REPUBLIQUE ALGERIENNE DEMOCRATIQUE ET POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET
DE LA RECHERCHE SCIENTIFIQUE
UNIVERSITÉ DES SCIENCES ET DE LA TECHNOLOGIE HOUARI
BOUMEDIENNE
FACULTÉ DE MATHÉMATIQUES

THÈSE

Présentée pour l'obtention du diplôme de DOCTORAT EN SCIENCES

EN : MATHÉMATIQUES

Spécialité : Recherche Opérationnelle

Par : GUEHAM Assia

Sujet

TECHNIQUES DE COLORATION ET DE GÉNÉRATION DE COLONNES, APPLICATIONS Á LA CONFECTION D'HORAIRES

Soutenue publiquement le 13 Mai 2017 , devant le jury composé de :

M. BERRACHEDI Abdelhafid	Professeur à l'USTHB	Président
M. AIT HADDADÈNE Hacène	Professeur à l'USTHB	Directeur de thèse
M. NAGIH Anass	Professeur à l'Université de Lorraine, France	Co-directeur de thèse
M. AIDER Meziane	Professeur à l'USTHB	Examinateur
M. KACEM Imed	Professeur à l'Université de Lorraine, France	Examinateur
Mme. AFFIF-CHAOUCHE Fatima	M.C.A à L'USTHB	Examinatrice
M. SADI Bachir	M.C.A à l'UMMTO	Examinateur

REMERCIEMENTS

Je remercie tout d'abord le professeur Hacène AIT HADDADENE, mon directeur de thèse, Professeur à l'université des sciences et de la technologie Houari Boumediene (USTHB), et le professeur Anass NAGIH, mon co-directeur, professeur à l'université de Lorraine-France (Metz), de m'avoir encadrée, pour l'intéressant sujet qu'ils m'ont proposé, qu'ils trouvent ici mes vifs et profonds respects, considérations, gratitude ainsi que reconnaissance.

Que Abdelhafid BERRACHEDI, Professeur à l'université des sciences et de la technologie Houari Boumediene (USTHB), trouve ici mes plus grands remerciements de m'avoir fait l'honneur de présider le jury de cette thèse, je lui suis très reconnaissante.

Que le Professeur Meziane AIDER de l'USTHB et le Professeur Imed KACEM de l'université de Lorraine-France (Metz), trouvent mes plus vifs remerciements et ma sincère reconnaissance d'avoir accepté d'examiner ce travail et d'être dans le jury de cette thèse. Je remercie également les maîtres de conférences Fatima AFFIF CHAOUCHE de l'USTHB et Bachir SADI de l'université de Tizi ouzou (UMMTO)qui m'ont fait l'honneur de prendre part de à mon jury de thèse.

Que Monsieur Malek Masmoudi, Maître de conférences à l'université de Jean Monnet-Saint Etienne, et Maître de recherche au Laboratoire d'Analyse des Signaux et des Processus Industriels (LASPI), trouve mes sincères remerciements et reconnaissances.

Mes grands remerciements vont à l'encontre de ma famille pour leurs encouragements et leurs compréhensions.

Finalement, je remercie tous ceux qui m'ont aidée de prés ou de loin et qui ne trouvent pas ici leurs nom. Qu'ils me pardonnent et trouvent toute ma reconnaissance et gratitude.

RÉSUMÉ

Nous nous sommes intéressés dans le cadre de cette thèse de doctorat à deux problèmes fondamentaux de la théorie des graphes, à savoir, le problème de la coloration des graphes et celui de la clique maximum. Au delà du large éventail d'applications pratiques qu'ils permettent de modéliser, leur complexité combinatoire rend difficile leur résolution exacte dès lors que la taille des problèmes à traiter est conséquente. Dans ce cas, le recours à des méthodes de résolution approchée s'avère une alternative nécessaire. Ainsi, nous avons conçu, dans un premier temps, un pré-traitement basé sur un algorithme efficace d'orientation des arêtes d'un graphe et un schéma d'ordre d'étiquetage des sommets d'un graphe. La méthode ainsi développée, s'est avérée efficace pour des graphes appartenant à certaines classes ou ayant certaines propriétés d'une part, et d'autre part pour des classes de graphes parfaits telles que les classes des graphes bipartis, $1-$scindé, les graphes parfaits à voisinages scindés et les graphes parfaits $2-$scindés possédant des caractéristiques bien définies. Par ailleurs, nous avons proposé une nouvelle approche de contraction basée sur la méthode d'orientation et le schéma d'ordre d'étiquetage. L'approche développée consiste à déterminer un encadrement de la taille de la clique. La qualité du minorant a été évaluée en utilisant des instances de la bibliothèque DIMACS. Cette approximation a permis de développer un algorithme de résolution exacte de type séparation et évaluation. En se basant sur l'algorithme d'orientation, nous avons proposé un majorant du nombre chromatique et implémenté une nouvelle méthode de coloration polynomial que nous avons nommée, la coloration par blocs. Cette dernière méthode de coloration nous a permis, de plus, d'améliorer le majorant précédemment proposé. Nous avons proposé aussi une comparaison théorique et empirique de notre majorant avec celui de Reed pour la classe des graphes sans triangle. Grâce à une méthode hybridant la génération de colonnes et la coloration par blocs, nous avons réussi à améliorer la qualité des solutions fournies par la coloration par blocs seule. Comme application, nous avons traité avec succès le problème d'emploi du temps pour la construction d'équipes d'infirmiers dans le service chirurgie de la clinique tunisienne "Soukra".

Mots clés : Schéma d'ordre d'étiquetage, Orientation des graphes, Problème de la clique maximum, Nombre de clique, Graphe scindé, Coloration des graphes, Nombre chromatique, Problème de la construction d'équipes, la construction d'équipes dans le domaine de la santé.

ABSTRACT

In this stady we aim to solve two fundamental problems of graph theory, namely the graph coloring problem and the maximum clique problem. Beyond, the wide range of practical applications that they allow to create a model, their combinatorial complexity makes difficult to find their exact resolution. Since the size of the problems to be treated is consequent. In this case, the use of approaches methods of resolution proves to be a necessary alternative. Thus, we conceive, in a first time, a pre-processing based on an efficient orientation algorithm of edges's graph and a labeling order scheme of graph vertices. The method thus developed, proves to be efficient for certain classes of graphs. On the one hand, graphs having certain properties and on the other hand, the perfect graph classes such as bipartite graph classes, $1-$ split, Perfect split neighborhood graphs and perfect $2-$ split graphs which has well-defined characteristics. In addition, we propose a new contraction approach based on the orientation method and the labeling order scheme. The developed approach consists to determine a framework for the size of the clique. The quality of the lower bound was evaluated by using instances of the DIMACS library. Also, this approach allowes to develope an exact resolution algorithm of branch and bound type. Depending on the orientation algorithm, we propose an upper bound on the chromatic number and we implement a new polynomial coloring method, called the block coloration method. This last method allowed us, moreover, to improve the previously proposed upper bound. We also propose a theoretical and empirical comparison of our upper bound with that of Reed conjecture for class of triangle-free graphs. Due to a hybridizing method of column generation and block coloration algorithm, we have been able to improve the quality of solutions provided by block coloration alone. As an application, we successfully deal with the timetabling problem for the team building of nurses in the surgery service of a private hospital (Clinic of Soukra in Tunis, Tunisia).

Keywords : Labeling order scheme, Orientation of graphs, the maximum clique problem, the clique number, Split graph, Graph coloring problem, Chromatic number, Team building problem, Team building problem in Healthcare domain.

Table des matières

Table des figures

Liste des tableaux

"Ne vous souciez pas d'être meilleur que vos contemporains ou vos prédécesseurs.
Essayez seulement d'être meilleur que vous-même."
William FAULKNER

INTRODUCTION GÉNÉRALE

L'homme est souvent confronté à des situations auxquelles il doit prendre des décisions instantanées, tout en essayant de s'organiser sur tous les plans. Néanmoins, il arrive qu'une telle ou telle décision puisse engendrer de grands profits ou d'énormes pertes. Ainsi, une étude scientifique devient par ce fait plus qu'indispensable, c'est justement l'objet de la théorie de la décision.

Choisir est une action centrale dans la théorie de la décision. Divers problèmes se présentent sous cette optique de choix. En effet, cette dernière devient le centre de l'étude dans les problèmes d'optimisation. Il est nécessaire de savoir que la solution optimale des problèmes liés à cette classe est rarement atteinte, voire quasi- impossible à atteindre. C'est justement dans la classe de ces problèmes, les plus difficiles, que se trouvent les problèmes traités dans cette thèse. Ceux-ci sont connus plus pour être des problèmes de choix. Ainsi, pour résoudre le problème de la confection d'horaire et le problème de la construction d'équipes, il serait heuristiquement bénéfique d'avoir recours à la théorie de la décision.

Avant, l'homme utilisait ses doigts comme moyen et outil de calcul, cela a bien évolué. S'inspirant de la nature, son outil devient les couleurs, ce qui a donné naissance à une nouvelle classe des problèmes, dont le principal est celui de la coloration des graphes. De nombreux problèmes peuvent être modélisés sous forme de problème de coloration ; il s'agit des problèmes se ramenant à la recherche d'une partition d'un ensemble d'objets en sous ensembles ne comportant que des éléments deux à deux compatibles. L'utilité pratique de ce problème de la coloration, ne cesse d'augmenter et d'avoir des extensions pratiques ; notamment les problèmes industriels et autres. Ainsi, la résolution des problèmes approchés par le biais de la coloration des sommets d'un graphe passe par l'étude des sous ensembles remarquables de sommets, représentant les points ou les arêtes, représentant et modélisant les liaisons entre les sommets du graphe. C'est dans cette classe de problèmes que s'inscrit la thématique de notre thèse « Techniques de coloration et de génération de colonnes, applications à la confection d'horaires ». En outre, nous étudions le problème de la confection d'horaires à travers la coloration des graphes afin d'améliorer la qualité des horaires pour les individus et d'optimiser le nombre de services offerts, notamment le problème de la construction d'équipes.

Notre recherche porte essentiellement sur deux problèmes fondamentaux de la théorie des graphes, à savoir, le problème de la coloration des graphes et celui de la clique maximum. Au delà du large éventail d'applications pratiques qu'ils permettent de modéliser, leur complexité combinatoire rend difficile leur résolution exacte dès lors que la taille des problèmes à traiter est conséquente. Dans ce cas, le recours à des méthodes de résolution approchée s'avère une alternative indispensable. Tout d'abord, nous nous intéressons à l'étude de la problématique d'ordre et à l'étude de l'orientation d'un graphe. Cette dernière consiste à attribuer une orientation à chaque arête du dit-graphe. Les propriétés qui en découlent ont été utilisées par plusieurs chercheurs afin de résoudre certains problèmes de littérature [24, 37–39, 41, 42, 66, 86, 98] et [101]. Dans notre cas, nous exploitons les caractéristiques du graphe ainsi orienté afin d'établir des résultats théoriques sur certaines classes de graphes concernant le problème de la clique maximum et celui de la coloration.

En effet, dans un premier temps, un pré-traitement basé sur un algorithme efficace d'orientation des arêtes d'un graphe et un schéma d'ordre d'étiquetage des sommets d'un graphe a été conçu. Cet algorithme permet de caractériser en temps polynomial la clique maximum et sa taille ω pour les graphes vérifiant une propriété d'ordre d'étiquetage. L'algorithme conçu consiste à parcourir un graphe en partant à chaque fois d'un sommet de degré maximum non encore traité et d'orienter les arêtes qui lui sont incidentes à l'extérieur. Une fois que tous les sommets sont traités, nous réitérons le même processus avec le sous graphe partiel engendré par les arêtes non orientées, tandis que le schéma d'ordre d'étiquetage consiste à définir une application I sur les sommets du graphe ainsi orienté par rapport à ses incidences. Notre méthode s'avère efficace pour les classes suivantes des graphes parfaits : bipartis, 1-scindé, à voisinage scindés et 2-scindés possédant des caractéristiques bien définies [51].

Nous étudions aussi les deux limites $\overline{\omega}$ et $\underline{\omega}$ qui correspondent au majorant et au minorant de ω tout en intégrant une nouvelle méthode de contraction qui fait l'objet de chapitre 3. Chacune de ces deux limites est une fonction de l'application I, où I représente le nombre d'arcs entrants. La qualité du minorant a été évaluée en utilisant des instances de la bibliothèque DIMACS. En fait, cette caractérisation découle des résultats obtenus dans [9]. Pour trouver la clique maximum et sa cardinalité, nous optons pour le développement d'une nouvelle méthode énumérative partielle par juxtaposition de l'algorithme d'orientation. Cette dernière consiste à énumérer certaines cliques maximales jusqu'à une itération donnée afin d'en tirer la clique maximum.

En se basant sur l'algorithme d'orientation [51], nous proposons un majorant du nombre chromatique et nous implémentons une nouvelle méthode de coloration polynomial que nous avons nommée, la coloration par blocs. Cette dernière méthode de coloration nous a permis, de plus, d'améliorer le majorant précédemment donné. Aussi, nous proposons une comparaison théorique et empirique de notre majorant avec celui de Reed pour la classe des graphes sans triangle. Grâce à une méthode hybridant la génération de colonnes et la coloration par blocs, nous avons réussi à améliorer la qualité des solutions fournies par la coloration par blocs seule. En effet, notre approche est générique et elle peut être adaptée à la résolution d'une grande partie des problèmes d'optimisation combinatoire

ayant un intérêt pratique, à savoir le problème de la construction d'équipes et ceci tout en assimilant les contraintes de problème de la construction d'équipes aux contraintes d'adjacence dans la coloration. Ce qui nous a conduit à proposer une version modifiée de notre algorithme de coloration par blocs qui est aussi polynomial et que nous avons baptisé la coloration par blocs de cardinalité limitée. Ce problème de la construction d'équipes est connu comme étant un problème difficile [4] et a suscité un très grand intérêt. Il a été appliqué avec succès dans le domaine de la santé [1], [4], le sport [34], l'éducation [92], militaire [30], etc. Comme application, nous avons présenté une illustration de problème de la construction d'équipes dans le domaine hospitalier [49] et [50].

Nous avons structuré notre thèse en cinq chapitres.

Le premier chapitre est consacré essentiellement aux définitions préliminaires et aux notions les plus utilisées. Il comprend les rappels qui seront nécessaires pour la suite du document, les notions fondamentales en optimisation combinatoire et en théorie de graphe. Nous y trouvons en particulier les principaux concepts et résultats, concernant le problème de la clique maximum ainsi que le problème de la coloration.

Le deuxième chapitre, est dédié à l'approche heuristique de résolution du problème de la clique maximum, au résultat fondamental définissant l'ordre d'étiquetage proposé ainsi qu'à l'algorithme d'orientation des arêtes et à la caractérisation du problème de la clique maximum dans des classes de graphes parfaits et d'autres graphes qui vérifient la propriété d'ordre d'étiquetage local. Nous y présentons aussi une nouvelle méthode heuristique pour la détermination de ω (le nombre de clique d'un graphe) ainsi la clique correspondante. La méthode ainsi développée, s'est avérée efficace pour certaines classes de graphes. D'une part, les graphes ayant certaines propriétés et d'autre part, les classes de graphes parfaits telles que les classes des graphes bipartis, $1-$scindé. D'autre part, les graphes parfaits à voisinage scindés et les graphes parfaits $2-$scindés possédant des caractéristiques bien définies.

Dans le troisième chapitre, une nouvelle méthode de contraction et un majorant $\overline{\omega}$ de ω ont été proposées tout en se basant sur l'algorithme d'orientation et le schéma d'ordre d'étiquetage développés au chapitre précédent. Utilisant cette méthode de contraction, une approche polynomiale d'encadrement pour la caractérisation du minorant, $\underline{\omega}$ du nombre chromatique *omega* a été proposée. Ceci nous a permis d'exhiber un algorithme exact pour le problème de la clique maximum, ce dernier consiste à énumérer certaines cliques maximales jusqu'à une itération donnée afin d'en tirer la clique maximum. Afin d'évaluer et de valider la qualité du minorant, nous avons utilisé des instances tests classiques de la bibliothèque DIMACS.

Nous avons introduit dans le quatrième chapitre une nouvelle méthode de coloration des sommets d'un graphe, qui est polynomial et que nous avons nommée la coloration par blocs, tout en intégrant notre méthode d'orientation. Cette méthode a permis de proposer un majorant pour le nombre chromatique. Par ailleurs, la qualité de ce majorant a été évaluée tout en établissant une comparaison théorique et empirique avec la borne de Reed pour la classe des graphes sans triangle. Grâce à une méthode hybridant la génération de colonnes et la coloration par blocs, nous avons réussi à améliorer la qualité des solutions

fournies par la coloration par blocs seule.

Enfin, dans le cinquième chapitre, nous avons illustré ce travail par des exemples pratiques tels que ; le problème de confection d'horaires dans le domaine de la pédagogie et la construction d'équipes en milieu hospitalier. Comme application, nous avons traité avec succès le problème d'emploi du temps pour la construction d'équipes d'infirmiers dans le service de chirurgie de la Clinique de la Soukra à Tunis (Tunisie).

Une conclusion finale, nous a permis de synthétiser les principaux résultats de cette recherche et les perspectives potentielles de ce travail.

Chapitre 1

Définitions et généralités

Nous présentons dans ce chapitre des rappels sur des concepts généraux qui sont abordés tout au long de ce document. Tout d'abord, nous commençons par un bref rappel sur la théorie de la complexité, les concepts de base de la théorie des graphes et ceux des graphes parfaits. Les définitions que nous présentons sont essentiellement inspirées de [12] et [60].

1.1 Rappels sur l'Optimisation Combinatoire

Un problème est dit combinatoire lorsqu'il comprend un grand nombre de solutions admissibles parmi lesquelles on cherche une ou plusieurs solutions qui vérifient une propriété donnée. En particulier, l'optimisation combinatoire s'intéresse à la détermination d'une solution optimale, c'est-à-dire la meilleure au sens d'un critère d'évaluation.
Les problèmes de l'optimisation combinatoire (O.C) se caractérisent par une formulation facile et souvent par une résolution très ardue.

Nous donnerons dans ce qui suit quelques définitions des notions fondamentales ayant trait à l'optimisation combinatoire.

Définition 1.1.1 *Une solution est un ensemble de valeurs données aux variables définissant le problème.*

Définition 1.1.2 *Une solution admissible (réalisable) est une solution d'un problème ne violant aucune contrainte de celui-ci.*

Définition 1.1.3 *Une solution optimale est une solution réalisable qui optimise une fonction objectif.*

Définition 1.1.4 *Une solution approchée est une solution réalisable qui ne garantit pas l'optimalité.*

Définition 1.1.5 *Étant donnés un ensemble fini de configurations S et une application f, un problème d'O.C (pour un problème de minimisation ou maximisation) consiste en la recherche d'un élément s^* ou plusieurs tel que :*

$$s^* = Arg\min_{s \in S} f(s)$$

$$c.à.d$$

$$f(s^*) = \min_{s \in S}(f(s)).$$

Il est très difficile d'énumérer et d'examiner exhaustivement toutes les solutions d'un problème d'optimisation combinatoire. Notamment, lorsque l'on souhaite le résoudre de manière exacte. Ainsi, s'impose la nécessité de distinguer entre les algorithmes selon leurs performances à nous procurer des solutions. L'analyse de la complexité des algorithmes permet de classer, comparer, évaluer, voire retenir les meilleurs algorithmes selon des critères de performance bien précis.

1.1.1 Sur l'analyse de la complexité

Avant d'aborder l'analyse de la complexité des algorithmes, il y a lieu de donner quelques rappels et définitions de base indispensables pour la compréhension de cette théorie.

1.1.1.1 Définitions de base

Les définitions données ci-dessous sont tirées des livres [12].

Définition 1.1.6 *On appelle instance I d'un problème (P) une situation particulière où est défini l'ensemble des paramètres fixés à des valeurs données.*

Définition 1.1.7 *Un algorithme de résolution d'un problème donné est une séquence finie d'opérations élémentaires, complémentaires, logiques et chronologiques.*

Définition 1.1.8 *Un algorithme est dit efficace s'il résout n'importe quelle instance I d'un problème (P) en un temps de calcul polynomial en la taille de ses données.*

Définition 1.1.9 *Un problème de décision est un problème posé sur un ensemble de concepts dont la solution est une réponse par* oui *ou par* non.

Définition 1.1.10 *La théorie de la complexité permet d'étudier d'une manière formelle la difficulté des problèmes pratiques. La complexité d'un algorithme se mesure par son coût en temps de calcul en fonction de la taille de l'instance du problème considéré, c'est à dire le nombre d'opérations élémentaires engendrées par l'exécution de l'algorithme, ce nombre étant exprimé en fonction de la taille de l'instance.*

Classe P

Ce sont les problèmes de décision pour lesquels la réponse par « oui »ou par « non »peut être trouvée en un temps polynomial, le sigle P signifie « Determinist Polynomial time, DP ».

La classe P correspond aux problèmes faciles dont la résolution peut être faite par des algorithmes efficaces.

Classe NP

Ce sont les problèmes de décision pour lesquels il est possible de vérifier une solution donnée en un temps polynomial. Le sigle NP provient de « Non Determinist Polynomial time, NDP ».

Il est important de remarquer à ce stade que la classe P est incluse dans la classe NP, J. Edmonds a conjecturé que $P \neq NP$ [33].

Dans la classe NP, une sous classe composée de problèmes ayant certaines propriétés et liés entre eux se caractérisent par une résolution difficile, il s'agit de la classe des problèmes NP-Complets.

Classe NP-complet

Un problème NP-complet possède la propriété que tout problème dans NP peut être transformé en celui-ci en temps polynomial.

Classe NP-difficile

Ce sont les problèmes d'optimisation dont le problème de décision approprié est NP-complet. Il est conjecturé qu'il n'existe pas d'algorithme polynomial pour les problèmes d'optimisation NP-difficiles.

Les méthodes de résolution exactes ou approchées proposées au cours de ces dernières années sont nombreuses, elles sont le reflet de l'éventail de méthodes dont on dispose pour traiter les problèmes d'optimisation combinatoire.

1.1.2 La programmation linéaire

C'est une partie de la programmation mathématique [1], de recherche d'extremum liée d'une fonction linéaire, dite fonction objectif, sur un ensemble défini par des relations linéaires de type équation ou inéquation (contraintes linéaires). La procédure suivie à ce niveau s'articule autour de deux étapes essentielles, la première consiste en la formulation mathématique du problème. Cette modélisation consiste en la détermination des variables, de leur nature, et des contraintes du problème sous formes d'équations et/ou d'inéquations linéaires ; le modèle obtenu est dit programme linéaire. Dans la seconde étape, il s'agit d'appliquer la méthode de résolution du modèle.

On envisage un programme linéaire en maximisation :

$$(PL) \qquad \begin{cases} \text{maximiser} & c^t x \\ \text{sous les contraintes} & x \in D(PL) \end{cases}$$

dans lequel c (les données de l'objectif) est un vecteur de R^n et le domaine des solutions réalisables $D(PL)$ est défini comme suit :

$$D(PL) = \{x \in \mathbf{R}^n \setminus Ax \,\mathcal{R}\, b; x \geq 0\}$$

Plus généralement, $x_j \geq 0$, $x_j \leq 0$, x_j quelconque.

– où

A est une matrice de $R^{m \times n}$ c'est-à-dire à m lignes et à n colonnes ;

b est un vecteur de R^m ;

$\mathcal{R}$ est un opérateur de $\mathbf{R}^m$ qui est composé d'au moins d'une des trois relations d'ordre suivantes : $\leq$, $\geq$ et $=$, sur les réels.

$D(PL)$ est un polyèdre de R^n qui peut être borné, non borné ou vide.

Un vecteur $\bar{x} \in R^n$ est dit solution optimale de (PL) si

$$c^t \bar{x} \geq c^t x \qquad \forall x \in D(PL).$$

1. Est une formulation mathématique d'un problème d'optimisation pour décrire ce que l'on cherche à optimiser ainsi que les relations entre les différentes données du problème.

La programmation linéaire (PL) est placée dans les classes P des problèmes polynomiaux. Le premier algorithme polynomial pour la programmation linéaire est la *méthode des ellipsoïdes* développée par Khachiyan en 1979 [61]. Malgré sa complexité polynomiale, cette méthode converge très lentement en pratique. De plus, elle est particulièrement sensible aux erreurs d'arrondis et requiert la mise en mémoire et la mise à jour à chaque itération d'une matrice $n \times n$ dense.

i- Méthodes de points intérieurs

En 1984, Karmarkar a proposé la méthode des points intérieurs qui sont connues comme étant des méthodes polynomiales et efficaces. La méthode de Karmarkar est une méthode itérative qui construit une suite de points convergeants vers une solution optimale, si elle existe.

Le premier algorithme décrit pour la résolution des programmes linéaires est la méthode du simplexe qui reste à ce jour la plus utilisée. Malgré son efficacité en pratique, l'algorithme s'avère non polynomial, pour certains cas. Cependant, la programmation linéaire est considérée comme un problème de la classe P, grâce à l'algorithme de Kachyian [61].

ii- Algorithme du simplexe

L'algorithme du simplexe permet de déterminer une solution optimale d'un programme linéaire donné sous sa forme standard $\min\{c^t x : Ax = b, x \geq 0\}$.

– Deux types d'algorithmes du simplexe sont présents, primal et dual, pour aboutir à un point extrême optimal (solution de base [2] réalisable et optimale) :

(i) d'une part, l'algorithme primal construit un chemin composé de points extrêmes (solutions de base réalisables) ;

(ii) d'autre part, l'algorithme dual qui part d'une solution de base optimale pour construire un chemin composé de points optimaux (solutions de base optimales).

La forme standard d'un programme linéaire est la suivante :

$$(PL) \quad \begin{cases} \text{maximiser} & c^t x & & \\ \text{sous les contraintes} & Ax & = & b \\ & x & \geq & 0. \end{cases}$$

2. Une base est une sous-matrice carrée régulière, c'est à dire inversible de dimension $m \times m$, extraite de la matrice A. La solution de base est dite réalisable si toutes ses composantes sont positives ou nulles. Une base correspondant à une solution de base réalisable est appelée base réalisable.

iii- Génération de colonnes

La génération de colonnes est une méthode privilégiée pour résoudre efficacement des programmes linéaires de grande taille. En effet, ces programmes linéaires ont trop de variables ou colonnes pour qu'on puisse les représenter toutes de manière explicite. A l'optimum, la plupart des variables sont hors base, qui sont toutes nulles, ainsi un seul sous ensemble de variables doit être pris en compte pour résoudre le problème. Elle repose sur une utilisation particulière de la méthode du simplexe sur un problème décomposé et restreint. Un problème auxiliaire permet de générer les variables non prises en compte initialement.

1.1.3 La programmation linéaire en nombres entiers

La forme générale d'un programme linéaire de maximisation en nombres entiers est la suivante :

$$(P) \qquad \begin{cases} \text{maximiser} & c^t x \\ \text{sous les contraintes} & x \in D(P) \end{cases}$$

dans laquelle l'objectif est un vecteur de $\mathbf{R}^n$ et le domaine des solutions réalisables $D(P)$ est défini comme suit :

$$D(P) = \{x \in X \setminus Ax \leq b\}$$

– où
$X = \{x \in \mathbf{R}^n \setminus x \geq 0 \text{ et } x_E \text{ entier}\}$ où E est un sous-ensemble non vide de $\{1, 2, ..., n\}$.

Si E est strictement inclus dans $\{1, 2, ..., n\}$ le problème (P) est dit à variables mixtes. Par contre, lorsque E est identique à $\{1, 2, ..., n\}$ le problème (P) est dit à variables entières ou en nombres entiers (PLNE), et dans le cas particulier où chaque x_j est dans $\{0, 1\}$, (P) est dit à variables 0-1 (PL 0-1).

i- Résolution exacte

Les méthodes exactes sont ainsi qualifiées en raison des solutions exactes qu'elles procurent. Elles nous permettent d'obtenir une solution optimale des instances des problèmes résolus. Leur principe général consiste en une énumération intelligente, le plus efficacement possible, de l'espace des solutions pour en extraire une solution optimale.

i.1- La méthode des plans sécants

C'est une approche qui se base sur la recherche d'une restructuration de l'espace des solutions réalisables et ce en imposant quelques contraintes supplémentaires à l'espace originel. L'idée générale est que ces contraintes additionnelles coupent des portions

de l'espace des solutions sans altérer ni exclure aucun point de l'espace des solutions réalisables.

i.2- Méthodes de coupes

Pour résoudre le $PLNE$ suivant :

$$(P) \quad \begin{cases} \text{maximiser} & c^t x & & \\ \text{sous les contraintes} & Ax & = & b \\ & x & \geq & 0 \\ & x \text{ entier} & & \end{cases}$$

On suppose le programme linéaire associé résolu :

$$(PL) \quad \begin{cases} \text{maximiser} & c^t x & & \\ \text{sous les contraintes} & Ax & = & b \\ & x & \geq & 0 \end{cases}$$

En notant I une base optimale de (PL), on rappelle qu'une solution optimale x^{PL} de (PL) est définie comme suit :

$$x^{PL} = \begin{cases} \bar{b}_i & \text{pour tout } i \in I \\ \\ 0 & \text{pour tout } i \in \bar{I} \end{cases}$$

Si $\bar{b}_i$ est un entier (nécessairement positif) pour tout $i \in I$, alors le problème (P) est résolu :

$$x^P = x^{PL} \text{ et } v(P) = v(PL).$$

On se place ensuite dans le cas où les composantes du vecteur $\bar{b}$ ne sont pas toutes entières ; cela implique l'existence d'au moins un indice i de I tel que $\bar{b}_i$ n'est pas dans N. Le but des contraintes valides est de couper le domaine de (PL) par une contrainte d'inégalité qui sépare x^{PL} et $D(P^*) = Conv$ [3]$(D(P))$.

La coupe proposée par Dantzig est la suivante :

$$\sum_{j \in \bar{I}} x_j \geq 1.$$

La coupe de Gomory exploite les relations exprimant les variables de base en fonction des variables hors-base :

$$x_i + \sum_{j \in \bar{I}} \bar{a}_{ij} x_j = \bar{b}_i$$

Pour un réel h non nul donné, sachant que toutes les variables doivent être positives dans toute solution réalisable de (P), on obtient

$$\lfloor h \rfloor \, [4] x_i + \sum_{j \in \bar{I}} \lfloor h \bar{a}_{ij} \rfloor x_j \leq h \bar{b}_i$$

3. Enveloppe convexe d'un ensemble discret $X = \{x^1, x^2, ..., x^p\}$ de points de R^n : $Conv(X) = \left\{ x \in R^n | x = \sum_{j=1}^p \lambda_j x^j, \ \sum_{j=1}^p \lambda_j = 1, \ \lambda_j \in [0,1], \ \forall j \in \{1, ..., p\} \right\}$

4. Représente la partie entière inférieure de h.

i.3- La programmation dynamique

L'origine des principales méthodes de la programmation dynamique revient à Richard BELLMAN. Ces méthodes sont utilisées pour résoudre des problèmes, dont la solution optimale s'obtient successivement en démarrant d'une solution réalisable et ceci en se basant sur le principe d'optimalité. Ainsi, la programmation dynamique est une méthode séquentielle, elle permet d'optimiser une fonction séparable de plusieurs variables, liées par des contraintes sous formes d'équations ou d'inéquations. La décomposition et la séquentialité sont les principes de base de la méthode de la programmation dynamique. Il est nécessaire, pour pouvoir l'appliquer que le problème puisse être décomposé en étapes, ce qui induit un gain de temps appréciable.

Cette approche a permis l'élaboration d'algorithmes de résolution pour un grand nombre de problèmes combinatoires.

i.4- Techniques de Séparation et Evaluation (SPE) "Branch and Bound"

Les techniques de SEP ont été les premières techniques utilisées lors de l'exploration d'arbres de possibilités trop complexes pour être parcourus intégralement, sur le domaine S des solutions d'un problème d'optimisation combinatoire. Ce domaine est décomposé progressivement sous forme d'arborescence dont la racine serait l'ensemble S.

La plupart des problèmes auxquels nous nous intéressons, sont dans la classe NP-Complet, c'est pourquoi on privilégie des heuristiques, et encore des métaheuristiques et ceci en raison de l'explosion combinatoire. Celles-ci permettent d'obtenir des solutions de bonne qualité, bien qu'elles ne soient pas nécessairement optimales. Lors de la résolution de certains problèmes, le nombre de solutions réalisables possibles peut évoluer rapidement avec leur taille, ce qui rend quasi-impossible l'exploration de l'ensemble de ces solutions en vue d'une optimisation, cette situation est qualifiée d'explosion combinatoire, en dépit de l'essor des moyens de calcul.

ii- Résolution approchée

Du fait des résultats de la NP-Complétude de certains problèmes de l'optimisation combinatoire, ceux qui jouissent d'un grand intérêt à la fois théorique et pratique, il est peu probable, d'envisager leur résolution à l'aide de méthodes exactes et ce en raison de leur temps d'exécution qui évolue exponentiellement avec la taille des instances du problème en question. La quasi-possibilité de trouver des algorithmes efficaces au sens complexité du terme, étant écartée, les chercheurs ont orienté leurs efforts vers l'élaboration des méthodes heuristiques, qui sont capables de fournir de « bonnes » solutions réalisables en un temps raisonnable, ainsi la raison d'être des heuristiques.

Définitions

Une méthode approchée est une méthode de recherche des solutions de bonnes qualités, quasi-optimales, en un temps de calcul raisonnable, sans toutefois pouvoir en garantir ni l'optimalité, ni même (dans de nombreux cas) l'éloignement de la solution par rapport à la plus proche solution réalisable ou optimale.
Dans les méthodes approchées nous distinguons deux types :

- Les méthodes approchées permettant de fournir une solution réalisable à savoir les heuristiques et les métaheuristiques dont la principale différence réside dans le mode opératoire.
- Certaines méthodes approchées permettant de fournir une solution optimale telles que les méthodes de relaxation.

Une heuristique est une méthode qui fournie une solution réalisable sans garantir l'optimalité. Elle est dédiée spécifiquement à la résolution d'un problème donné. Elle tente d'exploiter au mieux sa structure par des critères de décision déduits de la connaissance du problème à résoudre et dont la solution optimale n'est pas garantie.

Une métaheuristique est une méthode, ou plus précisément, un canevas de méthodes, pour résoudre de manière approchée tous les problèmes dont la solution optimale n'est pas garantie. Cependant, ces méthodes ne dépendent pas de type du problème que nous tentons de résoudre.

ii.1- Heuristiques

Les heuristiques constructives consistent à construire une solution pas à pas, son objectif est d'obtenir une bonne solution en utilisant des principes souvent très simples.

- Parmi ces principes, nous citons :

- les algorithmes gloutons : Ils sont caractérisés par le fait qu'il s'agit de procédure sans retour arrière : un choix fait à un moment n'est plus remis en question ultérieurement. Ces choix itératifs visent à construire une solution réalisable. Sans doute, le principe des méthodes gloutonnes est le plus utilisé. Nombreuses méthodes exactes sont basées sur le principe glouton. Elles tirent partie du fait que, dans un matroïde, toute heuristique gloutonne fournit une solution optimale.
- Le principe de construction progressive : C'est une extension du principe glouton dans la mesure où l'on s'autorise, cette fois-ci, de modifier des valeurs déjà assignées. Ceci correspond à accepter une remontée dans l'arbre décisionnel. Les algorithmes de backtracking sont des cas extrêmes puisqu'ils reposent sur un parcours exhaustif de l'arbre. Les heuristiques de construction de cycle hamiltonien pour le problème du voyageur de commerce (TSP), basées sur l'insertion itérative de nouveaux sommets sont à la fois des méthodes gloutonnes et à construction progressive selon qu'on se place sous l'angle des sommets insérés ou des arcs utilisés.
- Le principe de partitionnement : Il reprend le principe réductionniste de « diviser

pour régner » ; résoudre le problème global se révèle souvent plus complexe que résoudre la somme des sous problèmes qui le composent. Toute la difficulté réside alors dans la fusion des solutions de chaque sous problème (solutions approchées ou non et fusion exacte ou non). Plusieurs méthodes exactes sont basées sur cette approche (la décomposition).

- Heuristiques d'amélioration itérative (recherche locale) : Contrairement aux méthodes constructives dont l'objectif est de construire une solution, les méthodes d'amélioration modifient une solution initiale, en vue d'améliorer sa valeur. Cette solution initiale est souvent le résultat d'une méthode constructive ; un algorithme général sera composé de deux phases : une méthode constructive suivie d'une méthode d'amélioration. La plupart de ces méthodes utilise la notion de voisinage. Il s'agit de trouver, à chaque itération, une « bonne » solution parmi l'ensemble des solutions qui définissent un voisinage d'une solution courante. Parmi ces méthodes, nous distinguons celles qui améliorent, à chaque itération, la valeur de la fonction objectif, dites méthodes de descente, et celles qui permettent de choisir une solution qui n'améliore pas forcément la valeur de la fonction objectif. D'une manière générale, le principal inconvénient des méthodes de descente est qu'elles donnent souvent des solutions correspondant à des optima locaux qui ne sont pas de très bonne qualité, alors que, en choisissant une solution qui n'est pas de descente, on peut sortir des minima locaux.

ii.2- Métaheuristiques

Les métaheuristiques, quant à elles, reposent essentiellement sur le principe d'amélioration itérative. Elles nécessitent donc la possession d'une solution initiale.

– Dans ce qui suit, on présentera brièvement les méthodes métaheuristiques à savoir, la méthode de Recuit Simulé, la Recherche Tabou, les Algorithmes Génétiques et les algorithmes basés sur le principe des Colonies de Fourmis qui explorent également un voisinage.

- Le recuit simulé [20] et [65] : Il s'inspire du processus de recuit physique. Le processus du recuit simulé répète une procédure itérative qui cherche des configurations de coût plus faible (dans un problème de minimisation) tout en acceptant de manière contrôlée des configurations qui dégradent la fonction coût.
- La méthode tabou : Cette méthode, élaborée par Glover [48] , est basée sur la notion de mouvements interdits (tabou). Chaque itération consiste à trouver le mouvement qui nous donne la meilleure solution dans le voisinage de la solution courante ; sachant que certains mouvements sont interdits. Parfois, nous choisissons une solution qui détériore légèrement la solution courante pour s'échapper des minima locaux. L'inverse du mouvement effectué à chaque itération est rajouté dans une liste, appelée liste tabou, qui contient les mouvements interdits. Initialement, la liste tabou est vide [47].
- Les algorithmes génétiques : Ils sont proposés, pour la première fois, au milieu des

années 70 par J. Holland [58]. Ces algorithmes se proposent d'imiter la sélection naturelle et la génétique de la théorie de l'évolution. La terminologie de la génétique a été également empruntée pour la description de tels algorithmes. Contrairement aux méthodes Tabou et le Recuit Simulé qui manipulent une seule solution, les algorithmes génétiques considèrent un ensemble de solutions (individus) appelé population. Le but de la méthode est de faire évoluer la population en effectuant des mutations [5] et des croisements [6] suivis de sélection d'individus. L'idée de base de ces algorithmes réside dans le fait que travailler avec une population permet d'identifier et d'explorer les propriétés communes des bonnes solutions.

- Optimisation par colonies de Fourmis [25] : Cette métaheuristique, ACO (Ant Colony Optimization), s'inspire du comportement collectif des colonies de fourmis pour résoudre des problèmes à base de graphes. Une telle optimisation est basée sur le constat que, dans la nature, les fourmis (des animaux aveugles) sont capables d'établir par une somme d'interactions élémentaires le plus court chemin de leur nid à un objectif (une source de nourriture par exemple). Le but de cette approche est de trouver une solution annulant le coût et ceci en utilisant une multitude d'agents élémentaires (les fourmis) effectuant chacun de très simples actions.

5. La mutation est une opération unitaire qui modifié la structure d'un individu.
6. Le croisement est une opération binaire, qui à partir de deux individus, on produit deux nouveaux.

1.2 Rappels sur la théorie des graphes

Cette section rappelle quelques définitions classiques des graphes et des notations qui seront utilisées par la suite dans cette thèse. Pour la rédaction de toute cette partie, les définitions données, les graphes parfaits et les classes correspondantes.

1.2.1 Généralités

– ***1. Cas non orienté :***

- Un graphe G est la donnée de deux ensembles finis non vides V et E, où V représente l'ensemble des sommets noté $V(G)$ et E l'ensemble des paires des éléments de V noté $E(G)$ appelés arêtes.
- Un graphe simple est un graphe sans boucle (une arête dont les deux extrémités sont confondues) où tout couple de sommets est relié par au plus une arête.
- Deux sommets reliés par une arête sont dits adjacents ou voisins. Le degré d'un sommet u dans G, noté $d_G(u)$ est le nombre d'arêtes incidentes à ce sommet dans G.
- Un voisinage d'un sommet v, $N(v)$, est l'ensemble de ses sommets adjacents.
- Un graphe H est un sous-graphe induit de G s'il existe un sous-ensemble X de $V(G)$ tel que $H = G[X]$ tel que $E(G[X]) = \{u, v \in X, (u, v) \in E(G)\}$.
- Un graphe H est un sous-graphe partiel du graphe G si $V(H) \subseteq V(G)$ et $E(H) \subseteq E(G)$.
- Une chaîne de longueur k est une séquence $v_0, v_1, ..., v_k$ de sommets tels que v_i est adjacent à $v_{i+1}, \forall i = 0, 1, ..., k$ et v_0, v_k sont les deux extrémités de la chaîne, elle est dite élémentaire si elle n'utilise pas plus d'une fois le même sommet.
- Un cycle est une chaîne simple dont les extrémités sont confondues, il est élémentaire s'il n'utilise pas plus d'une fois le même sommet.
- Une corde est une arête qui relie deux sommets non consécutifs d'une chaîne ou d'un cycle.
- Une chaîne minimale de longueur k est une séquence $\{v_0, v_1, ..., v_k\}$ de sommets distincts tels que v_i est adjacent à v_{i+1}, $\forall i = \overline{0, k}$ sans corde et v_0, v_k représentent les deux extrémités de la chaîne, une telle chaîne est notée P_k. Un cycle sans corde ayant k sommets (de longueur k) est désigné par Cy_k.
- Le graphe complémentaire d'un graphe $G = (V, E)$ est le graphe $\overline{G} = (V, \overline{E})$ où $(u, v) \in \overline{E}$ si et seulement si $(u, v) \notin E$.
- Un cycle élémentaire de longueur au moins quatre sans corde est appelé trou et un anti-trou est le complémentaire d'un trou.
- Un graphe simple est dit complet si tous ses sommets sont adjacents entre eux.
- Un graphe est dit connexe si pour toute paire de sommets, il existe une chaîne joignant ces deux sommets.

Dans ce qui suit, nous rappelons quelques définitions importantes relatives aux problèmes classiques de la théorie des graphes.

Définition 1.2.1 *Une clique est un graphe dont tous les sommets sont adjacents entre eux.*

Définition 1.2.2 *Une clique C d'un graphe G est dite maximale s'il n'existe pas de clique C' de G telle que C soit strictement contenue dans C'.*

Définition 1.2.3 *Une clique C d'un graphe G est dite maximum s'il n'existe pas de clique C' de G telle que la taille de la clique C' est supérieure à la taille de la clique C, et $\omega(G)$ qui est appelé nombre de clique représente la taille de la plus grande clique.*

Définition 1.2.4 *La coloration des sommets d'un graphe consiste à affecter à chaque sommet une couleur de telle sorte que deux sommets adjacents aient des couleurs distinctes et le nombre de couleurs utilisées soit minimum. Ce nombre minimum de couleurs, $\chi(G)$, est appelé nombre chromatique du graphe G.*

Résoudre le problème de la coloration pour un graphe G revient à trouver le nombre de couleurs minimum, $\chi(G)$ tel qu'il existe une application $V \to \{1, 2, ..., \chi\}$ avec $(v_1, v_2) \in E \mapsto c(v_1) \neq c(v_2)$. Les ensembles de sommets ayant la même couleur sont, par définition, des stables de G.

Définition 1.2.5 *Un stable S est formé par un sous ensemble de sommets de V ne contenant aucune arête ; $\forall y_i, y_j \in S, (y_i, y_j) \notin E, i \neq j$ et la stabilité $\alpha(G)$ d'un graphe G dénote la taille d'un stable maximum dans G.*

– ***2. Cas orienté :***

- Un graphe orienté $G(V, A)$ est défini comme un ensemble V de n sommets ou nœuds et un ensemble $A \subset V^2$ de m arcs reliant ces sommets.
- Un arc (arête orientée) reliant le sommet v au sommet v_0 est noté (v, v_0).
- L'ensemble des successeurs de v_1, noté $\Gamma^+(v_1) = \{v_2 \in V \mid (v_1, v_2) \in A\}$.
- L'ensemble des prédécesseurs de v_1, noté $\Gamma^-(v_1) = \{v_2 \in V \mid (v_2, v_1) \in A\}$.
- Un chemin de longueur k est une suite de sommets $\{v_0, v_1, ..., v_k\}$ tels que $(v_i, v_{i+1}) \in A, i = 0, 1, ..., k$ et v_0, v_k sont les deux extrémités, initiale et finale de ce chemin, il est dit élémentaire s'il ne passe pas plus d'une fois par le même sommet.
- Un circuit est un chemin dont les deux extrémités sont confondues.

1.2.2 Cas des graphes parfaits

En 1960, l'historique des graphes parfaits a commencé quand Claude Berge a eu l'impression de ce qu'il nommait « *la belle propriété* », d'où vient le nom de *perfection*.

L'intérêt de C. BERGE dans l'introduction de ces concepts, a son origine les travaux de C.E. SHANNON [94] en théorie de l'information sur la capacité d'un canal de communication, à partir d'un ensemble de signaux que l'on peut émettre, on construit un graphe G dont l'ensemble des sommets est en bijection avec les différents signaux possibles. Dans le graphe G modélisant ce problème, deux sommets sont adjacents si et seulement si les signaux correspondant à ces sommets peuvent être confondus.

Un code est un ensemble de signaux qui ne peuvent être confondus. Le problème est de trouver le code le plus riche, le code transmettant un nombre maximum de signaux, ce qui revient à déterminer la cardinalité maximum $\alpha(G)$ d'un stable de G.

Soit $G(V, E)$ un graphe tel que V représente l'ensemble des sommets et E l'ensemble d'arêtes. Soient $\chi(G)$ le nombre chromatique du graphe G, $\omega(G)$ la taille de la plus grande clique, $\theta(G)$ la cardinalité minimum d'une couverture par des cliques de G et $\alpha(G)$ la taille du plus grand stable de G. Nous avons $\omega(G) \leq \chi(G) \leq |V|$ [77] et [60]. En effet, il faut au moins $\omega(G)$ couleurs pour colorier les sommets de G.

Berge a établi les concepts de graphe $\chi-$parfait et $\alpha-$parfait comme suit :

Définition 1.2.6 *(Graphe $\chi-$parfait)*
Un graphe est $\chi-$parfait si et seulement si pour tout sous graphe induit H de G, $\chi(H) = \omega(H)$.

Définition 1.2.7 *(Graphe $\alpha-$parfait)*
Un graphe est $\alpha-$parfait si et seulement si pour tout sous graphe induit H de G, $\theta(H) = \alpha(H)$.

La première conjecture qui en découle est la suivante :

Conjecture 1.2.1 *G est $\chi-$parfait si est seulement s'il est $\alpha-$parfait.*

Cette conjecture a été démontrée par Lovàsz [76] en 1972, depuis on désigne par un graphe parfait l'une des deux notions. Après définition de ces deux concepts, une recherche d'une caractérisation structurelle a été mise en place par Berge. Ainsi, les travaux de C.E. Shannon avaient encouragé Berge à s'intéresser aux graphes parfaits. Shannon avait constaté que le nombre de stabilité d'un C_5 est de 2 tandis que la taille minimale d'une partition en cliques de ce dernier est de 3, et que C_5 est le plus petit graphe ayant cette propriété. Le complémentaire de C_5 est $\overline{C}_5$, on aura aussi $\chi(\overline{C}_5) = 2$ et $\omega(\overline{C}5) = 3$. Globalement, pour les trous impairs C_{2k+1} pour $k \geq 2$, et les anti-trous, $\overline{C}_{2k+1}$ pour $k \geq 2$, on a : $\chi(C_{2k+1}) = 3$ et $\omega(C_{2k+1}) = 2$; $\omega(\overline{C}_{2k+1}) = k$ et $\chi(\overline{C}_{2k+1}) = k + 1$ pour $k \geq 2$.

Deux conjectures ont été proposées par C. Berge au début des années 60. La première conjecture a été démontré par Lovàsz en 1972. Cependant, la deuxième conjecture a suscité l'intérêt d'un très grand nombre de chercheurs, elle n'a été démontrée qu'en 2002

par Maria Chudnovsky et all dans un article d'un nombre de pages assez impressionnant (179 pages) [22], ce qui a donné les deux théorèmes suivants :

Théorème 1.2.1 *(Théorème faible des graphes parfaits [77])*
Un graphe est parfait si et seulement si son complémentaire est parfait.

Théorème 1.2.2 *(Théorème fort des graphes parfaits [8] et [7])*
Un graphe est parfait si et seulement si, ni lui ni son complémentaire ne contient un trou impair de longueur supérieure ou égale à 5*.*

Berge a conjecturé qu'un graphe est parfait si et seulement s'il ne contient pas de trous impairs et d'anti-trous impairs de longueur supérieure ou égale à 5.

Définition 1.2.8 *Un graphe de Berge est un graphe ne contenant pas comme sous graphe induit de trou ni d'anti-trou impair.*

Nous considérons dans ce qui suit d'un graphe G est parfait si et seulement si G est de Berge.

1.2.2.1 Classes des graphes parfaits

Nous citons dans ce qui suit quelques classes des graphes parfaits. Nous présentons quelques unes ; en particulier, celles qui feront l'objet d'analyse dans notre étude. La rédaction de cette partie est inspirée de [8, 17, 23, 26, 27, 45] et [46].

1.2.2.2 Les graphes bipartis

Un graphe G est dit biparti si l'ensemble de ses sommets peut être partitionné en deux sous ensembles stable S_1 et S_2. Si G contient toutes les arêtes reliant tous les sommets de S_1 à tous les sommets de S_2, alors le graphe G sera dit biparti complet. Ainsi, si S_1 contient p sommets et S_2 en contient q, alors G est noté $K_{p,q}$. Le $k-$parti (multiparti) complet noté $K_{p_1,...,p_k}$ a un ensemble de sommets qui peut être partitionné en k parties disjointes $S_1, ..., S_k$ telles que S_i possède P_i sommets, $i = 1, ..., k$, et deux sommets sont adjacents si et seulement s'ils appartiennent à deux parties distinctes.

1.2.2.3 Les graphes scindés

Un sommet scindé est un sommet dont le voisinage peut être partitionné en un stable S et une clique C_k. Un graphe G est dit à voisinage scindé, si tout sous graphe induit H de G possède un sommet scindé. Un graphe est un graphe scindé, dit aussi $1-$scindé, si ses sommets peuvent être partitionnés en une clique et un ensemble stable. Un graphe est un graphe k-scindés si ses sommets peuvent être partitionnés en k ensembles, chacun induit un graphe scindé. Un sommet v de G est dit $M-$scindé si v est un sommet scindé qui appartient à une clique maximum de G.

1.2.2.4 Les graphes triangulés

Un graphe est dit triangulé s'il ne possède pas de cycle induit de longueur supérieure ou égale à 4 sans corde. La perfection de cette classe a été établie par Berge ainsi que par Hajnal et Suranyi [54]. Ces deux derniers ont montré que le nombre de stabilité d'un graphe triangulé est égal à la taille minimale d'une partition en cliques. Berge a remarqué que cela signifie que les complémentaires des graphes triangulés sont χ-parfaits. Ensuite, Berge a démontré que dans un graphe triangulé, la taille maximale d'une clique est égale au nombre chromatique, comme tout sous graphe induit d'un graphe triangulé est triangulé. Une autre voie consiste à imposer des conditions sur les cycles impairs, par exemple sur leurs cordes éventuelles.

1.2.2.5 Les graphes i-triangulés

Un graphe est dit i-triangulé si tout cycle impair contient au moins deux cordes non croisées. La preuve de la perfection de cette classe revient à Gallai [40]. Les graphes i-triagulés contiennent la classe des graphes triangulés, ainsi que les graphes bipartis.

Un algorithme polynomial de reconnaissance a été construit par Burlet et Fonlupt (voir [27]). Ces deux dernières classes ont été généralisées par Meyniel.

1.2.2.6 Les graphes de Meyniel

Un graphe G est dit de Meyniel si tout cycle impair de G possède deux cordes. La perfection de cette classe fut établie par Meyniel en utilisant l'échange bi-chromatique. Une caractérisation, ainsi qu'un algorithme de reconnaissance, reviennent à Burlet et à Fonlupt. Le problème de la coloration est également résolu pour ces graphes (voir [17]).

Les conditions qui font intervenir deux cordes étant épuisées, tout en ayant fourni de bons résultats, il est naturel de considérer les conditions relatives à une seule corde. Cette classe généralise celle des graphes de parité ainsi que celle des graphes i-triangulés et les graphes de parité sont de Meyniel.
Une autre classe intéressante, contenant également les graphes triangulés, est la classe des graphes faiblement triangulés, introduite par Hayward [55].

1.2.2.7 Les graphes faiblement triangulés

Un graphe G est dit faiblement triangulé si ni G ni son complémentaire $\bar{G}$, ne contiennent des cycles induits de longueur au moins 5. Hayward a montré que ces graphes sont parfaits et a donné un algorithme polynomial pour les reconnaître. Les problèmes d'optimisation dans cette classe sont résolus par Hayward et all.

1.2.3 Quelques problèmes et méthodes classiques

Dans cette partie, nous présentons un aperçu sur la littérature des problèmes ayant trait principalement à : schéma d'ordre, orientation des arêtes, coloration des sommets et problème de la clique.

1.2.3.1 Problème d'ordre d'étiquetage des sommets

De nombreux travaux ont été réalisés afin d'établir un ordre des sommets d'un graphe G, afin d'aboutir à une caractérisation du graphe en question. Quelques stratégies sont proposées dans la littérature [24, 29, 31, 41, 42, 85, 86, 90] et [98], nous en citons :

1.2.3.2 Schéma d'élimination parfait

Un schéma d'élimination parfait dans un graphe à n sommets est un ordre des sommets $v_1, v_2, ..., v_n$, tel que v_i est simplicial[7] (voir [15]).

En effet, l'application du schéma d'élimination parfait sur le graphe de la figure ci-après (figure 2.1) donne l'ordre suivant $x_2, x_1, x_4, x_3, x_5, x_6$. Notons que, nous pourrions choisir le sommet x_1, x_4, x_5 ou x_6 dans la première position, mais pas le sommet x_3 car ses deux voisins x_5 et x_6 ne sont reliés aux autres sommets. Il devient simplicial dans le graphe résiduel en supprimant les sommets x_2, x_1 et x_4.

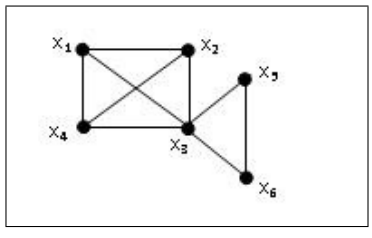

FIGURE 1.1 – Illustration pour le schéma d'élimination parfait

1.2.3.3 Graphes parfaitement ordonnable

Un graphe est parfaitement ordonnable [100] s'il admet un ordonnancement $v_1, v_2, ..., v_n$ des sommets tel qu'aucun $P4[a, b, c, d]$ n'ait $a < b$ et $d < c$; un tel ordre s'appelle *ordre parfait* et un $P4$ interdit s'appelle une *obstruction*.

1.2.3.4 Graphes de permutation

Un graphe $G(V, E)$ est dit de permutation s'il existe une permutation π des sommets telle que v_1 est adjacent à v_2 dans G si et seulement si $v_1 < v_2$ et $\pi(v_2) < \pi(v_1)$ ou $v_2 < v_1$ et $\pi(v_1) < \pi(v_2)$.

1.2.3.5 Ordre de contraction

Plusieurs classes de graphes parmi les graphes parfaitement contractiles, on trouve les graphes de Meyniel et les graphes faiblement triangulés (voir Hertz [56], aussi Hayward et all [55]). La contraction des graphes de Meyniel peut se faire en choisissant n'importe quel sommet, on le contracte jusqu'à l'obtention d'une clique. Ce n'est pas le cas pour les graphes faiblement triangulés.

7. Un sommet est appelé simplicial si le graphe induit par ses voisins est un graphe complet.

Un ordre de contraction de G est une notation de ses sommets sous la forme $V(G) = \{x_1^1, x_1^2, ..., x_1^{k_1}, x_2^1, ..., x_2^{k_2}, ..., x_l^1, ..., x_l^{k_l}\}$ telle que, pour $1 \leq c \leq l$, les ensembles $S_c = \{x_c^1, ..., x_c^{k_c}\}$ sont des stables.

Un ordre de contraction est dit amical (resp. P_4-libre) si la suite de contraction associée vérifie, pour tout $1 \leq c \leq l$ et $2 \leq i \leq k_c$.

Un graphe est dit ordre-contractile (resp P_4-libre-ordre-contractile) s'il possède un ordre de contraction amical (resp. P_4-libre).

1.2.3.6 L'algorithme LexBFS

L'algorithme LexBFS [90](Lexicografic Breadth-First Search) permet de donner une numérotation des sommets d'un graphe G de telle sorte qu'un sommet non numéroté reçoit une étiquette. Cette dernière correspond à l'ensemble de ses voisins déjà numérotés. On définit un ordre lexicographique sur les étiquettes, L, tel que $L(v_1) > L(v_2)$
si $\exists i \in L(v_1) - L(v_2)$ tel que $\forall j > i$, $j \in L(v_1) \cap L(v_2)$ où $j \notin L(v_1) \cup L(v_2)$. Le sommet suivant dans la numérotation est celui qui a l'étiquette maximale dans l'ordre lexicographique.

Algorithm 1 : Algorithme LexBFS

Entrées : $G(V, E)$.

Sorties : Un ordre d sur les sommets de G.

```
DEBUT
Pour tout sommet v ∈ V Faire
    L(v) ← ∅ ;
Fin pour
Pour i ← n à 1 Faire
    d(v) ← i tel que v est le sommet d'étiquette maximale ;
    Pour tout x non numéroté tel que (v, x) ∈ E Faire
        Ajouter i à L(x) ;
    Fin pour
Fin pour
FIN.
```

Un ordre d'élimination simplicial est un ordre des sommets de G, $(v_1, ..., v_n)$, tel que le sommet $v_i \setminus 1 \leq i \leq n$ est un sommet simplicial dans le graphe $G[v_1, ..., v_i]$.

L'algorithme LexFBS a été utilisé pour la reconnaissance des graphes triangulés, de telle sorte que si l'ordre trouvé par l'algorithme LexFBS est un ordre d'élimination simplicial alors le graphe en question est triangulé.

1.2.3.7 Problème d'orientation des arêtes

L'orientation des graphes est aussi utilisée par plusieurs chercheurs afin de résoudre quelques problèmes de la littérature [10, 37, 39, 44, 48, 66] et [101]

1.2.3.8 Graphes de comparabilité

Un graphe $G(V, E)$ est dit de comparabilité ([29] et [44]) s'il est possible d'orienter ses arêtes de façon transitive autrement dit :

$$\begin{cases} (v_1, v_2) \in A \\ \\ (v_2, v_3) \in A \end{cases} \Rightarrow (v_1, v_3) \in A$$

Algorithm 2 : Algorithme de reconnaissance d'un graphe de comparabilité

Entrées : $G(V, E)$.

Sorties : G un graphe de comparabilité ou non-comparabilité.

DEBUT
$F \leftarrow \emptyset$;
Tantque $F \neq E$ **Faire**
Choisir une arête e dans $E - F$, donner une orientation à e et propager cette orientation pour assurer une orientation transitive de G.
Si une arête doit être orientée dans les deux sens **Alors**
G n'est pas un graphe de comparabilité ;
Sinon
rajouter à F toutes les arêtes nouvellement orientées ;
Si $F = E$ **Alors**
G est un graphe de comparabilité.
Finsi
Finsi
Fin tantque
FIN.

Il est prouvé que les graphes de comparabilités sont parfaitement ordonnables car une orientation transitive induit un ordre parfait.

1.2.3.9 Graphes d'opposition

Un graphe G est dit graphe d'opposition si et seulement s'il admet une orientation de ses arêtes de telle sorte que les deux ailes [8] de tout P_4 [9] induit aient une orientation opposée.

1.2.3.10 Graphes de cordes

Un diagramme $C(V)$ est une système constitué par un cercle C et un ensemble fini non vide V de cordes de C. Un graphe est dit graphe de cordes si l'ensemble de ses sommets peut être mis en bijection avec l'ensemble des cordes d'un diagramme de telle sorte que deux sommets sont adjacents si et seulement si les cordes correspondantes se croisent. Un tel diagramme est appelé diagramme associé au graphe (voir [84]).

1.2.3.11 Caractérisation des graphes de cordes

Soit $G = (V, E)$ un graphe de cordes et soit $C(V)$ un diagramme associé [84]. Numérotons les différentes extrémités des cordes de $C(V)$ de 1 à $2n \setminus n = |V|$, dans l'ordre où elles sont rencontrées, en parcourant le cercle dans le sens des aiguilles d'une montre avec un point de départ arbitraire. On associe à chaque sommet x du graphe les numéros affectés aux extrémités de la corde correspondante, que l'on note $a(x)$ et $b(x)$ avec $a(x) < b(x)$. Nous définissons ensuite deux orientations L_1 et L_2 du graphe complet $(V, \wp_2(V))$ de la manière suivante :

$$(x, y) \in L_1 \Leftrightarrow a(x) < a(y) \text{ et } (x, y) \in L_2 \Leftrightarrow b(x) < b(y).$$

$M \subset (L_1 \cup L_2)$ et $F = (L_1 \cap L_2) - M$ de $V \times V$ est défini par : $(x, y) \in F \Leftrightarrow b(x) < a(y)$. Ainsi,

$$(F_1) : (x, y) \in F \text{ et } (y, z) \in L_1 \Rightarrow b(x) < a(y) < a(z) \Rightarrow (x, z) \in F\,;$$
$$(F2) : (x, y) \in L_2 \text{ et } (y, z) \in F \Rightarrow b(x) < b(y) < a(z) \Rightarrow (x, z) \in F.$$

Théorème 1.2.3 *[36] Un graphe $G = (V, E)$ est un graphe de cordes si et seulement s'il existe un triplet (M, L_1, L_2), où M est une orientation de G, L_1 et L_2 sont deux orientations transitives du graphe complet $(V, \wp_2(V))$ contenant M, et tel que la partie $F = (L_1 \cap L_2) - M$ vérifie (F_1) et (F_2).*

1.2.3.12 Graphes quasi-parfaitement ordonnables

- Un noyau d'un graphe orienté est un ensemble N de sommets qui est à la fois stable et absorbant : N est un ensemble stable et $\forall v_1 \in V - N, \exists v_2 \in N \setminus (v_1, v_2) \in A$. Un graphe est noyau-parfait [32] si chacun de ses sous-graphes induits possède un noyau.
- Une orientation est dite stricte si $\exists (a, b) \in A$ alors $(b, a) \notin A$.

8. Une aile est une arête définissant l'une des extrémités d'un chemin
9. Un P_4 est un chemin de longueur 4.

Théorème 1.2.4 *[31] : Un graphe non orienté est dit quasi-parfaitement ordonnable s'il admet une orientation stricte qui est un noyau parfait et sans fourche.*

Le graphe de la figure 1.2 est une fourche.

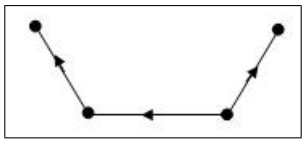

FIGURE 1.2 – Exemple d'une fourche

1.2.3.13 Problème de coloration des sommets d'un graphe

De nombreux problèmes peuvent être modélisés sous forme de problèmes de coloration de graphes, et ce dans divers domaines. Il s'agit des problèmes se ramenant à la recherche d'une partition d'un ensemble d'objets en sous ensembles ne comportant que des éléments compatibles deux à deux. A titre d'exemple, la détermination d'un horaire scolaire est une partition du temps en périodes durant lesquelles seuls des cours pouvant avoir lieu simultanément y sont donnés. Citons également, comme illustration, l'élimination des collisions dans des réseaux sans fils, le problème d'allocation des registres en informatique et le problème d'ordonnancement dont il existe souvent deux types de contraintes simultanées : des contraintes de précédence et des contraintes de disjonction.

Les algorithmes existants dans la littérature (voir [83]) peuvent être efficaces pour certaines classes de graphes ou pour des graphes de taille moindre.

1.2.3.14 Algorithme Glouton

L'algorithme glouton de coloration considère les sommets selon un ordre "<" et donne à chaque sommet la plus petite couleur non encore attribuée à l'un de ses voisins ;

Un graphe est parfaitement ordonné avec l'ordre "<" si et seulement si l'algorithme glouton appliqué à chaque sous-graphe induit utilisant l'ordre "< " colorie le sous-graphe avec un nombre minimum de couleurs.

Nous présentons dans ce qui suit quelques unes des méthodes de coloration des sommets d'un graphe pour certaines classes des graphes parfaits en se basant sur l'idée de l'orientation et le schéma d'ordre.

1.2.3.15 Graphes de comparabilité

L'algorithme 3 exploite l'orientation afin d'aboutir à une solution optimale pour les graphes de comparabilité [45, 91] et [99].

Algorithm 3 : Algorithme de coloration d'un graphe de comparabilité

Entrées : $G(V, E)$.

Sorties : k-coloration.

DEBUT

- Déterminer une orientation transitive de G ; $i \leftarrow 1$;

Tantque $\exists v \in V$ tel que v n'est pas colorié **Faire**

- Affecter la couleur i à toutes les sources ; $i \leftarrow i + 1$;

Fin tantque

FIN.

1.2.3.16 Graphes d'intervalles

Soit $I = \{I_1, I_2, ..., I_n\}$ une famille d'intervalles [44] sur une droite. On peut former un graphe G, dont les sommets $v_1, v_2, ..., v_n$ représentent respectivement les intervalles $I_1, I_2, ..., I_n$; deux sommets étant joints si les intervalles correspondants s'intersectent. Un tel graphe est dit représentatif de la famille I ou graphe d'intervalles (voir [44]).

Théorème 1.2.5 *[44] Les graphes d'intervalles appartiennent à la classe des graphes parfaits.*

Théorème 1.2.6 *[44] Un graphe G est d'intervalles si et seulement si G est triangulé et $\bar{G}$ est un graphe de comparabilité.*

Une méthode de coloration de cette classe de graphes est donnée par l'algorithme 4.

Algorithm 4 : Algorithme de coloration d'un graphe d'intervalles

Entrées : $G(V, E)$.

Sorties : k-coloration.

DEBUT

- Déterminer un schéma d'élimination parfait des sommets de G, soit $v_1, v_2, ..., v_n$;
- Colorier les sommets $v_n, v_{n-1}, ..., v_1$ en choisissant toujours la première couleur disponible.

FIN.

1.2.3.17 Problème de la clique

La recherche de la plus grande clique ou sa cardinalité dans un graphe est un problème difficile de la théorie des graphes, avec de nombreux domaines d'applications. Les algorithmes existants dans la littérature peuvent être efficaces pour certaines classes de graphes ou pour des graphes de taille moindre.

1.2.3.18 Heuristique pour le problème de la clique maximum

Ci-dessous une heuristique qui tente à déterminer la plus grande clique dans un graphe.

Algorithm 5 : Heuristique pour le problème de la clique [100]

Entrées : $G(V, E)$.

Sorties : C.

DEBUT

$S \leftarrow V$; $C \leftarrow \emptyset$;

Tantque $S \neq \emptyset$ **Faire**

- Choisir un sommet $v \in S$ de degré maximum dans G ;
- $C \leftarrow C \cup \{v\}$;
- $S \leftarrow S \cap N_G(v)$;

Fin tantque

FIN.

Dans cette partie une heuristique pour le problème de la coloration fractionnaire (PCF) est présentée, cette dernière fournit des bornes supérieures pour le problème de la clique maximum. Pour ce, faire l'auteur dans [100] a utilisé deux algorithmes pour décrire cette heuristique, l'algorithme COLOR et l'algorithme DSATUR.

COLOR est un algorithme de coloration séquentielle, il permet d'attribuer la première couleur disponible aux sommets de V puis à ré-exécuter la même procédure avec la couleur suivante jusqu'à ce que tous les sommets soient coloriés.

Algorithm 6 : L'algorithme COLOR [100]

Entrées : $G(V, E)$.

Sorties : Une classe de couleur C_k.

DEBUT

$C_k \leftarrow \emptyset$; $S \leftarrow V$;

Tantque $S \neq \emptyset$ **Faire**

$c(v) \leftarrow k$ tel que v est le sommet de degré maximum dans G ;

- $C_k \leftarrow C_k \cup \{v\}$;
- $S \leftarrow (S - \{v\}) - N_G(v)$;

Fin tantque

FIN.

Dans les références [32, 36] et [84], les bornes supérieures sont déterminées en utilisant l'heuristique de coloration DSATUR de Brélaz [13].

Algorithm 7 : L'algorithme DSATUR [100]

Entrées : $G(V, E)$.

Sorties : Une coloration-C.

DEBUT

Tantque $\exists v \in V \setminus v \notin C$ **Faire**

- Choisir un sommet non colorié adjacent au nombre maximum de différentes couleurs (appelé le degré de saturation de v ;

- Attribuer au sommet v la plus petite couleur non encore attribuée aux sommets voisins.

Fin tantque

FIN.

Après i itérations du PCF, chaque sommet est coloré exactement i fois, un poids est attribué à chaque classe de couleur $\frac{1}{i}$, ainsi $t_i = \frac{|C|}{i}$ est une borne supérieure de $\omega(G)$.

Algorithm 8 : FPC à l'itération i [100]

Entrées : $G(V, E)$.

Sorties : La borne supérieure B.

DEBUT

$C \leftarrow \emptyset$; $i \leftarrow 1$; $t_0 \leftarrow \infty$;

Répéter

Pour chaque sommet v **Faire**

Si $\exists v \setminus C_j \cup \{v\}$ est un ensemble indépendant où C_j est la première classe de couleur

Alors

$C_j \leftarrow C_j \cup \{v\}$;

Sinon

Soit U l'ensemble des sommets tel que $U \notin C_j$;

Trouver une $(C_1, C_2, ..., C_k)$ coloration de $G(U)$ en utilisant COLOR ou DASTUR ; $C \leftarrow C \cup \{C_1, C_2, ..., C_k\}$; $t_i \leftarrow \frac{|C|}{i}$

Finsi

Fin pour

$i \leftarrow i + 1$;

Jusqu'à $t_i \geq t_{i-1}$

$B \leftarrow \lfloor t_{i-1} \rfloor$

FIN.

Les auteurs dans [35] et [96] ont présentés une méthode de type séparation et évaluation pour le problème de la clique maximum utilisant une heuristique pour la borne supérieure.

Étant donné un graphe $G = (V, E)$, l'algorithme maintient les conditions suivantes :

1. Si h est la hauteur de l'arbre de recherche de l'ensemble des sommets $\{v_1, v_2, ..., v_{h-1}\}$ alos les sommets $v_i \setminus 1 \leq i \leq h - 1$ sont deux à deux adjacents.
2. M est la plus grande clique trouvée par l'algorithme ; $h - 1 \leq |M| \leq \omega(G)$.

3. Pour $1 \leq i \leq h$ l'ensemble des sommets $S_i \subseteq \cap_{j=1}^{i-1} N_G(v_j)$ comprend des candidats pour élargir $\{v_1, v_2, ..., v_{i-1}\}$ en une clique.
4. Pour $1 \leq i \leq h, C_1^i, C_2^i, ..., C_{k_i}^i$ est une coloration de $G(S_i)$. k_i et k_i' (trouvés par FCP) sont des bornes supérieures pour $\omega(G(S_i))$ avec $k_i' \leq k_i$.

Algorithm 9 : Algorithme MC [100]

Entrées : $G(V, E)$.

Sorties : clique Q.

DEBUT

Étape 0 **(Initialisation) :** Trouver une clique maximum M d'un sous-graphe triangulé d'arête-maximale de G. $h \leftarrow 1$; $S_h \leftarrow V$; Aller à l'étape 2.

Étape 1 **(Calcul de la borne inférieure) :** Trouver une clique Q de $G(S_h)$.

Si $h - 1 + |Q| > |M|$ **Alors**

$M \leftarrow \{v_1, v_2, ..., v_{h-1}\} \cup Q$;

Finsi

Aller à l'étape 2.

Étape 2 **(Calcul de la borne supérieure) :** Trouver une coloration $C_1^h, C_2^h, ..., C_{k_h}^h$ de $G(S_h)$.

Si $h - 1 + k_h \leq |M|$ **Alors**

Aller à l'étape 4.

Sinon

Appliquer FCP à $G(S_h)$ pour obtenir une borne supérieure $K'_h \geq \omega(G(S_h))$.

Si $h - 1 + k'_h \leq |M|$ **Alors**

Aller à l'étape 4.

Finsi

Finsi

Aller à l'étape 3.

Étape 3 **(Branchement) :** Choisir un sommet $v_h \in C_{k_h}^h$ de degré maximum dans G. $S_{h+1} \leftarrow S_h \cap N_G(v_h)$; $S_h \leftarrow S_h - \{v_h\}$; $C_{k_h}^h \leftarrow C_{k_h}^h - \{v_h\}$;

Si $C_{k_h}^h = \emptyset$ **Alors**

Décrémenter k_h ;

Finsi

Si $k_h < k'_h$ **Alors**

$k'_h \leftarrow k_h$;

Finsi

Incrémenter h et aller à l'étape 1.

Étape 4 **:**

Si $h = 1$ **Alors**

M est une clique maximum de G.

Sinon

Décrémenter h ;

Si $h - 1 + k'_h \leq |M|$ **Alors**

Aller à l'étape 4.

Sinon

Aller à l'étape 3.

Finsi

Finsi

FIN.

Chapitre 2

Un schéma d'ordre d'étiquetage pour le problème de la clique maximum

Sommaire

2.1 Introduction

Plusieurs problèmes réputés NP-complets tels que la recherche de : coloration minimum, clique maximum, ou stable maximum se résolvent en temps polynomial pour des classes de graphes parfaits.

Ce chapitre est consacré à l'approche heuristique de résolution du problème de la clique maximum et au résultat fondamental définissant l'ordre d'étiquetage proposé ainsi que l'algorithme d'orientation des arêtes. Les résultats ainsi obtenus sont exploités afin de caractériser la clique maximum pour certaines catégories de classes des graphes parfaits et d'autres graphes qui vérifient la propriété d'ordre d'étiquetage local. Nous présenterons aussi une nouvelle méthode heuristique pour la détermination de $\omega(G)$ ainsi que la clique correspondante.

Étant donné un graphe simple non orienté $G = (V, E)$ où V désigne l'ensemble des sommets et $E \subseteq V \times V$ désigne celui des arêtes. Soit A l'ensemble des arcs obtenus en orientant les arêtes de E. On note $G^O = (V, A)$ le graphe orienté associé au G. Les propriétés de G^O et la complexité des algorithmes pour les caractériser dépendent naturellement de celles du graphe d'origine G. Elles dépendent également de la procédure d'orientation des arêtes transformant E en A.

2.2 Algorithme d'orientation et ordre d'étiquetage

La méthode proposée pour déterminer la cardinalité de la clique maximum, $\omega(G)$, d'un graphe G procède en deux étapes : la première consiste à orienter le graphe puis en déduire un ordre local d'étiquetage des sommets ; tandis que la seconde détermine la cardinalité de la clique maximum.

2.2.1 Principe de l'algorithme d'orientation

Le principe de la méthode consiste à choisir un sommet non encore marqué ayant le plus grand degré, à orienter toutes les arêtes qui lui sont incidentes vers l'extérieur, puis à marquer tous ses voisins et le considérer comme traité. Réitérer ce processus sur le graphe partiel résiduel obtenu en retirant ce sommet traité. À l'issue de ce processus, tous les sommets sont soit marqués, soit traités. S'il existe encore des arêtes non orientées, ré-exécuter la même procédure sur ce sous graphe partiel résiduel (les marques des sommets non traités sont supprimées) en considérant les sommets dans l'ordre décroissant des degrés par rapport au graphe d'origine.

Remarque 1 *L'ordre initial des sommets de même degré dans le sous graphe partiel, doit être maintenu.*

2.2.2 Algorithme d'orientation

Algorithm 10 : Algorithme d'orientation

Entrées : $G(V, E)$

Sorties : $G^0(N, A)$

Variables utilisées :

V_T : L'ensemble des sommets traités tel que $V_T \subseteq V$;

$\tilde{V}$: L'ensemble des sommets voisins des traités, V_T ;

$\tilde{E}$: L'ensemble des arêtes transformées en arcs noté $\tilde{A}$ tel que $\tilde{E} \subseteq E$;

ORIENTE(x, y) : Fonction qui transforme l'arête (x, y) en l'arc (x, y), la fonction ORIENTE $(\tilde{E})$ donne en sortie $\tilde{A}$;

SGP$(\tilde{V}, \tilde{E})$: Est une fonction qui retourne (V_s, E_s) tel que $E_s = E - \tilde{E}$ et $V_s \subseteq \tilde{V} / V_s = \{v_i, v_j \in \tilde{V} / (v_i, v_j) \in E_s$ avec $i \neq j\}$;

$arret$: variable booléenne valant vrai si toutes les arêtes sont orientées.

DEBUT

1: $arret \leftarrow faux$; $N \leftarrow V$; $V_T \leftarrow \emptyset$; $\tilde{V} \leftarrow \emptyset$; $\tilde{E} \leftarrow \emptyset$; $\tilde{A} \leftarrow \emptyset$; $V_s \leftarrow V$; $E_s \leftarrow E$;

2: **Tantque** $arret = faux$ **Faire**

3: - Déterminer un sommet x dans $V_s - (V_T \cup \tilde{V})$ de degré maximum ; $V_T \leftarrow V_T \cup \{x\}$;

4: - Pour tout sommet $y \in V$ tel que $(x, y) \in E_s$:

5: $\tilde{E} \leftarrow \tilde{E} \cup (x, y)$; $\tilde{V} \leftarrow \tilde{V} \cup \{y\}$; ORIENTE(x, y) ; $\tilde{A} \leftarrow \tilde{A} \cup (x, y)$;

6: **Si** $\tilde{E} \neq E$ et $V_s = \tilde{V} \cup V_T$ **Alors**

7: $(V_s, E_s) \leftarrow$SGP$(\tilde{V}, \tilde{E})$; $V_T \leftarrow \emptyset$; $\tilde{V} \leftarrow \emptyset$;

8: **Sinon Si** $\tilde{E} = E$ **Alors**

9: $arret = vrai$;

10: **Finsi**

11: **Fin tantque**

12: $A \leftarrow \tilde{A}$;

FIN.

Illustration de l'algorithme d'orientation

L'exemple suivant donne une illustration de notre algorithme d'orientation.

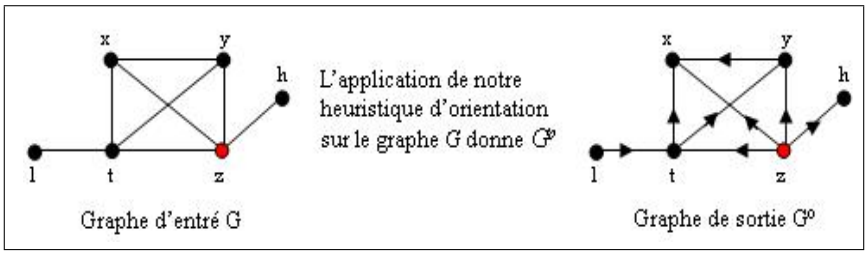

FIGURE 2.1 – Illustration de l'algorithme d'orientation

2.2.3 Analyse de l'algorithme

L'algorithme proposé, procède en un nombre fini d'étapes ($k < n$) et s'arrête quand les m arêtes sont orientées, ainsi la complexité de l'algorithme d'orientation des arêtes est d'ordre $k\dot{\Delta}$.
En effet, pour un sommet x du graphe G, on parcourt son voisinage, de taille au plus égale à Δ, non marqué et/ou non déjà traité tout en orientant les arêtes qui lui sont incidentes. On répète le processus avec k sommets tel que k représente le nombre de sommets qui peuvent être traités ($k < n$). Donc pour :

- $x_1 \xrightarrow{\text{Orientation de}} d_1$ arêtes.
- $x_2 \xrightarrow{\text{Orientation de}} d_2$ arêtes.
- .
- .
- .
- $x_k \xrightarrow{\text{Orientation de}} d_k$ arêtes.

avec $d_i \leq d(x_i) \leq \Delta$. Ce qui montre que notre algorithme est polynomial.

2.2.4 Ordre d'étiquetage

Un schéma d'ordre d'étiquetage consiste à définir une numérotation des sommets V, de 0 à l tel que $l \leq |V| - 1$. Cette numérotation définit une application surjective de l'ensemble des sommets de V dans l'ensemble des entiers $\{0, 1, ..., l\}$. Si $l = |V| - 1$, cette application est bijective et l'ordre d'étiquetage des sommets est dit sans redondance. L'ordre d'étiquetage est dit local lorsqu'il porte sur un sous ensemble de sommets.

Définition 2.2.1 *Nous définissons un schéma d'ordre d'étiquetage des sommets du graphe $G^o(V, A)$ par l'application d'incidence I ci-dessous :*

$$\begin{array}{rccl} I: & V & \longrightarrow & \{0, 1, ..., |V| - 1\} \\ & v & \longmapsto & \underset{G^o}{incidence}(v) \end{array}$$

Cet ordre d'étiquetage sera utilisé pour établir les principaux résultats portant sur le problème de la clique maximum.

2.2.5 Comparaison de schéma d'ordre d'étiquetage avec le schéma d'ordre simplicial

Dans cette partie, nous comparons notre schéma d'ordre d'étiquetage avec le schéma d'ordre classique d'élimination parfait nommé(PES), et appelé aussi le schéma simplicial. L'application du PES sur le graphe de la figure 2.2 donne $x_2, x_1, x_4, x_3, x_5, x_6$. Notons que, nous pourrions choisir x_1, x_4, x_5 ou x_6 en première position, mais pas x_3 et ceci car ses voisins ne sont pas reliés entre eux. Cependant, selon notre méthode le sommet x_3 doit occuper nécessairement la première position ainsi, l'ordre obtenu est $x_3, x_1, x_4, x_2, x_5, x_6$. Pour cette exemple, nous n'avons pas le choix pour le sommet de la première position.

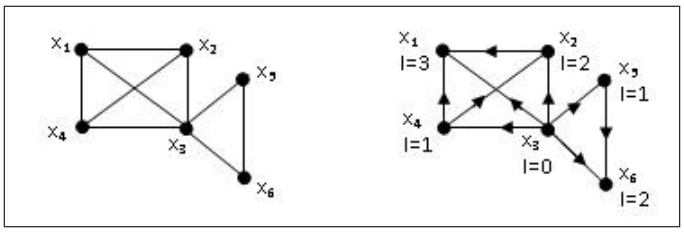

FIGURE 2.2 – Exemple 1 : Comparaison de l'ordre donné par le schéma d'ordre d'étiquetage avec l'ordre simplicial

D'une autre part, le graphe de la figure 2.3 ne possède pas un schéma d'élimination parfait du fait que ses voisins sont deux à deux disjoints. Toutefois, notre méthode donne l'ordre suivant : x_1, x_4, x_2, x_3, x_5.

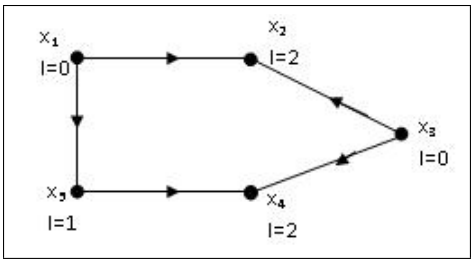

FIGURE 2.3 – Exemple 2 : Comparaison de l'ordre donné par le schéma d'ordre d'étiquetage avec l'ordre simplicial

2.3 Caractérisation de la clique maximum

Nous présentons dans ce qui suit nos principaux résultats en se basant sur l'approche développée dans ce chapitre, il s'agit de l'algorithme d'orientation et du schéma d'ordre d'étiquetage.

Théorème 2.3.1 *Toute orientation d'une clique C_k possède un schéma d'ordre d'étiquetage sans redondance [51].*

Preuve 1 *Soit x le premier sommet traité de la clique C_k. Toutes les arêtes provenant du sommet x sont orientées vers l'extérieur. Cela donne :*
$d^+_{C_k^o}(x) = 0 = incidence(x) \Rightarrow I(x) = 0$.

En retirant le sommet x de la clique C_k, le sous graphe partiel induit par les arêtes non traitées définit une clique de cardinalité $k-1$ telle que $\forall v \in C_{k-1}$ il existe une orientation de x vers v.

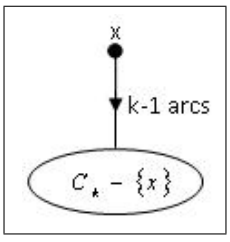

FIGURE 2.4 – La première étape de l'orientation récurrente de la clique C_k

En répétant le même processus avec le sommet y de la clique C_{k-1}, consécutivement à l'orientation, les arêtes issues du sommet y sont orientées à l'extérieur hormis celles découlant de x.

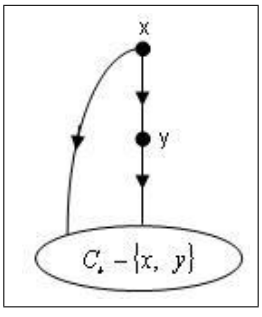

FIGURE 2.5 – La deuxième étape de l'orientation récurrente de la clique C_k

Ainsi, $d^+_{C^o_k}(y) = 1 =$*incidence*$(y) \Rightarrow I(y) = 1$.
Soit z le sommet suivant non encore traitées, de la clique C_{k-2}. En orientant les arêtes (non traitées) qui lui sont incidentes à l'extérieur sachant que (x, y) et (y, z) sont dans A.

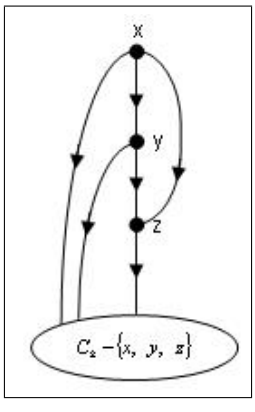

FIGURE 2.6 – La troisième étape de l'orientation récurrente de la clique C_k

Ceci montre que : $d^+_{C^o_k}(z) = 2 =$ *incidence*$(z) \Rightarrow I(z) = 2$.
En poursuivant le même procédé jusqu'à l'itération $k-2$ où le sous graphe partiel induit

par les arêtes non traitées est une clique de cardinalité 2 *(une arête formée par deux sommets* t *et* h*) ayant une orientation dans un sens. Considérons celle de* t *à* h*.*

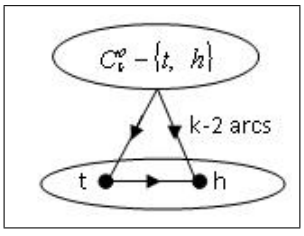

FIGURE 2.7 – L'étape $k-2$ de l'orientation récurrente de la clique C_k

Donc, $incidence(t) = k-2 \Rightarrow I(t) = k-2$ *;*
$Incidence(h) = k-1 \Rightarrow I(h) = k-1$.
Ceci nous amène à la propriété suivante :
Pour toute orientation d'une clique de cardinalité k*, il existe un seul sommet* $u \in C_k$ *tel que* $incidence(u) = k-1$*. Le sommet* u *étant celui d'incidence maximale, c'est à dire* $I(u) = k-1$*, Les autres sommets étant strictement ordonnés de* 0 *à* $k-2$ *ce qui établit un schéma d'ordre d'étiquetage des sommets de* C_k*. Nous pouvons conclure que la fonction d'incidence* I *prend ses valeurs dans* $\{0, 1, 2, ..., k-2, k-1\}$.

Par la suite, ce théorème sera utilisé pour caractériser la clique maximum, pour des graphes spécifiques.
Etant donné un sommet de transition maximale x et le sous graphe partiel G_{sp} induit par $\Gamma^+(x)$.

Théorème 2.3.2 *Si l'application* I *définit un schéma d'ordre d'étiquetage sans redondance sur le graphe* G^o_{sp} *alors* G^o_{sp} *est une clique maximum d'ordre* $\max_{v \in G^o_{sp}}(incidence(v)) + 1$ *[51].*

Preuve 2 *Voir la preuve du Théorème* 2.3.1.

Théorème 2.3.3 *Si l'application* I *définit un schéma d'ordre d'étiquetage sans redondance sur le graphe induit par les sommets de* $V_{sp} - \{x\}$ *alors* x *est voisin à une clique maximale ou maximum de taille :* $\max_{v \in G^o_{sp} - \{x\}}(incidence(v)) + 2$ *[51].*

Preuve 3 *Nous démontrons par l'absurde. De ce fait, nous supposons que le sommet de transition maximale* x *est un sommet d'une clique* C_k *non maximale (resp. non maximum). D'après le Théorème* 2.3.1*, l'application* I *définit un schéma d'ordre d'étiquetage local sur les voisins du sommet* x *et la taille de cette clique* C_k *est d'ordre* $\max_{v \in G^o_{sp} - \{x\}}(incidence(v)) + 2$*.*
C_k *est non maximale (resp. non maximum) donc il existe au moins un sommet* v *n'ayant pas une étiquette dans ce schéma d'ordre tel que* v *est voisin à tout sommet de la clique* C_k*. Or* x *est un sommet de transition maximale donc le sommet* v *est concerné par le schéma d'ordre d'étiquetage, ce qui est contradictoire.*

2.3.1 Certaines classes particulières de graphes

$G^o(V, E^o)$ est le graphe issu du graphe $G(V, E)$ obtenu en appliquant la méthode d'orientation de la première phase.
Soit x le sommet de transition maximale dans le graphe $G^o(V, E^o)$. Sachant que la transition d'un sommet v représente le nombre d'arcs entrants à celui-ci.
$G^o_{sp}(V_{sp}, E^o_{sp})$ est le sous graphe partiel engendré par cette transition, c'est à dire pour tout sommet $v \in V_{sp}$, $(x, v) \in E^o$.

Théorème 2.3.4 $\omega(G) = \max_{v \in G^o_{sp}}(incidence(v)) + 1$ *pour tout graphe [51] :*

1. *biparti ;*
2. 1*-scindé.*

Preuve 4 – ***Cas 1:*** *Un graphe biparti est un graphe partitionné en deux ensembles stables* S_1 *et* S_2.
Soit G *un graphe biparti.* x *étant le sommet de transition maximale et* G^o_{sp} *le graphe engendré par cette dernière. Donc le sommet* x *est soit dans* S_1 *soit dans* S_2. *Ainsi,* $\forall y_i, y_j \in V_{sp}, (x, y_i) \in E^o$ *et* $(x, y_j) \in E^o, i \neq j$ *tel que* $(y_i, y_j) \notin E$ *(sont deux à deux disjoints) donc* $\forall y_i \in V_{sp}, d^+(y_i) = 1 \Rightarrow incidence(y_i) = 1 \Rightarrow \omega(G) = 2$.

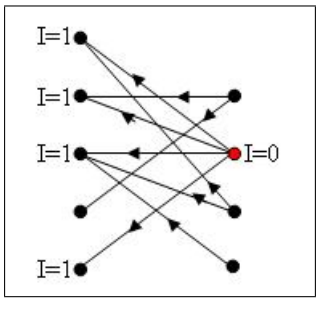

FIGURE 2.8 – Exemple d'orientation d'un graphe biparti

– ***Cas 2:*** *Un graphe* 1*-splite est un graphe scindé en un stable* S *et une clique* C_k, *de taille* k.
Soit G *un graphe* 1*-scindé. Pour tout sommet* $v \in C_k$ *et pour tout sommet* $y \in S : d_G(v) \geq d_G(y) \Rightarrow d_G(y) \leq k - 1$ *sinon* y *est un sommet de la clique* C_k. *S'il existe au moins un sommet* $y \in S$ *et un sommet* $v \in C_k \mid d_G(y) = d_G(v)$ *alors le graphe* G *contient au moins deux cliques de même taille ; chacune est formée par l'un des sommets* y *ou* v, *donc le sommet* y *peut appartenir au stable* S *ou à la clique* C_k, *de même pour le sommet* v. *Néanmoins, si l'un de ces sommets se présente l'autre s'absente de l'ensemble. Supposons qu'il n'existe aucun sommet* y *de* $S \mid d_G(y) = d_G(v)$ *avec* $v \in C_k$.
Il est clair que le sommet de transition maximale est un sommet la clique C_k.
Selon l'algorithme d'orientation, au voisinage de sommet de transition maximale x, *il n'existe aucun sommet* $y \in S$ *voisin à un sommet* $v \in C_k \mid (y, v) \in E^o$. *Dans ce voisinage le sommet d'incidence maximale est un sommet* v' *appartenant à la clique*

C_k et comme ce sommet ne reçoit que des incidences issues des sommets de la clique C_k et d'après le Théorème 2.3.1, $I(v\prime) = k-1 = \max_{v \in G^o_{sp}}(incidence(v))$.

Par conséquent, $\omega(G) = k-1+1 = k$.

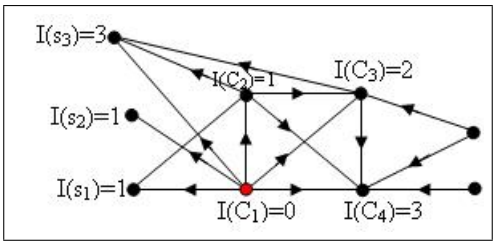

FIGURE 2.9 – Exemple d'orientation d'un graphe 1-splite

Remarque 2 *Une seule étape de l'algorithme d'orientation suffit pour les graphes bi-partis, le raisonnement peut être fait par rapport au premier sommet choisi pouvant être traité. Dans ce cas, le temps d'exécution est d'ordre Δ.*

2.3.2 Résultats sur des graphes non parfaits

La notion des graphes à voisinages scindés (SNAP) a été introduite par Maffray et Preissmann [81].

Théorème 2.3.5 *Pour tout graphe SNAP où le sommet de transition maximale x est un sommet M-scindé, $\omega(G) = \max_{v \in G^o_{sp}}(incidence(v)) + 1$ où G^o_{sp} est le graphe de transition induit par le sommet x [51].*

Preuve 5 *Soit $G(V,E)$ un graphe SNAP et le sommet x étant M-scindé tel que x est le sommet de transition maximale. Donc le sous graphe partiel $G^o_{sp}(V, E^o_{sp})$ induit par la transition de x peut être scindé en une clique maximum et un stable $S \mid V_{sp} = S \bigcup C_k$. Même démonstration que celle de 1-split (1-scindé) tout en considérant G^o_{sp} comme étant le graphe G^o.*

Un graphe 2-scindés (2-split graph) est un graphe $G(V,E)$ dont l'ensemble des sommets peuvent admettre une partition (S_1, S_2, C_1, C_2), où S_1 et S_2 sont des stables et C_1 et C_2 sont des cliques.

Théorème 2.3.6 *Pour tout graphe $2-$scindé dont $N(x)$ est une clique et peut être un stable : $\omega(G) = \max_{v \in G^o_{sp}-x}(incidence(v)) + 2$. Tel que le sommet d'incidence maximale x est un sommet de $C_{\max}$ [51].*

Preuve 6 *$N(x)$ est une clique C_k et peut être un stable S. Il y a deux cas possibles :*

Cas 1 :

(S n'existe pas)

$S = \{\emptyset\} \Rightarrow$ le voisinage du sommet x est la clique $C_k \Rightarrow I(x) = k-1$. En retirant le sommet x de G^o_{sp}, $G^o_{sp} - \{x\}$ est une clique de cardinalité $k-1 \Rightarrow \max_{v \in G^o_{sp}-\{x\}}(I(v)) = k-2$ (Voir Théorème 2.3.1), donc $\omega(G) = ((k-2)+1)+1 = k$.

Cas 2 :

(S existe)
$S \neq \{\emptyset\} \Rightarrow \forall y_i \in S, (y_i, x) \in E^o \Rightarrow d^+_{G^o_{sp}}(y_i) = 0$. *En supprimant le sommet* x, $G^o_{sp} - \{x\}$ *est une clique de cardinalité* $k-1$ *et un stable ce qui implique* $\max_{v \in G^o_{sp} - \{x\}}(I(v)) = k-2$ *(voir le Théorème* 2.3.1*). Ainsi,* $\omega(G) = ((k-2)+1)+1 = k$.

Théorème 2.3.7 *Étant donné un sommet* v *d'incidence maximale. Pour tout graphe* 2*-scindé dont le voisinage est une clique,* $\omega(G) = incidence(v) + 1$ *[51].*

Preuve 7 *Comme* v *est un sommet d'incidence maximale et* $N(v)$ *une clique donc* v *n'est voisin à aucun sommet* y *d'un stable. Ainsi, le sommet* v *ne reçoit que des incidences issues des sommets de la clique. D'après le Théorème* 2.3.2*, la taille de cette dernière est* $I(v)+1$ *et comme* v *est le sommet d'incidence maximale alors* $\omega(G) = I(v)+1$.

Soit x est le sommet de transition maximale qui appartient à la clique maximum et $G^o_{sp}(V_{sp}, E^o_{sp})$ est le sous graphe partiel induit par $\Gamma^+(x)$.

Proposition 2.3.1 *Pour tout graphe* 2*-scindé dont pour tout sommet* $v \in V_{sp}$*,* v *est scindé à une clique et peut être un stable,* $\omega(G) = \max_{v \in G^o_{sp}}(incidence(v)) + 1$ *[51].*

Preuve 8 *Ce résultat se démontre comme le Théorème* 2.3.4 *cas* 2.

Soit $G^o(V, A)$ le graphe orienté associé à $G(V, E)$ obtenu à l'issue de la phase 1.

2.4 Nouvelle heuristique pour la détermination de la clique maximum

notre algorithme d'orientation nous permet de proposer une nouvelle méthode pour le problème de la clique maximum.

Algorithm 11 : Algorithme de la clique maximum

Entrées : $G^o(V, E^o)$

Sorties : $\omega(G)$

DEBUT

Repérer le sommet d'incidence maximale, soit x tel que $\forall v \in V$, $d^+_{G^o}(x) \geq d^+_{G^o}(v)$;

2: Générer le sous graphe partiel engendré par l'incidence de x, soit $G^o_{sp}(V_{sp}, E^o_{sp})$;
Déterminer le sommet de transition maximale dans ce voisinage, il s'agit de G^o_{sp}, soit y tel que $T(y) = \max_{v \in V_{sp}}(T(v))$

4: $\omega(G) \leftarrow T(y) + 1$

FIN.

La clique maximum est générée par les voisins du sommet x ayant une transition descendante du sommet y.

Corollaire 2.4.1 *L'heuristique présentée fournie une solution optimale pour la classe des graphes bipartis.*

Preuve 9 *Soit G un graphe biparti, x un sommet d'incidence maximum et $G^o_{sp}(V_{sp}, E^o_{sp})$ le sous graphe partiel engendré par cette incidence.*

Le sommet x ne reçoit que des incidences d'un ensemble stable donc $\forall y \in V_{sp}$, $T(y) = 1 \Rightarrow \omega(G) = 1 + 1 = 2$.

Chapitre 3

Approche de contraction et encadrement du nombre de clique

Sommaire

3.1 Introduction

La recherche de la plus grande clique ou sa cardinalité dans un graphe est un problème réputé difficile de la théorie des graphes, avec de nombreux domaines d'applications. Les algorithmes existants dans la littérature peuvent être efficaces pour certaines classes de graphes ou pour des graphes de taille raisonnable. Notre motivation de ce travail se situe dans la même optique que celle qui a motivé nos travaux développés dans [43] ainsi que d'autres [14, 39, 41, 66, 70, 75, 86, 98] et [101].
Dans ce chapitre nous caractérisons un encadrement de ω ; un majorant $\overline{\omega}$ et un minorant $\underline{\omega}$ chacun est une fonction de l'application I, où I représente l'incidence associée à chaque sommets. En fait, cette caractérisation découle des résultats obtenus dans [51], il s'agit de la méthode d'orientation et du schéma d'ordre d'étiquetage. Ainsi, pour trouver la clique maximum et sa cardinalité nous avons opté pour le développement d'une nouvelle approche de contraction. Grâce à cette approche nous avons proposé un minorant pour le nombre de clique qui a été exploité afin d'établir une nouvelle méthode énumérative partielle. Cette dernière consiste à énumérer certaines cliques maximales jusqu'à une itération donnée afin d'en tirer la clique maximum.

3.2 Problèmes liés à la recherche de la clique maximum

Un graphe peut être utilisé pour modéliser des relations d'incompatibilité, de divergence, d'affinité ou de domination entre des personnes ou des objets. De nombreux problèmes pratiques qui apparaissent dans les domaines les plus divers peuvent être vus comme étant le problème de la recherche de la clique maximum. Nous en citons :

3.2.1 Analyse de la transmission de signaux

Une des modélisations de ce problème peut se faire par le problème de la clique maximum et ceci tout en considérant les sommets comme étant les signaux et les arêtes comme étant la relation entre les signaux pouvant être distingués naturellement l'un de l'autre [6] et [9].
En effet, trouver le plus grand ensemble de signaux clairement distinguables revient à la recherche de la plus grande clique.

3.2.2 Confection d'emploi du temps

Le problème de la clique associé au problème d'emploi du temps est le suivant : la cardinalité de la clique maximale du graphe de conflits d'un ensemble d'activités est une borne inférieure sur le nombre de périodes nécessaires pour ordonnancer ces activités [6]. Cependant, l'ensemble indépendant maximal représente une borne supérieure du nombre d'activités qui peuvent se dérouler simultanément.

3.2.3 Dessin expérimental

Selon le problème de la clique ce problème peut être modélisé de la manière suivante : les sommets peuvent être vus comme étant les sous-ensembles de taille n d'un ensemble de traitements d'une expérience statistique et les arêtes comme étant la relation entre k éléments étant en commun [6]. Déterminer la collection maximale de sous-ensembles ayant k élément en commun revient à déterminer la clique maximum du graphe le modélisant.

3.2.4 Simplification des machines incomplètement spécifiées

La modélisation adoptée à ce problème via le problème de la clique maximum est vu comme suit : Les sommets du graphe représentent les états des machines telles que deux sommets sont reliés par une arête si et seulement si les états associés sont compatibles. Ainsi, deux états sont compatibles si pour toute séquence d'entrée applicable aux deux états, la séquence de sortie est identique pour chacun d'entre eux. En effet, simplifier la machine revient à chercher le plus grand ensemble d'états compatibles pouvant se traduire par le problème de la clique maximum dans le graphe le modélisant.

3.2.5 Système de recherche d'informations

Aussi ce problème peut être considéré comme étant le problème de la clique maximum de sorte que les sommets du graphe associé sont les informations mémorisées et les arêtes sont les interrelations entre celles-ci. Ainsi, la collecte du nombre maximum d'informations entièrement liées entre elles et vu comme étant la recherche de la plus grande clique dans le graphe associé [6].

3.2.6 Diagnostic d'erreurs

Parmi les problèmes les plus importants lors de l'étude de comportement des systèmes multiprocesseurs figure celui de la tolérance aux erreurs, il consiste à identifier les processeurs qui sont en erreur dans le système. Ce problème peut se modéliser et se comprendre comme étant le problème de la clique maximum.

Nous proposons dans ce qui suit une approche polynomiale d'encadrement pour la caractérisation du majorant, $\overline{\omega}$, et du minorant, $\underline{\omega}$, nous permettant d'exhiber un algorithme exact pour le problème de la clique maximum.

3.3 Majorant de ω

Le théorème suivant propose un majorant de la valeur du nombre maximum de clique dans un graphe en utilisant la méthode d'orientation et le Théorème 3.2.1 du chapitre 2 et qui sont présentés dans [51]. Ce dernier stipule que toute orientation d'une clique C de taille k définit un schéma d'ordre d'étiquetage sans redondance, donc il existe un seul somment $v \in C$ d'incidence $I_C(v) = k - 1$. Pour la bonne compréhension de notre

méthode nous adoptons la notation suivante : $G^o(V, A)$ dénote le graphe orienté associé à $G(V, E)$ en orientant les arêtes par l'algorithme d'orientation.

Théorème 3.3.1 *Pour tout graphe G, $\omega \leq \max_{x \in G^o}(incidence(x)) + 1 = \overline{\omega}$.*

Preuve 10 *Nous démontrons par l'absurde. Nous supposons que G possède une clique de cardinalité maximum k tel que : $\omega = k > \max_{x \in G^o}(incidence(x)) + 1$. Selon l'algorithme d'orientation et le Théorème* 3.2.1 *de chapitre précédent, en se restreignant à la clique C_k il existe un sommet v dans C_k tel que : $I_C(v) = k - 1$ donc dans le graphe G^o, $I_{G^o}(v) = k - 1 + a$ avec $a \geq 0$ dont a correspond au nombre d'arcs entrant issus des sommets de $G - C$, $I_G(v)$ correspond à l'incidence du sommet v dans le graphe G et $I_C(v)$ correspond à l'incidence du sommet v dans le sous graphe partiel restreint à la clique C. On a : $\max_{x \in G^o}(incidence(x)) \geq I_{G^o}(v) = k - 1 + a | v$ est un sommet de la clique C_k, ceci implique que $\max_{x \in G^o}(incidence(x)) + 1 \geq I_{G^o}(v) + 1 = k - 1 + 1 + a \Rightarrow \omega = k > \max_{x \in G^o}(incidence(x)) + 1 \geq I_{G^o}(v) + 1 = k + a | a \geq 0 \Rightarrow \omega = k > k + a | a \geq 0$ ce qui est absurde.*

3.4 Contraction

De nombreux travaux ont été menés sur la contraction dans le but d'aboutir à une caractérisation du graphe en question [11, 39, 74, 86] [98]. Dans cette partie, nous utilisons la contraction de façon continue afin de combler tous les doublements générés par les sommets de même incidence.
La contraction utilisée dans ce texte consiste à fusionner les sommets ayant le même degré d'incidence dans le but de diminuer le nombre de sommets jusqu'à l'obtention d'une clique.

Illustration graphique

Le schéma ci-dessous (figure 3.1) illustre une étape de la contraction.

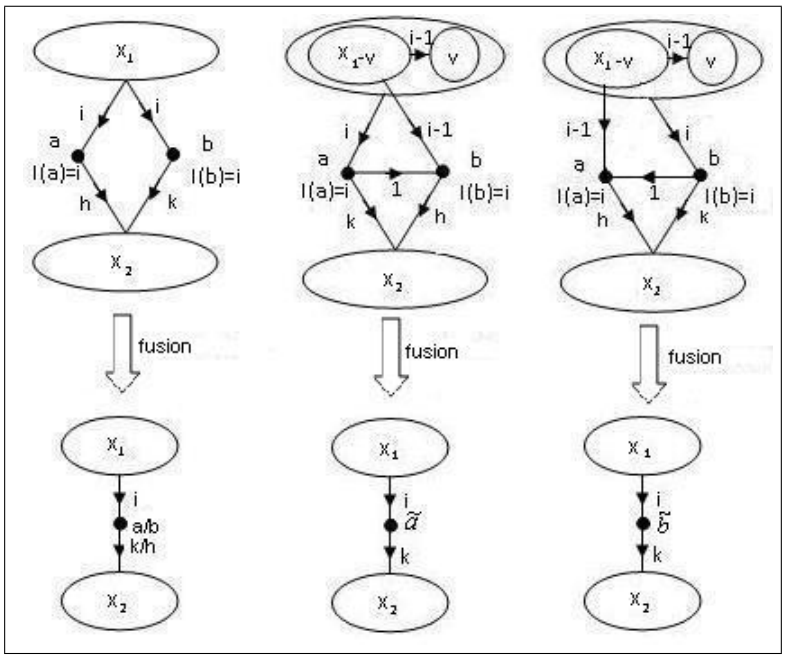

FIGURE 3.1 – Illustration de la contraction : fusion de deux sommets de même incidence

x étant le sommet de transition maximum dans le graphe G^o et G^o_{sp} le sous graphe partiel induit par cette transition, donc $I(x) = 0$.
y le sommet d'incidence maximum dans le graphe G^o_{sp} n'ayant aucune transition et $G^o_{sp}\{x \to y\}$ le sous graphe partiel induit par cette incidence, donc $I(y) = |V_{sp}\{x \to y\}|-1$.

Proposition 3.4.1 *Soit G un graphe, $G^o_{sp}\{x \to y\}(Vsp\{x \to y\}, A_{sp}\{x \to y\})$ est le graphe de transition par rapport à l'incidence obtenu à partir du graphe G et soit I l'application qui associe à chaque sommet son degré d'incidence.*

1. *Si l'application I définit un schéma d'ordre d'étiquetage alors $G^o_{sp}\{x \to y\}$ est une clique.*
2. *Si l'application I ne définit pas un schéma d'ordre d'étiquetage alors toute contraction de $G^o_{sp}\{x \to y\}$ induit une clique.*

Preuve 11 *Considérons la ré-numérotation des sommets de $V_{sp}\{x \to y\}$ selon leurs degrés d'incidence. Soit $X = \{x_0 = x, x_1, ..., x_{|V_{sp}\{x \to y\}|-1} = y\}$ et soient x_i et x_{i+1} le premier couple tel que $I(x_i) = I(x_{i+1})$.*
Soit $X_1 = \{x_0, x_1, ..., x_{i-1}\}$, comme $I(x_i) = I(x_{i+1}) = i$ avec i le plus petit indice ayant deux antécédents x_i et x_{i+1} alors la fusion de ces deux sommets diminue d'une unité le degré d'incidence de l'ensemble $X_2 = X - (X_1 \cup \{x_i, x_{i+1}\})$ ayant une incidence issue du sommet x_{i+1} y compris le sommet y. On réitère ce processus sur tout couple de sommets ayant le même degré d'incidence, qui sont nécessairement dans X_2. A l'issue de ce processus toutes les incidences sont différentes. Par conséquent on obtient un schéma d'ordre d'étiquetage, ainsi on retombe sur le premier cas de la proposition.

3.5 Minorant de ω

Tel qu'il résulte de la proposition précédente, le graphe $G^o_{sp}\{x \to y\}$ correspond à une clique qu'on appelle $\widetilde{G}^o_{sp}\{x \to y\}$ obtenue sans ou par contraction et ceci selon le schéma d'ordre d'étiquetage.

Théorème 3.5.1 *Pour tout graphe G, $\omega \geq \widetilde{I}(y) + 1$ tel que $\widetilde{I}$ représente l'incidence du sommet y dans la clique $\widetilde{G}^o_{sp}\{x \to y\}$ associée au graphe $G^o_{sp}\{x \to y\}$.*

Preuve 12 *Considérons la clique $\widetilde{G}^o_{sp}\{x \to y\}(\widetilde{V}_{sp}\{x \to y\}, \widetilde{A}_{sp}\{x \to y\})$. D'après le Théorème 3.2.1 du chapitre 2 la cardinalité de $\widetilde{G}^o_{sp}\{x \to y\}$ est $\max\limits_{v \in \widetilde{V}_{sp}\{x \to y\}} (incidence(v))+1$ qui est $\widetilde{I}(y) + 1$, d'où $\omega \geq \widetilde{I}(y) + 1$.*

Illustration graphique

Les schémas ci-dessous montrent le passage d'un graphe G^o à un graphe $\widetilde{G}^o_{sp}\{x \to y\}$ ainsi que le schéma d'ordre associé.

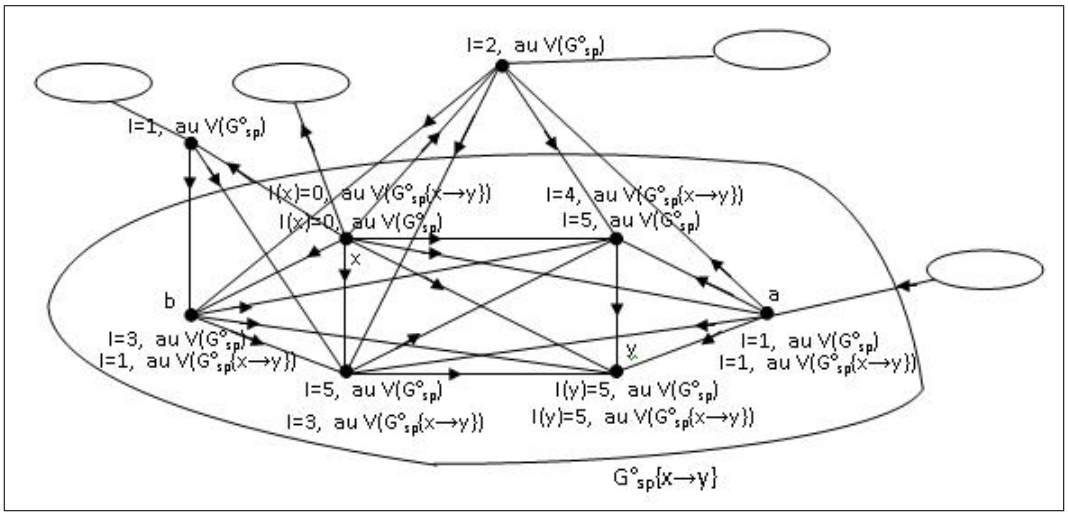

FIGURE 3.2 – Schéma 1 : G^o représente le graphe orienté par notre algorithme d'orientation

L'ordre obtenu dans $G^o_{sp}\{x \to y\}$ est : $0, 1, 1, 3, 4, 5$, l'ordre 1 se répète et l'ordre 2 ne figure pas. Le graphe obtenu après contraction, $\widetilde{G}^o_{sp}\{x \to y\}$, est le graphe présenté dans la figure suivante (figure 3.3) :

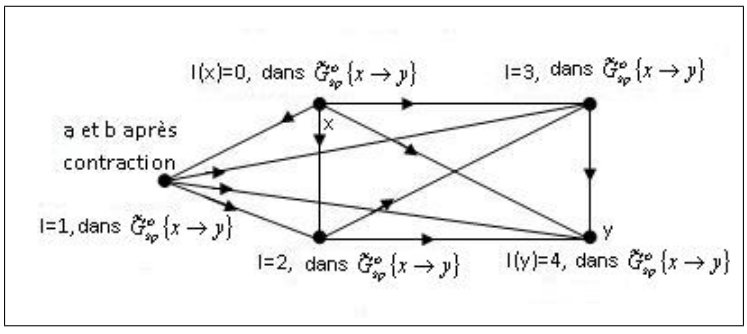

FIGURE 3.3 – Schéma 1 : $\widetilde{G}^o_{sp}\{x \to y\}$ représente le graphe contracté obtenu à partir du graphe G^o

L'ordre associé à ce graphe (figure 3.3) est : $0, 1, 2, 3, 4$ et $\widetilde{I}(y) = 4$. Cet ordre est un schéma d'ordre d'étiquetage dans le graphe $\widetilde{G}^o_{sp}\{x \to y\}$.

Remarque 3 *Afin de garder la plus grande clique incidente au sommet d'incidence maximale, une paire de sommets de même incidence $\{a, b\}$ dont a reçoit une incidence issue du sommet b est contractée de façon à remplacer le sommet a par le sommet b (voir figure 3.4).*

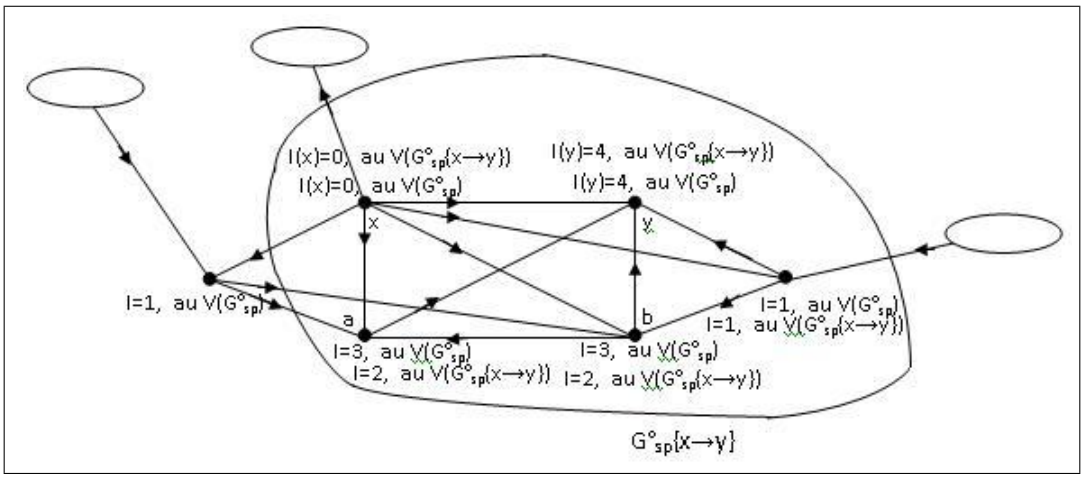

FIGURE 3.4 – Schéma 2 : G^o représente le graphe orienté par notre algorithme d'orientation

L'ordre obtenu dans $G^o_{sp}\{x \to y\}$ est : $0, 1, 2, 2, 4$, l'ordre 2 se répète et l'ordre 3 ne figure pas. Le graphe obtenu après contraction, $\widetilde{G}^o_{sp}\{x \to y\}$, est le graphe présenté dans la figure suivant (figure 3.5) :

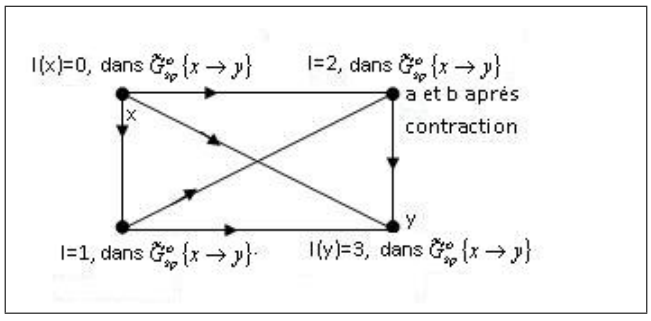

FIGURE 3.5 – Schéma 2 : $\widetilde{G}^o_{sp}\{x \to y\}$ représente le graphe contracté obtenu à partir du graphe G^o

L'ordre associé à ce graphe (figure 3.5) est : $0, 1, 2, 3$ et $\widetilde{I}(y) = 3$. Cet ordre est un schéma d'ordre d'étiquetage dans le graphe $\widetilde{G}^o_{sp}\{x \to y\}$.

3.5.1 Le minorant et ses extensions sur les graphes parfaits

Nous avons montré dans le chapitre précédent que pour tout graphe biparti et 1-scindé $\omega(G) = \underset{v \in V_{sp}}{max}(incidence(v))+1$ tel que $G^o_{sp}(V_{sp}, E^o_{sp})$ est le graphe de transition par rapport au sommet de transition maximale. Ici dans ce paragraphe et sous cet angle nous exhibons le sous graphe partiel par rapport à l'incidence du graphe de transition et nous faisons le test du schéma d'ordre d'étiquetage avant de passer à la contraction.

Théorème 3.5.2 *Le minorant est atteint pour les deux classes de graphes parfaits : biparti et 1-scindé.*

Preuve 13

1. *Soit x un sommet de transition maximum dans le graphe G dont G est biparti, donc le sommet x est soit dans le stable S_1 soit dans le stable S_2.*
   Supposons qu'il est dans le stable S_1, ainsi tous ses voisins sont dans le stable S_2. Par conséquent, l'incidence maximum dans ce voisinage $G^o_{sp}\{x \to y\}$, est d'ordre 1. De cette façon $G^o_{sp}\{x \to y\}$ est un graphe formé par les deux sommets x et y tel que $I(y) = 1$.
   D'où l'application I définit un schéma d'ordre d'étiquetage sur le graphe $G^o_{sp}\{x \to y\}$ ce qui implique que la taille de la clique incidente est : $I(y) + 1 = 1 + 1 = 2$ (voir Théorème 3.2.2 du chapitre précédent [51]). On en déduit que $\omega(G) = \underline{\omega}(G)$
2. *Soit x un sommet de transition maximum dans le graphe G dont G est 1-scindé. Pour tout sommet $v \in C_k$ et pour tout sommet z du stable S on a : $d_G(v) \geq d_G(z) \Rightarrow d_G(z) \leq k-1$ sinon z est un sommet de la clique C_k. S'il existe au moins un sommet z dans S et un sommet v dans C_k tel que $d_G(y) = d_G(v)$ alors le graphe G contient au moins deux cliques de même taille ; chacune est formée par l'un des sommets donc le sommet z peut appartenir au stable S ou à la clique C_k, respectivement le sommet v. Néanmoins, si l'un de ces sommets se présente l'autre s'absente de l'ensemble. Supposons qu'il n'existe aucun sommet z de S tel que $d_G(z) = d_G(v)$ avec $v \in C_k$.*

Il est clair que le sommet de transition maximum est un sommet de la clique C_k. Selon l'algorithme d'orientation, au voisinage de sommet de transition maximum x, il n'existe aucun sommet $z \in S$ voisin à un sommet v de la clique C_k tel que $(z, v) \in E^o$. Dans ce voisinage le sommet d'incidence maximum est un sommet y appartenant à la clique C_k et ne recevant que des incidences issues des sommets de la clique C_k. Donc le graphe $G^o_{sp}\{x \to y\}$ est la clique C_k.
Par conséquent, l'application I définit un schéma d'ordre d'étiquetage sur le graphe $G^o_{sp}\{x \to y\}$ tel que $I(y) = k - 1$ donc $\omega(G) \geq I(y) + 1 = k - 1 + 1 = k \Rightarrow \omega(G) = \underline{\omega}(G)$.

Remarque 4 *Le majorant est atteint si le sommet d'incidence maximum ne reçoit que des incidences issues d'une clique, le sous graphe partiel induit par l'incidence de ce sommet forme une clique. D'après le Théorème* 3.2.2 *du chapitre précédent [51] la taille de la clique est d'ordre I*

3.5.2 Analyse des écarts

L'analyse d'écart de la cardinalité de la clique maximum permet d'évaluer la qualité des bornes proposées. En effet, nous passons en revue dans les lignes qui suivent le calcul de l'écart entre les deux limites $\overline{\omega}(G)$ et $\underline{\omega}(G)$ correspondant au majorant et minorant et ceci pour les deux classes des graphes scindé et biparti.

3.5.2.1 Graphes 1-scindé

Pour les graphes 1-scindé, le sommet x d'incidence maximale est un sommet de la clique maximum C_k (voir preuve du Théorème 2.3.4 du chapitre 2 dans [51]), donc $\max_{v \in G^o}(I(v)) = k - 1 + b$ tel que b représente les incidences extérieures issues du stable S étant non voisin au sommet initial de transition maximale. Par conséquent $\overline{\omega} = k - 1 + 1 + b = k + b$ ce qui donne $\overline{\omega} = \omega + b$ cela implique que $\overline{\omega} \leq \omega + \frac{|S|}{2}$, d'après le Théorème 3.5.2 on a $\underline{\omega} = \omega$ donc $\overline{\omega} - \underline{\omega} \leq \frac{|S|}{2}$.
On en déduit que l'écart entre le majorant et le minorant pour la classe 1-scindé est d'ordre $\frac{|S|}{2}$.

3.5.2.2 Graphes bipartis

Soit G un graphe biparti, pour un sommet v de G^o on a $\max_{v \in G^o}(I(v)) \leq \min(|S_1|, |S_2|) \Rightarrow \overline{\omega} \leq \min(|S_1|, |S_2|)+1$, d'après le Théorème 3.5.2 on a $\underline{\omega} = \omega$ donc $\overline{\omega} - \underline{\omega} \leq \min(|S_1|, |S_2|) + 1 - \omega$. Étant donné que G est biparti, $\omega \in \{1, 2\}$ ainsi,
$\overline{\omega} - \underline{\omega} \leq \min(|S_1|, |S_2|) - 1$ ou bien $\overline{\omega} - \underline{\omega} \leq \min(|S_1|, |S_2|)$.

3.6 Résultats numériques

Nous présentons dans ce qui suit une étude expérimentale et numérique afin de comparer le minorant obtenu par notre méthode avec ω, et ceci pour des graphes où la

taille de la clique maximum est connue. Les tableaux ci-dessous résument les résultats expérimentaux sur 50 instances de la bibliothèque DIMACS pour lesquelles la solution optimale est connue. L'écart en pourcentage est calculé comme suit : $\frac{100*(\omega-\underline{\omega})}{\omega}$.

n, m, d et $\underline{\omega}$ désignent respectivement le nombre de sommets, le nombre d'arêtes, la densité et le minorant obtenu par notre méthode.

TABLE 3.1 – Comparaison de minorant obtenu par notre méthode avec le nombre de clique pour certaines classes de graphes

Instance	n(sommets)	m(arrêtes)	d(densité)	ω	$\underline{\omega}$	Gap(%)
c-fat500-1	500	4459	0,03	14	**14**	0,00
c-fat500-2	500	9139	0,07	26	**26**	0,00
c-fat200-1	200	1534	0,07	12	**12**	0,00
c-fat200-2	200	3235	0,16	24	**24**	0,00
c-fat500-5	500	23191	0,18	64	**64**	0,00
c-fat500-10	500	46627	0,37	126	**126**	0,00
c-fat200-5	200	8473	0,42	58	**58**	0,00
myciel7	191	2360	0,13	2	**2**	0,00
myciel6	95	755	0,16	2	**2**	0,00
myciel5	47	263	0,24	2	**2**	0,00
myciel4	23	71	0,28	2	**2**	0,00
myciel3	11	20	0,36	2	**2**	0,00
myciel2	5	5	0,50	2	**2**	0,00
brock200-2	200	9876	0,49	12	8	33,33
brock200-3	200	12048	0,60	15	11	26,66
brock800-4	800	207643	0,65	26	17	34,61
brock800-2	800	208166	0,65	24	16	33,33
brock200-4	200	13089	0,65	17	13	23,52
brock200-1	200	14834	0,74	21	16	23,81
brock400-3	400	59681	0,74	31	22	29,03
brock400-1	400	59723	0,74	27	18	33,33
brock400-2	400	59786	0,74	29	21	27,58
san400-51	400	39900	0,50	13	**13**	0,00
san200-72	200	13930	0,70	18	**18**	0,00
san400-73	400	55860	0,70	22	**22**	0,00
san400-71	400	55860	0,70	40	**40**	0,00
san200-71	200	13930	0,70	30	26	13,33
san400-72	400	55860	0,70	30	25	16,66
san400-91	400	71820	0,90	100	87	13,00
san200-92	200	17910	0,90	60	47	21,66
san200-91	200	17910	0,90	70	51	27,14
san200-93	200	17910	0,90	44	30	31,81

keller4	171	9435	0,64	11	9	18,18
keller5	776	225990	0,75	27	19	29,63
queen7-7	49	952	0,81	7	6	14,28
queen6-6	36	580	0,92	6	**6**	0,00
gen200-55	200	17910	0,90	55	43	21,81
gen400-55	400	71820	0,90	55	42	23,63
gen200-44	200	17910	0,90	44	32	27,27
gen400-75	400	71820	0,90	75	52	30,66
gen400-65	400	71820	0,90	65	41	36,92
MANN-a9	45	918	0,92	16	14	12,50
MANN-a27	378	70551	0,99	126	119	5,55
MANN-a45	1035	533115	0,99	345	331	4,05
C125.9	125	6963	0,89	34	28	17,64
C250.9	250	27984	0,89	44	**44**	0,00
C2000.9	2000	1799532	0,90	77	60	22,07
C500.9	500	112332	0,90	57	51	10,52
C1000.9	1000	450079	0,90	68	52	22,07

Le minorant est atteint pour toutes les instances de deux familles c-fat et myciel de la bibliothèque DIMACS.

Les résultats numériques montrent que le minorant est atteint pour 19 instances de la littérature. A la valeur optimale, l'écart moyen est d'environ $22,85$.

3.7 Méthode d'énumération partielle

Notre méthode est un schéma d'énumération qui suit les principales stratégies d'un Branch & Bound. La procédure de branchement s'articule sur le choix du sommet de transition maximale tout en exploitant le Théorème 3.5.1 pour construire certaines cliques partielles $\widetilde{G_1}, \widetilde{G_2}, ..., \widetilde{G_k}$ et générer les valeurs : $\tilde{\omega_1}, \tilde{\omega_2}, ..., \tilde{\omega_k}$ associées.

3.7.1 Évaluation

Considérons $\widetilde{G}^o_{sp_i}\{x \to y\}$ le graphe obtenu à partir du graphe $G^o_{sp_i}\{x \to y\}$ à l'issue de la proposition précédente, x étant le sommet de transition maximale dans G^o_i et y le sommet d'incidence maximale dans le graphe $G^o_{sp_i}\{x\}$ à l'étape i. La fonction d'évaluation $\widetilde{I}_i(y)$ consiste à calculer la taille de la clique $\tilde{\omega}_i$ donnée par $\widetilde{G}^o_{sp_i}\{x \to y\}$.

3.7.2 Phase de branchement

En partant du sommet de transition maximale $x = x_0$, fixons le branchement à chaque étape $i | 1 \leq i \leq k$ sur un élément de la suite ordonnée, selon l'ordre décroissant de leurs degrés de transitions maximales, des sommets distincts : $x_1, x_2, ..., x_k$ et générons les sous graphes partiels $G^o_{sp_i}\{x \to y\}$ tel que k désigne l'itération à laquelle le critère d'arrêt est vérifié. Ainsi, à chaque étape i de branchement, on effectue $(n-i)^2$ comparaisons pour la détermination du sommet de transition maximale, $(|V_{sp_i}(x)| - 1)^2$ pour la recherche du sommet d'incidence maximale dans $G^o_{sp}(x)$ et au pire des cas $|\widetilde{V}_{sp_i}\{x \to y\}| - l$ tels que $n = |V|$ et l représentent la position du premier couple ayant la même incidence dans le schéma d'ordre d'étiquetage.

3.7.3 Test d'arrêt

Le processus s'arrête à l'itération k à laquelle la transition maximale du sommet suivant, étant x_{k+1}, est inférieure à $\tilde{\omega_k}$ tel que $\tilde{\omega_k}$ représente la plus grande valeur dans l'ensemble des $\tilde{\omega_i}$.

3.7.4 Identification du test d'arrêt

Soit x_{k+1} le sommet de transition maximale à l'itération $k+1$, ce dernier est voisin à $T(x_{k+1}) - 1$ sommets donc la taille de la clique qui pourrait être incidente ne peut pas dépasser $T(x_{k+1}) + 1$. Supposons que $\tilde{\omega}_k > T(x_{k+1})$ et que $\tilde{\omega}$ n'est pas maximum alors il existe une clique de taille $\tilde{\omega}_{k+1}$ tel que $\tilde{\omega}_k < \tilde{\omega}_{k+1}$. On a $\tilde{\omega}_k > T(x_{k+1})$ sachant que la plus grande clique qui pourrait y être incidente est de taille $\tilde{\omega}_{k+1} = T(x_{k+1}) + 1$ donc $\tilde{\omega}_k \geq T(x_{k+1}) + 1 \Rightarrow \tilde{\omega}_k \geq \tilde{\omega}_{k+1}$ ce qui est absurde.
La formulation algorithmique de cette approche est présentée dans le paragraphe suivant.

3.7.5 Formulation algorithmique

Algorithm 12 : Algorithme d'Enumération Partiel

Entrées : $G^o(V, A)$

Sorties : La taille de la clique maximum $\omega(G)$ et la clique correspondante K_ω

Variables utilisées :

$x(\max_{v\in V}(T(v))$: le sommet de transition maximale dans l'ensemble V ;

$y(\max_{v\in V}(I(v))$: le sommet d'incidence maximale dans l'ensemble V ;

T : L'ensemble des sommets traités ;

$G^o_{sp}\{x\}(V_1, A_1)$: Le graphe de transition par rapport au sommet x ;

$G^o_{sp}\{x \to y\}(V_2, A_2)$: Le graphe de transition par rapport l'incidence du sommet y ;

contract(V_2, A_2) : Fonction qui retourne $(\widetilde{V}_2, \widetilde{A}_2)$ par contraction des sommets de même incidence à l'issue de la proposition de la section 4 ;

$\tilde{K}$: La clique partielle retournée par la fonction contract(V_2, A_2) et $\tilde{\omega}$ sa taille ;

DEBUT

1: $x \leftarrow x(\max_{v\in V}(T(v))$;$stop \leftarrow faux$; $T \leftarrow \emptyset$; $K_\omega \leftarrow \emptyset$; $\omega \leftarrow 0$;

2: **Tantque** $stop = faux$ **Faire**

3: $T \leftarrow T \cup \{x\}$;$V_1 \leftarrow \{x\}$;$A_1 \leftarrow \emptyset$;$V_2 \leftarrow \emptyset$; $A_2 \leftarrow \emptyset$;

4: - Pour tout sommet $v \in V | (x, v) \in A$ faire $V_1 \leftarrow V_1 \cup \{v\}$; $A_1 \leftarrow A_1 \cup (x, v)$;
$y \leftarrow y(\max_{v\in V_1}(I(v))$; $V_2 \leftarrow \{y\}$;

5: - Pour tout sommet $v \in V_1 | (v, y) \in A_1$ faire $V_2 \leftarrow V_2 \cup \{v\}$; $A_2 \leftarrow A_2 \cup (v, y)$;

6: - $(\widetilde{V}_2, \widetilde{A}_2) \leftarrow$contract$(V_2, A_2)$; $\tilde{\omega} \leftarrow |\widetilde{A}_2|$; $\tilde{K} \leftarrow \widetilde{A}_2$;

7: **Si** $\tilde{\omega} > \omega$ **Alors**

8: $\omega \leftarrow \tilde{\omega}$; $K_\omega \leftarrow \tilde{K}$;

9: **Finsi**

$x \leftarrow x(\max_{v\in V-T}(T(v))$; $V_1 \leftarrow \emptyset$;

10: **Si** $\omega > \max_{v\in V-T}(T(v))$ **Alors**

11: $stop = vrai$;

12: **Finsi**

13: **Fin tantque**

14: **FIN.**

Chapitre 4

Approche de la coloration des sommets d'un graphe et encadrement du nombre chromatique

Sommaire

4.1 Introduction

De nombreux problèmes peuvent être modélisés sous forme de problème de coloration des sommets d'un graphe. Il s'agit des problèmes se ramenant à la recherche d'une partition d'un ensemble d'objets en sous ensembles ne comportant que des éléments deux à deux compatibles. Ce problème est connu dans le milieu de l'optimisation comme étant un problème difficile [60].

Dans ce chapitre, nous nous focalisons essentiellement sur l'approche de la coloration des sommets d'un graphe, ce dernier a fait l'objet de plusieurs travaux de recherche (voir [16,56,62,68]et [82]) ainsi que l'estimation du nombre chromatique [3,59,63,64,67,69,88,89] et [93]. De nombreux travaux de littérature cherchent à valider la conjecture de Reed [89] sur certaines classes de graphes [3, 63, 64] et [67]. Cette conjecture est formulée comme suit :

Conjecture 4.1.1 *[89]*
Pour tout graphe G,

$$\chi(G) \leq \lceil \frac{\omega(G) + \Delta(G) + 1}{2} \rceil.$$

tel que $\chi(G)$, $\omega(G)$ et respectivement $\Delta(G)$ sont le nombre chromatique, la taille de la clique maximum et le degré maximum du graphe G. La conjecture de Reed a été validée pour certaines classes telles que : la classe des graphes de line graphes de multi graphes [64] et la classe des graphes de quasi-line graphes [63].

Nous proposons un premier majorant du nombre chromatique basé sur l'algorithme d'orientation présentée dans le chapitre 2. Grâce à un nouvel algorithme polynomial de coloration, nous avons amélioré le premier majorant. Nous proposons aussi une comparaison théorique et empirique de notre borne avec celle de Reed pour la classe des graphes sans triangle dans ce cas la conjecture de Reed devient $\chi(G) \leq \lceil \frac{\Delta}{2} \rceil + 2$.

On dit qu'un graphe G est K-coloriable si $\chi(G) = K$.
Nous rappelons dans cette partie des résultats classiques sur les bornes du nombre chromatique.

- Pour tout graphe G,

$$1 \leq \chi(G) \leq n.$$

- Si tous les sommets du graphe G sont disjoints, c'est à dire $m = 0$, alors $\chi(G) = 1$.
- Si G est biparti ou arbre alors $\chi(G) = 2$.
- Si G est complet alors $\chi(G) = n$.
- Si G contient une clique de taille K alors au moins K couleurs sont nécessaires pour la colorier, ainsi $\chi(G) \geq \omega(G)$.
- Pour tout graphe G de degré maximum $\Delta(G)$, nous avons $\chi(G) \leq \Delta(G) + 1$.

Afin de se comparer à la borne de Reed, notre travail est axé sur l'étude heuristique du problème de la coloration, ce qui nous a conduit au développement d'une nouvelle méthode de coloration d'un graphe quelconque qui est de complexité polynomiale et que nous avons nommé coloration par blocs tout en intégrant la méthode d'orientation présentée dans le chapitre 2. Cette dernière nous a permet de déterminer un majorant pour le nombre

chromatique. L'heuristique ainsi développée sera utilisée aussi pour initialiser la solution de départ de la génération de colonnes afin de réduire le nombre d'itérations.

4.2 Nouvel algorithme de coloration par blocs

Nous présentons un algorithme de coloration polynomial que nous avons nommé coloration par blocs [49], celui-ci procède en deux phases.

1. Phase 1 : Construction d'un graphe orienté ;
2. Phase 2 : Construction des groupement de sommets (blocs) permettant de caractériser une coloration valide.

Principe de la méthode

Notre méthode consiste à classer les sommets du graphe G selon l'ordre croissant de leurs degrés d'incidence afin de construire les ensembles L_i tout en ordonnant les éléments d'un même ensemble suivant l'ordre décroissant de leurs degrés de transition. Ce qui définit un ordre lexicographique sur les sommets du graphe L où, L est la concaténation des sous-ensembles L_i. Elle consiste ensuite à attribuer une couleur valide respectant la contrainte d'adjacence aux sommets de même ensemble L_i, il s'agit des ensembles C_i.

La formulation algorithmique de cette méthode qui représente la phase 2 de la coloration est donnée dans ce qui suit, (l'algorithme 13).

Algorithm 13 Phase 2 : Algorithme de colration par blocs

Entrées : $G^o(V, A)$ un graphe, où A représente l'ensemble des arcs.

Sorties : C, l'ensemble des couleurs C_i.

Début
$k \leftarrow 0$;
$C_k \leftarrow \{L(1)\}$;
Pour $v \leftarrow L(2)$ à $L(n)$ tel que $n = |V|$ **Faire**
$j \leftarrow 0$;
$affect \leftarrow false$;
Tantque $affect = false$ **Faire**
Si $(x, v) \notin A$, $\forall x \in C_j$ **Alors**
$C_j \leftarrow C_j \cup \{v\}$;
$affect \leftarrow true$;
Sinon
Si $j = k$ **Alors**
$k \leftarrow k + 1$, $j \leftarrow k$;
$C_j \leftarrow \{v\}$;
$affect \leftarrow true$;
Finsi
Finsi
$j \leftarrow j + 1$;
Fin tantque
Fin pour
Fin.

A l'issu de l'algorithme de la coloration par blocs $L_0 = \{v \in V$ tel que $I(v) = 0\}$ et $C_0 = L_0$.

La complexité de cette approche dépend de celle de la phase 1 (l'algorithme d'orientation) qui est polynomiale (voir chapitre 2).

La figure ci-dessous illustre un exemple de coloration d'un graphe par notre méthode de coloration par blocs.

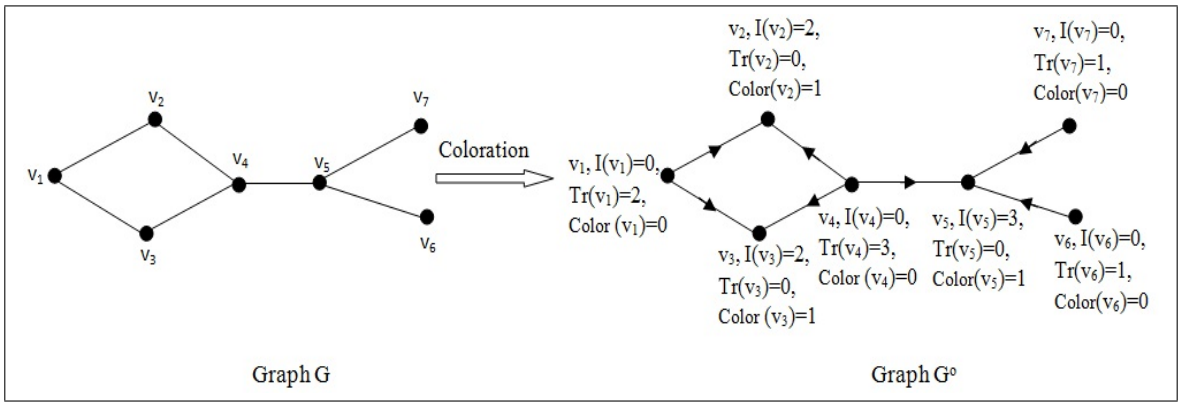

FIGURE 4.1 – Une illustration graphique de notre algorithme de coloration par blocs

Selon l'ordre lexicographique, nous avons $L_0 = \{v_4, v_1, v_6, v_7\}$, $L_1 = \{v_2, v_3\}$, $L_2 = \{v_5\}$ ce qui définit l'ensemble trié L tel que $L \leftarrow L_0 \cup L_1 \cup L_2$, $L = \{v_1, v_4, v_6, v_7, v_2, v_3, v_5\}$. Par ailleurs, l'algorithme de coloration par blocs donne $C_0 = \{v_1, v_4, v_6, v_7\}$ and $C_1 = \{v_2, v_3, v_5\}$.

La méthode conçue donne une coloration propre ne violant aucune contrainte d'adjacence.

Dans cette section, un ensemble de trois lemmes et de trois conséquences, est considéré afin de prouver l'efficacité de notre approche.

Lemme 4.2.1 *Les sommets de l'ensemble L_0 sont deux à deux disjoints [49].*

Preuve 14 *Selon l'algorithme d'orientation, toutes les arêtes incidentes au sommet initial qui peut être traité sont orientées à l'extérieur, donc son incidence est égale à 0. Le sommet suivant qui peut être traité est un sommet non marqué (un sommet marqué est un sommet ayant une incidence d'au moins d'un sommet traité) alors son incidence est aussi égale à 0, et ainsi de suite jusqu'à ce que tous les sommets sont soit marqués soit traités. Il en résulte que l'ensemble des sommets traité qui sont représentés par l'ensemble L_0 sont deux à deux disjoints.*

Conséquence 4.2.1 *$\forall v \in V - L_0, \exists \grave{v} \in L_0$ tel que $(\grave{v}, v) \in A$.*

Lemme 4.2.2 *Si $\exists v_1, v_2 \in V - L_0$ tel que $(v_1, v_2) \in E$ alors soit $T(v_1) > 0$ soit $T(v_2) > 0$ [49].*

Preuve 15 *Selon l'algorithme d'orientation, à une itération k tous les sommets sont soit marqués soit traités. Si $|E| \neq |A|$ alors $\exists$ des arêtes non encore orientées reliant des sommets qui sont marqués. De ce faire, nous considérons le sous graphe partiel engendré par les arêtes non encore orientées. Donc si $\exists v_1, v_2 \in V - L_0$ tel que $(v_1, v_2) \in E$ alors (v_1, v_2) est orientée dans un seul sens, soit $(v_1, v_2) \in A$ soit l'inverse et ceci selon leurs degré dans le graphe G, donc soit $T(v_1) > 0$ soit $T(v_2) > 0$.*

Conséquence 4.2.2 *$\forall v_1, v_2 \in V - L_0$ tel que $T(v_1) = T(v_2) = 0$ alors $(v_1, v_2) \notin E$ donc v_1 et v_2 peuvent avoir la même couleur.*

Lemme 4.2.3 *Tous les sommets d'un ensemble C_i sont deux à deux disjoints [49].*

Preuve 16 *Selon la condition "tant que" de l'algorithme de la coloration, deux sommets voisins de même incidence ne peuvent pas être dans le même ensemble C_i.*

Il en découle la conséquence suivante.

Conséquence 4.2.3 *Chaque bloc de sommets C_i est un ensemble stable.*

Soient $G^o(V, A)$ le graphe obtenu à partir du graphe $G(V, E)$ par l'algorithme d'orientation, C_i les ensembles obtenus à l'issu de l'algorithme de la coloration par blocs et $\chi(G)$ le nombre chromatique du graphe G.

4.3 Deux majorants du nombre chromatique

Dans cette section, nous présentons deux résultats basé sur de nouveaux majorants pour le nombre chromatique $\chi(G)$, et deux autres théorèmes montrant les performances de ces majorants.

Théorème 4.3.1 *Pour tout graphe $G(V,E)$, $\chi(G) \leq \max_{v\in V}(I(v)) + 1$ [49].*

Preuve 17 *Nous distinguons trois cas possibles :*
Cas1 : Supposons que le sommet d'incidence maximale est un sommet d'un anti-trou impair.
Selon l'algorithme d'orientation, en se restreignant aux anti-trous impair de longueur $2k+1$, $\exists$ au moins un sommet $x \in V_{anti-trou} \setminus I(x) = 2k-2$ ainsi, $I_G(x) = 2k-2+a$ où a représente les incidences extérieurs alors $\max_{v\in V}(I(v)) + 1 \geq 1 + I_G(x) = 2k-1+a \geq \chi = k+1$.
Cas2 : Supposons que le sommet d'incidence maximale est un sommet d'un trou impair.
Selon l'algorithme d'orientation, en se restreignant aux trous impair de longueur $2k+1$, $\exists$ au moins un sommet $x \in V_{trou} | I(x) = 2$ ainsi, $I_G(x) = 2+a$ où a représente les incidences extérieurs. Ainsi $\max_{v\in V}(I(v)) \geq I_G(x) = 2+a$ ce qui implique que $\chi = 3 \leq 2+1+a$.
Cas3 : Supposons que G est un graphe de Berge sachant que pour tout graphe de Berge $\omega = \chi$.
D'après le Théorème 3.2.1 du chapitre 2, $\omega \leq \max_{v\in V}(I(v)) + 1$ et comme $\omega = \chi$ donc $\chi \leq \max_{v\in V}(I(v)) + 1$.

Théorème 4.3.2 *Pour tout graphe $G(V,E)$ [49],*

$$\begin{cases} \chi(G) \leq |C| & (1) \\ |C| \leq \max_{v\in V}(I(v)) + 1 & (2) \end{cases}$$

Preuve 18 .
Cas (1) : Il est évident.
Cas (2) : Selon la coloration par blocs, les ensembles L_j sont les ensembles de sommets de même degré d'incidence. Ainsi, $L_0 = \{v \in V \text{ tel que } I(v) = 0\}$.
Soit $\rho = \{j \leq \max_{v\in V}(I(v)); \exists x \in L_j \text{ tel que } \Gamma(x) \cap L_j \neq \emptyset\}$ où $\Gamma(x) = \{y \in V | (y,x) \in E\}$.
Si $\rho = \emptyset$ alors
Les blocs C_j sont construits à partir des ensembles L_{s_i} de la manière suivante :

$$\begin{cases} C_0 = L_0 \\ C_j = L'_j \cup B_j | L'_j = L_j - [(\cup_{i=0}^{j-1} B_i) \cap L_j], \\ \quad B_j = \{x \in \cup_{i=j+1}^{n} L_i | \Gamma(x) \cap L'_j = \emptyset\}; \\ \quad B_0 = \emptyset, n \geq j \geq 1. \end{cases}$$

Où $n = |L| - 1$ et C_j non vide. Donc, $|C| = i + 1 \leq n + 1 = |L|$ ce qui donne $|C| \leq \max_{v\in V}(I(v)) + 1$.

Si $\rho \neq \emptyset$ *alors*
On pose $h = \min\{j \in \rho\}$.
Il est claire que $|\rho| > 0$. *On peut toujours se ramener au cas* $1 \notin \rho$.
On considère l'ensemble $V_1 = \{v \in L_h | I_{L_h}(v) \neq 0\}$.
Pour tout $v \in V_1, I_G(v) = h = I_{L_h}(v) + I_{\cup_{i=0}^{h-1} L_i}(v) + I_{\cup_{i=h+1}^{n} L_i}(v)$. *Ainsi,* $\forall v \in V_1, \exists (h-j)$ *ensembles pouvant l'accueillir or* $(h-j)$ *blocs pouvant être crées.*
D'où $|C| \leq \max_{v \in V}(I(v)) + 1$.

Afin d'évaluer l'efficacité de ces deux majorants, une comparaison formelle avec les bornes classiques est établie ; il s'agit de la conjecture de Reed.

Dans le théorème suivant, un premier majorant du nombre chromatique $\chi(G)$, est fourni en se basant sur l'algorithme d'orientation. En outre, nous montrons les performances de ce majorant et nous fournissons une comparaison avec $\delta + 1$:

Théorème 4.3.3 *Pour tout graphe* $G(V, E)$, $\max_{v \in V}(I(v)) + 1 \leq \Delta + 1$ *[49].*

Preuve 19 *Selon notre algorithme d'orientation [51], le sommet de degré maximum,* Δ, *est le premier sommet qui peut être traité tout en orientant toutes les arêtes incidentes à l'extérieur alors* $\forall v \in V, d(v) \leq \Delta$ *ainsi* $\max_{v \in V}(I(v)) \leq \Delta$ *donc* $\max_{v \in V}(I(v)) + 1 \leq \Delta + 1$.

Reed a introduit une conjecture qui présente une borne, de $\chi(G)$. Dans le théorème suivant, nous présentons un deuxième majorant pour le nombre chromatique $\chi(G)$ basé sur l'algorithme de la coloration par blocs (deuxième phase). Par ailleurs, nous montrons la performance de ce majorant tout en le comparant avec la borne introduite par Reed pour une classe spécifiques des graphes à savoir ; la classe des graphes sans-triangle :

Théorème 4.3.4 *Pour tout graphe* $G(V, E)$ *sans triangle, tel que* $\Delta(G) \leq 4$ *[49] :*

$$\chi(G) \leq |C| \leq \lceil \frac{\Delta}{2} \rceil + 2.$$

Preuve 20
- *Si* $\Delta(G) = 1$ *alors* $|C| = 2 < \lceil \frac{\Delta(G)}{2} \rceil + 2 = 3$ *;*
- *Si* $\Delta(G) = 2$ *alors*
  1. *Pour un graphe de Berge nous avons :* $|C| = 2 < \lfloor \frac{\Delta}{2} \rfloor + 2 = 3$ *;*
  2. *Pour un graphe non Berge nous avons :* $|C| = 3 = \lfloor \frac{\Delta}{2} \rfloor + 2$.
- *Si* $\Delta(G) = 3$ *alors*
  1. *Soit* $|C| = 2 < \lceil \frac{\Delta(G)}{2} \rceil + 2 = 4$ *;*
  2. *Soit* $|C| = 3 < \lceil \frac{\Delta(G)}{2} \rceil + 2 = 4$.
- *Si* $\Delta(G) = 4$ *alors*
  1. *Soit* $|C| = 2 < \lceil \frac{\Delta(G)}{2} \rceil + 2 = 4$ *;*
  2. *Soit* $|C| = 3 < \lceil \frac{\Delta(G)}{2} \rceil + 2 = 4$.
  3. *Soit* $|C| = 4 = \lceil \frac{\Delta(G)}{2} \rceil + 2$ *(cas de graphe de chvàtal).*

Il en résulte que notre borne domine celle de Reed pour $\Delta \leq 4$.

Remarque 5 *Si* $\exists x \in V | d_G(x) = \Delta$ *tel que* Δ *est assez grand et* $\forall v \in V - \{x\}, d_G(v) = 1$ *alors* $|C| = 2 << \lfloor \frac{\Delta}{2} \rfloor + 2$ *;*

En l'absence de preuve formelle, pour $\Delta \geq 5$, les tests numériques confirment la dominance de notre borne par rapport à celle de Kostochk et Reed. De ce fait Nous proposons ci-dessous une comparaison empirique basée sur des instances générées aléatoirement.

Notons que *na*, *da* et *RC* sont respectivement, la moyenne du nombre de sommets, la moyenne de la densité et la borne donné par la conjecture de Reed pour la classe des graphes sans-triangle, qui est $\lceil \frac{\Delta}{2} \rceil + 2$.

TABLE 4.1 – Comparaison entre $|C|$ et *RC* pour des instances de petite tailles.

Instances	Δ	**na**	**da**	**RC**	$\lvert C \rvert$
Instances1	5	8.5	0.413	4	Moyenne= 2.88 $\min = 2; \max = 4$
Instances2	6	9.5	0.387	5	Moyenne= 3.87 $\min = 2; \max = 5$
Instances3	7	10.5	0.358	5	Moyenne= 3.83 $\min = 2; \max = 5$
Instances4	8	11.5	0.325	6	Moyenne= 4.03 $\min = 2; \max = 6$
Instances5	9	12.5	0.332	6	Moyenne= 3.86 $\min = 2; \max = 6$
Instances6	10	13.5	0.321	7	Moyenne= 3.8 $\min = 2; \max = 7$
Instances7	11	14.5	0.302	7	Moyenne= 4 $\min = 2; \max = 7$
Instances8	12	15.5	0.323	8	Moyenne= 4.66 $\min = 2; \max = 8$
Instances9	13	16.5	0.307	8	Moyenne= 4.63 $\min = 2; \max = 8$

Dans le tableau 4.1 nous avons présenté 9 blocs d'instances, chaque ligne de ce tableau est une moyenne de 30 instances. 5 instances sur 30 ayant les mêmes caractéristiques

(même Δ et même nombre de sommets).

Nous présentons dans le tableau 4.2 les résultats des expériences sur des instances quelconques de graphes, sans-triangle, pour lesquelles le nombre de sommets n et Δ sont assez grands.

TABLE 4.2 – Comparaison entre $|C|$ et RC pour des instances de grande tailles.

Instance	**n**	**m**	$\Delta(G)$	**RC**	$\|C\|$	**Instance**	**n**	**m**	$\Delta(G)$	**RC**	$\|C\|$
Instance1	100	235	80	42	9	Instance26	200	860	43	24	22
Instance2	100	230	75	40	40	Instance27	200	1100	39	22	20
Instance3	100	240	85	45	9	Instance28	200	1226	37	21	20
Instance4	100	254	99	52	10	Instance29	200	1400	40	22	21
Instance5	100	368	16	10	10	Instance30	200	1500	43	24	21
Instance6	100	380	20	12	11	Instance31	400	720	280	142	9
Instance7	100	520	30	17	13	Instance32	400	849	390	197	12
Instance8	100	550	36	20	15	Instance33	400	830	360	182	10
Instance9	100	620	33	19	16	Instance34	400	800	320	162	10
Instance10	100	647	39	22	16	Instance35	400	1000	37	21	17
Instance11	150	275	70	37	7	Instance36	400	1500	46	25	22
Instance12	150	280	76	40	7	Instance37	400	3800	76	40	39
Instance13	150	320	100	52	11	Instance38	400	6500	80	42	40
Instance14	150	300	90	47	8	Instance39	400	7000	82	43	40
Instance15	150	400	40	22	13	Instance40	400	7500	88	46	41
Instance16	150	580	33	19	17	Instance41	500	690	310	157	8
Instance17	150	610	38	21	19	Instance42	500	750	380	192	9
Instance18	150	789	41	22	22	Instance43	500	780	380	192	10
Instance19	150	1200	40	22	20	Instance44	500	800	400	202	9
Instance20	150	1330	42	23	20	Instance45	500	810	395	200	10
Instance21	200	206	199	102	5	Instance46	500	824	405	205	10
Instance22	200	600	32	18	16	Instance47	500	5500	76	40	33
Instance23	200	690	35	20	16	Instance48	500	6300	81	43	37
Instance24	200	790	42	23	20	Instance49	500	7800	78	41	38
Instance25	200	800	48	26	20	Instance50	500	8000	85	47	40

Les tests numériques confirment la domination de notre borne face à celle de Reed.

4.4 Adaptation à la construction d'équipes

De nombreux problèmes peuvent être modélisés sous forme de problèmes de coloration de graphes [9, 16, 56, 62, 68], [16] et [82], il s'agit des problèmes se ramenant à la recherche d'une partition d'un ensemble d'objets en sous ensembles ne comportant que des éléments compatibles deux à deux. La NP- complétude de ce problème fait que la solution optimale est rarement atteinte voir quasi- impossible [60]. Au delà du large éventail d'applications pratiques qu'ils permettent de modéliser, à savoir le problème de la construction d'équipes [2], [72] et [92], leur complexité combinatoire rend difficile leur résolution exacte dès lors que la taille des problèmes à traiter est conséquente. En effet, nous considérons dans cette partie une adaptation de notre algorithme de coloration par blocs pour le problème de la construction d'équipes.

Coloration par blocs de cardinalité limitée pour le problème de la construction d'équipes

Plusieurs travaux ont été produits pour tenter la résolution de problème de la construction d'équipes, et ce dans divers domaines, il s'agit des problèmes se ramenant à la recherche d'un sous ensemble de couples formant les affectations optimales ou quasi-optimales tout en respectant les contraintes dures et certaines contraintes de préférence. Ce problème s'est révélé difficile [4] et a suscité un très grand intérêt. Il a été appliquée avec succès dans le domaine de ; la santé [1] et [4], le sport [34], l'éducation [92], militaire [30], etc.

Dans ce travail, nous nous proposons d'apporter une contribution à la résolution de problème de la construction d'équipes par l'approche de la coloration des sommets d'un graphe. Cette approche a été adopter dans la littérature par plusieurs chercheurs, nous en citons [2], [71]et [92]. Ceci en assimilant les contraintes de problème de la construction d'équipes aux contraintes d'adjacence dans la coloration. Les auteurs dans [92] ont utilisé l'approche de la coloration comme étant un outil de modélisation et de résolution de problème de la construction d'équipes. Ils ont considéré ce problème comme étant un graphe pondéré et ils ont utilisé pour sa résolution une heuristique itérative basée sur une méthode de contraction "vertex contraction technique". Tandis que dans [2] et [71], les auteurs se sont basés sur le principe glouton afin de construire une coloration.

Généralement les équipes ont une cardinalité prédéfinie. Ainsi, pour faire face au problème de la construction d'équipes, nous proposons une version modifiée de notre algorithme de coloration par blocs qui demeure polynomiale, où les stables ont des cardinalités fixes (identiques ou non identiques). Nous présentons dans ce qui suit deux versions modifiées de notre algorithme de coloration par blocs. La première consiste à créer des blocs de cardinalité fixe, p. Cependant, la deuxième version consiste à créer des blocs j chacun d'une cardinalité $p[j]$ et ceci selon les spécificités de chaque bloc. Pour ce faire, la condition "$(x, v) \notin A$, $\forall x \in C_j$" dans l'algorithme de la coloration par blocs devient "$(x, v) \notin A$, $\forall x \in C_j$ et $C_j \leq p$ (or $C_j \leq p[j]$)", où $p[j]$ dépend des caractéristiques

du bloc C_j", [49]. Afin d'améliorer la qualité de travail effectuée par chaque équipe et créer des équipes cohérentes ayant des compétences complémentaires, nous faisons classer les ressources dans un ensemble de catégories et ceci en fonction de leurs compétences. Pour ce faire, nous considérons T comme étant l'ensemble de h catégories de ressources T_t ($T = \cup_{t=1}^{h} T_t$). En général, parmi les h catégories, un ensemble de b catégories ($b \leq h$) sont considérés critiques. La spécificité de ces b catégories réside dans la nécessité d'avoir au moins un élément de ces catégories dans chaque équipe, ils peuvent représenter les chefs de département, superviseurs ou autres. Ce qui représente une contrainte dure dans le problème de construction d'équipes (voir la contrainte 2). L'objectif de notre travail est de construire des équipes cohérentes C_i respectant les contraintes suivantes :

1. $\forall C_i : |C_i| \leq p$ (or $|C_i| \leq p[i]$) ; cela signifie que chaque équipe doit avoir au plus p éléments ;
2. $\forall C_i : C_i \cap (\cup_{t=1}^{b} T_t) \neq \emptyset$ où T_t représentent l'ensemble des sommets (éléments) de même catégorie t ; cela signifie que chaque équipe doit avoir au moins un élément de $\cup_{t=1}^{b} T_t$;
3. Le nombre d'équipes, $q \leq \sum_{t=1}^{b} |T_t|$; ceci signifie que le nombre d'équipes est inférieur ou égal au nombre d'éléments dans $\cup_{t=1}^{b} T_t$;
4. Les contraintes dures correspondent aux arêtes de type $(T_t, T_t), t = 1...b$; à savoir, les arêtes entre les sommets de même catégorie ;
5. Les contraintes souples correspondent aux arêtes potentielles, ce sont toutes les arêtes sauf celles de types $(T_t, T_t), t = 1...b$;

Utilisant les contraintes définies précédemment et notre algorithme de coloration par blocs, nous obtenons un algorithme adapté aux stables de cardinalité limitée, il s'agit de la construction d'équipes de cardinalité limitée. Ainsi, à l'issu de notre algorithme la configuration (décomposition) trouvée n'est pas nécessairement complète [1], ceci est dû au fait que deux sommets $v_1, v_2 \in V | (v_1, v_2) \in (\times_{t=1}^{b} T_t), (v_1, v_2) \notin E$ peuvent être dans la même équipe, ce qui peut donner des équipes C_i tel que $\forall v \in C_i, v \notin (\cup_{t=1}^{b} T_t)$. Ainsi, il existe toujours une configuration complète à une permutation près.

Proposition 4.4.1 *Toute configuration initiale, obtenue par la coloration par blocs de cardinalité limitée, est complète à une permutation près, [49].*

Preuve 21 *Supposons que la solution initiale est non complète,*
donc $\exists i \in \{1, ..., k\}$ *tel que* C_i *ne satisfait pas la condition* (2)*, et comme les deux conditions* (3) *et* (4) *sont vérifiées,*
alors $\exists$ *au moins* $j \in \{1, ..., k\}, j \neq i$ *such that* $C_j \cap (\cup_{t=1}^{b} T_t) = b$*, donc, pour* $v \in C_i$ *et* $y \in C_j$ *tel que* $y \in (\cup_{t=1}^{b} T_t)$*,*
Si $(\forall v_1 \in C_j - \{y\}, (v_1, v) \notin E)$ *et* $(\forall v_2 \in C_i - \{v_1\}, (v_2, y) \notin E)$*,*
alors $C_j \leftarrow (C_j - \{y\}) \cup \{v\}$ *et* $C_i \leftarrow (C_i - \{v\}) \cup \{y\}$*,*

1. Une configuration complète est un ensemble d'équipes comportant au moins un sommet $v \in V$ tel que $v \in (\cup_{t=1}^{b} T_t)$

sinon si $\exists h \in \{1, ..., k\}, h \neq i, j$ *tel que pour* $y_2 \in C_h | y_2 \in (\cup_{t=1}^{b} T_t)$, $(\forall v_3, v_1, v_2 \in C_h - \{y_2\}, C_j - \{y\}, C_i - \{v\}, (y, v_2), (v, v_1), (y, v_3) \notin E)$,
alors $C_h \leftarrow (C_h - \{y_2\}) \cup \{y\}$, $C_j \leftarrow (C_j - \{y\}) \cup \{v\}$ *et* $C_i \leftarrow (C_i - \{v\}) \cup \{y_2\}$,
sinon annuler l'arête de préférence entre le sommet v *et l'ensemble* C_j, *et posons* $C_j \leftarrow (C_j - \{y\}) \cup \{v\}$ *et* $C_i \leftarrow (C_i - \{v\}) \cup \{y\}$.

Dans ce qui suit, deux stratégies sont proposées, [49], pour le choix de sommet v qui peut quitter son bloc, afin d'améliorer la cohésion et la performance des équipes :

Stratégie 1 *:* Dans cette stratégie, le sommet qui est favorisé à quitter son bloc est le sommet de plus faible transition car ce sommet a une forte probabilité de ne pas avoir des conflis avec les blocs suivants.

Stratégie 2 *:* Dans cette stratégie, le sommet qui est favorisé à quitter son bloc est celui de plus faible préférence. De ce fait, nous considérons la fonction f comme étant la fonction de préférence qui est définie sur un intervalle fermé. Cette fonction associe à chaque arête de préférence un entier (poids) w, tel que $f(x_i, x_j) = w \geq 1 | (x_i, x_j) \in E$. Il est noté que si le degré (niveau) de conflit est très faible alors $w = 1$. Notre but est de construire des équipes (stables) cohérentes de poids (conflit) faible. Donc, le choix du sommet v qui peut quitter son bloc se fait sur le sommet de plus faible poids de transition.

L'algorithme de la coloration par blocs de cardinalité limitée permet de respecter la cardinalité des équipes tout en réalisant les souhaits exprimés par les employés (sommets). Ainsi, il garantit toutes les conditions à part la condition 2. Donc, à une permutation près l'algorithme génère une solution qui respecte toutes les conditions à part la condition 5, il est noté que la permutation ne dépassera jamais $\sum_{t=1}^{b} |T_t|$.

Notre approche est générique. La stratégie 1 ou la stratégie 2 est sélectionnée pour l'application considérée, dans le cas où la solution trouvée par l'algorithme de coloration par blocs de cardinalité limitée est non complète. Dans ce qui suit, nous présentons une illustration de notre algorithme pour le problème de construction d'équipes dans le domaine de la santé ; la construction d'équipes d'infirmiers dans le service de chirurgie de la Clinique de la Soukra à Tunis (Tunisie).

En s'inspirant de la formulation classique du problème de la coloration, nous proposons une nouvelle formulation mathématique relative à notre heuristique de coloration par blocs.

4.5 Modélisation Mathématique liée au problème de la coloration

La modélisation adoptée dans cette partie est faite à travers un programme mathématique s'inspirant de la contribution de Malagui et Toth [82]. Ceci en assimilant les contraintes de notre modèle aux relations d'adjacence entre les sommets. En effet, il suffit de construire une bijection entre les blocs C_i obtenus à l'issue de notre méthode de coloration et l'ensemble des blocs de couleurs.

4.5.1 Définition des variables

Soit V l'ensemble des sommets du graphe support tel que chaque élément de cet ensemble est en relation avec un et un seul bloc C_h avec $h = \{1, ..., k\}$ où k désigne le nombre de blocs trouvés par notre algorithme à une itération donnée. Selon notre méthode, les blocs C_h représentent l'ensemble des couleurs pouvant être affectées aux sommets du graphe G. En effet, nous attribuons à chaque sommet i le numéro du bloc h où i se trouve. Ainsi, pour chaque sommet i dans le bloc h on associe une variable de décision x_{ih} telle que $x_{ih} \in \{0, 1\}$ où $x_{ih} = 1$ si et seulement si le sommet i appartient au bloc h. Aussi, on fait correspondre la variable de décision y_h aux blocs C_h tel que $y_h = 1$ si le bloc h existe.

4.5.2 Définition des contraintes

Nous définissons la contrainte d'adjacence entre deux sommets dans la coloration par blocs comme suit : deux sommets i et j tels que $(i, j) \in E$ ne peuvent être dans un même bloc h, ainsi $x_{ih} + x_{jh} \leq y_h$.
Chaque sommet i de l'ensemble V correspond à un et un seul bloc h, cette contrainte pour être traduite comme suit : $\sum_{h=1}^{k} x_{ih} = 1$.

4.5.3 Définition de la fonction objectif

La contrainte de la minimisation du nombre de blocs peut être considérée comme étant le nombre chromatique du graphe G. Donc cette contrainte est formulée par $\sum_{i=1}^{n} y_h$.

4.5.4 Modèle mathématique

Il en résulte le modèle suivant :

$$\begin{cases} MinZ & = \sum_{i=1}^{n} y_h, \forall h = 1, ..., k & (1) \\ \sum_{h=1}^{k} x_{ih} & = 1, \forall i = 1, ..., n & (2) \\ x_{ih} + x_{jh} & \leq y_h, (i,j) \in E, \forall h = 1, ..., k & (3) \\ x_{ih} \in \{0,1\} & i \in V, h \in \{1,2,...,k\}, & (4) \\ y_h \in \{0,1\} & h \in \{1,2,...,k\}. & (5) \end{cases}$$

La fonction objectif (1) minimise le nombre de blocs utilisées, ainsi le nombre de couleurs. La contrainte (2) exige qu'un sommet i se trouve dans un seul bloc h, tandis que la contrainte (3) impose que deux sommets adjacents ne peuvent pas être dans le même bloc. Enfin, les contraintes (5) et (6) imposent les variables à être binaires.

4.5.5 Méthode de génération de colonnes

La génération de colonnes est l'une des principales méthodes utilisée pour résoudre des problèmes linéaires de grande taille. La programmation linéaire et la programmation en nombres entiers sont des domaines d'optimisation fortement liés.

Le principe de la méthode consiste à résoudre itérativement une série de problème de taille raisonnable, problème auxiliaire. Dans le but d'obtenir une solution optimale du programme linéaire initial qui est nommé (MP) autrement dit problème maître. Ainsi, le terme génération de colonnes provient de l'ajout d'une colonne (solution du problème auxiliaire) à la matrice des contraintes du problème maître et ceci à chaque itération.

Pour mieux comprendre son déroulement, considérons le programme linéaire (problème maître) de minimisation suivant :

$$\begin{cases} min & \sum_{j=1}^{n} c_j x_j & \\ sc: & \sum_{j=1}^{n} A^j x_j & = a \\ & x_j \geq 0 & j = 1, 2, ..., n. \end{cases}$$

où A^j et a sont des $m-$vecteurs $(m < n)$.

- x_I solution de base réalisable associé à la base I.
- c_I le coût associé.
- $\pi = c_I(A^I)^{-1}$ le multiplicateur du simplexe, où chaque composante π_i est la variable duale associée à la contrainte $A_i x = a_i, i = 1, 2, ..., m$.

Considérons les coûts réduits des variables hors-base :

$$\bar{c}_j = c_j - \pi(A^J) = c_j - c_i(A^I)^{-1}A^j$$

Si

$$min\{\bar{c}_j \setminus j \in \bar{I}\} = \bar{c}_s < 0$$

alors la solution courante peut améliorée en faisant entrer x_s dans la base via un pivotage.

Plus généralement, supposons que les colonnes A^j sont dans un ensemble S contenant les $m-$vecteurs satisfaisant les contraintes. Le sous-problème générateur de colonnes, appelé aussi problème auxiliaire, s'écrit :

$$min\{c_j - \pi A^j \setminus A^j \in S\} = v(\pi)$$

Schéma de résolution

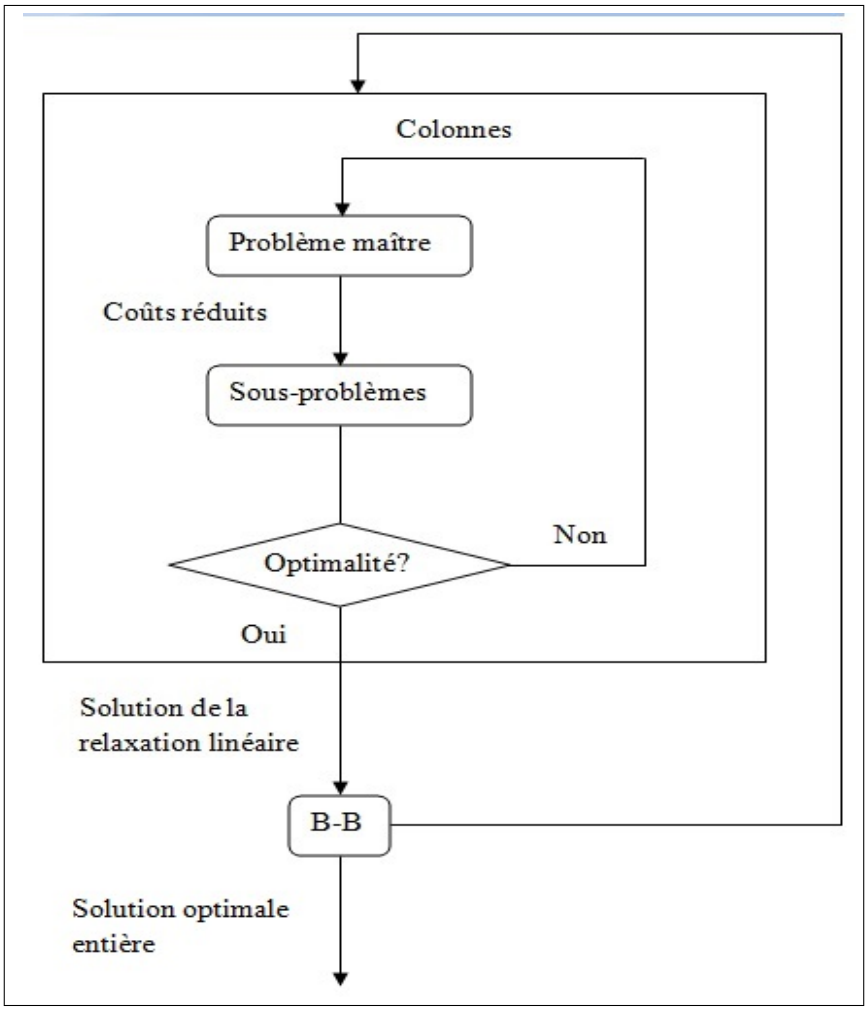

FIGURE 4.2 – Schéma de génération de colonnes

Cas d'une matrice bloc-diagonale

Considérons une matrice de contraintes à p blocs, avec $p \geq 1$:

$$\begin{pmatrix} A \\ B \end{pmatrix} = \begin{pmatrix} A_1 & A_2 & . & . & . & A_p \\ B_1 & 0 & . & . & . & 0 \\ 0 & B_2 & . & . & . & 0 \\ . & & . & & & . \\ . & & & . & & . \\ . & & & & . & . \\ 0 & 0 & . & . & . & B_p \end{pmatrix}$$

Soit alors le programme linéaire (P) réécrit sous la forme :

$$\begin{cases} minz = & c_1x_1 & c_2x_2 & . & . & . & c_px_p & & \\ s.c. & A_1x_1 & A_2x_2 & . & . & . & A_px_p & = & a \\ & B_1x_1 & & & & & & = & b_1 \\ & & B_2x_2 & & & & & = & b_2 \\ & & & & & & & & . \\ & & & . & . & . & & = & . \\ & & & & & & & & . \\ & & & & & & B_px_p & = & b_p \\ & x_1 & x_2 & . & . & . & x_p & \geq & 0 \end{cases}$$

Le sous-problème s'écrit :

$$\begin{cases} min & \sum_{i=1}^{p}(c_i - \pi_i A_i)x_i & & & \\ s.c. & B_ix_i & = & b_i, & i = 1, ..., p \\ & x_i & \geq & O, & i = 1, ..., p \end{cases}$$

Ainsi, pour résoudre le problème maître on obtient p problèmes indépendants (problèmes auxiliaires).

4.6 Le problème de la coloration et la génération de colonnes

Dans notre heuristique de la génération de colonnes, le problème auxiliaire construit les stables, autrement dit les blocs de sommets qui sont deux à deux disjoints. Donc les blocs de sommets pouvant avoir la même couleur.Ce processus permet d'améliorer la qualité des solutions générées.

Nous utilisons notre heuristique de coloration par blocs pour initialiser la solution de départ de la génération de colonnes afin de réduire le nombre de couleurs.

Notre génération de colonnes s'appuie donc, sur la méthode de coloration par blocs.

Soient $S^1, S^2, ..., S^t$ l'ensemble des stables possibles et A^h, $n-$vecteur ou colonne indicatrice représentant le stable S^h. Nous définissons une variable de décision y_h telle que

$y_h = 1$ si le stable S^h est dans la coloration. Le problème maitre (PM) sera défini comme suit :

$$\begin{cases} MinZ & = \sum_{h=1}^{t} y_h \\ sc: & \sum_{h=1}^{t} A^h y_h = 1 \\ y_h \in \{0,1\} & \forall h \in \{1,2,...,t\}. \end{cases}$$

Le problème maitre relaxé (PMR) ainsi obtenu est donné par :

$$\begin{cases} MinZ & = \sum_{h=1}^{t} y_h \\ sc: & \sum_{h=1}^{t} A_i^h y_h = 1 \quad \forall i = 1,...,n \\ & y \in [0,1]. \end{cases}$$

Étant donné que le nombre de stable est potentiellement exponentiel, nous considérons un sous ensemble restreint de stables $S^1, S^2, ..., S^{t_1}$ où $t_1 << t$. A chaque itération le problème maitre relaxé sera enrichit par de nouveaux stables S^j par génération de colonnes.

Le coût réduit d'une variable y_h est donné par la formule : $\bar{c}_h = c_h - \Pi A^h$ ce qui donne $\bar{c}_h = 1 - \sum_{i=1}^{n} \Pi A_i^h$. En effet, le problème auxiliaire détermine un stable tel que $\bar{c}_h$ soit minimum. Ainsi, pour un stable S^{h_0} si $\bar{c}_{h_0} > 0$ alors le stable S^{h_0} est optimale sinon il sera introduit dans le (PMR). Soit x_i une variable de décision tel que $x_i \in \{0,1\}$ où $x_i = 1$ si le stable $i \in S^*$.

le problème auxiliaire est donné par le modèle mathématique suivant :

$$\begin{cases} MinZ & = 1 - \sum_{i=1}^{n} \Pi_i x_i \\ sc: & x_i + x_j \leq 1 \qquad (i,j) \in E \end{cases}$$

Chapitre 5

Approche de la coloration des sommets d'un graphe pour des problèmes pratiques

Sommaire

5.1 Approche de la coloration des sommets d'un graphe et ses applications pour la confection d'horaires

La planification des horaires est un sujet très important et toujours d'actualité. Elle consiste à allouer des ressources données à des objets dans un intervalle de temps, de façon à satisfaire au mieux un ensemble d'objectifs. Un de ces problèmes est celui de la confection d'emploi du temps. Celui-ci consiste à définir un certain nombre d'affectations permettant d'assigner plusieurs ressources humaines, matérielles ou autres sur une période de temps, tout en respectant les contraintes imposées par les entités telles que, la disponibilité des ressources humaines, matérielles, etc.

On se trouve confronté aux problèmes pouvant se définir en termes de contraintes de temps dans divers domaines comme celui de la pédagogie, notamment dans les universités qui consomment de nombreuses ressources humaines. Citons également d'autres domaines classiques tels que le transport, la logistique ou la santé. Dans ce qui suit, nous présenterons quelques exemples des problèmes d'emploi du temps.

5.1.1 Emploi du temps dans le domaine de la santé

Déterminer le planning des médecins et/ou des infirmiers dans un hôpital donné doit permettre de déterminer le nombre minimum de médecins et/ou d'infirmiers nécessaires pour couvrir tous les besoins sur des périodes de temps bien définies. Une journée de travail dans un hôpital est divisée en plusieurs créneaux, vacations, chacun ayant un certain nombre d'heures sachant que les besoins en personnel varient d'un créneau à un autre. Par exemple, pour satisfaire les besoins de l'hôpital, l'effectif doit être renforcé le matin afin d'assurer les différents soins apportés aux patients.

5.1.2 Emploi du temps dans le domaine de transport

La confection d'horaires trouve aussi son application dans le transport. Son importance se justifie par le large éventail des problèmes d'intérêt pratique et/ou opérationnel qu'il englobe tels que : la gestion des flux de trafic aérien ou ferroviaire, le transport maritime et le transport routier. Dans certains cas, il est nécessaire de procéder à une re-planification en permanence et ceci en temps réel.

5.1.3 Emploi du temps dans le domaine de la pédagogie

La planification des cours est un problème classique rencontré dans les établissements scolaires. Un modèle général unifié pour ce type de problème est difficile à mettre en évidence car les contraintes qui lui sont associées peuvent différer grandement d'un établissement à un autre.

La détermination d'un horaire scolaire est une partition du temps en périodes durant lesquelles seuls des cours pouvant avoir lieu simultanément y sont donnés.

5.2 Adaptation de la coloration par blocs à la construction d'équipes en milieu hospitalier

La construction d'équipes d'infirmiers a fait l'objet de nombreux travaux [1], [4], [18], [21].

Nous nous focalisons dans cette partie sur la composition d'équipes d'infirmiers non polyvalents étant vu en Recherche Opérationnelle comme étant un ensemble de contraintes dures et de contraintes de préférences (souples) (à savoir ; {jour, nuit} [5] ou {matin, soir, nuit} [97]).
Notre objectif est de proposer aux cadres de la santé une meilleure composition d'équipes d'infirmiers tout en utilisant l'approche de la coloration des sommets d'un graphe. Il s'agit de l'algorithme de la coloration par blocs de cardinalité limitée de telle sorte à ce que les sommets représentent les infirmiers et les arêtes la relation d'adjacence.

Le service de chirurgie de la clinique de Soukra, de Tunis (Tunisie) travaille avec deux vacations {matin, soir} par 24 heures et 4 équipes d'infirmiers. Selon les expériences et les compétences, il existe 7 classes d'infirmiers :

- Infirmier stagiaire, Student Nurse (SN) : de 0 à 2 ans d'expérience.
- Infirmier, Nurse (N) : de 2 à 5 ans d'expérience, c'est celui qui fait le travail principalement.
- Infirmier principal, Principal Nurse (PN) : plus que 5 ans d'expérience. Il a une responsabilité de coordination entre les membres de l'équipe.
- Surveillant adjoint, Secondary Overseer Nurse (SON) : infirmier principal avec des compétences en management, du charisme, le sens de la responsabilité et la gestion d'équipes. Il remplace l'infirmier principal en cas d'absence.
- Surveillant, Overseer Nurse (ON) : plus que 5 ans d'expérience en tant que surveillant adjoint.
- Surveillant général nuit, Night General Overseer (NGO) : surveillant (ON) avec des compétences en management.
- Surveillant général, General Overseer (GO) : surveillant (ON) avec une vision globale sur les pathologies, gestionnaire de première classe, connaissance de toute la clinique (admissions, entrées des patients, synchronisation entre les services, polyvalent, relationnel).

Le surveillant général contrôle tous les services et travaille de $7am$ à $4pm$. Ainsi, pour chaque service, les équipes sont contrôlées par un surveillant qui doit être présent de $6am$ à $3pm$. Dans le but d'améliorer la qualité de travail, les infirmiers pouvant être dans la même équipe doivent avoir différents niveaux de compétence. Dans notre cas d'application, nous pouvons distinguer 3 types de catégories :

- T_1 représente l'ensemble des surveillants adjoints ;
- T_2 représente l'ensemble des infirmiers principaux ;
- T_3 représente l'ensemble des infirmiers et des infirmiers stagiaires.

Pour la construction des blocs (équipes) C_i, un ensemble de contraintes dures et simples doit être réalisé.

- le nombre de vacations par jour (à savoir ; deux vacations {matin, soir}) ;
- le nombre d'infirmiers par vacation (par exemple pour le service médical nous avons, 4 infirmiers le matin (de $7a.m.$ à $7p.m$) et 3 à 4 infirmiers le soir (de $7p.m.$ à $7a.m$)), cela ce traduit par $|C_i| \leq p[i]$;
- le nombre d'équipes q ($q = 4$), $q \leq |T_1| + |T_2|$;
- au moins un infirmier principal (PN) le soir, ce qui donne $\forall C_i : C_i \cap (T_2) \neq \emptyset$ et $\nexists v_1, v_2 \in T_2$ tel que v_1 et v_2 peuvent être dans la même équipe ;
- un surveillant adjoint (SON) dans une vacation de matin, celui-ci peut remplacer l'infirmier principal (PN), ce qui implique que $\forall C_i : C_i \cap (T_1 \cup T_2) \neq \emptyset$ et $\nexists v_1, v2 \in T_1 \times T_2$ tel que (v_1, v_1) et (v_2, v_2) peuvent être dans la même équipe. Il en découle que $\exists v_1, v_2, v_3 \in T_1 \times T_2 \times T_3$ tel que (v_1, v_2), (v_1, v_3), (v_2, v_3) and (v_3, v_3) peuvent être dans la même équipe où $|T_1| \leq |T_2|$.

Construisons le graphe G associé à la matrice d'adjacence donnée dans la Table 5.1, tout en considérant les sommets du graphe G comme étant les infirmiers. Tandis que, les données de la matrice correspondent aux conflits qui peuvent se présenter entre les infirmiers de différents niveaux. Le graphe associé à notre cas d'application contient $|V| = 14$ sommets (4 équipes, $p = [4, 4, 3, 3]$, à savoir ; une équipe de 4 infirmiers à une vacation de matin et une équipe de 3 infirmiers à une vacation de soir (une équipe travaille tous les jours).

TABLE 5.1 – Matrice d'adjacence (conflits entre les infirmiers)

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}
v_1	0	1	0	0	0	1	1	0	0	0	0	0	0	0
v_2	1	0	1	0	0	0	0	0	0	0	0	0	0	0
v_3	0	1	0	1	1	0	0	0	1	0	1	0	0	0
v_4	0	0	1	0	1	0	0	0	0	0	0	0	1	1
v_5	0	0	1	1	0	0	0	0	0	0	0	0	1	1
v_6	1	0	0	0	0	0	0	0	1	0	0	0	0	0
v_7	1	0	0	0	0	0	0	0	1	0	0	0	0	0
v_8	0	0	0	0	0	0	0	0	0	0	0	0	0	0
v_9	0	0	1	0	0	1	1	0	0	1	1	0	0	0
v_{10}	0	0	0	0	0	0	0	0	1	0	0	0	0	0
v_{11}	0	0	1	0	0	0	0	0	1	0	0	0	0	0
v_{12}	0	0	0	0	0	0	0	0	0	0	0	0	0	0
v_{13}	0	0	0	1	1	0	0	0	0	0	0	0	0	0
v_{14}	0	0	0	1	1	0	0	0	0	0	0	0	0	0

Deux sommets de G sont reliés entre eux si et seulement si les deux infirmiers correspondant ne peuvent être dans la même équipe. Ainsi, ces deux sommets reçoivent deux couleurs différentes.

Soient $\{v_1, v_2\}$ représente l'ensemble des surveillants adjoints (SON), $\{v_3, v_4, v_5\}$ l'ensemble des infirmiers principaux (PN), $\{v_6, .., v_{12}\}$ l'ensemble des infirmiers (N) et $\{v_{13}, v_{14}\}$

l'ensemble des infirmiers stagiaires (SN).

La coloration donnée par notre algorithme de coloration par blocs de cardinalité limitée pour les donnée présentées dans la Table 5.1, est représentée sur la Figure 5.1.

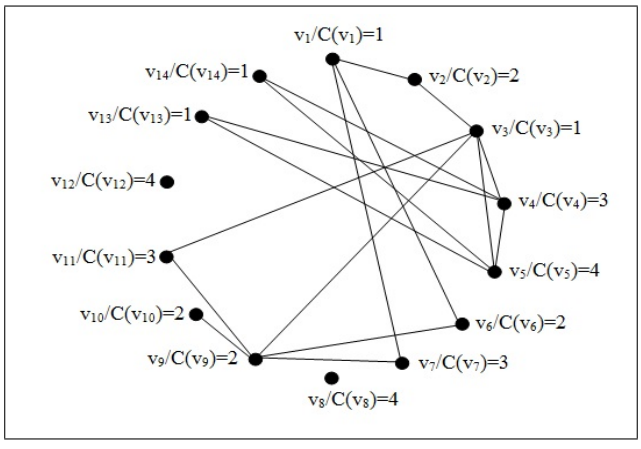

FIGURE 5.1 – Le graphe G représentant le modèle et la coloration obtenue par l'algorithme de coloration par blocs de cardinalité limitée

La solution trouvée par l'algorithme de coloration par blocs de cardinalité limitée est une configuration complète. Ainsi, la répartition des différents niveaux d'infirmiers dans les équipes est représentée dans la Table5.2.

TABLE 5.2 – Les équipes obtenues par l'algorithme de la coloration de cardinalité limitée

équipe 1	**équipe** 2	**équipe** 3	**équipe** 4
v_3, v_1, v_{13}, v_{14}	v_{10}, v_9, v_2, v_6	v_{11}, v_7, v_4	v_5, v_8, v_{12}
1 SON 1 PN 2 SN	1 SON 3 N	1 PN 2 N	1 PN 2 N

Les deux équipes 1 et 2, contiennent notamment un surveillant adjoint (SON). Ces deux équipes vont être programmées dans les vacations de matin (D), tandis que les deux équipes 3 et 4 vont être programmées dans les vacations de soir (N).
Chaque infirmier est déjà affecté à l'une des quatre équipes et travaille selon le planning (modèle) de quatre jours, voir la table 5.3. Le modèle ainsi présenté est acyclique.

TABLE 5.3 – Exemple de quatre jours de travail pour les équipes d'infirmiers

Jour	1	2	3	4
équipe 1	D	-	D	-
équipe 2	-	D	-	D
équipe 3	N	-	N	-
équipe 4	-	N	-	N

Nous présentons dans la section suivante un exemple illustratif de la modélisation par la coloration des sommets d'un graphe, d'un problème d'emploi du temps dans un établissement universitaire.

5.3 Illustration de la résolution d'un problème d'emploi du temps par l'approche de la coloration

Considérons le problème de confection d'horaires dans un établissement universitaire. Il consiste à attribuer des cours à un nombre précis d'intervalles de temps et de salles. Ainsi, trouver une solution réalisable pour ce problème permet de satisfaire un certain nombre de contraintes pouvant être difficiles.

Problème proposé

Nous proposons dans ce qui suit une modélisation et une résolution d'une situation pouvant se poser à l'Université de Lorraine, Faculté de Mathématiques, Informatique et Mécanique (MIM). Le problème concerne la construction d'un emploi du temps pour Master 1 informatique du semestre 2.

Description du problème proposé

La formation de première année du master informatique est composée de deux types d'unités d'enseignement UE, à savoir les UE fondamentales (cours obligatoires) et les UE optionnelles.

Le semestre comprend deux périodes, chacune s'étend sur 7 semaines : 6 semaines de cours et 1 semaine consacrée aux examens. Ainsi, chaque cours dure 6 semaines. Il se déroule donc sur l'une des 2 périodes. Aussi, nous pouvons distinguer deux types d'étudiants ; les étudiants en formation classique (FC) et les étudiants en formation en alternance (FA). L'alternance permet de suivre une formation tout en travaillant à temps partiel. Les étudiants en FA ne sont présents que 4 semaines sur 6. De ce fait, chaque semaine où ils sont présents, des créneaux doivent être réservés pour dédoubler certains

cours et chacun de ces créneaux dure 2 semaines afin d'assurer le nombre total de séances par période.

Dans ce cas, nous devons concevoir un emploi du temps séparé des étudiants en FC et des étudiants en FA. Les étudiants en FC ont $4UE$ fondamentales et $8UE$ optionnelles à choisir parmi $22UE$ optionnelles au total proposées. Tandis que les étudiants en FA ont les 4 UE fondamentales et pour les UE optionnelles ils n'ont pas le choix, 8 UE optionnelles sont affectées pour tous les étudiants de cette formation.

Nous disposons de 11 créneaux par semaine et certains cours sont incompatibles (même enseignant ou même étudiant en commun). Ainsi, nous pouvons placer 11 cours sans conflits par période.

Pour concevoir un planning qui répond à tous les besoins durant la période de la formation, nous devons respecter les contraintes suivantes :

1. Toutes les UE sont effectuées ;
2. Chaque UE est affectée à l'enseignant prévu pour l'assurer ;
3. A un enseignant donné, à un créneau bien précis, est affectée une unique UE ;
4. Deux UE optionnelles ayant un étudiant en commun ne peuvent se dérouler en même temps.

Modélisation du problème proposé par l'approche de la coloration des sommets d'un graphe

Nous construisons un graphe $G(V, E)$ dont l'ensemble des sommets V est en bijection avec les différents cours. Ainsi, pour chaque cours, nous associons un sommet v_i et deux sommets sont reliés entre eux si et seulement si les cours correspondant ne peuvent pas se dérouler en même temps, du fait qu'ils ont le même enseignant ou un étudiant en commun.

Dans ce cas, le nombre minimum de couleurs nécessaire pour colorier les sommets de G sans que deux sommets de même couleur ne soient adjacents est le nombre minimum de créneaux pour le déroulement de tous les cours, donc $\min(\text{créneaux}) = \chi(G)$. Ainsi, tous les cours dont les sommets correspondant ont une même couleur peuvent se dérouler en même temps.

De ce fait, nous considérons l'ensemble des sommets $V = V^C \cup V^A$ tel que V^C représente l'ensemble des UE en formation classique tandis que V^A représente l'ensemble des UE en formation en alternance. L'ensemble $V^C = V^{CTC} \cup V^{CO}$ où l'ensemble V^{CTC} représente les UE fondamentales et l'ensemble V^{CO} représente les UE optionnelles pour la formation classique.

Pour une période de la formation classique, nous avons 2 cours obligatoires, donc $V^{CTC} = \{v_1^{CTC}, v_2^{CTC}\}$ et un nombre fini d'options, disant 10 options, alors $V^{CO} = \{v_1^{CO}, v_2^{CO}, ..., v_{10}^{CO}\}$. Cependant, pour une période de la formation en alternance, les étudiants sont présents 4 semaines sur 6 et trois créneaux flottants sont proposés chaque deux semaines pour dédoubler les séances et ceci afin d'assurer le déroulement de tous les cours. De ce fait, nous associons à l'ensemble des cours obligatoires pour cette formation l'ensemble des sommets V^{ATC}, à l'ensemble des cours optionnelles l'ensemble des

sommets V^{AO} et à l'ensemble des cours flottants l'ensemble des sommets V^{AF} tels que : $V^A = V^{ATC} \cup V^{AO} \cup V^{AF}$ où $V^{ATC} = \{v_1^{ATC}, v_2^{ATC}\}$, $V^{AO} = \{v_1^{AO}, v_2^{AO}, v_3^{AO}, v_4^{AO}\}$ et $V^{AF} = \{v_1^{AF}, v_2^{AF}, v_3^{AF}, v_4^{AF}, v_5^{AF}, v_6^{AF}\}$ tel que l'ensemble $\{v_1^{AF}, v_2^{AF}, v_3^{AF}$ est le dupliqué [1] de l'ensemble $\{v_1^{ATC}, v_1^{AO}, v_2^{AO}\}$ et l'ensemble $v_4^{AF}, v_5^{AF}, v_6^{AF}\}$ est le dupliqué de de l'ensemble $\{v_2^{ATC}, v_3^{AO}, v_4^{AO}\}$ sachant que $\forall i \in \{1,2,3\}$ et $j \in \{4,5,6\}$, $(v_i^{AF}, v_j^{AF}) \notin E$.

Nous disposons de 11 créneaux par semaine, donc 11 cours peuvent se placer sans conflits. Ainsi, le nombre chromatique $\chi(G)$ du graphe modélisant le problème en question est majoré par 11, ainsi $\chi(G) \leq 11$.

Tous les étudiants de la formation en alternance passent les 6 cours ensemble, donc la taille de la clique maximum pouvant se présenter dans le graphe G est minorée par 6. Ainsi $\omega(G) \geq 6$ tel que $\omega(G) \leq \chi(G)$, d'où :

$$6 \leq \chi(G) \leq 11$$

.

Approche de résolution

Pour la résolution nous adoptons l'approche développée dans cette thèse, à savoir la coloration par blocs sur le tableau5.4. L'application de la méthode de coloration par blocs donne le résultat (planning) présenté sur le tableau 5.5.

TABLE 5.4 – Matrice de conflits

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}	v_{17}	v_{18}	v_{19}	v_{20}	v_{21}	v_{22}
v_1	-	1	-	1	-	-	1	-	1	-	-	-	1	-	1	-	-	1	-	1	-	-
v_2	-	-	1	-	-	1	-	1	-	-	-	1	-	1	-	-	1	-	1	-	-	-
v_3	-	-	-	-	1	-	1	-	-	1	-	-	1	-	-	1	-	1	-	-	1	-
v_4	-	-	-	-	1	1	-	-	-	1	-	1	-	-	-	1	1	-	-	-	1	-
v_5	-	-	-	-	-	-	-	1	1	-	-	-	-	1	1	-	-	-	1	1	-	-
v_6	-	-	-	-	-	-	1	1	-	1	1	-	1	-	1	-	1	1	1	1	1	1
v_7	-	-	-	-	-	-	-	1	-	-	1	1	1	1	-	-	1	1	1	1	1	1
v_8	-	-	-	-	-	-	-	-	-	-	1	-	1	-	-	1	1	1	1	1	1	1
v_9	-	-	-	-	-	-	-	-	-	1	1	1	-	-	-	1	-	-	-	-	-	1

1. Un sommet v_1 est le dupliqué d'un sommet v si et seulement s'ils ont les mêmes voisins. Un ensemble E_1 est le dupliqué d'un ensemble E si chaque élément de l'ensemble E_1 est le dupliqué de son image dans l'ensemble E.

	v_1	v_2	v_3	v_4	v_5	v_6	v_7	v_8	v_9	v_{10}	v_{11}	v_{12}	v_{13}	v_{14}	v_{15}	v_{16}	v_{17}	v_{18}	v_{19}	v_{20}	v_{21}	v_{22}
v_{10}	-	-	-	-	-	-	-	-	-	-	1	-	-	1	1	-	-	-	1	-	-	1
v_{11}	-	-	-	-	-	-	-	-	-	-	-	-	1	-	-	-	1	1	1	1	1	-
v_{12}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
v_{13}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	1	1	1	1	-
v_{14}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
v_{15}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
v_{16}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
v_{17}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	1	1	1	-
v_{18}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	1	1	-
v_{19}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	1	-
v_{20}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	1	-
v_{21}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
v_{22}	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

TABLE 5.5 – Emploi du temps associé

	cours				
créneau 1	v_{11}	v_1	v_3	-	-
créneau 2	v_4	v_{22}	v_2	v_{15}	v_{13}
créneau 3	v_{16}	v_5	v_{10}	v_6	-
créneau 4	v_9	v_{14}	v_8	-	-
créneau 5	v_{12}	v_{20}	-	-	-
créneau 6	v_{19}	-	-	-	-
créneau 7	v_{17}	-	-	-	-
créneau 8	v_{18}	-	-	-	-
créneau 9	v_{21}	-	-	-	-
créneau 10	v_7	-	-	-	-

CONCLUSION GÉNÉRALE ET PERSPECTIVES

Arrivé à terme de notre travail, il y a lieu de faire un bilan récapitulatif.

De nombreux algorithmes et méthodes de résolution exactes ont été développés suite aux travaux de recherche entrepris sur la coloration des sommets d'un graphe et le problème de la clique maximum. Ces algorithmes concernent certaines classes complètement caractérisées pour lesquelles la solution optimale est totalement déterminée par l'application de la méthode adéquate.

Dans la première partie de ce travail, présentée dans [51], nous avons présenté une contribution à la résolution du problème de la clique maximum. Cette contribution consiste en l'implémentation d'une nouvelle méthode efficace pour ce problème, basée essentiellement sur une méthode polynomiale d'orientation des arêtes et un schéma d'ordre d'étiquetage des sommets. En particulier, nous avons montré que la méthode conçue permet d'avoir une solution optimale pour quelques classes des graphes tels que les graphes biparti, 1-scindé, les graphes à voisinage scindé et les graphes 2-scindé ayant certaines propriétés. Nous avons également présenté une nouvelle heuristique exploitant l'algorithme d'orientation et le schéma d'ordre d'étiquetage afin de déterminer la clique maximum et sa cardinalité.

Dans la deuxième partie de ce travail, nous avons mis l'accent sur la recherche de la clique maximum dans un graphe et sa cardinalité. Nous avons présenté une nouvelle méthode de contraction qui nous a permis de caractériser un encadrement pour le nombre de clique. En exploitant le minorant $\underline{\omega}$, nous avons proposé une nouvelle méthode exacte pour la détermination de certaines cliques maximales, de la clique maximum et de sa cardinalité ω. Nous avons présenté dans celui-ci une étude expérimentale numérique afin de comparer les résultats obtenus par notre méthode avec la solution optimale sur la base des instances de la bibliothèque DIMACS. Les résultats expérimentaux obtenus sur ces instances montrent l'efficacité de notre algorithme.

Dans la troisième partie de ce travail, présentée dans [49] et [50], nous avons traité le problème de la coloration des sommets d'un graphe. Nous avons proposé une nouvelle méthode de coloration, qui est polynomiale et que nous avons nommée la coloration par blocs et un encadrement pour le nombre chromatique. En outre, nous avons proposé une comparaison théorique et empirique de notre majorant avec la borne de Reed dans le cas des graphes sans triangle. Les tests numériques confirment la domination de notre majorant face à celui de Reed.

Par ailleurs, nous avons présenté une version modifiée de notre algorithme de coloration

par blocs, il s'agit de l'algorithme de coloration par blocs de cardinalité limitée qui fourni des stables avec une cardinalité prédéfinie pour tenter la résolution du problème de la construction d'équipes présenté dans [49]. Afin d'améliorer la solution obtenue par la coloration par blocs, nous avons établi une nouvelle méthode de génération de colonnes.

La dernière partie de notre travail a été consacrée à l'étude d'un ensemble d'exemples pratique, il s'agit du problème d'emploi du temps dans le domaine de la pédagogie et celui de la construction d'équipes d'infirmiers pour le service de chirurgie d'un hôpital privé (Clinique de la Soukra à Tunis, Tunisie) [49].

Perspectives

Comme perspective, nous prévoyons dans de futurs travaux de recherche :

- Comparer notre méthode exacte, pour la recherche de la clique maximum et sa cardinalié, aux méthodes existantes. ;
- Valoriser la coloration pour la de génération de colonnes ;
- Adapter notre algorithme de coloration pour traiter des graphes pondérés et de l'appliquer pour le problème de la planification dans le domaine hospitalier.

Bibliographie

[1] B. Addis, R. Aringhieri, G. Carello, A. Grosso, F. Maffioli, E. Tanfani, A. Testi, Workforce management based on forecasted demand, in Advanced Decision Making Methods Applied to Health Care, Springer Milan, 1, 11, 2012.

[2] G. Anane, A nurse scheduling using graph colouring, Doctoral dissertation, Kwame Nkrumah University of Science and Technology, 2013.

[3] N.R. Aravind, T. Kartchik, and C.R. Subramanian, Bounding χ in terms of ω and Δ for some classes of graphs, Discrete Mathematics, 311 :911-920, 2011.

[4] R. Aringhieri, Composing medical crews with equity and efficiency, Central European Journal of Operations Research, 17(3), 343–357, 2009.

[5] M.N. Azaiez, S.S Al Sharif, A 0-1 Goal Programming Model for Nurse Scheduling, Computers & Operations Research, 32(3), 491-507, 2005

[6] E.Balas, C. S. Xu, Finding a maximum clique in an arbitrary graph. SIAM J. comuting, 15 (4) : 1054-1069, 1986.

[7] C. Berge, Motivations and history of some of my conjectures, Discrete Mathematics 165-166, 61-70, 1997.

[8] C. Berge, The history of the perfect graphs, Southeast Asian Bull. Math. 20 No 1, 1996.

[9] C. Berge, Théorie des graphes et ses applications, Dunod, Paris, 1967.

[10] L.A. Berry, F. Havet, C.L. Sales, B. Reed and S. Thomasse, Oriented trees in digraphs, Discrete Mathematics 313, 967-974, 2013.

[11] M.E. Bertschi, Perfectly contractile graphs, Journal of Combinatorial Theory B 50 (1990) 109.

[12] A. Billionnet, Optimisation discrète, De la modélisation à la résolution par des logiciels de programmation mathématique,livre, Dunod, Paris 2007.

[13] D. Brelaz, New methods to color the vertices of a graph, Comm. ACM 22 (1979).

[14] I. M. Bomze, M. Budinich, P.M. Pardalos, M. Pelillo, in : D. Z. Du, P. M. Pardalos (Eds.), The Maximum Clique Problem, in : Handbook of Combinatorial Optimization, vol. 4, Kluwer Academic Publishers, 1999, pp. 1–74.

[15] A. Boussaïri, A. Chaïchaâ, Les graphes 2-reconstructibles indécomposables, C. R. Acad. Sci. Paris, Mathématique, Série I 345 (2007) 1-4.

[16] R.L. Brooks, On colouring the nodes of a network, Proc. Cambridge Philosophical Society, 1941.

[17] M. Burlet, J. Fonlupt, Polynomial algorithm to recognize a Meyniel graph, Ann. Discrete Math. 21 (1984), p. 225-252.

[18] E. K. Burke, P. De Causmaecker, G. V. Berghe, H. Van Landeghem, The state of the art of nurse rostering. Journal of scheduling, 7(6), 441-499, 2004.

[19] L. Cavique, C.J. Luz, it A heuristic for the stability number of a graph based on convex quadratic programming and tabu search, Journal of Mathematical Sciences 2009 :161(6) :944-55.

[20] V. Cern*ý*, A thermodynamical approach to the traveling salesman problem : an efficient simulation algorithm, Journal of Optimization Theory and Applications, 45(1) :41-51, 1985.

[21] B. Cheang, H. Li, A. Lim, B. Rodrigues, Nurse rostering problems, bibliographic survey, European Journal of Operational Research, Volume 151, Issue 3, Pages 447-460, 2003.

[22] M. Chudnovsky, N. Robertson, P. Seymour, R. Thomas, The strong perfect graph theorem, Ann. of Math. (2) 164 (2006), no. 1, p. 51-229.

[23] V. Chv*á*tal, J. Fonlupt, L. Sun, A. Zemirline, Recognizing dart-free perfect graphs, SIAM J. Comput. 31 (2002), no. 5, p. 1315-1338.

[24] V. Chv*á*tal, Perfectly ordered graphs, in : C. Berge and V. Chvatal eds, Topics on Perfect Graphs, Ann. Discrete Mathematics. 21, North-Holland, Amsterdam, 1984.

[25] A. Colorni, M. Dorigo, V. Maniezzo, An investigation of some proprieties of an ant algorithm, In Manner and Manderick, 1992.

[26] M. Conforti, G. Cornuéjols, K. Vu*š*ković, Decomposition of oddhole- free graphs by double star cutsets and 2-joins, Discrete Appl. Math. 141 (2004), no. 1-3, p. 41-91.

[27] M. Conforti, G. Cornuéjols, Graphs without odd holes, parachutes or proper wheels : a generalization of Meyniel graphs and of line graphs of bipartite graphs, J. Combin. Theory Ser. B 87 (2003), no. 2, p. 300-330.

[28] A. Dammak, A. Elloumi, H. Kamoun, An enterprise system component based on graph colouring for exam timetabling : A case study in a Tunisian university, Transforming Government : People, Process and Policy, 1(3), 255-270, 2007.

[29] R. P. Dilworth. A decomposition theorem for partially ordered sets. Annals of Mathematics second Series, 51(1) :161-166, 1950.

[30] J. S. Donsbach, S.I. Tannenbaum, G.A. Alliger, J.E. Mathieu, E. Salas, G. F. Goodwin, Team composition optimization : The Team Optimal Profile System (TOPS), ARI Technical Review, Alexandria, VA : U.S. Army Research Institute for the Behavioral and Social Sciences, 2009.

[31] P. Duchet, S. Olariu, Graphes parfaitement ordonnables généralisés, Discrete Mathematics 90 (1991) 99-101.

[32] P. Duchet, Graphes noyau-parfaits, Ann. Discret Math. 9, North-Holland, Amsterdam, 1980, 93-101.

[33] J. Edmonds. Paths, treescand flowers, Canadian Journal of Mathematics, 1965- vol. 17- P. 449-467.

[34] A. Faez, D. Kalyanmoy, J. Abhilash, Multi-objective optimization and decision making approaches to cricket team selection, Applied Soft Computing, 13(1), 402–414, 2013.

[35] T. Fahle, Simple and fast : improving a branch and bound algorithm for maximum clique, In : Lecture notes in computer science, vol. 2461. Berlin : Springer ; 2002. p. 485-98.

[36] J.C. Fournier, Line caractérisation des graphes de cordes, C.R. Acad. Sci. Paris, (1978) 811-813.

[37] A. Frank, T. Kiraly and Z. Kiraly, On the orientation of graphs and hypergraphs, Discrete Applied Mathematics 131 (2003) 385-400.

[38] D. R. Fulkerson and O. A. Gross, Incidence matrices and interval graphs, Pacific Journal of Mathematics Vol. 15, No. 3, 1965.

[39] A. Frank, T. Kiraly, Z. Kiraly, On the orientation of graphs and hypergraphs, Discrete Applied Mathematics 131 (2003) 385-400.

[40] T. Gallai, Graphen mit triangulierbaren vielecken. Magyar Tud. Akad. Mat. Kutato Int. kozl. 7 (1962) A. 3-36.

[41] C. Gavoille, D. Peleg, S. Perennes and R. Raz, Distance labeling in graphs, Journal of Algorithms 53 (2004) 85-112.

[42] C. Gavoille, M. Katz, N.A. Katz, C. Paul and D. Peleg, Approximate distance labeling schemes, in : Proceedings of the 9th Annual European Symposium on Algorithms (ESA), 2001, pp. 476-487.

[43] B. Gendron, A. Hertz, P. St-Louis, A sequential elimination algorithm for computing bounds on the clique number of a graph, Discrete Optimization 2008 ;5(3) :615-28.

[44] A. Ghouilà-Houri. Caractérisation des graphes non orientés dont on peut orienter les arêtes de manière à obtenir le graphe d'une relation d'ordre. Compte-rendu de l'Académie des Sciences, Paris, 254 :1370-1371, 1962.

[45] M. C. Golumbic, Algorithmic Graph Theory and Perfect Graphs, Computer Science and Applied Mathematics. Academic Press, 1980.

[46] M. C. Golumbic, W. Rheinbolds, Algorithmic Graph Theory and Perfect Graphs. ISBN : 978-0-12-289260-8.

[47] F. Glover, Tabu search : part I, ORSA Journal on Computing, Vol. 1 (3), 1989, p. 190-206.

[48] F. Glover, Future paths for integer programming and links to artificial intelligence, Computer and Operations Research, 13 :533-549, 1986.

[49] A.Gueham, A. Nagih, H. Ait haddadène and M. Masmoudi, Graph coloring approach with new upper bounds for the chromatic number : Team building application, RAIRO- Operations Research, Doi 10.1051/ro/2016069.

[50] A. Gueham, A. Nagih, H. Ait Haddadene, Two bounds of chromatic number in graphs coloring problem, 2014 International Conference on Control, Decision and Information Technologies (CoDIT), pp.292,296, 3-5 Nov. 2014.

[51] A. Gueham, H. Ait haddadene and A. Nagih, A labeling order scheme for the maximum clique problem, Applied Mathematical Sciences, 6, 5439–5452, 2012.

[52] M. Grötschel, L. Lovász and A. Schrijver, Polynomial algorithms for perfect graphs, Annals of Discrete Mathematics, 21, 325-356,(1984).

[53] P.L. Hammer and Bruno Simeone, The splittance of a graph, Combinatorica 1(3)(1981) 275–284.

[54] A. Hajnal, J. Suranyi, Uber die Auflosung von graphenin 1, 1958, 113-121.

[55] R.B. Hayward, C.T. Hoàng, F. Maffray, Optimizing weakly triangulated graphs, Graph and Combinatorics 5(1989) 339-349.

[56] A. Hertz, A fast algorithm for coloring Meyniel graphs, Journal of Combinatorial Theory B 50 (1990) 213-240.

[57] C. T. Hoàng and V. B. Le, Recognizing perfect 2-split graphs, SIAMJ. Discret Math, (2000) 48-55.

[58] J. Holland, Adaptation in natural and artificial systems, University of Michigan Press, Ann Arbor, 1975.

[59] T. Ito, W.S. Kennedy and B.A. Reed, A characterization of graphs with fractional total chromatic number equal to $\Delta + 2$, Electronic Notes in Discrete Mathematics 35, 235-240, 2009.

[60] R. M. Karp, Reducibility among combinatorial problems, Complexity of computer computations, 1972.

[61] L.G. Khachiyan, A polynomial algorithm in linear programming, Soviet Mathematics Doklady 20 (1979) 191-301.

[62] H.A. Kierstead and J.H. Schmerl, The chromatic number of graphs which induce neither K1,3 nor K5-e, Discrete mathematics 58, 253-262, 1986.

[63] A.D. King and B.A. Reed, Bounding χ in terms of ω and Δ for quasi-line graphs, Journal of Graph Theory 59, no. 3, 215-228, 2008.

[64] A.D. King, B.A. Reed, A. Vetta, An upper bound for the chromatic number of line graphs, European Journal of Combinatorics 28, 2182-2187, 2007.

[65] S. Kirkpatrik, C.D. Gelatt, K.P. Vecchi, Optimization by simulated annealing, Science, 220 :671-680, 1983.

[66] K.M. Koh, E.G. Tay, On optimal orientations of cartesian products with a bipartite graph, Discrete Applied Mathematics 98 (1999) 103-120.

[67] A. Kohl, I. Schiermeyer, Some results on Reed's Conjecture about ω, Δ and χ with respect to α, Discrete Mathematics 310, 1429-1438, 2010.

[68] A.V. Kostochka, L. Rabern, M. Stiebitz, Graphs with chromatic number close to maximum degree, Discrete Mathematics 312, 1273-1281, 2012.

[69] A.V. Kostochka, B.Y. Stodolsky, An upper bound on the domination number of n-vertex connected cubic graphs, Discrete Mathematics vol.309, 1142-1162, 2009.

[70] D. E. Knuth, The sandwich theorem, Electronic Journal of Combinatorics 1 (1994) 48.

[71] B. T. G. S. Kumara , A. A. I. Perera, Automated system for nurse scheduling using Graph Colouring, Indian Journal of Computer Science and Engineering, 2(3), 476–485, 2011.

[72] T. Lapegue, Planification de personnel avec affectation de taches fixées : méthodes et application dans un contexte médical, PhD thesis, University of Nante, October 2014.

[73] C. Linhares-Sales, F. Maffray, Even pairs in claw-free perfect graphs, J. Combin. Theory Ser. B 74(1998) 169-191.

[74] C. Linhares-Sales, F. Maffray, B. Reed-On planar perfectly contractile graphs, Graphs and Combinatorics 13 (1997), 167-187.

[75] L. Lovász, On the Shannon capacity of a graph, IEEE Transactions on Information Theory 25 (1) (1979) 1–7.

[76] L. Lovász, Normal hypergraphs and the perfect graph conjecture, Discrete Mathematics.2, 253-267, 1972.

[77] L. Lovász, A characterization of perfect graphs, journal of combinatorial theory (B), 13, 95-98, 1972.

[78] R. D. Luce, A. D. Perry, A method of matrix analysis of group structure, Psychometrika 1949 ;14 :95-116.

[79] C. J. Luz, A. Schrijver, A convex quadratic characterization of the Lovasz theta number, SIAM Journal on Discrete Mathematics 2005 ;19(2) :382-7.

[80] P. R. J. Ostergard, A fast algorithm for the maximum clique problem, Discrete Applied Mathematics 2002 ;120 :197-207.

[81] F. Maffray and M. Preissmann, Split-Neighbourhood Graphs and the Strong Perfect Graph Conjecture, Journal of Combinatorial Theory, Serie B (1995).

[82] E. Malaguti, P. Toth, A survey on vertex coloring problems, International Transactions in Operational Research, Volume 17, Number 1, 2010 , pp. 1-34(34)

[83] D. Matula, G. Marble, J. Isaacson, Graph coloring algorithmis. In Graph Theory and Computing, Academic Press, New York, 1972 (109-122).

[84] W. Naji, Reconnaissance des graphes de cordes, Discrete Mathematics 54 (1985) 329-337.

[85] J.A. Noel, B.A. Reed, D.B. West, H. Wu and X. Zhu, Choosability of graphs with bounded order : Ohba's conjecture and beyond, Electronic Notes in Discrete Mathematics 43, 89-95, 2013.

[86] D. Peleg, Proximity-preserving labeling schemes, Journal of Graph Theory 33 (2000) 167-176.

[87] J. Pena, J. Vera, L. F. Zuluaga, Computing the stability number of a graph via linear and semidefinite programming, SIAM Journal on Optimization 2007 ; 18(1) :87-105.

[88] L. Rabern, A note on Reed's conjecture, SIAM J. Discrete Math. 22, 820-827, 2008.

[89] B. Reed, ω, Δ, and χ, Journal of Graph Theory 27, 177-212, 1998.

[90] D.J. Rose, R.E. Tarjan, G.S. Lueker, Algorithmic aspects of vertex elimination of graphs, SIAM Journal on Computing 5 (1976) 266-283.

[91] I. Rusu, Graphes parfaits : étude structurelle et algorithmiques de coloration.

[92] Y.G. Sahin, A team building model for software engineering courses term projects, Computers & Education, 56(3), 916–922, 2011.

[93] I. Schiermeyer, A new upper bound for the chromatic number of a graph, Discuss. Math. Graph Theory 27, 137-142, 2007.

[94] C. E. Shannon, The zero error capacity of a noisy channel, Inform. Theory IT-2 (1956), 8-19.

[95] K. K. Singh, A. K. Pandey, Survey of Algorithms on Maximum Clique Problem, International Advanced Research Journal in Science, Engineering and Technology, Vol. 2, Issue 2, February 2015.

[96] E. Tomita, T. Kameda, An efficient branch and bound algorithm for finding a maximum clique with computational experiments, Journal on Global Optimization 2007, 37 :95-111.

[97] C. Valouxis, E. Housos, Hybrid Optimization Techniques for the Workshift and Rest Assignment of Nursing Personnel, Artificial Intelligence in Medicine, 20, 155-175, 2000.

[98] O. Weimann and D. Peleg, A note on exact distance labeling, Information Processing Letters 111 (2011) 671-673.

[99] D. West. Introduction to Graph Theory. Prentice Hall, 2001.

[100] D. R. Wood, An algorithm for finding a maximum clique in a graph, Operations Research Letters 21 (1997) 211-217.

[101] W. C. K. Yen, The edge-orientation problem and some of its variants on weighted graphs, Information Sciences 176 (2006) 2791-2816.

ANNEXE

Dans cette partie, nous présentons la simulation de nos méthodes qui font l'objet des chapitres 3 et 4 à savoir l'approche de contraction qui induit le minorant du nombre de clique $\underline{\omega}$ ainsi que la clique associée, et la méthode de la coloration par blocs. Cette simulation a donnée lieu à deux applications. A partir d'un graphe G, la première application fournie un minorant du graphe G sur la base de la méthode de contraction. En exploitant ce minorant, la méthode permet de proposer la clique maximum. Cependant, la deuxième application permet de colorier le graphe G et d'en extraire un majorant pour le nombre chromatique. Ces deux applications ont été programmées en langage de programmation C++ et ont été exécutées sur une machine CORE i5. La moyenne du temps d'exécution des instances déjà présentées dans les chapitres 3 et 4 est estimée à 5mn.

Nous présentons dans ce qui suit certaines instances de graphes pour lesquelles la méthode considérée fournie une solution qui peut être optimale ou non.

5.4 Instances d'application pour le problème de la clique

5.4.1 Instance pour laquelle la solution optimale est atteinte

Nous considérons l'instance myciel5 pour 47 sommets et 236 arêtes dont le nombre de clique est 2 (SOURCE : Michael Trick (trick@cmu.edu) DESCRIPTION : Graphe sans Triangle basé sur Mycielski transformation).

e1	1	2	e2	1	4	e3	1	7	e4	1	9	e5	1	13	e6	1	15
e7	1	18	e8	1	20	e9	1	25	e10	1	27	e11	1	30	e12	1	32
e13	1	36	e14	1	38	e15	1	41	e16	1	43	e17	2	3	e18	2	6
e19	2	8	e20	2	12	e21	2	14	e22	2	17	e23	2	19	e24	2	24
e25	2	26	e26	2	29	e27	2	31	e28	2	35	e29	2	37	e30	2	40
e31	2	42	e32	3	5	e33	3	7	e34	3	10	e35	3	13	e36	3	16

e37	3	18	e38	3	21	e39	3	25	e40	3	28	e41	3	30
e42	3	33	e43	3	36	e44	3	39	e45	3	41	e46	3	44
e47	4	5	e48	4	6	e49	4	10	e50	4	12	e51	4	16
e52	4	17	e53	4	21	e54	4	24	e55	4	28	e56	4	29
e57	4	33	e58	4	35	e59	4	39	e60	4	40	e61	4	44
e62	5	8	e63	5	9	e64	5	14	e65	5	15	e66	5	19
e67	5	20	e68	5	26	e69	5	27	e70	5	31	e71	5	32
e72	5	37	e73	5	38	e74	5	42	e75	5	43	e76	6	11
e77	6	13	e78	6	15	e79	6	22	e80	6	25	e81	6	27
e82	6	34	e83	6	36	e84	6	38	e85	6	45	e86	7	11
e87	7	12	e88	7	14	e89	7	22	e90	7	24	e91	7	26
e92	7	34	e93	7	35	e94	7	37	e95	7	45	e96	8	11
e97	8	13	e98	8	16	e99	8	22	e100	8	25	e101	8	28
e102	8	34	e103	8	36	e104	8	39	e105	8	45	e106	9	11
e107	9	12	e108	9	16	e109	9	22	e110	9	24	e111	9	28
e112	9	34	e113	9	35	e114	9	39	e115	9	45	e116	10	11
e117	10	14	e118	10	15	e119	10	22	e120	10	26	e121	10	27
e122	10	34	e123	10	37	e124	10	38	e125	10	45	e126	11	17
e127	11	18	e128	11	19	e129	11	20	e130	11	21	e131	11	29
e132	11	30	e133	11	31	e134	11	32	e135	11	33	e136	11	40
e137	11	41	e138	11	42	e139	11	43	e140	11	44	e141	12	23
e142	12	25	e143	12	27	e144	12	30	e145	12	32	e146	12	46
e147	13	23	e148	13	24	e149	13	26	e150	13	29	e151	13	31
e152	13	46	e153	14	23	e154	14	25	e155	14	28	e156	14	30
e157	14	33	e158	14	46	e159	15	23	e160	15	24	e161	15	28
e162	15	29	e163	15	33	e164	15	46	e165	16	23	e166	16	26
e167	16	27	e168	16	31	e169	16	32	e170	16	46	e171	17	23

e172	17	25	e173	17	27	e174	17	34	e175	17	46	e176	18	23
e177	18	24	e178	18	26	e179	18	34	e180	18	46	e181	19	23
e182	19	25	e183	19	28	e184	19	34	e185	19	46	e186	20	23
e187	20	24	e188	20	28	e189	20	34	e190	20	46	e191	21	23
e192	21	26	e193	21	27	e194	21	34	e195	21	46	e196	22	23
e197	22	29	e198	22	30	e199	22	31	e200	22	32	e201	22	33
e202	22	46	e203	23	35	e204	23	36	e205	23	37	e206	23	38
e207	23	39	e208	23	40	e209	23	41	e210	23	42	e211	23	43
e212	23	44	e213	23	45	e214	24	47	e215	25	47	e216	26	47
e217	27	47	e218	28	47	e219	29	47	e220	30	47	e221	31	47
e222	32	47	e223	33	47	e224	34	47	e225	35	47	e226	36	47
e227	37	47	e228	38	47	e229	39	47	e230	40	47	e231	41	47
e232	42	47	e233	43	47	e234	44	47	e235	45	47	e236	46	47

Après l'exécution de notre programme nous avons abouti à :

Soit $G(V, E)$ un graphe tel que $|V| = N = 47$ et $|E| = M = 236$ Soit $G(V, A)$ le graphe oriente associe au graphe G, tel que A est l'ensemble d'arc défini de la manière suivante :

L'arc (47, 24)	L'arc (47, 25)	L'arc (47, 26)	L'arc (47, 27)
L'arc (47, 28)	L'arc (47, 29)	L'arc (47 , 30)	L'arc (47 , 31)
L'arc (47 , 32)	L'arc (47 , 33)	L'arc (47 , 34)	L'arc (47 , 35)
L'arc (47 , 36)	L'arc (47 , 37)	L'arc (47 , 38)	L'arc (47 , 39)
L'arc (47 , 40)	L'arc (47 , 41)	L'arc (47 , 42)	L'arc (47 , 43)
L'arc (47 , 44)	L'arc (47 , 45)	L'arc (47 , 46)	L'arc (23 , 12)
L'arc (23 , 13)	L'arc (23 , 14)	L'arc (23 , 15)	L'arc (23 , 16)
L'arc (23 , 17)	L'arc (23 , 18)	L'arc (23 , 19)	L'arc (23 , 20)
L'arc (23 , 21)	L'arc (23 , 22)	L'arc (23 , 35)	L'arc (23 , 36)
L'arc (23 , 37)	L'arc (23 , 38)	L'arc (23 , 39)	L'arc (23 , 40)
L'arc (23 , 41)	L'arc (23 , 42)	L'arc (23 , 43)	L'arc (23 , 44)

L'arc (23 , 45)	L'arc (11 , 6)	L'arc (11 , 7)	L'arc (11 , 8)
L'arc (11 , 9)	L'arc (11 , 10)	L'arc (11 , 17)	L'arc (11 , 18)
L'arc (11 , 19)	L'arc (11 , 20)	L'arc (11 , 21)	L'arc (11 , 29)
L'arc (11 , 30)	L'arc (11 , 31)	L'arc (11 , 32)	L'arc (11 , 33)
L'arc (11 , 40)	L'arc (11 , 41)	L'arc (11 , 42)	L'arc (11 , 43)
L'arc (11 , 44)	L'arc (1 , 2)	L'arc (1 , 4)	L'arc (1 , 7)
L'arc (1 , 9)	L'arc (1 , 13)	L'arc (1 , 15)	L'arc (1 , 18)
L'arc (1 , 20)	L'arc (1 , 25)	L'arc (1 , 27)	L'arc (1 , 30)
L'arc (1 , 32)	L'arc (1 , 36)	L'arc (1 , 38)	L'arc (1 , 41)
L'arc (1 , 43)	L'arc (3 , 2)	L'arc (3 , 5)	L'arc (3 , 7)
L'arc (3 , 10)	L'arc (3 , 13)	L'arc (3 , 16)	L'arc (3 , 18)
L'arc (3 , 21)	L'arc (3 , 25)	L'arc (3 , 28)	L'arc (3 , 30)
L'arc (3 , 33)	L'arc (3 , 36)	L'arc (3 , 39)	L'arc (3 , 41)
L'arc (3 , 44)	L'arc (4 , 5)	L'arc (4 , 6)	L'arc (4 , 10)
L'arc (4 , 12)	L'arc (4 , 16)	L'arc (4 , 17)	L'arc (4 , 21)
L'arc (4 , 24)	L'arc (4 , 28)	L'arc (4 , 29)	L'arc (4 , 33)
L'arc (4 , 35)	L'arc (4 , 39)	L'arc (4 , 40)	L'arc (4 , 44)
L'arc (2 , 6)	L'arc (2 , 8)	L'arc (2 , 12)	L'arc (2 , 14)
L'arc (2 , 17)	L'arc (2 , 19)	L'arc (2 , 24)	L'arc (2 , 26)
L'arc (2 , 29)	L'arc (2 , 31)	L'arc (2 , 35)	L'arc (2 , 37)
L'arc (2 , 40)	L'arc (2 , 42)	L'arc (22 , 6)	L'arc (22 , 7)
L'arc (22 , 8)	L'arc (22 , 9)	L'arc (22 , 10)	L'arc (22 , 29)
L'arc (22 , 30)	L'arc (22 , 31)	L'arc (22 , 32)	L'arc (22 , 33)
L'arc (22 , 46)	L'arc (34 , 6)	L'arc (34 , 7)	L'arc (34 , 8)
L'arc (34 , 9)	L'arc (34 , 10)	L'arc (34 , 17)	L'arc (34 , 18)
L'arc (34 , 19)	L'arc (34 , 20)	L'arc (34 , 21)	L'arc (15 , 5)
L'arc (15 , 6)	L'arc (15 , 10)	L'arc (15 , 24)	L'arc (15 , 28)

L'arc (15 , 29)	L'arc (15 , 33)	L'arc (15 , 46)	L'arc (13 , 6)
L'arc (13 , 8)	L'arc (13 , 24)	L'arc (13 , 26)	L'arc (13 , 29)
L'arc (13 , 31)	L'arc (13 , 46)	L'arc (27 , 5)	L'arc (27 , 6)
L'arc (27 , 10)	L'arc (27 , 12)	L'arc (27 , 16)	L'arc (27 , 17)
L'arc (27 , 21)	L'arc (25 , 6)	L'arc (25 , 8)	L'arc (25 , 12)
L'arc (25 , 14)	L'arc (25 , 17)	L'arc (25 , 19)	L'arc (45 , 6)
L'arc (45 , 7)	L'arc (45 , 8)	L'arc (45 , 9)	L'arc (45 , 10)
L'arc (38 , 5)	L'arc (38 , 6)	L'arc (38 , 10)	L'arc (36 , 6)
L'arc (36 , 8)	L'arc (43 , 5)	L'arc (5 , 8)	L'arc (5 , 9)
L'arc (5 , 14)	L'arc (5 , 19)	L'arc (5 , 20)	L'arc (5 , 26)
L'arc (5 , 31)	L'arc (5 , 32)	L'arc (5 , 37)	L'arc (5 , 42)
L'arc (46 , 12)	L'arc (46 , 14)	L'arc (46 , 16)	L'arc (46 , 17)
L'arc (46 , 18)	L'arc (46 , 19)	L'arc (46 , 20)	L'arc (46 , 21)
L'arc (7 , 12)	L'arc (7 , 14)	L'arc (7 , 24)	L'arc (7 , 26)
L'arc (7 , 35)	L'arc (7 , 37)	L'arc (28 , 8)	L'arc (28 , 9)
L'arc (28 , 14)	L'arc (28 , 19)	L'arc (28 , 20)	L'arc (10 , 14)
L'arc (10 , 26)	L'arc (10 , 37)	L'arc (30 , 12)	L'arc (30 , 14)
L'arc (39 , 8)	L'arc (39 , 9)	L'arc (33 , 14)	L'arc (16 , 8)
L'arc (16 , 9)	L'arc (16 , 26)	L'arc (16 , 31)	L'arc (16 , 32)
L'arc (24 , 9)	L'arc (24 , 18)	L'arc (24 , 20)	L'arc (12 , 9)
L'arc (12 , 32)	L'arc (21 , 26)	L'arc (35 , 9)	L'arc (18 , 26)

Le majorant= 13 lié à l'incidence du sommet v_6

Le minorant = 2

La clique max est formée par les sommets :

v_{47}

v_{24}.

5.4.2 Instance pour laquelle la solution optimale n'est pas atteinte

Dans cette section, nous considérons l'instance MANN-a9 pour laquelle la solution optimale n'est pas atteinte. La source de cette instance Carlo Mannino (mannino@iasi.rm.cnr.it), DESCRIPTION : Clique formulation of the Steiner Triple Problem, translated from the set covering formulation, Graph Size :45, Clique Size : 16.

e1	2	1	e2	3	1	e3	3	2	e4	4	1	e5	4	2
e6	4	3	e7	5	1	e8	5	2	e9	5	3	e10	5	4
e11	6	1	e12	6	2	e13	6	3	e14	6	4	e15	6	5
e16	7	1	e17	7	2	e18	7	3	e19	7	4	e20	7	5
e21	7	6	e22	8	1	e23	8	2	e24	8	3	e25	8	4
e26	8	5	e27	8	6	e28	8	7	e29	9	1	e30	9	2
e31	9	3	e32	9	4	e33	9	5	e34	9	6	e35	9	7
e36	9	8	e37	10	2	e38	10	3	e39	10	4	e40	10	5
e41	10	6	e42	10	7	e43	10	8	e44	10	9	e45	11	1
e46	11	3	e47	11	4	e48	11	5	e49	11	6	e50	11	7
e51	11	8	e52	11	9	e53	12	1	e54	12	2	e55	12	4
e56	12	5	e57	12	6	e58	12	7	e59	12	8	e60	12	9
e61	13	1	e62	13	2	e63	13	3	e64	13	5	e65	13	6
e66	13	7	e67	13	8	e68	13	9	e69	13	10	e70	13	11
e71	13	12	e72	14	1	e73	14	2	e74	14	3	e75	14	4
e76	14	6	e77	14	7	e78	14	8	e79	14	9	e80	14	10
e81	14	11	e82	14	12	e83	15	1	e84	15	2	e85	15	3
e86	15	4	e87	15	5	e88	15	7	e89	15	8	e90	15	9
e91	15	10	e92	15	11	e93	15	12	e94	16	1	e95	16	2
e96	16	3	e97	16	4	e98	16	5	e99	16	6	e100	16	8
e101	16	9	e102	16	10	e103	16	11	e104	16	12	e105	16	13
e106	16	14	e107	16	15	e108	17	1	e109	17	2	e110	17	3
e111	17	4	e112	17	5	e113	17	6	e114	17	7	e115	17	9

e116	17	10	e117	17	11	e118	17	12	e119	17	13	e120	17	14
e121	17	15	e122	18	1	e123	18	2	e124	18	3	e125	18	4
e126	18	5	e127	18	6	e128	18	7	e129	18	8	e130	18	10
e131	18	11	e132	18	12	e133	18	13	e134	18	14	e135	18	15
e136	19	2	e137	19	3	e138	19	4	e139	19	5	e140	19	6
e141	19	7	e142	19	8	e143	19	9	e144	19	10	e145	19	11
e146	19	12	e147	19	13	e148	19	14	e149	19	15	e150	19	16
e151	19	17	e152	19	18	e153	20	1	e154	20	2	e155	20	3
e156	20	4	e157	20	6	e158	20	7	e159	20	8	e160	20	9
e161	20	10	e162	20	11	e163	20	12	e164	20	13	e165	20	14
e166	20	15	e167	20	16	e168	20	17	e169	20	18	e170	21	1
e171	21	2	e172	21	3	e173	21	4	e174	21	5	e175	21	6
e176	21	7	e177	21	8	e178	21	10	e179	21	11	e180	21	12
e181	21	13	e182	21	14	e183	21	15	e184	21	16	e185	21	17
e186	21	18	e187	22	2	e188	22	3	e189	22	4	e190	22	5
e191	22	6	e192	22	7	e193	22	8	e194	22	9	e195	22	10
e196	22	11	e197	22	12	e198	22	13	e199	22	14	e200	22	15
e201	22	16	e202	22	17	e203	22	18	e204	22	19	e205	22	20
e206	22	21	e207	23	1	e208	23	2	e209	23	3	e210	23	5
e211	23	6	e212	23	7	e213	23	8	e214	23	9	e215	23	10
e216	23	11	e217	23	12	e218	23	13	e219	23	14	e220	23	15
e221	23	16	e222	23	17	e223	23	18	e224	23	19	e225	23	20
e226	23	21	e227	24	1	e228	24	2	e229	24	3	e230	24	4
e231	24	5	e232	24	6	e233	24	7	e234	24	9	e235	24	10
e236	24	11	e237	24	12	e238	24	13	e239	24	14	e240	24	15
e241	24	16	e242	24	17	e243	24	18	e244	24	19	e245	24	20
e246	24	21	e247	25	2	e248	25	3	e249	25	4	e250	25	5

e251	25	6	e252	25	7	e253	25	8	e254	25	9	e255	25	10
e256	25	11	e257	25	12	e258	25	13	e259	25	14	e260	25	15
e261	25	16	e262	25	17	e263	25	18	e264	25	19	e265	25	20
e266	25	21	e267	25	22	e268	25	23	e269	25	24	e270	26	1
e271	26	2	e272	26	3	e273	26	4	e274	26	5	e275	26	7
e276	26	8	e277	26	9	e278	26	10	e279	26	11	e280	26	12
e281	26	13	e282	26	14	e283	26	15	e284	26	16	e285	26	17
e286	26	18	e287	26	19	e288	26	20	e289	26	21	e290	26	22
e291	26	23	e292	26	24	e293	27	1	e294	27	2	e295	27	3
e296	27	4	e297	27	5	e298	27	6	e299	27	8	e300	27	9
e301	27	10	e302	27	11	e303	27	12	e304	27	13	e305	27	14
e306	27	15	e307	27	16	e308	27	17	e309	27	18	e310	27	19
e311	27	20	e312	27	21	e313	27	22	e314	27	23	e315	27	24
e316	28	1	e317	28	3	e318	28	4	e319	28	5	e320	28	6
e321	28	7	e322	28	8	e323	28	9	e324	28	10	e325	28	11
e326	28	12	e327	28	13	e328	28	14	e329	28	15	e330	28	16
e331	28	17	e332	28	18	e333	28	19	e334	28	20	e335	28	21
e336	28	22	e337	28	23	e338	28	24	e339	28	25	e340	28	26
e341	28	27	e342	29	1	e343	29	2	e344	29	3	e345	29	5
e346	29	6	e347	29	7	e348	29	8	e349	29	9	e350	29	10
e351	29	11	e352	29	12	e353	29	13	e354	29	14	e355	29	15
e356	29	16	e357	29	17	e358	29	18	e359	29	19	e360	29	20
e361	29	21	e362	29	22	e363	29	23	e364	29	24	e365	29	25
e366	29	26	e367	29	27	e368	30	1	e369	30	2	e370	30	3
e371	30	4	e372	30	5	e373	30	6	e374	30	7	e375	30	8
e376	30	10	e377	30	11	e378	30	12	e379	30	13	e380	30	14
e381	30	15	e382	30	16	e383	30	17	e384	30	18	e385	30	19

e386	30	20	e387	30	21	e388	30	22	e389	30	23	e390	30	24
e391	30	25	e392	30	26	e393	30	27	e394	31	1	e395	31	3
e396	31	4	e397	31	5	e398	31	6	e399	31	7	e400	31	8
e401	31	9	e402	31	10	e403	31	11	e404	31	12	e405	31	13
e406	31	14	e407	31	15	e408	31	16	e409	31	17	e410	31	18
e411	31	19	e412	31	20	e413	31	21	e414	31	22	e415	31	23
e416	31	24	e417	31	25	e418	31	26	e419	31	27	e420	31	28
e421	31	29	e422	31	30	e423	32	1	e424	32	2	e425	32	3
e426	32	4	e427	32	6	e428	32	7	e429	32	8	e430	32	9
e431	32	10	e432	32	11	e433	32	12	e434	32	13	e435	32	14
e436	32	15	e437	32	16	e438	32	17	e439	32	18	e440	32	19
e441	32	20	e442	32	21	e443	32	22	e444	32	23	e445	32	24
e446	32	25	e447	32	26	e448	32	27	e449	32	28	e450	32	29
e451	32	30	e452	33	1	e453	33	2	e454	33	3	e455	33	4
e456	33	5	e457	33	6	e458	33	8	e459	33	9	e460	33	10
e461	33	11	e462	33	12	e463	33	13	e464	33	14	e465	33	15
e466	33	16	e467	33	17	e468	33	18	e469	33	19	e470	33	20
e471	33	21	e472	33	22	e473	33	23	e474	33	24	e475	33	25
e476	33	26	e477	33	27	e478	33	28	e479	33	29	e480	33	30
e481	34	1	e482	34	3	e483	34	4	e484	34	5	e485	34	6
e486	34	7	e487	34	8	e488	34	9	e489	34	10	e490	34	11
e491	34	12	e492	34	13	e493	34	14	e494	34	15	e495	34	16
e496	34	17	e497	34	18	e498	34	19	e499	34	20	e500	34	21
e501	34	22	e502	34	23	e503	34	24	e504	34	25	e505	34	26
e506	34	27	e507	34	28	e508	34	29	e509	34	30	e510	34	31
e511	34	32	e512	34	33	e513	35	1	e514	35	2	e515	35	3
e516	35	4	e517	35	5	e518	35	7	e519	35	8	e520	35	9

e521	35	10	e522	35	11	e523	35	12	e524	35	13	e525	35	14
e526	35	15	e527	35	16	e528	35	17	e529	35	18	e530	35	19
e531	35	20	e532	35	21	e533	35	22	e534	35	23	e535	35	24
e536	35	25	e537	35	26	e538	35	27	e539	35	28	e540	35	29
e541	35	30	e542	35	31	e543	35	32	e544	35	33	e545	36	1
e546	36	2	e547	36	3	e548	36	4	e549	36	5	e550	36	6
e551	36	7	e552	36	9	e553	36	10	e554	36	11	e555	36	12
e556	36	13	e557	36	14	e558	36	15	e559	36	16	e560	36	17
e561	36	18	e562	36	19	e563	36	20	e564	36	21	e565	36	22
e566	36	23	e567	36	24	e568	36	25	e569	36	26	e570	36	27
e571	36	28	e572	36	29	e573	36	30	e574	36	31	e575	36	32
e576	36	33	e577	37	1	e578	37	2	e579	37	4	e580	37	5
e581	37	6	e582	37	7	e583	37	8	e584	37	9	e585	37	10
e586	37	11	e587	37	12	e588	37	13	e589	37	14	e590	37	15
e591	37	16	e592	37	17	e593	37	18	e594	37	19	e595	37	20
e596	37	21	e597	37	22	e598	37	23	e599	37	24	e600	37	25
e601	37	26	e602	37	27	e603	37	28	e604	37	29	e605	37	30
e606	37	31	e607	37	32	e608	37	33	e609	37	34	e610	37	35
e611	37	36	e612	38	1	e613	38	2	e614	38	3	e615	38	4
e616	38	5	e617	38	7	e618	38	8	e619	38	9	e620	38	10
e621	38	11	e622	38	12	e623	38	13	e624	38	14	e625	38	15
e626	38	16	e627	38	17	e628	38	18	e629	38	19	e630	38	20
e631	38	21	e632	38	22	e633	38	23	e634	38	24	e635	38	25
e636	38	26	e637	38	27	e638	38	28	e639	38	29	e640	38	30
e641	38	31	e642	38	32	e643	38	33	e644	38	34	e645	38	35
e646	38	36	e647	39	1	e648	39	2	e649	39	3	e650	39	4
e651	39	5	e652	39	6	e653	39	7	e654	39	8	e655	39	10

e656	39	11	e657	39	12	e658	39	13	e659	39	14	e660	39	15
e661	39	16	e662	39	17	e663	39	18	e664	39	19	e665	39	20
e666	39	21	e667	39	22	e668	39	23	e669	39	24	e670	39	25
e671	39	26	e672	39	27	e673	39	28	e674	39	29	e675	39	30
e676	39	31	e677	39	32	e678	39	33	e679	39	34	e680	39	35
e681	39	36	e682	40	1	e683	40	2	e684	40	4	e685	40	5
e686	40	6	e687	40	7	e688	40	8	e689	40	9	e690	40	10
e691	40	11	e692	40	12	e693	40	13	e694	40	14	e695	40	15
e696	40	16	e697	40	17	e698	40	18	e699	40	19	e700	40	20
e701	40	21	e702	40	22	e703	40	23	e704	40	24	e705	40	25
e706	40	26	e707	40	27	e708	40	28	e709	40	29	e710	40	30
e711	40	31	e712	40	32	e713	40	33	e714	40	34	e715	40	35
e716	40	36	e717	40	37	e718	40	38	e719	40	39	e720	41	1
e721	41	2	e722	41	3	e723	41	4	e724	41	6	e725	41	7
e726	41	8	e727	41	9	e728	41	10	e729	41	11	e730	41	12
e731	41	13	e732	41	14	e733	41	15	e734	41	16	e735	41	17
e736	41	18	e737	41	19	e738	41	20	e739	41	21	e740	41	22
e741	41	23	e742	41	24	e743	41	25	e744	41	26	e745	41	27
e746	41	28	e747	41	29	e748	41	30	e749	41	31	e750	41	32
e751	41	33	e752	41	34	e753	41	35	e754	41	36	e755	41	37
e756	41	38	e757	41	39	e758	42	1	e759	42	2	e760	42	3
e761	42	4	e762	42	5	e763	42	6	e764	42	7	e765	42	9
e766	42	10	e767	42	11	e768	42	12	e769	42	13	e770	42	14
e771	42	15	e772	42	16	e773	42	17	e774	42	18	e775	42	19
e776	42	20	e777	42	21	e778	42	22	e779	42	23	e780	42	24
e781	42	25	e782	42	26	e783	42	27	e784	42	28	e785	42	29
e786	42	30	e787	42	31	e788	42	32	e789	42	33	e790	42	34

e791	42	35	e792	42	36	e793	42	37	e794	42	38	e795	42	39
e796	43	1	e797	43	2	e798	43	4	e799	43	5	e800	43	6
e801	43	7	e802	43	8	e803	43	9	e804	43	10	e805	43	11
e806	43	12	e807	43	13	e808	43	14	e809	43	15	e810	43	16
e811	43	17	e812	43	18	e813	43	19	e814	43	20	e815	43	21
e816	43	22	e817	43	23	e818	43	24	e819	43	25	e820	43	26
e821	43	27	e822	43	28	e823	43	29	e824	43	30	e825	43	31
e826	43	32	e827	43	33	e828	43	34	e829	43	35	e830	43	36
e831	43	37	e832	43	38	e833	43	39	e834	43	40	e835	43	41
e836	43	42	e837	44	1	e838	44	2	e839	44	3	e840	44	5
e841	44	6	e842	44	7	e843	44	8	e844	44	9	e845	44	10
e846	44	11	e847	44	12	e848	44	13	e849	44	14	e850	44	15
e851	44	16	e852	44	17	e853	44	18	e854	44	19	e855	44	20
e856	44	21	e857	44	22	e858	44	23	e859	44	24	e860	44	25
e861	44	26	e862	44	27	e863	44	28	e864	44	29	e865	44	30
e866	44	31	e867	44	32	e868	44	33	e869	44	34	e870	44	35
e871	44	36	e872	44	37	e873	44	38	e874	44	39	e875	44	40
e876	44	41	e877	44	42	e878	45	1	e879	45	2	e880	45	3
e881	45	4	e882	45	5	e883	45	6	e884	45	8	e885	45	9
e886	45	10	e887	45	11	e888	45	12	e889	45	13	e890	45	14
e891	45	15	e892	45	16	e893	45	17	e894	45	18	e895	45	19
e896	45	20	e897	45	21	e898	45	22	e899	45	23	e900	45	24
e901	45	25	e902	45	26	e903	45	27	e904	45	28	e905	45	29
e906	45	30	e907	45	31	e908	45	32	e909	45	33	e910	45	34
e911	45	35	e912	45	36	e913	45	37	e914	45	38	e915	45	39
e916	45	40	e917	45	41	e918	45	42	-	-	-	-	-	-

Lors de l'exécution de notre programme nous avons trouvé ce qui suit :
Soit $G(V, E)$ un graphe tel que $|V| = N = 45$ et $|E| = M = 918$

Soit $G(V, A)$ le graphe oriente associe au graphe G, tel que A est l'ensemble d'arc défini de la manière suivante :

L'arc (10 , 2)	L'arc (10 , 3)	L'arc (10 , 4)	L'arc (10 , 5)
L'arc (10 , 6)	L'arc (10 , 7)	L'arc (10 , 8)	L'arc (10 , 9)
L'arc (10 , 13)	L'arc (10 , 14)	L'arc (10 , 15)	L'arc (10 , 16)
L'arc (10 , 17)	L'arc (10 , 18)	L'arc (10 , 19)	L'arc (10 , 20)
L'arc (10 , 21)	L'arc (10 , 22)	L'arc (10 , 23)	L'arc (10 , 24)
L'arc (10 , 25)	L'arc (10 , 26)	L'arc (10 , 27)	L'arc (10 , 28)
L'arc (10 , 29)	L'arc (10 , 30)	L'arc (10 , 31)	L'arc (10 , 32)
L'arc (10 , 33)	L'arc (10 , 34)	L'arc (10 , 35)	L'arc (10 , 36)
L'arc (10 , 37)	L'arc (10 , 38)	L'arc (10 , 39)	L'arc (10 , 40)
L'arc (10 , 41)	L'arc (10 , 42)	L'arc (10 , 43)	L'arc (10 , 44)
L'arc (10 , 45)	L'arc (11 , 1)	L'arc (11 , 3)	L'arc (11 , 4)
L'arc (11 , 5)	L'arc (11 , 6)	L'arc (11 , 7)	L'arc (11 , 8)
L'arc (11 , 9)	L'arc (11 , 13)	L'arc (11 , 14)	L'arc (11 , 15)
L'arc (11 , 16)	L'arc (11 , 17)	L'arc (11 , 18)	L'arc (11 , 19)
L'arc (11 , 20)	L'arc (11 , 21)	L'arc (11 , 22)	L'arc (11 , 23)
L'arc (11 , 24)	L'arc (11 , 25)	L'arc (11 , 26)	L'arc (11 , 27)
L'arc (11 , 28)	L'arc (11 , 29)	L'arc (11 , 30)	L'arc (11 , 31)
L'arc (11 , 32)	L'arc (11 , 33)	L'arc (11 , 34)	L'arc (11 , 35)
L'arc (11 , 36)	L'arc (11 , 37)	L'arc (11 , 38)	L'arc (11 , 39)
L'arc (11 , 40)	L'arc (11 , 41)	L'arc (11 , 42)	L'arc (11 , 43)
L'arc (11 , 44)	L'arc (11 , 45)	L'arc (12 , 1)	L'arc (12 , 2)
L'arc (12 , 4)	L'arc (12 , 5)	L'arc (12 , 6)	L'arc (12 , 7)
L'arc (12 , 8)	L'arc (12 , 9)	L'arc (12 , 13)	L'arc (12 , 14)
L'arc (12 , 15)	L'arc (12 , 16)	L'arc (12 , 17)	L'arc (12 , 18)
L'arc (12 , 19)	L'arc (12 , 20)	L'arc (12 , 21)	L'arc (12 , 22)
L'arc (12 , 23)	L'arc (12 , 24)	L'arc (12 , 25)	L'arc (12 , 26)

L'arc (12 , 27)	L'arc (12 , 28)	L'arc (12 , 29)	L'arc (12 , 30)
L'arc (12 , 31)	L'arc (12 , 32)	L'arc (12 , 33)	L'arc (12 , 34)
L'arc (12 , 35)	L'arc (12 , 36)	L'arc (12 , 37)	L'arc (12 , 38)
L'arc (12 , 39)	L'arc (12 , 40)	L'arc (12 , 41)	L'arc (12 , 42)
L'arc (12 , 43)	L'arc (12 , 44)	L'arc (12 , 45)	L'arc (1 , 2)
L'arc (1 , 3)	L'arc (1 , 4)	L'arc (1 , 5)	L'arc (1 , 6)
L'arc (1 , 7)	L'arc (1 , 8)	L'arc (1 , 9)	L'arc (1 , 13)
L'arc (1 , 14)	L'arc (1 , 15)	L'arc (1 , 16)	L'arc (1 , 17)
L'arc (1 , 18)	L'arc (1 , 20)	L'arc (1 , 21)	L'arc (1 , 23)
L'arc (1 , 24)	L'arc (1 , 26)	L'arc (1 , 27)	L'arc (1 , 28)
L'arc (1 , 29)	L'arc (1 , 30)	L'arc (1 , 31)	L'arc (1 , 32)
L'arc (1 , 33)	L'arc (1 , 34)	L'arc (1 , 35)	L'arc (1 , 36)
L'arc (1 , 37)	L'arc (1 , 38)	L'arc (1 , 39)	L'arc (1 , 40)
L'arc (1 , 41)	L'arc (1 , 42)	L'arc (1 , 43)	L'arc (1 , 44)
L'arc (1 , 45)	L'arc (19 , 2)	L'arc (19 , 3)	L'arc (19 , 4)
L'arc (19 , 5)	L'arc (19 , 6)	L'arc (19 , 7)	L'arc (19 , 8)
L'arc (19 , 9)	L'arc (19 , 13)	L'arc (19 , 14)	L'arc (19 , 15)
L'arc (19 , 16)	L'arc (19 , 17)	L'arc (19 , 18)	L'arc (19 , 22)
L'arc (19 , 23)	L'arc (19 , 24)	L'arc (19 , 25)	L'arc (19 , 26)
L'arc (19 , 27)	L'arc (19 , 28)	L'arc (19 , 29)	L'arc (19 , 30)
L'arc (19 , 31)	L'arc (19 , 32)	L'arc (19 , 33)	L'arc (19 , 34)
L'arc (19 , 35)	L'arc (19 , 36)	L'arc (19 , 37)	L'arc (19 , 38)
L'arc (19 , 39)	L'arc (19 , 40)	L'arc (19 , 41)	L'arc (19 , 42)
L'arc (19 , 43)	L'arc (19 , 44)	L'arc (19 , 45)	L'arc (20 , 2)
L'arc (20 , 3)	L'arc (20 , 4)	L'arc (20 , 6)	L'arc (20 , 7)
L'arc (20 , 8)	L'arc (20 , 9)	L'arc (20 , 13)	L'arc (20 , 14)
L'arc (20 , 15)	L'arc (20 , 16)	L'arc (20 , 17)	L'arc (20 , 18)

L'arc (20 , 22)	L'arc (20 , 23)	L'arc (20 , 24)	L'arc (20 , 25)
L'arc (20 , 26)	L'arc (20 , 27)	L'arc (20 , 28)	L'arc (20 , 29)
L'arc (20 , 30)	L'arc (20 , 31)	L'arc (20 , 32)	L'arc (20 , 33)
L'arc (20 , 34)	L'arc (20 , 35)	L'arc (20 , 36)	L'arc (20 , 37)
L'arc (20 , 38)	L'arc (20 , 39)	L'arc (20 , 40)	L'arc (20 , 41)
L'arc (20 , 42)	L'arc (20 , 43)	L'arc (20 , 44)	L'arc (20 , 45)
L'arc (21 , 2)	L'arc (21 , 3)	L'arc (21 , 4)	L'arc (21 , 5)
L'arc (21 , 6)	L'arc (21 , 7)	L'arc (21 , 8)	L'arc (21 , 13)
L'arc (21 , 14)	L'arc (21 , 15)	L'arc (21 , 16)	L'arc (21 , 17)
L'arc (21 , 18)	L'arc (21 , 22)	L'arc (21 , 23)	L'arc (21 , 24)
L'arc (21 , 25)	L'arc (21 , 26)	L'arc (21 , 27)	L'arc (21 , 28)
L'arc (21 , 29)	L'arc (21 , 30)	L'arc (21 , 31)	L'arc (21 , 32)
L'arc (21 , 33)	L'arc (21 , 34)	L'arc (21 , 35)	L'arc (21 , 36)
L'arc (21 , 37)	L'arc (21 , 38)	L'arc (21 , 39)	L'arc (21 , 40)
L'arc (21 , 41)	L'arc (21 , 42)	L'arc (21 , 43)	L'arc (21 , 44)
L'arc (21 , 45)	L'arc (22 , 2)	L'arc (22 , 3)	L'arc (22 , 4)
L'arc (22 , 5)	L'arc (22 , 6)	L'arc (22 , 7)	L'arc (22 , 8)
L'arc (22 , 9)	L'arc (22 , 13)	L'arc (22 , 14)	L'arc (22 , 15)
L'arc (22 , 16)	L'arc (22 , 17)	L'arc (22 , 18)	L'arc (22 , 25)
L'arc (22 , 26)	L'arc (22 , 27)	L'arc (22 , 28)	L'arc (22 , 29)
L'arc (22 , 30)	L'arc (22 , 31)	L'arc (22 , 32)	L'arc (22 , 33)
L'arc (22 , 34)	L'arc (22 , 35)	L'arc (22 , 36)	L'arc (22 , 37)
L'arc (22 , 38)	L'arc (22 , 39)	L'arc (22 , 40)	L'arc (22 , 41)
L'arc (22 , 42)	L'arc (22 , 43)	L'arc (22 , 44)	L'arc (22 , 45)
L'arc (23 , 2)	L'arc (23 , 3)	L'arc (23 , 5)	L'arc (23 , 6)
L'arc (23 , 7)	L'arc (23 , 8)	L'arc (23 , 9)	L'arc (23 , 13)
L'arc (23 , 14)	L'arc (23 , 15)	L'arc (23 , 16)	L'arc (23 , 17)

L'arc (23 , 18)	L'arc (23 , 25)	L'arc (23 , 26)	L'arc (23 , 27)
L'arc (23 , 28)	L'arc (23 , 29)	L'arc (23 , 30)	L'arc (23 , 31)
L'arc (23 , 32)	L'arc (23 , 33)	L'arc (23 , 34)	L'arc (23 , 35)
L'arc (23 , 36)	L'arc (23 , 37)	L'arc (23 , 38)	L'arc (23 , 39)
L'arc (23 , 40)	L'arc (23 , 41)	L'arc (23 , 42)	L'arc (23 , 43)
L'arc (23 , 44)	L'arc (23 , 45)	L'arc (24 , 2)	L'arc (24 , 3)
L'arc (24 , 4)	L'arc (24 , 5)	L'arc (24 , 6)	L'arc (24 , 7)
L'arc (24 , 9)	L'arc (24 , 13)	L'arc (24 , 14)	L'arc (24 , 15)
L'arc (24 , 16)	L'arc (24 , 17)	L'arc (24 , 18)	L'arc (24 , 25)
L'arc (24 , 26)	L'arc (24 , 27)	L'arc (24 , 28)	L'arc (24 , 29)
L'arc (24 , 30)	L'arc (24 , 31)	L'arc (24 , 32)	L'arc (24 , 33)
L'arc (24 , 34)	L'arc (24 , 35)	L'arc (24 , 36)	L'arc (24 , 37)
L'arc (24 , 38)	L'arc (24 , 39)	L'arc (24 , 40)	L'arc (24 , 41)
L'arc (24 , 42)	L'arc (24 , 43)	L'arc (24 , 44)	L'arc (24 , 45)
L'arc (25 , 2)	L'arc (25 , 3)	L'arc (25 , 4)	L'arc (25 , 5)
L'arc (25 , 6)	L'arc (25 , 7)	L'arc (25 , 8)	L'arc (25 , 9)
L'arc (25 , 13)	L'arc (25 , 14)	L'arc (25 , 15)	L'arc (25 , 16)
L'arc (25 , 17)	L'arc (25 , 18)	L'arc (25 , 28)	L'arc (25 , 29)
L'arc (25 , 30)	L'arc (25 , 31)	L'arc (25 , 32)	L'arc (25 , 33)
L'arc (25 , 34)	L'arc (25 , 35)	L'arc (25 , 36)	L'arc (25 , 37)
L'arc (25 , 38)	L'arc (25 , 39)	L'arc (25 , 40)	L'arc (25 , 41)
L'arc (25 , 42)	L'arc (25 , 43)	L'arc (25 , 44)	L'arc (25 , 45)
L'arc (26 , 2)	L'arc (26 , 3)	L'arc (26 , 4)	L'arc (26 , 5)
L'arc (26 , 7)	L'arc (26 , 8)	L'arc (26 , 9)	L'arc (26 , 13)
L'arc (26 , 14)	L'arc (26 , 15)	L'arc (26 , 16)	L'arc (26 , 17)
L'arc (26 , 18)	L'arc (26 , 28)	L'arc (26 , 29)	L'arc (26 , 30)
L'arc (26 , 31)	L'arc (26 , 32)	L'arc (26 , 33)	L'arc (26 , 34)

L'arc (26 , 35)	L'arc (26 , 36)	L'arc (26 , 37)	L'arc (26 , 38)
L'arc (26 , 39)	L'arc (26 , 40)	L'arc (26 , 41)	L'arc (26 , 42)
L'arc (26 , 43)	L'arc (26 , 44)	L'arc (26 , 45)	L'arc (27 , 2)
L'arc (27 , 3)	L'arc (27 , 4)	L'arc (27 , 5)	L'arc (27 , 6)
L'arc (27 , 8)	L'arc (27 , 9)	L'arc (27 , 13)	L'arc (27 , 14)
L'arc (27 , 15)	L'arc (27 , 16)	L'arc (27 , 17)	L'arc (27 , 18)
L'arc (27 , 28)	L'arc (27 , 29)	L'arc (27 , 30)	L'arc (27 , 31)
L'arc (27 , 32)	L'arc (27 , 33)	L'arc (27 , 34)	L'arc (27 , 35)
L'arc (27 , 36)	L'arc (27 , 37)	L'arc (27 , 38)	L'arc (27 , 39)
L'arc (27 , 40)	L'arc (27 , 41)	L'arc (27 , 42)	L'arc (27 , 43)
L'arc (27 , 44)	L'arc (27 , 45)	L'arc (2 , 3)	L'arc (2 , 4)
L'arc (2 , 5)	L'arc (2 , 6)	L'arc (2 , 7)	L'arc (2 , 8)
L'arc (2 , 9)	L'arc (2 , 13)	L'arc (2 , 14)	L'arc (2 , 15)
L'arc (2 , 16)	L'arc (2 , 17)	L'arc (2 , 18)	L'arc (2 , 29)
L'arc (2 , 30)	L'arc (2 , 32)	L'arc (2 , 33)	L'arc (2 , 35)
L'arc (2 , 36)	L'arc (2 , 37)	L'arc (2 , 38)	L'arc (2 , 39)
L'arc (2 , 40)	L'arc (2 , 41)	L'arc (2 , 42)	L'arc (2 , 43)
L'arc (2 , 44)	L'arc (2 , 45)	L'arc (28 , 3)	L'arc (28 , 4)
L'arc (28 , 5)	L'arc (28 , 6)	L'arc (28 , 7)	L'arc (28 , 8)
L'arc (28 , 9)	L'arc (28 , 13)	L'arc (28 , 14)	L'arc (28 , 15)
L'arc (28 , 16)	L'arc (28 , 17)	L'arc (28 , 18)	L'arc (28 , 31)
L'arc (28 , 32)	L'arc (28 , 33)	L'arc (28 , 34)	L'arc (28 , 35)
L'arc (28 , 36)	L'arc (28 , 37)	L'arc (28 , 38)	L'arc (28 , 39)
L'arc (28 , 40)	L'arc (28 , 41)	L'arc (28 , 42)	L'arc (28 , 43)
L'arc (28 , 44)	L'arc (28 , 45)	L'arc (29 , 3)	L'arc (29 , 5)
L'arc (29 , 6)	L'arc (29 , 7)	L'arc (29 , 8)	L'arc (29 , 9)
L'arc (29 , 13)	L'arc (29 , 14)	L'arc (29 , 15)	L'arc (29 , 16)

L'arc (29 , 17)	L'arc (29 , 18)	L'arc (29 , 31)	L'arc (29 , 32)
L'arc (29 , 33)	L'arc (29 , 34)	L'arc (29 , 35)	L'arc (29 , 36)
L'arc (29 , 37)	L'arc (29 , 38)	L'arc (29 , 39)	L'arc (29 , 40)
L'arc (29 , 41)	L'arc (29 , 42)	L'arc (29 , 43)	L'arc (29 , 44)
L'arc (29 , 45)	L'arc (30 , 3)	L'arc (30 , 4)	L'arc (30 , 5)
L'arc (30 , 6)	L'arc (30 , 7)	L'arc (30 , 8)	L'arc (30 , 13)
L'arc (30 , 14)	L'arc (30 , 15)	L'arc (30 , 16)	L'arc (30 , 17)
L'arc (30 , 18)	L'arc (30 , 31)	L'arc (30 , 32)	L'arc (30 , 33)
L'arc (30 , 34)	L'arc (30 , 35)	L'arc (30 , 36)	L'arc (30 , 37)
L'arc (30 , 38)	L'arc (30 , 39)	L'arc (30 , 40)	L'arc (30 , 41)
L'arc (30 , 42)	L'arc (30 , 43)	L'arc (30 , 44)	L'arc (30 , 45)
L'arc (4 , 3)	L'arc (4 , 5)	L'arc (4 , 6)	L'arc (4 , 7)
L'arc (4 , 8)	L'arc (4 , 9)	L'arc (4 , 14)	L'arc (4 , 15)
L'arc (4 , 16)	L'arc (4 , 17)	L'arc (4 , 18)	L'arc (4 , 31)
L'arc (4 , 32)	L'arc (4 , 33)	L'arc (4 , 34)	L'arc (4 , 35)
L'arc (4 , 36)	L'arc (4 , 37)	L'arc (4 , 38)	L'arc (4 , 39)
L'arc (4 , 40)	L'arc (4 , 41)	L'arc (4 , 42)	L'arc (4 , 43)
L'arc (4 , 45)	L'arc (13 , 3)	L'arc (13 , 5)	L'arc (13 , 6)
L'arc (13 , 7)	L'arc (13 , 8)	L'arc (13 , 9)	L'arc (13 , 16)
L'arc (13 , 17)	L'arc (13 , 18)	L'arc (13 , 31)	L'arc (13 , 32)
L'arc (13 , 33)	L'arc (13 , 34)	L'arc (13 , 35)	L'arc (13 , 36)
L'arc (13 , 37)	L'arc (13 , 38)	L'arc (13 , 39)	L'arc (13 , 40)
L'arc (13 , 41)	L'arc (13 , 42)	L'arc (13 , 43)	L'arc (13 , 44)
L'arc (13 , 45)	L'arc (9 , 3)	L'arc (9 , 5)	L'arc (9 , 6)
L'arc (9 , 7)	L'arc (9 , 8)	L'arc (9 , 14)	L'arc (9 , 15)
L'arc (9 , 16)	L'arc (9 , 17)	L'arc (9 , 31)	L'arc (9 , 32)
L'arc (9 , 33)	L'arc (9 , 34)	L'arc (9 , 35)	L'arc (9 , 36)

L'arc (9 , 37)	L'arc (9 , 38)	L'arc (9 , 40)	L'arc (9 , 41)
L'arc (9 , 42)	L'arc (9 , 43)	L'arc (9 , 44)	L'arc (9 , 45)
L'arc (18 , 3)	L'arc (18 , 5)	L'arc (18 , 6)	L'arc (18 , 7)
L'arc (18 , 8)	L'arc (18 , 14)	L'arc (18 , 15)	L'arc (18 , 31)
L'arc (18 , 32)	L'arc (18 , 33)	L'arc (18 , 34)	L'arc (18 , 35)
L'arc (18 , 36)	L'arc (18 , 37)	L'arc (18 , 38)	L'arc (18 , 39)
L'arc (18 , 40)	L'arc (18 , 41)	L'arc (18 , 42)	L'arc (18 , 43)
L'arc (18 , 44)	L'arc (18 , 45)	L'arc (14 , 3)	L'arc (14 , 6)
L'arc (14 , 7)	L'arc (14 , 8)	L'arc (14 , 16)	L'arc (14 , 17)
L'arc (14 , 31)	L'arc (14 , 32)	L'arc (14 , 33)	L'arc (14 , 34)
L'arc (14 , 35)	L'arc (14 , 36)	L'arc (14 , 37)	L'arc (14 , 38)
L'arc (14 , 39)	L'arc (14 , 40)	L'arc (14 , 41)	L'arc (14 , 42)
L'arc (14 , 43)	L'arc (14 , 44)	L'arc (14 , 45)	L'arc (15 , 3)
L'arc (15 , 5)	L'arc (15 , 7)	L'arc (15 , 8)	L'arc (15 , 16)
L'arc (15 , 17)	L'arc (15 , 31)	L'arc (15 , 32)	L'arc (15 , 33)
L'arc (15 , 34)	L'arc (15 , 35)	L'arc (15 , 36)	L'arc (15 , 37)
L'arc (15 , 38)	L'arc (15 , 39)	L'arc (15 , 40)	L'arc (15 , 41)
L'arc (15 , 42)	L'arc (15 , 43)	L'arc (15 , 44)	L'arc (15 , 45)
L'arc (5 , 3)	L'arc (5 , 6)	L'arc (5 , 7)	L'arc (5 , 8)
L'arc (5 , 16)	L'arc (5 , 17)	L'arc (5 , 31)	L'arc (5 , 33)
L'arc (5 , 34)	L'arc (5 , 35)	L'arc (5 , 36)	L'arc (5 , 37)
L'arc (5 , 38)	L'arc (5 , 39)	L'arc (5 , 40)	L'arc (5 , 42)
L'arc (5 , 43)	L'arc (5 , 44)	L'arc (5 , 45)	L'arc (32 , 3)
L'arc (32 , 6)	L'arc (32 , 7)	L'arc (32 , 8)	L'arc (32 , 16)
L'arc (32 , 17)	L'arc (32 , 34)	L'arc (32 , 35)	L'arc (32 , 36)
L'arc (32 , 37)	L'arc (32 , 38)	L'arc (32 , 39)	L'arc (32 , 40)
L'arc (32 , 41)	L'arc (32 , 42)	L'arc (32 , 43)	L'arc (32 , 44)

L'arc (32 , 45)	L'arc (31 , 3)	L'arc (31 , 6)	L'arc (31 , 7)
L'arc (31 , 8)	L'arc (31 , 16)	L'arc (31 , 17)	L'arc (31 , 34)
L'arc (31 , 35)	L'arc (31 , 36)	L'arc (31 , 37)	L'arc (31 , 38)
L'arc (31 , 39)	L'arc (31 , 40)	L'arc (31 , 41)	L'arc (31 , 42)
L'arc (31 , 43)	L'arc (31 , 44)	L'arc (31 , 45)	L'arc (33 , 3)
L'arc (33 , 6)	L'arc (33 , 8)	L'arc (33 , 16)	L'arc (33 , 17)
L'arc (33 , 34)	L'arc (33 , 35)	L'arc (33 , 36)	L'arc (33 , 37)
L'arc (33 , 38)	L'arc (33 , 39)	L'arc (33 , 40)	L'arc (33 , 41)
L'arc (33 , 42)	L'arc (33 , 43)	L'arc (33 , 44)	L'arc (33 , 45)
L'arc (6 , 3)	L'arc (6 , 7)	L'arc (6 , 8)	L'arc (6 , 16)
L'arc (6 , 17)	L'arc (6 , 34)	L'arc (6 , 36)	L'arc (6 , 37)
L'arc (6 , 39)	L'arc (6 , 40)	L'arc (6 , 41)	L'arc (6 , 42)
L'arc (6 , 43)	L'arc (6 , 44)	L'arc (6 , 45)	L'arc (35 , 3)
L'arc (35 , 7)	L'arc (35 , 8)	L'arc (35 , 16)	L'arc (35 , 17)
L'arc (35 , 37)	L'arc (35 , 38)	L'arc (35 , 39)	L'arc (35 , 40)
L'arc (35 , 41)	L'arc (35 , 42)	L'arc (35 , 43)	L'arc (35 , 44)
L'arc (35 , 45)	L'arc (34 , 3)	L'arc (34 , 7)	L'arc (34 , 8)
L'arc (34 , 16)	L'arc (34 , 17)	L'arc (34 , 37)	L'arc (34 , 38)
L'arc (34 , 39)	L'arc (34 , 40)	L'arc (34 , 41)	L'arc (34 , 42)
L'arc (34 , 43)	L'arc (34 , 44)	L'arc (34 , 45)	L'arc (36 , 3)
L'arc (36 , 7)	L'arc (36 , 16)	L'arc (36 , 17)	L'arc (36 , 37)
L'arc (36 , 38)	L'arc (36 , 39)	L'arc (36 , 40)	L'arc (36 , 41)
L'arc (36 , 42)	L'arc (36 , 43)	L'arc (36 , 44)	L'arc (36 , 45)
L'arc (7 , 3)	L'arc (7 , 8)	L'arc (7 , 17)	L'arc (7 , 37)
L'arc (7 , 38)	L'arc (7 , 39)	L'arc (7 , 40)	L'arc (7 , 41)
L'arc (7 , 42)	L'arc (7 , 43)	L'arc (7 , 44)	L'arc (16 , 3)
L'arc (16 , 8)	L'arc (16 , 37)	L'arc (16 , 38)	L'arc (16 , 39)

L'arc (16 , 40)	L'arc (16 , 41)	L'arc (16 , 42)	L'arc (16 , 43)
L'arc (16 , 44)	L'arc (16 , 45)	L'arc (17 , 3)	L'arc (17 , 37)
L'arc (17 , 38)	L'arc (17 , 39)	L'arc (17 , 40)	L'arc (17 , 41)
L'arc (17 , 42)	L'arc (17 , 43)	L'arc (17 , 44)	L'arc (17 , 45)
L'arc (8 , 3)	L'arc (8 , 37)	L'arc (8 , 38)	L'arc (8 , 39)
L'arc (8 , 40)	L'arc (8 , 41)	L'arc (8 , 43)	L'arc (8 , 44)
L'arc (8 , 45)	L'arc (38 , 3)	L'arc (38 , 40)	L'arc (38 , 41)
L'arc (38 , 42)	L'arc (38 , 43)	L'arc (38 , 44)	L'arc (38 , 45)
L'arc (39 , 3)	L'arc (39 , 40)	L'arc (39 , 41)	L'arc (39 , 42)
L'arc (39 , 43)	L'arc (39 , 44)	L'arc (39 , 45)	L'arc (37 , 40)
L'arc (37 , 41)	L'arc (37 , 42)	L'arc (37 , 43)	L'arc (37 , 44)
L'arc (37 , 45)	L'arc (3 , 41)	L'arc (3 , 42)	L'arc (3 , 44)
L'arc (3 , 45)	L'arc (40 , 43)	L'arc (40 , 44)	L'arc (40 , 45)
L'arc (41 , 43)	L'arc (41 , 44)	L'arc (41 , 45)	L'arc (42 , 43)
L'arc (42 , 44)	L'arc (42 , 45)	- - -	- - -

Le majorant = 42 lié à l'incidence du sommet v_{43}.
Le minorant = 14.
La clique max est formée par les sommets :
v_{10}
v_2
v_7
v_{14}
v_{15}
v_{16}
v_{19}
v_{22}
v_{25}
v_{29}
v_{32}
v_{35}
v_{37}
v_{40}

5.5 Application de la coloration par blocs pour quelques instances

Nous présentons dans ce qui suit des résultats d'implémentations de notre méthode de coloration par blocs sur des instances de graphes sans triangle qui sont générées aléatoirement et leurs comparaison avec la borne proposée par Reed. Lors de l'exécution de notre programme pour les instances ci-dessous nous avons trouvé :

5.5.1 Instance pour laquelle le majorant trouvé par notre méthode est différent de RC (conjecture de Reed)

Soit $G(V, E)$ un graphe tel que $|V| = N = 10$ et $|E| = M = 18$.
Soit $G(V, A)$ le graphe au graphe G, tel que A est l'ensemble d'arc défini de la manière suivante :
L'arc $(1, 2)$
L'arc $(1, 3)$
L'arc $(1, 4)$
L'arc $(1, 8)$
L'arc $(1, 9)$
L'arc $(1, 10)$
L'arc $(5, 3)$
L'arc $(5, 9)$
L'arc $(5, 10)$
L'arc $(6, 4)$
L'arc $(6, 8)$
L'arc $(6, 10)$
L'arc $(7, 4)$
L'arc $(7, 8)$
L'arc $(7, 9)$
L'arc $(2, 3)$
L'arc $(2, 4)$
L'arc $(3, 4)$
Les sommets de la couleur C_1 sont :
v_1
v_5
v_6
v_7
Les sommets de la couleur C_2 sont :
v_2
v_8
v_9
v_{10}

Les sommets de la couleur C_3 sont :
v_3
Les sommets de la couleur C_4 sont :
v_4
le majorant $= 4$ tandis que $RC = 6$.

5.5.2 Instance pour laquelle le majorant trouvé par notre méthode est identique à RC (conjecture de Reed)

Soit $G(V, E)$ un graphe tel que $|V| = N = 5$ et $|E| = M = 8$.
Soit $G(V, A)$ le graphe au graphe G, tel que A est l'ensemble d'arc défini de la manière suivante :
L'arc $(3, 1)$
L'arc $(3, 2)$
L'arc $(3, 4)$
L'arc $(3, 5)$
L'arc $(4, 1)$
L'arc $(4, 2)$
L'arc $(1, 2)$
Les sommets de la couleur C_1 sont :
v_3
Les sommets de la couleur C_2 sont :
v_4
Les sommets de la couleur C_3 sont :
v_1
v_5
Les sommets de la couleur C_4 sont :
v_2.