêtes sont orientées (fléchées). On distingue alors le sommet origine de l'arête et son extrémité terminal.

I.3 Quelques notions de la théorie des graphes

Commençons par des notations générales de la théorie des ensembles. Soit A un ensemble fini, nous notons $|A|$ le cardinal de A. L'inclusion d'un ensemble A dans un ensemble B est représentée par $A \subset B$ ou $B \supset A$ et par $\subseteq$ ou $\supseteq$ pour préciser lorsqu'il s'agit d'une inclusion stricte. La différence symétrique de deux ensembles A et B s'écrit $A\Delta B = (A \setminus B)) \bigcup (B \setminus A)$. Nous rappelons maintenant des définitions de la théorie des graphes. La plupart d'entre elles sont classiques et se trouvent dans [2].

I.3.1 Notions de base et notations

Nous définissons un **graphe simple** non orienté G par son ensemble de sommets $V(G)$ et son ensemble d'arêtes $E(G)$, avec G n'a ni arêtes multiples,ni boucles. Nous notons $V = V(G)$ et $E = E(G)$. Avec ces définitions, un graphe G est alors noté $G = (V, E)$. $n(G) = n$ désigne le cardinal de V et $m(G) = m$ représente le nombre d'arêtes du graphe G. Une **arête** est une paire non ordonnée de sommets distincts de V. Une arête de G est donc un ensemble u, v où u, $v \in V$ et $u \neq v$. Cependant, nous adoptons la notation uv pour représenter une arête, vu représentant la même arête. Soient u et v deux sommets de G, si uv appartient à E, alors u et v sont **adjacents**, nous dirons également que u et v sont **voisins**. L'arête uv est **incidente** à u et v. Le nombre d'arêtes **incidentes** à un sommet u est appelé le **degré** de u et est noté $\deg(u)$. Soit un entier k, un graphe G est **k-régulier** si tout sommet est de degré k. Un graphe 3-régulier est dit **cubique**. Une **chaîne** de longueur k entre les sommets u et v dans le graphe G est une séquence de sommets successifs $v_0v_1...v_k$ telle que $v_0 = u, v_k = v$ et $v_{i-1}v_i \in E$. La longueur de la chaîne est le nombre d'arêtes qu'elle contient. Une chaîne est en particulier un cycle lorsque $v_0 = v_k$ On appelle P_k (respectivement C_k) une **chaîne (respectivement un cycle)** de longueur $k-1$ (respectivement k), ayant k sommets distincts (on dit que la chaîne/le cycle est élémentaire). Nous appellerons en particulier un C_3 **un triangle et un** C_4 **un carré**. Un **cycle hamiltonien** dans G est un cycle passant une unique fois par chaque sommet. Un graphe possédant un cycle hamiltonien est un graphe hamiltonien. Les **composantes connexes** d'un graphe sont définies comme les classes d'équivalence de sommets pour la relation d'accessibilité, c'est-à-dire s'il existe une chaîne entre les sommets. Un graphe est alors **connexe** s'il comporte une unique composante connexe. Un graphe est k**-sommet-connexe** ou plus simplement k-connexe s'il reste **connexe** pour toute suppression de $(k-1)$ sommets. De manière équivalente, G est k-connexe si entre toute paire de sommets de G il existe k chaînes disjointes par les sommets. **La distance** entre deux sommets u, v, notée d (u, v) :est le nombre minimal d'arêtes d'une chaîne $u...v$. **L'excentricité (ou écartement)** d'un sommet u, notée $e(u) = max_{v \in V} d(u, v)$, est la distance maximale d'un sommet de G à u. **Le diamètre** diam (G) du graphe G est alors l'excentricité maximale d'un sommet de G. Pour $u \in V$ et $i \in N, i \leq e(u)$, nous désignons par $N_i^u = \{v \in V, d(u, v) = i\}$ l'ensemble des sommets de G à distance i de u Un graphe est **acyclique** s'il ne contient pas de cycle. L'**acyclicité** est une propriété monotone [2],[15],[24],[14].



V	V(v)
a	$\{b, c, f\}$
b	$\{a, c\}$
c	$\{a, b, d\}$
d	$\{c, e\}$
e	$\{d, f, g\}$
f	$\{a, e, g\}$
g	$\{e, f\}$

FIGURE I.1 – Un graphe simple et son tableau de voisins

•**Un graphe complémentaire** de $G = (V, E)$ est le graphe $\bar{G} = (V, \bar{E})$ avec $\bar{E} = V \times V \backslash E$, c'est-à-dire $\bar{G}$ a les mêmes sommets que G et deux sommets distincts de $\bar{G}$ sont adjacents si et seulement s'ils ne sont pas adjacents dans G.

• **Ordre d'un graphe :** Le nombre de sommets d'un graphe correspond à l'ordre du graphe.

• **Sous-graphe** : Un sous-graphe de G consiste à considérer seulement une partie des sommets de V et les liens induits par E. Par exemple si G représente les liaisons aériennes journalières entre les principales villes du monde, un sous-graphe possible est de se restreindre aux liaisons journalières entre les principales villes européennes.

• **Graphe partiel** : Un graphe partiel de G consiste à ne considérer qu'une partie des arêtes de E. En reprenant le même exemple, un graphe partiel possible est de ne considérer que les liaisons journalières de moins de 3 heures entre les principales villes du monde.

• **Graphe complet** : Un graphe est dit complet s'il existe une arête entre toute paire de sommets distincts.

•**Une Clique** : Une clique est un graphe tel que pour toute paire de sommets x, y l'arête xy existe. On parle aussi de graphe complet.

Remarque I.3.1. Une clique est toujours un graphe complet, alors que le graphe complet n'est pas toujours une clique.

- *Une clique est un sous-graphe complet.*

•**Un Stable** : Un sous-ensemble S des sommets d'une graphe $G = (V, E)$ est un stable s'il n'existe pas d'arete xy de E pour tout x et y appartiennent à S.
• *Un stable est un sous-graphe sans arête.*
•**Problème de la clique de taille maximale** : Pour un graphe $G = (V, E)$ donné, trouver un sous-ensemble C inclus dans V de cardinalité maximale tel que le graphe induit par C est une clique.
•**Problème du stable de taille maximale** : Pour un graphe $G = (V, E)$ donné, trouver un stable S de cardinalité maximale.

I.4 Représentation du graphe en machine

Il existe de nombreuses façons de représenter des graphes en machine. On en présente ici deux : les représentations par matrice d'adjacence et par listes de successeurs.
Pour la représentation par matrice d'adjacence d'un graphe G à n sommets, on suppose que les sommets du graphe sont numérotés arbitrairement de 1 à n : $V(G) = \{x_1, ..., x_n\}$.La matrice d'adjacence de G est alors la matrice M de taille $n \times n$, à valeurs dans $\{1..n\}$ définie comme suit :

Définition I.4.1. Matrice d'adjacence d'un graphe d'ordre n est la matrice carrée A de taille $n \times n$ telle que $a_{ij} = 1$ s'il existe une arête entre les sommets i et j,et $a_{ij} = 0$ sinon.

Notons que la matrice d'un graphe non orienté est symétrique. De plus, une matrice d'adjacence dépend de la numérotation des sommets du graphe qu'elle représente. Ainsi la notion de matrice d'adjacence n'a-t-elle de sens que pour un graphe étiqueté, et l'on parle alors bien de la matrice d'adjacence du graphe.

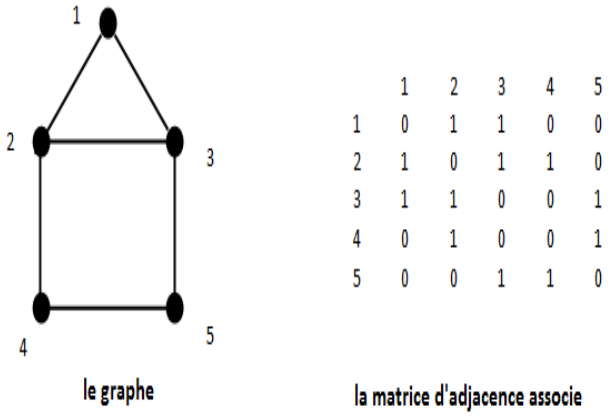


FIGURE I.2 – Un graphe (la maison) et sa matrice d'adjacence.

Définition I.4.2. Matrice d'incidence La matrice d'incidence (incidence matrix) d'un graphe d'ordre n est la matrice B de taille $n \times m$ (m désignant comme vu plus haut le nombre d'arêtes du graphe) telle que $b_{ij} = 1$ si le sommet i est une extrémité initiale de l'arête j, $b_{ij} = -1$ si le sommet i est une extrémité terminale de l'arête j, 0 sinon. Les représentations par matrice ont de nombreuses propriétés. Par exemple, une matrice d'adjacence permet de tester l'adjacence de deux sommets en une seule opération (il suffit de regarder la valeur de l'entrée correspondante). Elles sont cependant mal adaptées pour représenter des graphes peu denses (c'est-à-dire ayant peu d'arêtes en regard du nombre de sommets), car elles sont alors très creuses (c'est-à-dire contiennent beaucoup de zéros). Pour ces graphes, on leur préfère souvent une représentation par listes d'adjacence[9].

Définition I.4.3. Liste d'adjacence (adjacency list) est une structure de données dans laquelle on associe à chaque sommet sa liste de voisins.

Ainsi, on ne stocke que les informations utiles concernant l'adjacence dans le graphe, ce qui permet un gain d'espace mémoire non négligeable par rapport à une représentation par matrice. En revanche, déterminer si deux sommets sont adjacents requiert en général plus d'opérations avec une liste d'adjacence qu'avec une matrice d'adjacence, puisqu'il peut parcourir la liste entière des voisins d'un sommet. représentation de G par listes de successeurs consiste en un tableau T de n listes, une pour chaque sommet de G. Pour chaque $x \in V(G)$, la liste des successeurs de x est une liste chaînée des sommets y tels que $(x, y) \in E(G)$. Autrement dit, T(x) est constitué de tous les successeurs de x dans G.

Les sommets de chaque liste de successeurs sont généralement chaînés suivant un ordre arbitraire. Si G est un graphe orienté, la somme de toutes les longueurs des listes de successeurs vaut $| E(G) |$. Si G est non orienté, cette somme vaut $2 | E(G) |$. Dans les deux cas, l'encombrement mémoire de la représentation d'un graphe à n sommets et m arêtes est en O(n+m). La représentation par listes d'adjacence est souvent privilégiée parce qu'elle permet de représenter d'une façon plus compacte les graphes peu denses – ceux pour lesquels $| E(G) |$ est beaucoup plus petit que $| V(G) |^2$. La représentation par matrice d'adjacence est néanmoins intéressante si le graphe est dense ou si l'on doit souvent tester la présence d'un arc entre deux sommets. Dans ce dernier cas, la représentation par listes d'adjacence est particulièrement mal adaptée.

I.5 Propriétés de graphes

Nous identifions une propriété de graphes P et la classe des graphes satisfaisants cette propriété. Une propriété de graphes est non triviale si elle est vraie pour un ensemble infini de graphes et fausse pour un ensemble infini de graphes. Une propriété de graphes P est **héréditaire** si elle est close par suppression de sommets, c'est-à-dire pour tout graphe G satisfaisant P, tout sous-graphe induit de G satisfait P. Une propriété de graphes est **monotone** si elle est close par suppression de sommets ou d'arêtes, c'est-à-dire, pour tout graphe de P, tout sous-graphe (non nécessairement induit) satisfait cette propriété. Ainsi, *une propriété monotone est héréditaire*. De *nombreuses classes de graphes très étudiées* sont monotones ou héréditaires.

I.6 Quelques classes de graphes classiques

Il est intéressant d'étudier des classes de graphes. D'un point de vue théorique afin de comprendre ce qui fait la difficulté d'un problème, mais également d'un point de vue pratique. En effet en modélisant un problème il est tout à fait possible que le graphe obtenu ait des propriétés spécifiques et qu'il soit alors possible d'avoir des algorithmes efficaces.

I.6.1 Graphe parfait

Un graphe G est parfait si, et seulement si, pour tout sous-graphe induit H de G, $\omega(H) = \chi(H)$. Un graphe est de Berge si ni lui ni son complémentaire ne contiennent de cycle impair induit de longueur supérieure ou égale à cinq.

I.6.2 Graphes de comparabilité

Un graphe de comparabilité possède un ordre partiel sur les sommets tel qu'il existe une arête entre toute paire d'éléments comparables. Ils sont également appelés transitivement orientables. Cette classe est une sous-classe des graphes parfaits. Tester la transitivité d'une orientation est polynomial (équivalent à un produit matriciel) et la construction d'une orientation transitive d'un graphe, quand elle existe, est linéaire. Ainsi la reconnaissance d'un graphe de comparabilité est polynomiale.[24].

Remarque I.6.1. Pour définir le graphe d'incompatibilité noté Inc(P)=(X ; Inc(P)), il suffit de passer au complémentaire qui est donné par : Inc(P)=$\overline{Comp(P)}$.

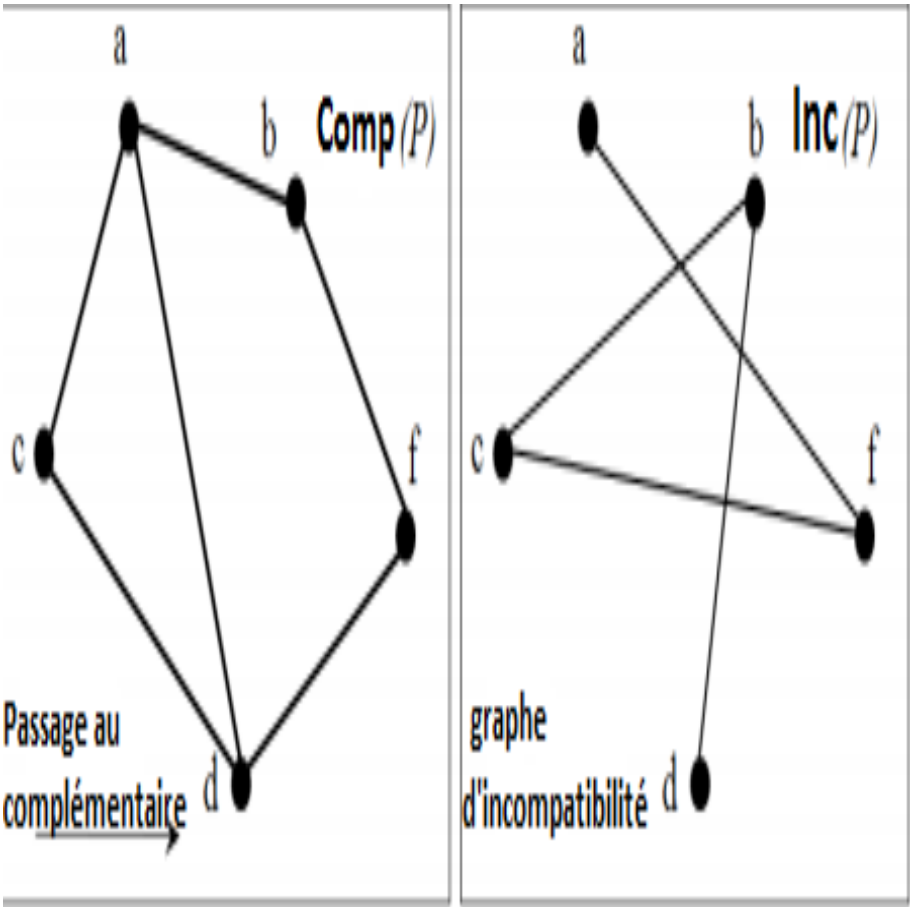


FIGURE I.3 – Diagramme de Comparabilité et D'incompatibilité.

I.6.3 Graphes bipartis

Un graphe est biparti lorsque l'ensemble des sommets V peut être partitionné en deux ensembles $V1$ et $V2$ tels que $uv \in E$ implique $u \in V1$ et $v \in V2$ ou $u \in V2$ et $v \in V1$. Tout graphe biparti est de comparabilité, en prenant par exemple, une orientation des arêtes de $V1$ vers $V2$. Cette propriété est monotone [24].

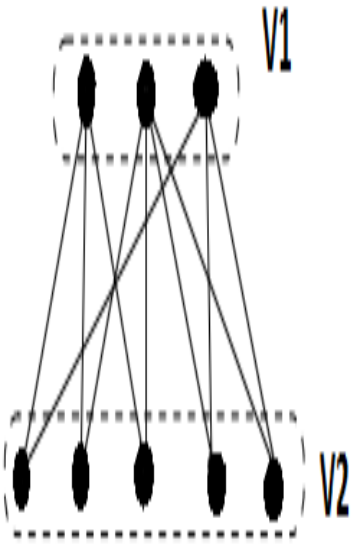


FIGURE I.4 – Un Graphe biparti.

I.6.4 Graphes de Chain ou (graphes de différences)

Un graphe G = (V, E) est un graphe de différence s'il existe des nombres réels a_v pour chaque $v \in V$ et un nombre réel t, tel que $\mid a_v \mid < t$ t pour $v \in V$ et $uv \in E$ si et seulement si $\mid a_u - a_v \mid \geq t$. Un graphe dont l'ensemble des sommets peut être divisé en deux ensembles indépendants est appelé un graphe biparti. Cette partition de l'ensemble des sommets en deux ensembles indépendants est appelé bipartition, et il est unique si et seulement si le graphe est connexe.[19] Yannakakis a donné un autre nom pour les graphes de différences. Il les a appelée graphes de chains, et il a définit un graphe biparti d'un graphe de chaîne si l'un des côtés de la bipartition a la propriété des voisins imbriqué. Il a également montré

que d'un côté on a cette propriété si et seulement si les deux parties ont la propriété [25].

I.6.5 Graphes Triangulés

Un graphe est triangulé s'il ne possède pas de cycle induit de longueur supérieure ou égale à 4. Les graphes triangulés sont parfaits et cette propriété est également héréditaire.

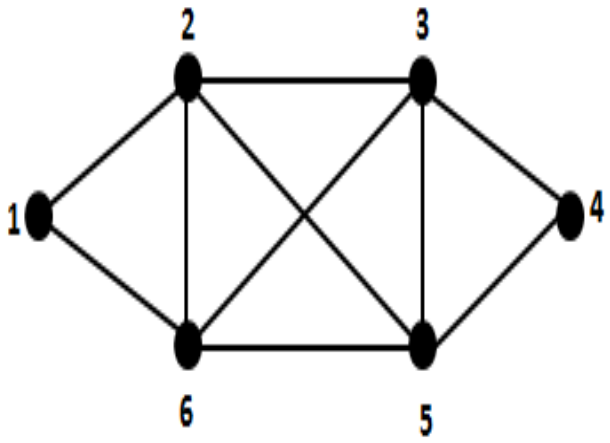


FIGURE I.5 – Un graphe triangulé

I.6.6 Graphes d'intervalles

Un graphe d'intervalles est le graphe d'intersection d'un ensemble d'intervalles de la droite réelle (ou de manière équivalente le graphe sous-arbre d'un graphe chaîne). Chaque sommet représente un intervalle et une arête relie deux sommets lorsque les intervalles correspondants sont d'intersection non vide [24].

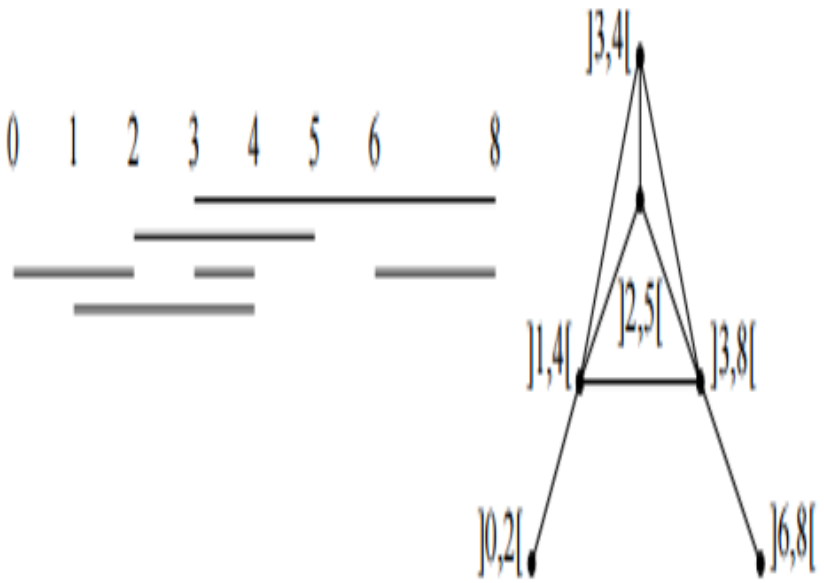


FIGURE I.6 – Un graphe d'ntervalle

I.6.7 Graphes planaires

Un graphe est planaire si on peut le dessiner sur une feuille de papier sans que les arêtes se croisent. Il est bien connu que la taille d'un graphe planaire G d'ordre $n \geq 3$, est liée par $m \leq 3(n-2)$[24].

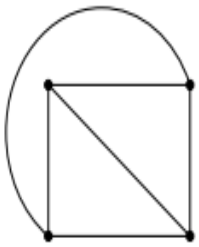


FIGURE I.7 – Un graphe planaire

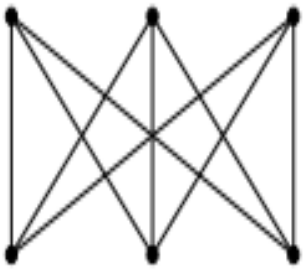


FIGURE I.8 – Un graphe non planaire

I.6.8 Arbres

Un arbre est un graphe connexe sans cycle. Par définition, un arbre$T$ de taille n et d'order m est un graphe connexe sans cycle avec $m = n - 1$.Pour tout couple de sommets, il existe une chaîne unique qui les relie. Il est connu que les arbres avec leurs propriétés occupent une place importante dans les graphes.

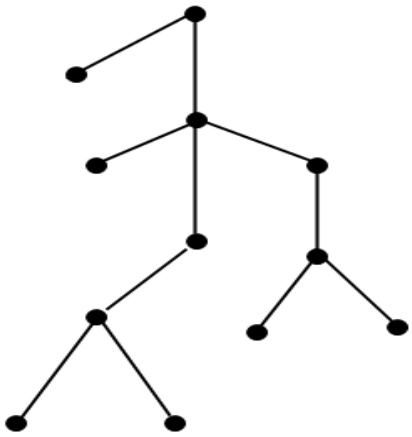


FIGURE I.9 – Un arbre

I.6.9 Cactus

Un cactus est un graphe connexe dans lequel deux cycles simples ont au plus un sommet commun. De manière équivalente, un cactus est un graphe tel que chaque arête appartient à au plus un cycle. Les cactus sont des graphes planaires. Un graphe est un cactus si et seulement si toute composante 2-connexe est soit un cycle, soit une arête.

I.6.10 Graphe à distance héréditaire

Un graphe G est à distance héréditaire si la distance entre deux sommets quelconques d'un sous-graphe induit connexe de G est égale à la distance entre ces sommets dans G. Ces graphes ont été introduits par Howorka dans [17]. Nous notons **DistH** la classe des graphes à distance héréditaire. Ils sont caractérisés par l'existence de deux cordes croisées pour tout cycle de longueur au moins cinq, toute chaîne induite est une plus courte chaîne. Comme son nom le laisse entendre, cette propriété de graphes est héréditaire. Ces graphes sont parfaits et la reconnaissance d'un graphe à distance héréditaire est linéaire [24].

Dans l'exemple suivant, nous allons illustrer les relations d'inclusion entres les différentes classes de graphes.

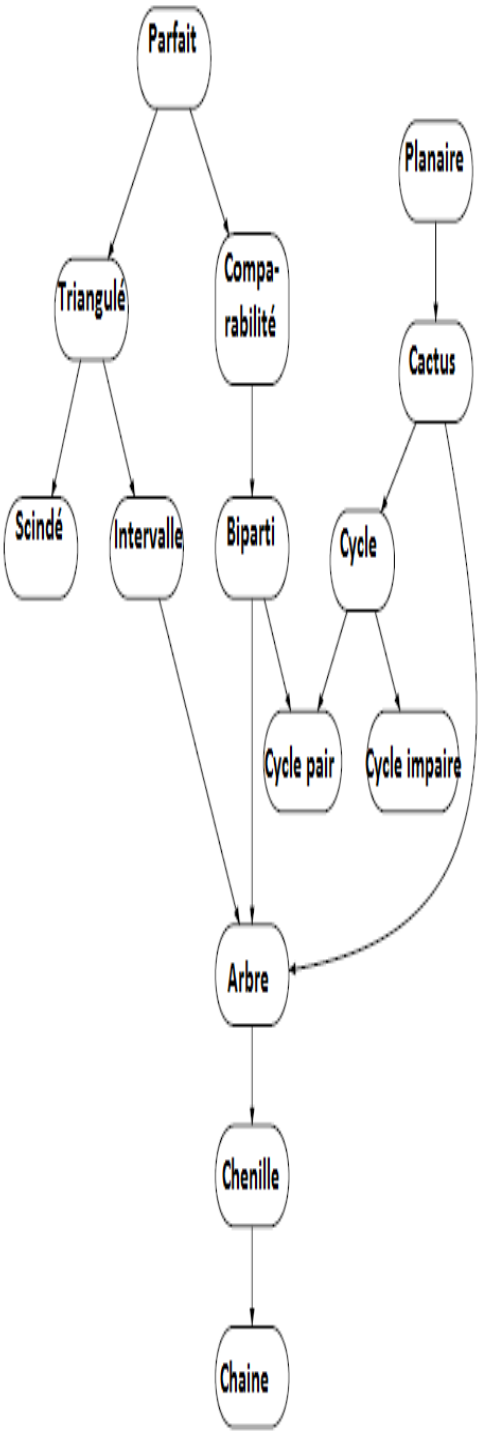


FIGURE I.10 – Relations d'inclusion de différentes classes de graphes classiques [24],[3]

Chapitre II

Complexité Algorithmique

II.1 Introduction

Nous introduisons ici dans ce deuxième chapitre, quelques notions de complexité. Pour toute notion non développée ici, nous renvoyons le lecteur intéressé vers l'excellent ouvrage de Michael R. Garey et David S. Johnson[11].
Quant au chapitre suivant, il fera l'objet d'une etude sur les split graphes, et leurs propriétés.

II.2 Définition de la Complexité

Nous nous intéressons, pour un problème donné, à déterminer la **difficulté** de résolution de ce problème. Pour cela, nous distinguons les problèmes de décision dont la réponse est **oui** ou **non** et les problèmes de recherche d'une solution pour lesquels il faut construire une solution. La théorie de **la complexité** est une branche des mathématiques et de l'informatique ayant pour cadre l'étude de la difficulté intrinsèque des problèmes algorithmiques, et qui vise à classer ces problèmes en fonction de cette difficulté. Ici, les mots **complexité** et **difficulté** ne se rapportent pas à la mise au point d'un algorithme de résolution, ou aux concepts avancés auxquels il peut faire appel (comme une structure de données élaborée), mais plutôt à la quantité de ressources à utiliser pour résoudre le problème. En ce qui nous concerne, les ressources consistent en le temps que met un algorithme à résoudre le problème c'est sa **complexité temporelle** et l'espace mémoire qu'il utilise au cours de son exécution sa **complexité spatiale** ; mais il existe d'autres mesures de complexité, comme le nombre de portes logiques à utiliser pour la réalisation d'un circuit, ou encore la quantité d'informations à transmettre dans le cadre de la théorie de la complexité de la communication. Nous distinguons deux classes de problèmes. La classe NP (pour Non determinist Polynomial) est définie comme la classe des problèmes de décision que l'on peut résoudre,

à l'aide d'un algorithme non déterministe, en temps polynomiale par rapport à la taille de la donnée. À l'intérieur de cette classe, la classe P des problèmes polynômiaux est l'ensemble des problèmes qui peuvent être décidés sur une machine déterministe polynômial. Une célèbre conjecture en informatique théorique est que $P \neq NP$; cela signifierait que certains problèmes de NP ne peuvent pas être résolus par des algorithmes efficaces.

Les problèmes les plus **difficiles** à résoudre parmi les problèmes de NP sont les problèmes NP-complets. Un problème Π est NP**-difficile** si l'existence d'un algorithme polynomial (déterministe) pour Π implique l'existence d'un algorithme polynomial (déterministe) pour n'importe quel problème de NP. Un problème est NP-complet s'il est NP-difficile et appartient à NP. En 1971, Stephen A. Cook est le premier à avoir prouvé l'existence de tels problèmes, en montrant que le problème de satisfaction des formules booléennes Sat est NP-complet [5]. Une façon de montrer la NP-complétude d'un problème est d'utiliser la notion de réduction polynômial : étant donné un problème $\Pi \in NP$et un problème NP-difficile Π^*, si Π^* se réduit principalement à Π, alors Π est NP-complet (ou NP-difficile s'il n'est pas dans NP). En règle générale, on montre qu'un problème d'optimisation est NP-difficile en montrant que le problème de décision associé est NP-complet.

II.3 Extensions et généralisations

II.3.1 Définitions de bases

• **Un probleme** est une question générale possédant des paramètres dont le résultats n'est pas connue.

• **Une instance** d'un problème est obtenue en affectant une valeur à chacun de ses paramètres.

•**La taille d'une instance** désigne généralement la quantité de cases mémoires nécessaires pour décrire les paramètres.

• **Un problème de décision** ou problème de reconnaissance est un problème auquel la réponse est oui ou non.

• **Un problème d'optimisation** consiste à déterminer la meilleure solution parmi toutes les solutions réalisables.

•**La complexité temporelle** d'un algorithme correspond au nombre d'instructions élémentaires (opérations arithmétiques,comparaison,affectation...)effectuées au cours de son exécution.

• **La complexité spatiale** d'un algorithme correspond au nombre de cases mémoires

occupées par les données manipulées par l'algorithme au cours de son exécution.

Définition II.3.1. Un Algorithme de résolution d'un problème (P) donnée est une procedure, décomposable en opérations élémentaires transformant une chaîne de caractères representant les données de n'importe quel exemple du problème (P) en une chaîne de caractère representant les résultats de (P).

Définition II.3.2. un algorithme est dit **polynômial** si le nombre d'opérations élémentaires nécessaire pour résoudre un exemple de taille n est une fonction polynômial en n. un algorithme est considéré comme efficace si et seulement s' il est polynomial.

Définition II.3.3. La complexité d'un algorithme est une évaluation du nombre d'opérations élémentaires qu'effectue l'algorithme en fonction de la taille et de la nature des données.

II.3.2 Ordre d'une fonction

Définition II.3.4. La notation **grand O**, dite aussi symbole de **Landau**, décrit le comportement asymptotique d'une fonction, exprimé à l'aide d'une autre fonction généralement plus simple. Plus formellement,
étant données deux fonctions f, g : $N \longrightarrow N$, on dit que f est O(g) s'il existe une constante c telle que :

- $f(n) \leq g(n)$ pour tout $n \in N$.

on dit que f est polynômial si elle est O(g) et g est un polynôme en n, ou encore s'il existe deux constantes c et k telles que :

- $f(n) \leq cn^k$, pour tout $n \in N$.

Remarque II.3.1. On notera que dans ces définitions visant à caractériser la croissance d'une fonction,la constante c ne joue aucun role.on s'interesse uniquement a la forme de la croissance de la fonction

Exemple II.3.1. : Soit $f(n) = 6n^4 - 2n^3 + 5$. Choisissons $n_0 = 1$. Alors pour tout $n \geq n_0$, on a $6n^4 - 2n^3 + 5 \leq 6n^4 + 2n^4 + 5n^4 = 13n^4$ Ainsi, en prenant $c = 13$, on a $f = O(n^4)$. Autrement dit, à un facteur constant prés, $f(n)$ ne croît pas plus rapidement que n^4. Il est facile de voir qu'un polynôme $P(n)$ de degré k est toujours en $O(n^k)$.

Définition II.3.5. (Notations Ω, Θ)

Soient f et g deux fonctions $f; g : \mathbb{N} \to \mathbb{R}_+$.

- On note $f = \Omega(g)$ lorsqu'il existe un entier n_0 et une constante réelle $c^{'}$ tel que pour tout $n \geq n_0$, $f(n) \geq c^{'}g(n)$.
- On note $f = \Theta(g)$ lorsque $f(n) = O(g)$ et $f(n) = \Omega(g(n))$, c'est-à-dire lorsque il existe un entier n_0 et deux constantes réelles c et $c^{'}$ tel que $cg(n) \leq f(n) \leq c^{'}g(n)$.

$\Omega(g)$ (respectivement $\Theta(g)$) est l'ensemble des fonctions d'ordre inférieur (respectivement équivalentes) à g pour n assez grand.

Exemple II.3.2. Soit $f(n) = n^3 sinn$, on a $\forall n$, $-n^3 \leq f(n) \leq n^3$. Ainsi $f(n) = \Theta(n^3)$.

II.4 Les Classes de Complexité

Nous allons maintenant nous intéresser à l'étude de la difficulté intrinsèque des problèmes de décision, ce que l'on appelle complexité des problèmes (non pas des algorithmes), et on va les classer selon la complexité des algorithmes les résolvant. Un grand nombre d'entre eux sont des problèmes faciles car on connaît des algorithmes polynômiaux pour les résoudre. Cependant, il existe aussi un grand nombre de problèmes pour lesquels on ne connaît pas d'algorithmes polynômiaux. On ne peut pas prouver qu'il n'en existe pas, mais on peut cependant montrer que l'existence d'un algorithme polynomial pour l'un d'entre eux impliquerait l'existence d'un algorithme polynomial pour presque tous les problèmes.

II.4.1 La Classe P

Définition II.4.1. Un problème de décision P est dans la classe P si, pour chacune de ses instances, dont la taille est notée n, il existe un réel positif k tel qu'il peut être résolu par un algorithme de complexité temporelle $O(n^k)$, c'est-à-dire qu'il peut être décidé en temps polynomial.

Les problèmes de la classe P sont dits faciles. Ce sont ceux que l'on sait résoudre efficacement.

Exemple II.4.1. Soit le problème du plus court chemin entre deux sommets dans un graphe orienté et valué par des coûts positifs.

- Entrée : un graphe value $G = (V, E, u)$, deux sommets s et t, et une constante k positive ;
- Sortie : existe-t-il un chemin allant de s à t de longueur au plus k.

Ce problème est dans la classe P car l'algorithme de Dijkstra, qui est un algorithme polynomial, peut résoudre ce problème.

Cependant, on ne sait pas si les problèmes de décision associés au problème du cycle hamiltonien au problème du stable maximum appartiennent à P ou pas.

II.4.2 La classe NP

Un problème de reconnaissance est dans la classe NP si pour toute instance de ce problème, on peut vérifier, en un temps polynomial par rapport à la taille de l'instance, qu'une solution proposée ou devinée permet d'affirmer que la réponse est "OUI" pour cette instance.

Définition II.4.2. Un problème de décision est dans la classe NP si l'on peut vérifier en temps polynomial qu'une solution pour une instance donnée est valide (ce que l'on appelle un **certificat du oui**).

La classe des problèmes NP est la classe des problèmes pour lesquels on peut vérifier si la solution est celle d'un problème est facile.

Proposition II.4.1. *$P \subseteq NP$ Effectivement, la classe des problème P est incluse dans la classe NP. cela veut dire que la classe NP est composée de problèmes polynômiaux P dits faciles et des problèmes NP-Complets dit difficiles.*

Exemple II.4.2.

pour la plupart des problèmes de la classe NP, on ne sait pas s'il peuvent ou ne peuvent pas être résolus par un algorithme polynomial. pour ne pas rester dans un flou total, on va montrer qu'il existe une large classe de problèmes qui sont équivalents du point de vue de l'existence d'un algorithme polynomial pour les résoudre dans le sens suivant : si un seul de ces problèmes peut être résolu par un algorithme polynomial alors tous le peuvent.

Remarque II.4.1. • Les problèmes de la classe NP sont ceux que l'on peut résoudre par énumération complète de toutes les solutions possibles (méthode "brutale") et en les testant à l'aide d'un algorithme polynomial.

• On a clairement $P \subseteq NP$. En effet, si on peut résoudre un problème par un algorithme polynomial, alors on peut aussi vérifier en temps polynomial que la solution fournie est bien une solution du même problème.

• La question de savoir si $P = NP$ est un problème ouvert, le plus important, de la théorie de la complexité. Cela revient à savoir si le fait de chercher une solution est aussi simple que de vérifier une solution.

De nombreuses personnes pensent que $P \neq NP$.

II.4.3 La classe NPC

Elle est constituée par les problèmes les plus difficiles de NP. Les problèmes $NP - complet$ sont tous équivalents en termes de difficultés. Pour affirmer que certains problèmes sont les plus difficiles, il faut pouvoir comparer les problèmes entre eux. On définit pour cela la notion de réduction polynômial sur la classe NP.

Définition II.4.3. Une réduction est une transformation d'un problème en un autre ; ceci permet de capturer la notion informelle de « problème au moins aussi difficile qu'un autre problème ». Plus précisément, si un problème X peut être résolu en utilisant un algorithme permettant de résoudre un problème Y, c'est que X n'est pas plus difficile que Y ; on dit alors que X se réduit à Y. Par exemple, le problème consistant à élever un nombre au carré se réduit au problème plus général de multiplication de deux nombres (ici, aucune transformation n'est nécessaire). Une réduction est polynômial lorsque le processus de transformation peut se faire en temps polynomial.

Définition II.4.4. Soient p et p' deux problèmes de décision. On dit que p se transforme (ou se réduit) polynomialement en p' s'il existe un algorithme polynomial transformant toute instance x de p en une instance x' de p' admettant la même réponse que x. On écrit alors $p \prec p'$. Autrement dit, les instances **oui** sont transformées en instances **oui**, et les instances **non** sont transformées en instances **non**.

Proposition II.4.2 (Optimisation combinatoire)**.** *Si p se réduit polynomialement à p' et s'il existe un algorithme polynomial pour p', alors il existe un algorithme polynomial pour p.*

Remarque II.4.2. • La relation $\prec$ est transitive et $p \prec p'$ signifie que p n'est pas plus difficile que p'.
•Ainsi, p **est polynomialement réductible par** p' signifie que si l'on connaît un algorithme polynomial résolvant p', on en déduit que $p \in P$. En effet, on traduit les instances de p en instances de p', puis on résout p' et enfin on retraduit les solutions de p' en solutions de p, tout cela se fait en temps polynomial.

Définition II.4.5. Un problème de décision est NP- complet s'il vérifie les deux conditions suivantes :
• il appartient à la classe NP,
• tous les problèmes de la classe NP se ramènent à celui-ci via une réduction polynômial.

Il est facile de voir que la restriction de la notion de **réduction polynômial** sur les problèmes $NP - complet$ est une relation d'équivalence,donc s'il existe un algorithme

polynomiale pour un seul élément de la classe $NP - complet$, on pourrait en déduire un algorithme polynomial pour n'importe quel autre élément dans la même classe, et d'après la proposition précédente, on aurait aussi $P = NP$.

II.5 Quelques problèmes de la théorie des graphes et leur complexité

Dans cette partie, nous définissons quelques problèmes de la théorie des graphes. Pour les problèmes d'optimisation, nous associons un problème de décision.

II.5.1 Quelques problèmes classiques

Les premiers problèmes décrits ici sont des problèmes classiques de théorie des graphes. Leur complexité pour différentes classes de graphes P est donnée dans [11].

P-Clique Maximum On s'intéresse à la recherche d'une clique de cardinalité maximale dans un graphe G satisfaisant P. Dans le problème de décision associé, soit k un entier, la question est celle de l'existence d'une clique de cardinalité supérieure ou égale à k. Ce problème est NP-complet de manière générale. Cependant pour toute valeur d fixée, $\{\Delta \leq d\}-k-$Clique Maximum est polynomial.

P-Stable Maximum On s'intéresse à la recherche d'un stable de cardinalité maximale dans un graphe G satisfaisant P. Dans le problème de décision associé, soit k un entier, la question est celle de l'existence d'un stable de cardinal supérieur à k. Ce problème est NP-complet pour les graphes cubiques planaires.

Chapitre III

Split Graphes

III.1 Introduction

Dans ce troisième chapitre, nous définissons la classe des graphes qui est les split, tout en donnant des généralités et extensions ainsi que l'algorithme de reconnaissance de cette classe de graphe.
Le chapitre suivant sera consacré aux bisplit graphes et leurs extensions.

III.2 Préliminaires et définitions

Définition III.2.1. [15] Un graphe est appelé Split graphe ou (graphe scindé) si son ensemble de sommets peut être partitionné en deux ensembles $V1$ et $V2$ tels que $V1$ induit un ensemble stable et $V2$ induit une clique, La perfection des split graphes découle du fait qu'ils sont triangulés.

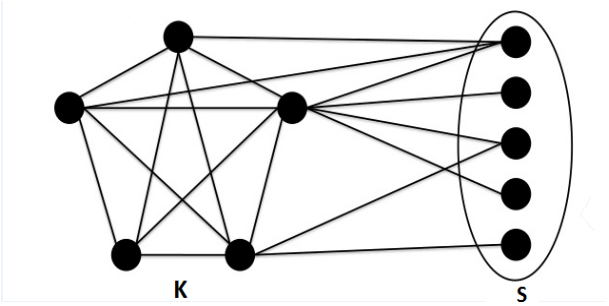


FIGURE III.1 – Un split graphe

Définition III.2.2. [13],[15] Un graphe non orienté G=(V,E) est dit split s'il existe une partition $V = I \cup K$ de son ensemble de sommets V en un ensemble indépendant I et

un graphe complet K. Étant donnée la sequence de degré des sommets du graphe, il est possible de reconnaître s'il est split en un temps $O(|V|)$.

Les split graphes sont définis par Hammer et Foldes 1977 qui ont prouvé en particulier qu'un graphe est split si et seulement s'il ne contient pas les graphes induits suivants : $C_4, C_5, 2K_2$, où C_k est un cycle sans corde avec k sommets, et $2K_2$ est formé de deux cliques de longueur 2, Il n'y a pas de restriction sur les arêtes entre les sommets de K. En générale, la partition $V(G) = K + S$, des split graphes n'est pas unique. Aussi K et S ne sont pas, nécessairement, maximums. L'exemple de la figure suivant le montre.

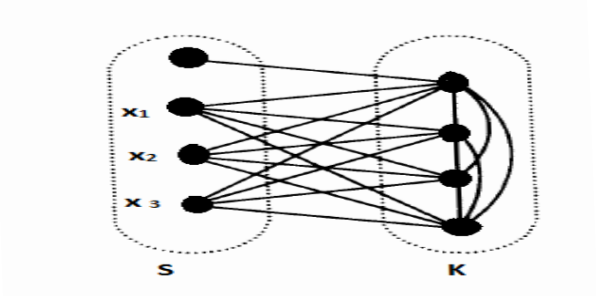


FIGURE III.2 – Split graphe

Remarque III.2.1. Dans les deux exemples précédents, on remarque bien que S est l'ensemble Stable, et K est l'ensemble **Clique.**

Remarque III.2.2. $\bar{G}$ est aussi un split graphe, car la clique de G est un stable de $\bar{G}$ et le stable de G est une clique de $\bar{G}$.

Définition III.2.3. split graphe complet
un graphe est split complet s'il peut être divisé en un ensemble indépendant S et un ensemble clique K tel que chaque sommet dans S soit adjacent à chaque sommet dans la clique.

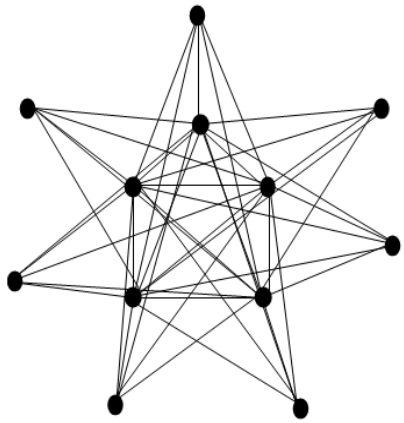


FIGURE III.3 – Un split graphe complet.

Théorème III.2.1. *[15] Un graphe non orienté G est split graphe si et seulement si son complémentaire $\bar{G}$ est split.*

Théorème III.2.2. *[15] Les points suivants sont équivalents :*

1. G est un split graphe.

2. G et $\bar{G}$ sont triangulés.

3. G ne contient aucun sous graphe induit isomorphique à C_4, C_5, $2K_2$.

Preuve. *[15]* 1. $\Rightarrow$ 2. *Soit $G = (V, E)$ ayant la partition de sommets $V = S + K$ avec S stable et K complet,*
supposons que G contient un cycle sans corde de longueur ≥ 4, Au moins un et au plus deux sommets (adjacentes) de G doivent être dans K,
les deux cas implique que S contient une pair de deux sommets adjacents, contradiction, par consequent G doit être triangulé, d'après le théorème 1 $\bar{G}$ est split donc $\bar{G}$ est triangulé.

Théorème III.2.3. *[13] Soit G un split graphe , $V(G) = K + S$. exactement une des deux conditions est prise :*

1. $|K| = \omega(G)$ et $|S| = \alpha(G)$ (dans ce cas la partition $K + S$ est unique).

2. $|K| = \omega(G) - 1$ et $|S| = \alpha(G)$ (dans ce cas il existe un $y \in S$ telque $K + y$ est complet).

3. $|K| = \omega(G)$ et $|S| = \alpha(G) - 1$ (dans ce cas il existe un $x \in K$ telque $S + x$ est stable).

Preuve. *[13] Comme l'ensemble stable et l'ensemble complet ont aux plus un sommet en commun, alors le split graphe vérifie la somme* $\alpha(G) + \omega(G) = |V|$ *ou* $|V| + 1$
Si $\alpha(G) + \omega(G) = |V|$ *alors on est dans le cas (i) ;*
On suppose dans ce cas, il y a une autre partition $V = S' + K'$ *Soit* $x = S \cup K'$ *et* $y = S \cup K$ *Si x et y sont adjacents dans G, alors* $x + k$ *est une clique de cardinalité* $\alpha(G) + 1$*, Ce qui est impossible.*
Si x et y sont non adjacents dans G, alors $y + s$ *est un ensemble stable de cardinalité* $\alpha(G) + 1$*, ce qui est impossible, donc la partition* $V = S + K$ *doit être unique.*
Si $\alpha(G) + \omega(G) = |V|$ *alors on est dans le cas (ii) ou bien le cas (iii)*
Soit $|S| = \alpha(G)$, $|K| = \omega(G) - 1$ *Soit* $K^{'}$ *une clique de cardinalité* $\omega(G)$ *comme* $S + K$ *est une partition et K' est plus grand que K,* $S \cap K'$ *doit être non vide c'est pourquoi la caridinalité 1 Soit* $x = S \cap K'$ *il suit que* $k' = k + x$ *qui est complet.*

Corollaire III.2.1. *[13] Soit G un split graphe avec la partition* $V(G) = K + S$ *telque :* $K = \{x_1, x_2, \ldots\ldots, x_n\}$ *et* $S = \{y_1, y_2, \ldots\ldots, y_m\}$*. Alors cette partition est unique si et seulement si* $d_G(x_i) \geq n$ *pour* $i = 1, 2, \ldots\ldots n$ *et* $d_G(y_i) \leq n - 1$ $i = 1, 2, \ldots\ldots m$*.*

Théorème III.2.4. *[13] Un graphe est split si et seulement s'il ne contient aucun sous ensemble de sommets qui induit un* C_4,C_5 *ou* $2K_2$*.*

Lemme III.2.1. *[13] Pour chaque split graphe G, un ordre de degré décroissant est un ordre d'élimination simplicical.*

Preuve. *[13] Soit* $G = (V, E)$ *un split graphe,* (K, I) *est une split partition de G, et* α *un ordre croissant de G.*
Noter que chaque sommet de I est simplicial dans G. chaque sommet v avec $d(v) >|$ *K* $|$ *appartient à K, et chaque sommet v avec* $d(v) <|$ *K* $|$ *appartient à I, ainsi il est simplicial. par consequent, s'il n y a pas de sommets de degré égale alors* α *est trivialement une peo. Supposons que G a des sommets de degré* $|$ *K* $|$*. s'ils sont tous simpliciaux dans G, alors* α *devrai être peo.*
Soit x un sommet non simplicial de degré $|$ *K* $|$ *avec le plus petit indice dans* α*. Ainsi tous les sommets qui apparaît avant x dans* α *sont simplicial dans G.puisque x n'est pas simplicial, x appartient à K et a un voisin w dans Iqui n'est pas adjacent aux autres sommets de K. puisque x est dans K, et a un degré* $|$ *K* $|$*, et le seul voisin de x dans I. d'ailleurs,* $d(w) <|$ *K* $|$ *puisque w n'est pas adjacent à tous les sommets de K. ainsi w est ordonné avant x dans* α*.*
Après l'elimination de w, x devient simplicial. cet argument peut être répété pour tous les

sommets non simpliciaux de degré | K |. par consequent, α est un ordre d'elimination simplicial(peo).♦

III.2.1 Séquence de degrés

Définition III.2.4. [15] Une séquence $\Delta = \{d_1, ...d_n\}$ de nombres entiers,
$n-1 \geq d_1 \geq d_2..... \geq d_n \geq 0$, est appelée graphique s'il existe un graphe non orienté ayant une séquence de degré Δ . Par exemple , la séquence $[2,2,2,2]$ correspond à un cycle sans corde de longueur 4 (C_4), tandis que la séquence $[2,2,2,2,2,2]$ correspond à la fois à $2K_3$ et C_6 . Il est facile de construire des séquences qui ne sont pas graphiques, telles que $[1,1,1]$ et $[4,4,2,1,1]$.

Définition III.2.5. Arc incident à un sommet : Si un sommet x est I'extrémité initiale d'un arc u, on dit que I'arc u est incident à x vers I'extérieur. Dans un graphe G, le nombre d'arcs incidents à x vers I'extérieur se note $d_G^+(x)$, et s'appelle le demi-degré extérieur de x. On définit de même un arc incident à x vers l'intérieur, et le demi-degré intérieur $d_G^-(x)$.

Définition III.2.6. Degré d'un sommet : $d_G(x) = d_G^+(x) + d_G^-(x)$: nombre d'arcs ayant une extrémité en x (chaque boucle est comptée deux fois).

Définition III.2.7. Le degré d'un sommet v,noté $d_G(x)$, est égal au nombre $|N(v)|$ de ses voisins.

Théorème III.2.5. *[15] Pour tout graphe simple G on a :*

$$\sum_{u \in V} d_G(u) = 2m$$

La somme des degrés des sommets d'un graphe est égale à deux fois le nombre d'arêtes.

Preuve. *[15] Une arête **uv** a deux extrémités, par consequent chaque arête est comptée deux fois, une pour **u** et l'autre pour **v**.*

Corollaire III.2.2. *le nombre de sommets de degré impair est pair.*

Preuve. *[18] Diviser V(G) en V_{impair}=$\{u \mid d_G(u)$ est impair $\}$ et V_{pair}=$\{u \mid d_G(u)$ est pair $\}$. Alors :*
$2m = \sum_{u \in V_{impair}} d_G + \sum_{u \in V_{pair}} d_G$

il est claire que $\sum_{u \in V_{pair}} d_G$ *est pair.*

Donc, $\sum_{u \in V_{impair}} d_G$ *est aussi pair.*

Définition III.2.8. [18] Un graphe est dit régulier si tous ses sommets ont le même degré. G est k-régulier si et seulement s'il est régulier de degré k.

Théorème III.2.6. *[18] La séquence d'entiers*
n-1$\geq d_1 \geq d_2 \succeq ... \succeq d_n \succeq 0$. *est un graphique si et seulement si :*

i) $\sum_{i=1}^{n} d_i$ *est pair*

ii) $\sum_{i=1}^{r} d_i = r(r-1) + \sum_{i=r+1}^{n} min\{r_i d_i\}$ *pour r=1,2,..,n-1.*

L'inégalité (ii) est appelée l'inégalité de **Erdos-Gallai (EGI**).

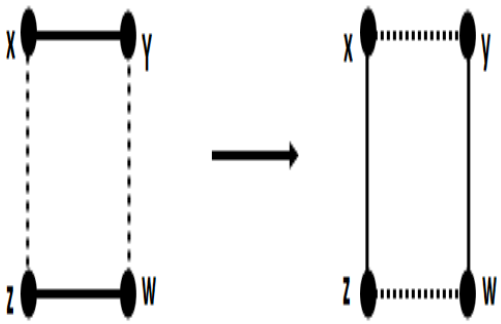


FIGURE III.4 –

Dans l'exemple précèdent on a :

Soit x, y, z et w des sommets distincts de G, avec xy et zw des aretes de G et xz et yw sont des non arêtes de G c'est à dire n'y appartiennent pas.
Si on remplace les deux arêtes avec les deux non-arêtes, le graphe obtenu G' va avoir la même séquence de degré que G, tel que le remplacement sera appelé permutation.

Remarque III.2.3. [18] Si deux graphes ont la même séquence de degré, alors chacun peut être obtenu à partir de l'autre avec une séquence finie de permutations.

Théorème III.2.7. *[18] Soit G=(V,E) un graphe non orienté avec une séquence de degré* $d_1 \geq d_2 \succeq ... \succeq d_n$

soit $m = max\{i | d_i \geq i-1\}$ *alors, G est split graphes si et seulement si :*
$\sum_{i=1}^{m} d_i = m(m-1) + \sum_{i=m+1}^{n} d_i$
d'ailleurs, si c'est la cas alors $\omega(G) = m$.

Preuve. *[18] Le théorème est vrai si G est un graphe complet. Alors on peut supposer que* $d_m \geq m-1$ *et* $d_m < m$.
puisque Δ *est croissant,* $min\{m, d_i\} = d_i$ *pour* $i \geq m+1$, *par conséquence le mth EGI se simplifie à :*

$S = \sum_{i=1}^{m} d_i \leq m(m-1) + \sum_{i=m+1}^{n} d_i$ (1)

soit K un ensemble des premiers m sommets du plus grand degré, la somme de gauche va être divisée en deux contributions :
$S = S_1 + S_2$, *ou*
$S_1 = \sum_{x \in K} \mid \{z \in K / xz \in E\} \mid \leq m(m-1)$ *(2)*

$S_2 = \sum_{x \in K} \mid \{y \notin K / xy \in E\} \mid$

$= \sum_{y \notin K} \mid \{x \in K / xy \in E\} \mid \leq \sum_{i=m+1}^{n} d_i$ *(3)*

$\Longrightarrow$ *l'inégalité tient en (2) si et seulement si K est complet.*
L'inégalité tient en (3) si et seulement si V-K est stable par consequent l'inégalité (1) alors G est split graphe.

$\Longleftarrow$ *par contre, pour prouver l'inverse, on suppose que G=(V,E) est split graphe. D'après le théorème 2.2.3 on peut diviser V en un ensemble stable et un ensemble complet K tel que :* $\mid K \mid = \omega(G)$.
chaque sommet dans K a au moins un degré de $\mid K \mid -1$ *et tant que K est maximum, alors chaque sommet dans S a un degré au plus* $\mid K \mid -1$.
c'est pourquoi, on peut supposer que les sommets sont ordonnés tels que : $K = \{v_1, .., v_{|k|}\}$ *et* $S = \{v_{|k|} .., v_n\}$ *ou* $deg(v_i) = d_i$.
De plus $d_{|k|} \geq \mid K \mid -1$ *et* $d_{|k|+1} \leq \mid K \mid -1 \leq \mid K \mid$

alors $\omega(g) =\mid K \mid= m$.
Finalement, vu que K est complet S=V-K est un stable.
On conclut que l'inégalité tient en (2) et (3) par conséquent en (1).♦

Corollaire III.2.3. *si G est split graphe alors chaque graphe avec la même séquence de degré est split graphe.*

Remarque III.2.4. [15] Hammer et simeone [1977] ont exploré davantage de problèmes dans les graphes, Ils ont défini la **splittance** d'un graphe aléatoire non orienté, qui est le nombre minimum d'aretes qui seront ajoutées ou supprimées pour obtenir un split graphe. c'est clair que les splits graphes sont ceux qui ont leur splittance égale à zero. leur résultat principal montre que la splittance dépend seulement de la séquence de degré d'un graphe.

III.2.2 Ordre d'elimination simplicial (peo)

Définition III.2.9. Dans un graphe G, un sommet v est dit simplicial si le sous graphe de G induit par l'ensemble des sommets $\{v\} \cup N(v)$ est un graphe complet.

Par exemple, dans le graphe ci-dessous, le sommet 3 est simplicial, tandis que le sommet 4 ne l'est pas.[14]

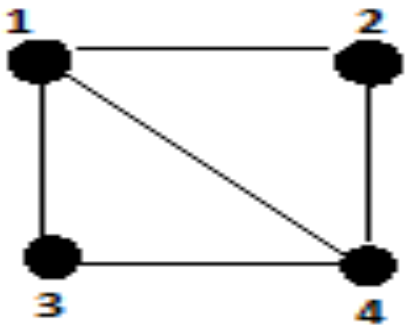


FIGURE III.5 – Exemple d'une PEO

Définition III.2.10. [22],[21] Un graphe G à n sommets a un ordre d'élimination simplicial si et seulement s'il y a un ordre $\{v_1, ..., v_n\}$ des sommets de G, de sorte que chaque v_i est simplicial dans le graphe induit par les sommets $\{v_1, ..., v_n\}$. A titre d'exemple ,le graphe précèdent ayant comme ordre d'élimination simplicial l'ordre $(2, 1, 3, 4)$.

Proposition III.2.1. *[22],[21] Si G admet un ordre d'élimination simplicial, donc tous ses sous graphes induits admettent eux aussi un ordre d'élimination simplicial.*

Preuve. *[22],[21],[14] soit* $\{v_1, ..., v_k\}$ *l'ordre d'elimination simplicial de* G $i_1 < i_1 < ..... < i_k$*, une sous sequence de* $\{1, 2, ..., n\}$*, et* H *correspondant a un sous graphe de* G *dans* $v_{i_1}, ..., v_{i_k}$*, d'après la definition nous avons chacun des graphes*

- $(\{v_{i_k}\} \cup N(v_n)) \cap \{v_1, ..., v_{i_k}\}$,
- $(\{v_{i_{k-1}}\} \cup N(v_{n-1})) \cap \{v_1, ..., v_{i_{k-1}}\}$,
- .....
- $(\{v_{i_1}\} \cup N(v_{i_1})) \cap \{v_1, ..., v_{i_1}\}$,

étaient cliques dans G, Par conséquent, en nous limitant à H, nous savons que tous les jeux :

- $(\{v_{i_k}\} \cup N(v_n)) \cap \{v_1, ..., v_{i_k}\}$,
- $(\{v_{i_{k-1}}\} \cup N(v_{n-1})) \cap \{v_1, ..., v_{i_{k-1}}\}$,
- ......,
- $(\{v_{i_1}\}$.

sont toujours cliques, cependant, ces sous graphes H admettent toujours un ordre d'elimination simplicial.

Proposition III.2.2. *Si G admet un ordre d'élimination simplicial, alors G est parfait.*

Preuve. *[22],[21],[14] Par notre proposition ci-dessus, il suffit juste de montrer que* $\gamma(G) = \omega(G)$ *pour tout graphe G avec un ordre d'élimination simplicial. Nous procédons par récurrence sur le nombre de sommets dans G. Si* $|V(G)| = 1$ *, G est trivialement parfait, car il est* $K1$*.*
Supposons maintenant que $V(G) = n > 1$*, pour un certain n, et que* $\{v_1...v_n\}$ *soit l'ordre d'elimination simplicial . Regardez le graphe* $G \setminus \{v_n\}$*, formé par suppression* $\{v_n\}$ *de G. Par notre proposition,* $G \setminus \{v_n\}$ *admet encore l'ordre d'elimination simplicial . Par conséquent , nous pouvons appliquer notre hypothèse inductive de voir que* $G \setminus \{v_n\}$ *est parfait : à savoir que* $\gamma(G \setminus \{v_n\}) = \omega(G \setminus \{v_n\})$*.*

définir $k = \omega(G \setminus \{v_n\})$*, Dans G lui-même, par définition , nous savons que la collection de sommets* $\{v_n\} \cup N(v_n)$ *induit une clique comme un sous-graphe : par conséquent, nous savons que* $\{v_n\} \cup N(v_n)$ *lui-même induit une clique, et donc que*
$d(v_n) = N|(v_n)| \leq \omega(G \setminus \{v_n\}) = k$*. Alors* v_n *a moins de k voisins.*
supposons que $d(v_n) < k$*. alors, compte tenu de tout k-coloration de* $G \setminus \{v_n\}$*. En accordant à une coloration de G simplement laisser quelque soit la couleur de* $\{v_n\}$ *dans* $\{1, 2, ...k\}$ *ne se présente pas dans son voisin. Cela signifie que* $\gamma(G) = k = \omega(G \setminus \{v_n\}) \leq \omega(G)$*, et*

donc G est parfait.
A l'inverse , supposons que $d(v_n) = k$. Puis $\{v_n\} \cup N(v_n)$ forme une clique de taille $k+1$, donc $\omega(G) \geq k+1$. Enfin , parce que $\gamma(G \setminus \{v_n\}) = k$, nous pouvons étendre toute k-coloration de$(G \setminus \{v_n\})$ à k+1-coloration de G en colorant v_n avec la couleur $k+1$; ceci montre que $\gamma(G) \leq k+1 \leq \omega(G)$, et donc (encore une fois) que G est parfait .

La seule partie un peu insatisfaisante de cette nouvelle famille de graphes est que leur propriété (cet ordre d'elimination simplicial) n'est pas facile à vérifier.

Théorème III.2.8. *Pour tout split graphe G, le degré d'ordre croissant est un ordre d'élimination simplicial.*

Soit l'ordre $\alpha = (v_1, v_2, ..., v_n)$ si α est tel que $d_{v_1} \leq d_{v_2} \leq ..... \leq d_{v_n}$, il est appelé un ordre croissant. si α est tel que $d_{v_1} \geq d_2 \geq ..... \geq d_{v_n}$ c'est un ordre décroissant.

III.2.3 L'Algorithme de l'ordre d'elimination simplicial

Fonction ordre d'elimination simplicial(G,σ)

1. **Pour** i =1 à n-1 **faire**
2. definir V dans $\sigma^{-1}(i)$
3. definir $m[v]$ dans $\sigma^{-1}(min\{\sigma(w)|w \in adj(v), \sigma(w) > \sigma(v)\})$
4. **Pour tout** les sommets w adjacents au sommet v **faire**
5. **si** $\sigma(m(v)) < \sigma(w) etw \notin adj(m(v))$ **alors**
6. retourne Faux
7. **fin if.**
8. **fin for.**
9. **retourne vrai**
10. **Fin fonction.**

TABLE III.1 – Algorithme **peo**

III.2.4 L'Algorithme de Reconnaissance des Split Graphes

ENTREE : un graphe $G = (V, E)$.
SORTIE : retourne "oui" si G est split graphe avec la partition V en un ensemble indépendant I et une clique K, ou bien
retourne un "non" si G n'est pas split graphe avec un sous ensemble de sommets incluant un $2K_2$, C_4, ou C_5.

calculer l'ordre de degree décroissant $\alpha = (v_1, v_2, ..., v_n)$ de G ;
Si α n'est pas une **peo** de G **alors**
trouver deux voisins v_j et v_k de v_i tels que $v_j v_k \notin$ E et $i < j < k$;
Si $v_j et v_k$ ont un voisin en commun z avec $v_i z \notin$ E
Alors $\{v_i, v_j, z, v_k\}$ induit un C_4 ;
Sinon trouver deux sommets x et y tels que $v_j x, v_k y \in E$ $v_j y, v_k x, v_i x, v_i y \notin E$;
Si $xy \in E$ **alors**
$\{v_i, v_j, x, v_k, y\}$ forment un C_5 ;
Sinon
$\{v_i, v_j, x, v_k, y\}$ induit un $2K_2$;
Fin-si.
Fin-si.
afficher "non" et l'ensemble des sommets induit $2K_2$, C_4, ou C_5. constaté ci-dessus ;
Retourner ;(1)
Fin-si.
calculer la longueur de la plus grande clique, et la noter k ;
$K = 0; I = 0; i = n;$
Tant que $|K| \leq k - 1$ **faire**
$A = N(v_i) \cap K$;
Si $|A| = |K|$ **alors** $K = K \cup v_i$ **Sinon** trouver un voisin $x \notin K$ et un non voisin $y \in K$ de v_i ; trouver un voisin z de y tel que $v_j z, xz \notin E$; afficher "non" et $\{v_i, x, y, z\}$ induit un $2K_2$; **Retourner ; (2)**
Fin-si.
$i = i - 1$; **Fin- tant que** .
Tant que $i \geq 1$ **faire** $A = N(v_i) \cap (K \cup I)$;
Si $A \subseteq K$
Alors $I = I \cup \{v_i\}$;
Sinon Désignons par x l'unique sommet dans $A \cap I$; Trouver un sommet $y \in K$ tel que $xy, v_i y \notin E$; trouver un voisin z de y avec $xz \notin E$; retourner "non" et $\{v_i, x, y, z\}$ induit un

$2K_2$; **Retourner ; (3)**
Fin-si.
$i = i - 1$; **Fin-tant que** .
retourner "oui", K et I.
Fin de l'algorithme.

Remarque III.2.5. Dans l'algorithme, "**retourne**", veut dire que l'algorithme se termine dans ce point et les lignes suivantes ne sont pas exécutées.

III.2.5 Organigramme de l'algorithme

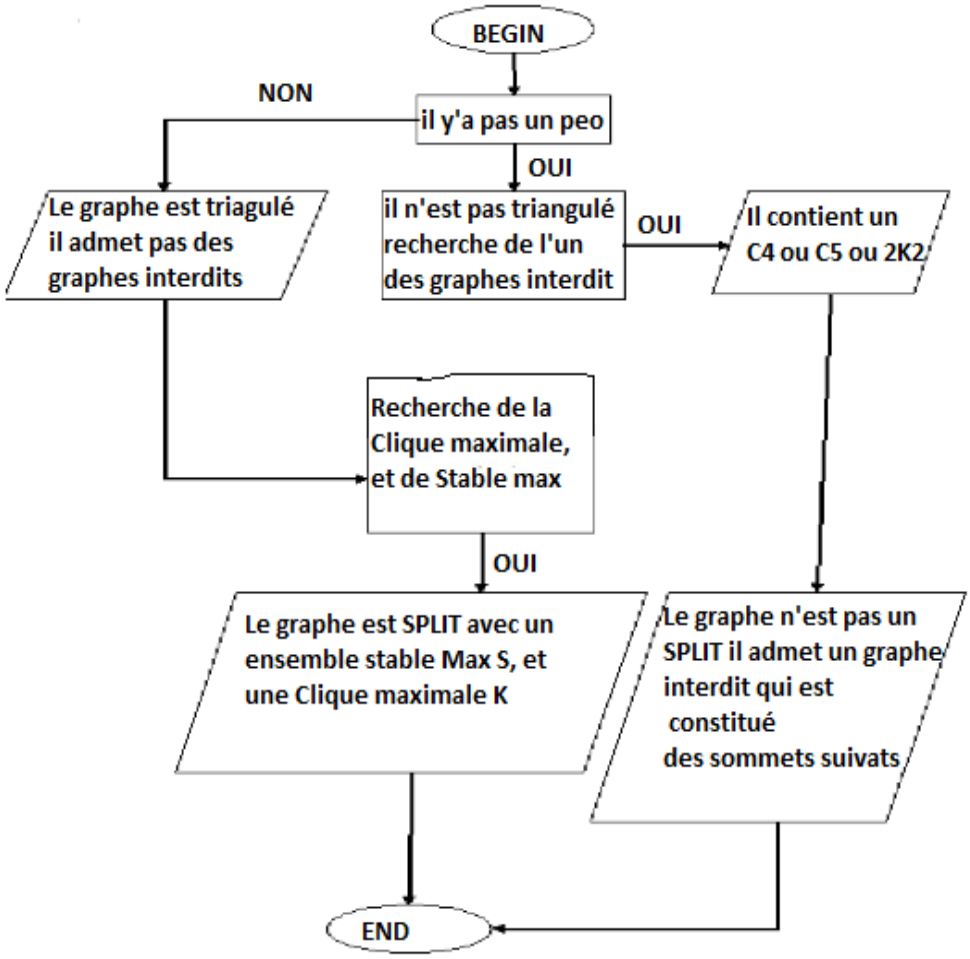


FIGURE III.6 – Organigramme de l'algorithme

Lemme III.2.2. *L'algorithme de reconnaissance de split graphe est correct.*

Preuve. *[16] si l'algorithme retourne OUI, alors le graphe est split, puisque chaque sous graphe induit de chaque itération est split, attester par les ensembles calculer K et I.*
Noter en particulier que tous les sommets isolés sont placés dans I, sauf quand G consiste seulement en les sommets isolés, dans ce cas, un seul sommet seulement placé dans K et le reste dans I.
Si le graphe est split, alors la séquence de degré décroissante est ***peo****(ordre d'élimination simplicial) d'après le théorème 2.2.3*
et l'algorithme place les derniers k sommets de cet ordre dans la clique obtenue K.Donc le cas "NON" ne s'offrira pas et l'algorithme retourne un "OUI" avec la correcte partition split.
Ainsi, l'algorithme délivre(retourne) les correctes "OUI" ou bien sans réponse.
On a besoin d'argumenter que dans le cas du résultat "NON" le sous ensemble de sommets obtenu est un sous graphe interdit de G.
les cas numéroter (1),(2),(3) dans l'algorithme, on va considérer chaque cas séparément.

(1). Dans ce cas, il existe un sommet v_i avec les voisins $v_j$et v_k tels que : $v_jv_k \notin E$, et $i < j < k$
tant que les degrés des v_j et v_k sont en moins aussi grand que v_i, les deux $v_j$et v_k doivent avoir un autre voisin que v_i. pour chaque voisin en commun z de v_j et v_k, aussi l'arete $v_iz \in E$, sinon G aura un cycle c_4 sans corde induit par les sommets $\{v_i, v_j, z, v_k\}$.
On suppose que chaque voisin en commun de v_j et v_k est aussi un voisin de v_i. Dans ce cas, chacun de v_j et v_k doit avoir un autre voisin que v_i qui n'est pas adjacents aux autres et à v_i. Ainsi il y a les arêtes v_jx et v_ky telles que : $x \neq y \neq v_i$, si non plus x ou y est adjacent à v_i.
Maintenant, soit $xy \in E$ et on a un c_5 induit par $v_jx, v_k, y\}$. ou $xy \notin E$ et on a un $2K_2$ induit par $\{v_j, x, v_k, y\}$. pour le reste de la preuve, on sait que α est peo, puisque l'algorithme ne se termine en (1). c'est pourquoi on sait aussi que G est triangulé, et la taille maximale k d'une clique de G peut être calculer en O(n+m).

(2).Dans ce cas, $\mid K \mid \leq k-1$ on sait qu'il y a k sommets de degré en moins k-1, dans le graphe, et puisque v_i est choisi avant quelque sommets $d(v_i) \geq k-1$
$d(v_i) \geq k-1$, ainsi puisque v_i a en moins un non-voisin $y \in K$ et il en moins un voisin $x \notin K$. remarque que un voisin x de v_i en dehors de K peut être adjacent à y, cela parce que chaque tel voisin a une antérieure position en α et son plus grand voisin en $K \cup \{v_i\}$

induit une clique tant que α est un ordre d'élimination simplicial.
Ainsi, $v_i y, xy \notin E$ puisqu'il y a un degré en moins aussi grand que v_i $d(y) \geq k-1$, et puisqu'il y a au plus k-2 sommets en K,avec y (en ajoutant y),le sommet y doit avoir un voisin z en dehors de K. La position de x et z en α est avant v_i et y. C'est pourquoi xz ne peut pas être une arete de G,autrement $v_i y$ peut être une arete dans G par la propriété de ***peo*** *ainsi $\{v_i, x, y, z\}$ induit un $2K_2$.*

(3).Dans ce cas, v_i a exactement un voisin x dans I, puisque I est un ensemble indépendant. Le grand nombre de voisins de v_i induit une clique par la propriété ***peo*** *de α. on remarque qu'un non-voisin de x dans K peut être un voisin de v_i,puisque x a la plus grande position en α(qui est* ***peo****)*
Donc l'ensemble des voisins de v_i en K est un sous ensemble de voisins de x en K, en plus, x a au plus k-1 voisins dans K parce que autrement on a une clique de taille k+1, ainsi il y a un sommet y dans K, tel que $xy, v_i, y \notin E$.
Maintenant, on va argumenter qu'il y a un voisin z qui n'est pas adjacent à x ou à v_i,on suppose d'abord qu'il y a un voisin z en dehors de K, le sommet z ne peut être adjacent à x ou v_i puisque chaque z est déjà placé dans I, ou bien il a une petite position en α et cela va violer la propriété ***peo*** *de α.*
On suppose pour l'autre cas qu'il n'y a aucun voisin en dehors de K, cela veut dire que $d(y) = k-1$ puisqu'il y a la plus grande position en α que x, on peut conclure que $d(y) \leq k-1$ puisque v_i est voisin de x, x a au plus k-2 voisins en K et aussi il y a au moins deux sommets en K qui sont non adjacents à x, et séquentiellement à v_i, donc, dans ce cas, y a un voisin z en K qui n'est pas adjacent à x ou à v_i, dans chaque cas on a le $2K_2$ induit par $\{v_i, x, y, z\}$ ♦

Lemme III.2.3. *[16] Le temps d'exécution de l'algorithme de reconnaissance de split graphe est O(n+m).*

Preuve. *[16]* ***Golumbic*** *décrit la procédure de temps linéaire qui vérifie si un ordre donné est peo, et si ce n'est pas le cas, procurer les sommets v_i, v_j, v_k qui sont utilisés dans l'algorithme de reconnaissance split, d'ailleurs[15], il est expliqué comment la taille de la clique maximale dans les graphes triangulés peut être calculée en temps linéaire. pour le reste de l'algorithme, pour chaque v_i le temps nécessaire est linéaire en nombre d'arêtes entre v_i et le processus des sommets. tant que v_i est placé en K ou en I.Ainsi le temps total obligatoire par toutes les étapes qui placent un sommet en K ou en I est O(n+m). A chaque fois qu'on commence la recherche des sous graphes interdits on est déjà sure que la réponse est NON. ainsi on cherche un sous graphe interdit au plus une fois pendant l'execution de l'al-*

gorithme. trouver un tel sous graphe interdit demande un temps O(n), puisque il consiste juste a verifier les voisins d'au plus 4 sommets. ♦

Lemme III.2.4. *[16] Les certificats retournés par Algorithme reconnaissance-Split peuvent être authentifiés en $O(n+m)$ pour les graphes entrés qui sont split, et en $O(n)$ pour ceux entrés qui ne sont pas split.*

Preuve. *[16] Lorsque l'algorithme conclut que le graphe est split, il retourne K et I.Il peut être vérifié en O (n + m) le temps de prouver que K est en effet une clique et I est en effet un ensemble indépendant, et qu'il n'y a pas d'autres sommets du graphe. Quand l'algorithme conclut que le graphe n'est pas split, un sous-ensemble de 4 ou 5 sommets est retourné comme un certificat . Par observation 2.2, il peut être vérifié en temps O (n) que le sous-ensemble de sommets induit en effet un sous-graphe interdit.*

Par le lemme ci-dessus, nous avons prouvé le résultat suivant,

Théorème III.2.9. *[16] Il y a un temps de certification linéaire de l'algorithme de reconnaissance de split graphe qui délivre des certificats qui peuvent être authentifiés en un temps de O (n + m) pour l'adhésion et en temps O (n) pour les non appartenances.*

III.2.6 Absolute retract de split graphes

Dans cette section on veut caractériser absolute retract de split graphe et les splits graphes qui sont absolument rétractés, i.e., absolute retract dans toutes les classes de graphe.

Lemme III.2.5. *Soit G un split graphe,V (G) = K + S, est un sous graphe complet maximum.soit H un sous graphe de G avec une partition induite $V(H) = K' + S'$, alors H est un absolute retract de G si et seulement si $\omega(H) = \omega(G)$ et si $x \in K - K'$ alors $N_{G(x)} \cap S' = \emptyset$.*

Théorème III.2.10. *Un graphe split est un absolute retract de split graphe si et seulement si la partition de son ensemble de sommets en un ensemble stable et un ensemble complet est unique ou bien il est un split graphe complet*

Corollaire III.2.4. *soit G un split graphe et soit G et son complement $\bar{G}$ relié. alors G est un retract absolu de split graphe si et seulement si $\bar{G}$ est un retract absolu de split graphe.*

soit un retract absolu, $\omega(G) = n$. alors G est (n-1) connecté.

III.2.7 Partitionement complet des split graphes

Il existe des questions sur le partitionnement d'un split graphe en des parties connexes. Le résultat principal est de savoir si un split graphe est complètement décomposable.
Soit $\alpha = (\alpha_1, .., \alpha_n)$ les partitions des n sommets du graphe.
Le graphe G est appelé α-décomposable s'il existe une partition de V en des ensembles disjoints $A_1, .., A_k$ de cardinalité $|A_i| = \alpha_i$ pour $1 \leq i \leq k$ telle que chaque ensemble A_i induit le sous graphe connexe de G.
Une telle partition est appelée α-décomposable.
Un graphe est appelé complètement décomposable s'il est α-décomposable pour chaque partition α de n.
Il existe deux questions algorithmiques importantes pour la α-décomposition du graphe.
• Q_1. Décider si un graphe donné est α-décomposable pour une partition donnée α.
• Q_2.Décider si un graphe donné est complètement décomposable

Théorème III.2.11. *Il est NP-difficile de décider si un split graphe donné avec n sommets est α-décomposable pour une partition donnée α.*

Théorème III.2.12. *Il peut être décidé en un temps polynomial si un split graphe avec n sommets est complètement décomposable.*

Et puisque le split graphe est triangulé, alors on pourra parler du partitionement de ce dernier en des ensembles indépendants et cliques.

Remarque III.2.6. Un graphe est dit (k,l)-graphe si son ensemble de sommets peut être partitionné en K ensembles indépendants et l cliques(le split graphe est obtenu quand k = l = 1).

Chapitre IV

Bisplit graphes

IV.1 Introduction

Dans ce quatrième chapitre, nous définissons une autre classe qui est celle des bisplit graphes, tout en donnant des généralités et extensions ainsi quel l'algorithme de reconnaissance de cette classe de graphes.
Dans le chapitre qui suit, nous exposerons un programme codé sous logiciel Pascal ainsi qu'une application des split graphes.

IV.2 Préliminaires et définitions

Dans cette section, nous présentons plusieurs définitions équivalentes d'un graphe bisplit tout en donnant des exemples d'illustration.

Définition IV.2.1. [4] Un graphe non orienté G = (V, E) est un graphe bisplit si son ensemble de sommets peut être partitionné en un ensemble stable S et un graphe biparti complet. Nous proposons un temps O $(|V||E|)$ algorithme de reconnaissance pour ces graphes et les caractériser en termes de sous-graphes induits interdits. Nous discutons aussi le problème de reconnaître si un graphe G a un ensemble stable S de telle sorte que les composantes connexes de G [V\S] soient plus d'un sous-graphe biparti complet.

Définition IV.2.2. [4] Un graphe non orienté G = (V, E) est un graphe bisplit si son ensemble de sommets peut-entre partitionné en trois ensembles stables X, Y et Z tel que $Y \cup Z$ induit un sous-graphe biparti complet une $(bi - clique)$ dans G. Une telle partition de V est dit appropriée. On note la classe de tous les graphes bisplit par ϕ. Dans le même sens, la classe des bisplit graphes ressemble à celle des splits graphes dont l'ensemble des sommets peut être partitionné en un ensemble stable et une clique.

Définition IV.2.3. [4] Si G est un graphe bisplit et abc est un **triangle** dans G, alors

(*i*) AB, AC, BC et O sont des ensembles stables, et $G[A]$, $G[B]$ et G $[C]$ sont des graphes bipartits.

(*ii*) Supposons que (X, Y, Z) est une partition correcte de G. Alors $O \subset X$. En outre si $a \in X, b \in Y, c \in Z$, Alors $BC \subset X, AC \subset Y, etAB \subset Z$.

Depuis les bisplit graphes sont transitivement orientables, un triangle T doit avoir une orientation transitive. On notera que dans toutes les orientations transitives d'un triangle, un sommet a 2 et degré entrant 0 degré sortant, un sommet a 0 et degré entrant 2 degré sortant, et le troisième sommet a 1 degré entrant 1 degré sortant.

Définition IV.2.4. [4] Soit G un bisplit graphe et soit $\vec{G}$ une orientation transitive de G. Supposons que abc est un triangle dans G telque $\overrightarrow{ab}, \overrightarrow{ca}, \overrightarrow{cb} \in \overrightarrow{G}$.

Donc G admet une partition propre (X, Y, Z) telle que :

$$\{a\} \cup BC \cup U \cup O \cup B^+ \cup C^- \subseteq X, \tag{IV.1}$$

$$\{b\} \cup AC \cup C^+ \subseteq Y, \tag{IV.2}$$

$$\{c\} \cup AB \cup B^- \subseteq Z, \tag{IV.3}$$

IV.3 Extensions et généralisations

Dans cette section, nous présentons quelques extensions et généralisations des bisplits graphes

IV.3.1 k-Bisplit graphes et les bisplit faibles

On dit qu'un graphe G est k-bisplit s'il a un ensemble stable X tel que le graphe induit $G[V \backslash X]$ a au plus k composantes connexes, et sont tous des bi-cliques. A noter que l'ensemble de tous les sommets isolés dans $G[V \backslash X]$ est considéré comme une seule bi-clique.

On dit également qu'un graphe G est un bisplit faible si il est k-bisplit pour un k Notez que les faibles bisplit graphes sont encore 3-colorables, mais aucun graphe de comparabilité plus (les cycles sans fil chordless de longueur impaire au moins cinq sont des graphes 2-bisplit), Contrairement aux bisplits graphes, il sera montré ci-dessous que la reconnaissance de la faiblesse des graphes bisplits est un problème NP-complet. De plus, nous allons montrer que la reconnaissance des graphes k-bisplit peut être faite en un temps polynomial pour chaque k fixe [4].

Pour les deux résultats nous avons besoin des variantes du problème de SATISFAISABILITÉ :

2SAT : Soit C un ensemble de clauses sur un ensemble de variables booléennes, dont chacun contient exactement deux littéraux. Y-t-il un C assignment satisfaisant C. Il est bien connu que 2SAT peut être résolu en un temps linéaire[20],[1].

$1 - IN - 3$ 3SAT : Soit C une clause de collection des plus d'un ensemble de variables booléennes, chacune qui contient exactement trois littéraux unnegated, à savoir, les variables. Y at-il une mission de vérité satisfaisant C de telle sorte que chaque article ait exactement une vraie variable Cette variante du problème est connue pour être NP-complet[11].

Théorème IV.3.1. *[4] Pour tout $k \geq 1$ fixé, k-Bisplit graphes est reconnaissable en un temps polynomial.*

Preuve. *[4] Pour développer une reconnaissance de temps polynômial des k-bisplit graphes, nous avons besoin d'introduire encore plus de terminologie. Par analogie avec les les graphes bisplit, nous appelons une partition $V = X \cup W$ bon si X est un ensemble stable de telle sorte que le sous-graphe H induite par W a au plus k connecté composants dont chacun est un bi-clique. Appelons un sous-ensemble maximal de sommets de H avec le même quartier (en H) une classe canonique. De toute évidence, toute la classe canonique est un ensemble stable, toutes deux classes canoniques sont disjoints et H a une partition unique, dans les classes canoniques. En choisissant un sommet arbitraire dans chaque classe canonique de H nous obtainaninduit sous-graphe qui est appelé le graphique caractéristique de H. En variante, le graphe caractéristique peut être obtenu en contractant chaque classe canonique en un seul sommet.*

Avec cette terminologie, nous pouvons dire qu'un graphe $G = (V, E)$ est k-bisplit si et seulement si G admet une partition $V = X \cup W$ dans un ensemble stable X et un graphe $H = G[W]$ de telle sorte que le graphique caractéristique de H a au plus sommets 2k dont chacun est de degré 1, à l'exception éventuellement, pour un seul sommet qui est isolé dans H. Si M est un sous-ensemble de W tel que $H[M]$ est le graphe caractéristique de H, nous dirons que G admet une partition correcte par rapport au sous-graphe induit par M.

Théorème IV.3.2. *[4] Reconnaissons que la faiblesse des Bisplit graphes est NP-complet, même pour la comparabilité 3-colorables graphes*

Preuve. *[4] De toute évidence, le problème appartient à NP. Pour prouver qu'il est NP-complexe, nous allons utiliser une réduction de $1-IN-3$ 3Sat sans littéraux négatifs. Soit $\zeta = C_1, C_2, ..., C_m$ un m clauses avec un ensemble de variables $U = v_1, v_2, ..., V_n$ tel que*

chaque clause C_i contient exactement trois variables, $C_i = \{c_{i1}, c_{i2}, c_{i3}\}$ où chaque literal c_{ij} $(1 \leq i \leq m, 1 \leq j \leq 3)$, est un v_k pour un certains k approprié. Nous construisons un graphe de comparabilité 3-colorable $G = G(C)$ tel que G est un bisplit faible si et seulement si $\zeta \in 1 - IN - 3$ 3Sat. Pour chaque v_k variables $\in U$, soit $G(k)$ un triangle avec un v_k sommet marqué. Pour chaque clause $Ci = \{c_{i1}, c_{i2}, c_{i3}\}$, soit $G(Ci)$ le shownin graphique.

IV.4 Reconnaissance des Bisplit graphes

Entrée : un graphe G=(V, E).

Sortie : "OUI" si $G \in \Phi$ "NON"sinon

1 : prendre un sommet arbitraire $v \in V$ et partitioner V en un sous-ensembles $V_0 = \{v\}, V_1, V_2, ..., V_p$,

où V_j désigne le sous-ensemble de sommets à une distance j de v

2 : Si pour chaque $j \in \{1, ..., p\}$,V_j est un ensemble stable, Alors Sortie Afficher "OUI" et Arrêter. G [est un Bipartite] ; autrement j est le plus grand indice pour lequel V_j contient une arete e

3 : Si les extrémités de e n'ont pas de voisin commun dans V_{j-1} Alors afficher "NON" et Arrêter [G contient un cycle induit de longueur impaire supérieur a trois 3] ; Autrement soit T un triangle formé par les extrémités de e conjointement avec un voisin commun dans V_{j-1}

4 : Effectuer la phase d'orientation de la procédure "transitive orientation" pour G. Si l'orientation du triangle T n'est pas transitive, alors Afficher "NON" et STOP [G n'est pas un graphique de comparabilité] ; autrement, on note par a un sommet de degré entrant 1 et un sommet de degré sortant 1 dans T. On note par b le sommet de degré entrant 2, et le sommet sortant 0 dans T Et on note par c le sommet de degré entrant 0 et le sommet sortant 2. Ainsi, T a une orientation $a \longrightarrow b, c \longrightarrow a, c \longrightarrow b$.

5 : Si $A \neq \varnothing$, Alors la Sortie "NON" et Arrêter[G n'est pas un graphe de comparabilité]. Si T a un 3-Sommet ou si l'un de ses ensembles O, AB, AC, BC est non stable, Alors "NON" a la Sortie et Arrêter[G n'est pas un bisplit graphe]. Si $G[B]$ ou $G[C]$ n'est pas un graphe bipartit, Alors "NON" a la Sortie et Arrêter[Gn'est pas un graphe bipartit]. Sinon determiner les ensembles $B^0, B^-, B^+, C^0, C^+, C^-$ donnés par la phase d'orientation. Soit
X_0 :=$\{a\} \cup BC \cup O \cup B^+ \cup C^-$,
$Y_O := \{b\} \cup AC \cup C^+$,
$Z_O := \{c\} \cup AB \cup B^-$.

6 :

(a) déterminer tous B^0 et C^0 sommets en conflit avec X_0, Y_0, Z_0 :

B_X^1 := les sommets de B^0 qui ont un voisin dans Z_0 Ou ceux qui n'ont pas de voisin dansY_0 ;
B_Z^1 := les sommets de B^0 qui ont un voisin dans X_0 ;
C_X^1 := les sommets de C^0 qui ont un voisin dansY_0 Ou ceux qui n'ont pas de voisin dansZ_0 ;
C_Y^1 := les sommets de C^0 qui ont un voisin dansX_0 ;
(b) Si $B_X^1 \cap B_Z^1 \neq \varnothing$ ou $C_X^1 \cap C_Y^1 \neq \varnothing$, Alors la Sortie "NON" et Arrêter [le sommet est a double conflit].
(c) $X_1 := X_0 \cup B_X^1 C_X^1$, $Y_1 := Y_0 \cup C_Y^1$, $Z_1 := Z_0 \cup B_Z^1$.
(d) $B^1 := B^0 \setminus (B_X^1 \cup B_Z^1), C^1 := C^O \setminus (C_X^1 \cup C_Y^1)$ Si $B^1 = B^0$ et $C^1 = C^0$, Alors ALLER en(8).

7 : $k := 1$;

répéter

$k := k + 1$

(a) [determiner un nouveau conflit des sommets pour le prochaine round]
B_X^K := les sommets de B^{k-1} qui n'ont pas de voisins dans C_Y^{k-1} [les sommets B_X^k doivent être attribuer a X]

B_Z^K := les sommets de B^{k-1} qui ont un voisin dans C_X^{k-1} [les sommets B_Z^k doivent être attribués a Z]

C_X^K := les sommets de C^{k-1} qui n'ont pas de voisins dans B_Z^{k-1} [les sommets C_X^k doivent être attribuer a X]

C_Y^K := les sommets de C^{k-1} qui ont un voisin dans B_X^{k-1} [les sommets C_Y^k doivent être attribuer a Y]

(b) Si $B_X^k \cap B_Z^k \neq \varnothing$ ou $C_X^k \cap C_Z^k \neq \varnothing$, Alors la Sortie "NON" et Arrêter [le sommet a un double conflit]

(c) $X_k := X_{k-1} \cup B_X^k \cup C_X^k$, $Y_k := Y_{k-1} \cup C_Y^k$, $Z_k := Z_{k-1} \cup B_Z^k$

(d) $B^k := B^{k-1} \setminus (B_X^k \cup B_Z^k)$, $C^k := C^{k-1} \setminus (C_X^k \cup C_Y^k)$, [Les sommets restants soient attribuer]

Jusqu'a $B_X^k \cup B_Z^k \cup C_X^k \cup C_Y^k = \varnothing$.

8 : $X := X_k \cup B^k, Y := Y_k \cup C^k, Z := Z_k$.

9 : If (X, Y, Z) est une partition correcte de G, Alors la Sortie "OUI", autrement "NON".

Chapitre V

Application des split graphes

V.1 Introduction

Dans ce chapitre, nous allons presenter notre programme qui représente un codage de l'algorithme de reconnaissance des split graphes.
Pour cela, nous avons opté pour l'un des langages de programmation les plus performants qui est Pascal.

V.2 Presentation du logiciel

Définition V.2.1. Le langage de programmation Pascal (dont le nom vient du mathématicien français **BLAISE PASCAL**) a été inventé par **NIKLAUS WIRTH** dans les années 1970. Il a été conçu pour servir à l'enseignement de la programmation de manière rigoureuse mais simple, en réaction à la complexité d'Algol 68. Le premier compilateur a été conçu sur un $CDC6400$[1],[10] .

Ce langage est l'un de ceux qui ont servi à enseigner la programmation structurée. Le goto ou saut n'importe où dans le programme (dit « branchement ») est fortement déconseillé dans ce langage, le programme est un assemblage de procédures et de fonctions, dans lesquelles on peut utiliser des blocs conditionnels (if, case) et répétitifs (while, for, repeat) ayant chacun une entrée et une sortie afin de faciliter les contrôles, ce qui aboutit à des mises au point rapides et sûres.

Le langage est de plus fortement et statiquément typé, c'est-à-dire que toutes les variables doivent avoir un type défini au moment de la compilation. En revanche son manque de souplesse pour gérer les passages du type caractère au type chaîne de caractères est l'un de ses points faibles.

Il a largement pénétré le monde de l'éducation et de la recherche (universités), puis dans une moindre mesure celui de l'industrie et de l'édition logicielle.

V.2.1 Example

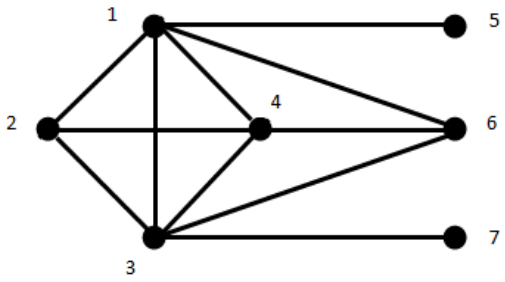


FIGURE V.1 – Le graphe au depart

• Comme première étape, on fait la saisie de la matrice d'adjacence de notre graphe, si il y'a une arête entre deux sommet on tape 1 sinon 0.
• La matrice d'adjacence du graphe précèdent est donee comme suit :

Les sommets	**1**	**2**	**3**	**4**	**5**	**6**	**7**
1	0	1	1	1	1	1	0
2	1	0	1	1	0	0	0
3	1	1	0	1	0	1	0
4	1	1	1	0	0	1	0
5	1	0	0	0	0	0	0
6	1	0	1	1	0	0	0
7	0	0	1	0	0	0	0

• La premiere partie de l'algorithme : est ce que le degré sequence du graphe est une **peo** ?

les degrés des sommets graphe sont donné dans le tableau suivant :

Sommet	1	2	3	4	5	6	7
Degré	4	3	4	4	1	3	1

utiliser l'algorithme **peo** pour savoir si l'ordre est un ordre d'éliminations simpliciale ou pas. notre programme affiche le message suivant **le graphe admet un ordre d'elimination simplicial donc il est triangulé**

si l'ordre n'est pas **peo** on suit l'algorithme pour savoir si le graphe contient un C_4, C_5 où $2K_2$, ou non.

• la deuxième étape consiste à trouver la clique max en utilisant l'algorithme "all maximal clique" qu'on a définit.

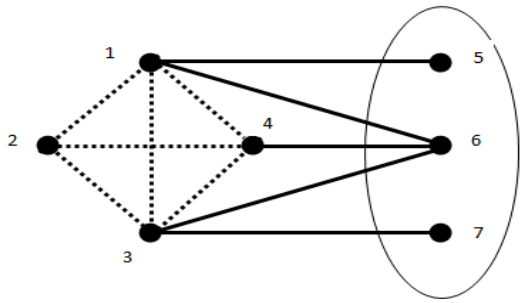


FIGURE V.2 – Le graphe partitionné

Les sommets qui constituent la clique max dans notre graphe sont les sommets : $1, 2, 3, 4$.

• la troisième étape consiste à trouver le stable max

Dans notre graphe, les sommets qui constituent le stable max sont :$5, 6, 7$.

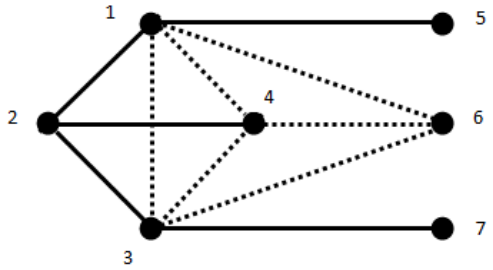


FIGURE V.3 – Le split graphe

• Comme dernier résultat, le programme affiche le texte suivant :
le graphe introduit est **un split graphe**, et il est composé du stable max suivant[5,6,7], et la clique max[1,2,3,4].
Les étapes ci-dessus, résument brièvement l'enchaînement de l'execution des instructions de l'algorithme que nous illustrerons dans la prochaine section.

V.3 Application des split graphes sur la phylogénétique

V.3.1 Introduction

Les split jouent un rôle important dans la théorie et le calcul de l'apparence (aspect) de la **phylogénétique**.
La phylogénétique Où phylogénie est l'étude des relations de parenté entre êtres vivants, individus, populations et espèces. Elle permet de reconstituer l'évolution des organismes vivants et représente les parentés par un arbre phylogénétique. Le nombre de noeuds entre les branches qui représentent autant d'ancêtres communs, indique le degré de parenté entre les individus[10],[7].

V.3.2 Les split graphes et la phylogénétique

Un split partage l'ensemble des **taxons** en deux. Ainsi, un ensemble de deux split incompatibles, qui se traduit sur le split graphe par l'apparition d'une boîte, permet de visualiser un conflit phylogénétique : Comme il y a un cycle, il existe plus d'un

chemin possible pour la phylogénie. Un split graphe peut donc servir comme consensus de plusieurs arbres phylogénétiques. Ce problème est connu en bio informatique comme celui du super arbre (ou encore du consensus network).
• **Un taxon** est un groupe à un niveau quelconque de catégories dans la classification hiérarchique des êtres vivants. Une feuille d'un arbre est appelée taxon.

Ainsi le splits graph est principalement un outil de visualisation. Encore faut il arriver à le dessiner de façon lisible. Le problème est donc de dessiner un graphe connaissant les longueurs des arêtes (elles sont en effet fixées par les données biologiques de départ), et sachant que certaines arêtes (celles de même coloration, c'est à dire correspondant à un même split) sont parallèles.
Le professeur Daniel Huson est à l'origine du logiciel splits tree.

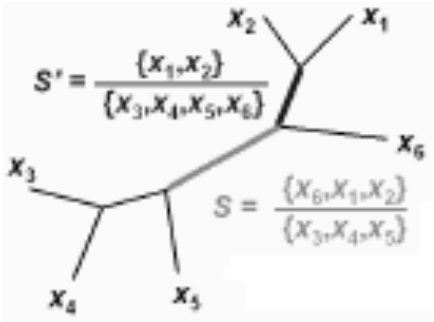


FIGURE V.4 – Un arbre phylogénétique et deux splits compatibles S et S'

Splits Tree propose d'utiliser la structure de splits graph pour en déduire un réseau réticuler, comme montré dans la précédente figure.
Le logiciel split tree est une innovation pour représenter non pas de simples arbres phylogénétiques comme c'est le cas habituellement, mais des réseaux phylogénétiques qui pouvaient représenter les recombinaisons.
Le logiciel Splits Tree permet de représenter et d'exploiter les conflits en construisant pour un ensemble de taxons leur réseau phylogénétique. Splits Tree propose aussi d'utiliser le splits graph, comme simple outil de visualisation de la robustesse d'une phylogénie[10],[7].

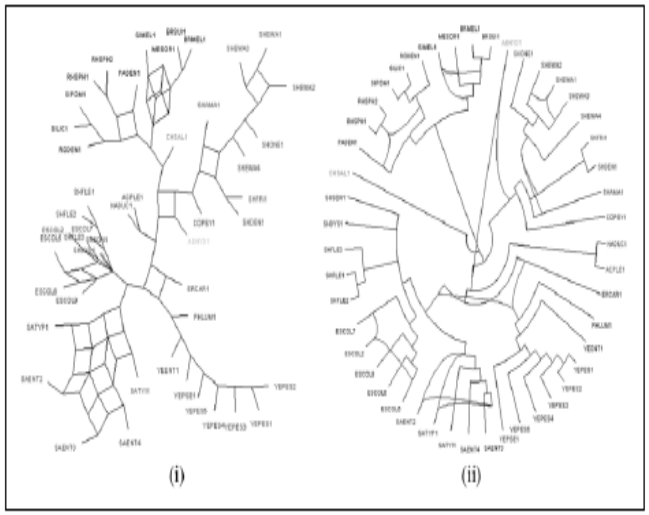


FIGURE V.5 – Reconstruction combinatoire de réseaux phylogénétiques

dans cette figure on a :

•**(i)**Un split network, exemple de réseau phylogénétique abstrait.

•**(ii)** Un réseau phylogénétique explicite construits respectivement par SplitsTree et Dendroscope, à partir de bipartitions et de clades apparaissant dans 20 de 16 arbres de gènes de la base HOGENOM concernant 46 espèces d'alpha et gamma-protéobactéries.

Conclusion

Ce mémoire fait l'objet d'une etude sur une classe de graphe, qui est les Split graphes et leurs propriétés, généralisations et extensions après avoir rappelé les notions de base de la théorie des graphes et de la théorie de la complexité, en particulier certaines classes de graphes classiques, nous avons mentionné quelques théorèmes et démonstrations sur les split graphes, mais aussi sur les bisplit graphes qui sont un cas particulier des split donc de la même famille.

L'objectif de cette étude est de faire connaître d'une manière générale le concept des split graphes avec leurs différentes variantes tout en mettant en évidence leurs principaux avantages notamment leurs applications en pratiques.
Ainsi, dans le cadre de ce travail, nous avons étudié les propriétés mathématiques des split graphes, tout en donnant des différentes définitions, extensions et des théorèmes avec leurs preuves.
En présentant une application des split graphes dans la phylogénétique nous espérons avoir donné un exemple concret et intéressant pour voir l'importance et mettre en évidence cette classe de graphes. Et avons programmé l'algorithme de reconnaissance des split graphes avec le langage de programmation PASCAL.

Ce travail nous a permis de dégager plusieurs perspectives qui nous semblent intéressantes à explorer dans les travaux futurs. Parmi ces perspectives, nous citons :

- Etude et analyse des split graphes dans l'analyse génétique des êtres humains.
- Synthèse et étude des applications des split graphes dans différents domaines.
- Etude des ensembles sécurisés dans les graphes avec le concepts des split.

Bibliographie

[1] M.F. Plass B. Aspvall and R.E. Tarjan. A linear-time algorithm for testing the truth of certain quantified boolean formulas, inform. *Process. Lett. 8* ***8****, pp. 121–123.*, (1979).

[2] C. Berge. Graphes gauthier-villars. *Discrete Mathematics*, (1983).

[3] A. Brandstadt and V.B. LE. Split-perfect graphs : Characterizations and algorithmic use. *SIAM J. DISCRETE MATH* ***17****, pp. 341–360.*, (2004).

[4] A. Brandstädt, P.L. Hammer, V.B. Le, and V.V. Lozin. Bisplit graphs. *J. Comb. Math. Combin. Comput.* ***48****, pp. 155-177*, (2004).

[5] S.A. Cook. The complexity of theorem-proving procedures. *In Proceedings of the third annual ACM symposium on Theory of computing,.* ***19****, pp. 151–158*, 1971.

[6] R. E. Tarjan D. J. Rose and G. Lueker. Algorithmic aspects of vertex elimination on graphs. *SIAM Journal on Computing,* ***15(3)****, pp. 146–160*, (1976).

[7] P. Darlu and P. Tassy. La reconstruction phylogénétique concepts et méthodes. *bhj*, (2004).

[8] S. Földes and P.L. Hammer. Split graphs. *Congr. Numer. 19* ***25****, pp. 311–315*, (1977).

[9] D. R. Fulkerson and O. A. Gross. Incidence matrices and interval graphs. *Pacific journal of mathematics,* ***15(3)****, pp. 835–855*, (1965).

[10] P. Gambette. Reconstruction combinatoire de réseayx phylogénétiques. *Biosystema Société Francaise de Systématique* ***28****, pp. 85-92*, (2011).

[11] M. R. Garey and D. S. Johnson. A guide to the theory of np-completeness. *Computers and Intractibility* ***309****, pp. 2140–2147*, (1979).

[12] F. Gavril. The intersection graphs of subtrees in trees are exactly the chordal graphs. *Journal of Combinatorial Theory, Series B,* ***16(1)****, pp. 47–56*, (1974).

[13] F. Gavril. Linear-time certifying algorithms for recognizing split graphs and related graph classes. *Journal of Combinatorial Theory, Series B,* ***16(1)****, pp. 47–56*, (2004).

[14] Martin Charles Golumbic. Algorithmic graph theory and perfect graphs. *Academic Press,*.

[15] Martin Charles Golumbic. *Algorithmic Graph Theory and Perfect Graphs*. Caesarea Rothschild Institute University o f Haifa Haifa, Israel, 2004.

[16] P. Heggernes and D.Kratsch. Linear-time certifiying algorithms for reconizing split graphs and related graph classes. *Department of informatics, University of Bergen. Norway.*

[17] E. Howorka. A characterization of distance-hereditary graphs. *The Quarterly Journal of Mathematics,* ***28(4)****, pp. 417*, (1977).

[18] W. Kocay and D.L. Kreher. Graphs, algoritms and optimization. *Discrte Mathematics and its Applications series editor KENNETH H. ROSEN.*

[19] D. Konig. Theorie der endlichen und unendlichen graphen. *Akademische Verlagsgesellschaft, Leipzig*, (1936).

[20] A. Itai S. Even and A. Shamir. On the complexity of timetable and multicommodity flow problems. *SIAM J. Comput. 5.* ***157****, pp. 691–703*, (1996).

[21] R. Shamir. Advanced topics in graph algorithms. *Technical report, Tel-Aviv University*, (1994).

[22] K. Simon. Algorithmen fur perfecte graphen. *Leinfaden und Monograhien der Informatik. B. G. Teubner Stuttgart*, (1992).

[23] R. E. Tarjan and M. Yannakakis. Simple linear-time algorithms to test chordality of graphs, test acyclicity of hypergraphs, and selectively reduce acyclic hypergraphs. *SIAM Journal on Computing,* ***13****, pp. 566*, (1984).

[24] Hélène TOPART. Étude d'une nouvelle classe de graphes les graphes hypotriangulés. 2011.

[25] M. Yannakakis. Node deletion problems on bipartite graphs. *SIAM J. Comput****10****, pp. 310-327.*, (1981).

RÉSUMÉ

Les split graphes ont reçu une grande attention de la part des chercheurs et ont été étudiées intensivement depuis des années dans la littérature. Ceci s'explique par leurs applications intéressantes dans divers domaines à savoir la médecines, la phylogénétique, les réseaux informatiques, le clustering des données, etc. Dans notre travail, nous avons effectué une étude sur les split graphes tout en donnant certaines de leurs extensions et généralisations, leurs propriétés mathématiques ainsi que leurs applications en pratique, mais aussi l'algorithme de reconnaissance associe que nous avons programmer sur un langage de programmation qui est le PASCAL.

Mots clés : Théorie des graphes, Split graphes, bisplit graphes, reconnaissance des split, les split et la phylogénétique.

ABSTRACT

The split graphs have received great attention from researchers and have been studied extensively for years in literature. This is due to their interesting applications in various fields namely medicines, phylogenetics, computer networks, clustering of data,etc. In our work, we conducted a study on split graphs while giving some of their extensions and generalizations, their mathematical properties and their applications in practice, but also the algorithm combines recognition that we have a language program programming is the PASCAL

Key words : Graphs theory, split graphs, bisplit graphs, reconnaissance of the split, the split and phylogénétique.