Optimizing the imbalances in a graph

Antoine Glorieux

THÈSE DE DOCTORAT DE TÉLÉCOM SUDPARIS

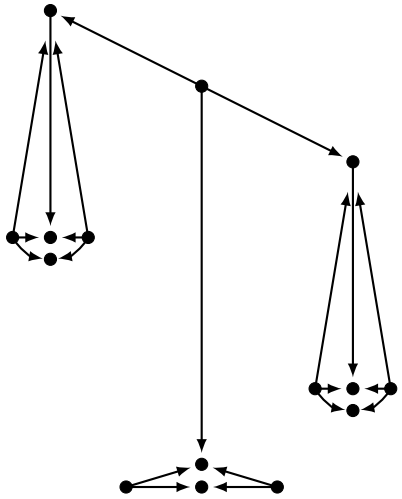
Spécialité
Mathématiques

École doctorale Informatique, Télécommunications et Électronique (Paris)

Présentée par
Antoine Glorieux

Pour obtenir le grade de
DOCTEUR DE TÉLÉCOM SUDPARIS

Optimiser les déséquilibres dans un graphe



soutenue le 19 juin 2017
devant le jury composé de :

Rapporteur :	Antoine Deza	Professeur, McMaster University, Chaire de recherche du Canada en optimisation combinatoire
Rapporteur :	Mourad Baïou	Directeur de recherche, CNRS, Université Blaise Pascal Clermont 2
Examinateur :	Frédéric Meunier	Professeur, École nationale des Ponts et Chaussées
Examinatrice :	Marie-Christine Costa	Professeur, ENSTA ParisTech
Examinateur :	Evripidis Bampis	Professeur, Université Pierre et Marie Curie
Examinateur :	Abdel Lisser	Professeur, Université Paris Sud
Encadrant de thèse :	José Neto	Maître de conférences, Télécom SudParis
Directeur de thèse :	Walid Ben-Ameur	Professeur, CNRS, Télécom SudParis

Numéro National de Thèse (NNT) : 2017TELE0011

TÉLÉCOM SUDPARIS DOCTORAL THESIS

Specialty
Mathematics

École doctorale Informatique, Télécommunications et Électronique (Paris)

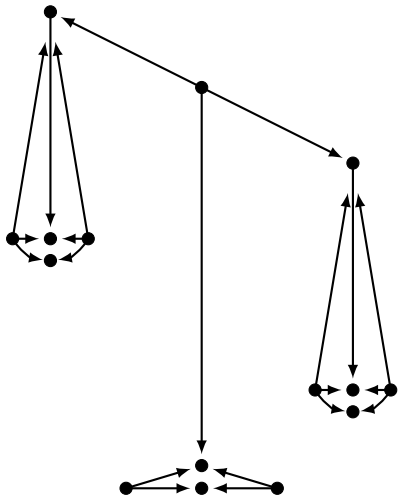
Presented by

Antoine Glorieux

submitted for the degree of

DOCTOR OF PHILOSOPHY AT TÉLÉCOM SUDPARIS

Optimizing the imbalances in a graph



Defense on June 29th, 2017
Defense comittee:

Reviewer:	Antoine Deza	Professor, McMaster University, Canada research chair in combinatorial optimisation
Reviewer:	Mourad Baïou	Research Director, CNRS, Université Blaise Pascal Clermont 2
Examiner :	Frédéric Meunier	Professor, École nationale des Ponts et Chaussées
Examiner :	Marie-Christine Costa	Professor, ENSTA ParisTech
Examiner :	Evripidis Bampis	Professor, Université Pierre et Marie Curie
Examiner :	Abdel Lisser	Professor, Université Paris Sud
Advisor:	José Neto	Assistant professor, Télécom SudParis
Advisor:	Walid Ben-Ameur	Professor, CNRS, Télécom SudParis

National Number of Thesis (NNT) : 2017TELE0011

Declaration

I declare that this thesis was composed by myself and that the work contained therein is my own, except where explicitly stated otherwise in the text.

Antoine Glorieux

Acknowledgements

I wish to express my most sincere gratitude and appreciation to my thesis advisors, Walid Ben-Ameur and José Neto for their constant and most valuable guidance, their precious encouragement, and their remarkable patience during my PhD study. Not only did they give me this great opportunity of doctoral study, but they created a friendly and motivating working environment with the perfect balance between autonomy and teamwork while allowing me to make room for my personal and parenting life. I will be forever honored and grateful for working with them and will always look back on these three years and half with a sense of accomplishment.

My gratitude extends to Prof. Antoine Deza and Dr. Mourad Baïou for their time in reviewing my manuscript and insightful comments. I also would like to thank Prof. Frédéric Meunier, Prof. Marie-Christine Costa, Prof. Evripidis Bampis and Prof. Abdel Lisser for accepting the invitation to participate in the defense.

It is a good opportunity to thank Tijani Chahed, Jeremie Jakubowicz, Valérie Mateus and Zahia Zendagui for their unlimited support, along with my PhD fellows, Pierre Bauguion, Mohamed Ahmed Mohamed Sidi, Jiao Yang, Guanglei Wang and Vincent Angilella, all craftsmen and craftswomen of the friendly working environment that is Télécom SudParis.

Finally, I would like to express nothing but love to my parents, brothers, sisters, niece and nephew for their relentless support, their unconditional love and for simply being who they are for they enabled me to become who I am now. And saving the best for last, my dearest Anne-Lise, love of my life, and my most beloved Tahar, greatest of publications, I owe you two all the happiness each day holds for me. You are my future and I love you.

La mathématique est l'art de donner le même nom à des choses différentes.
-Mathematics is the art of giving the same name to different things-

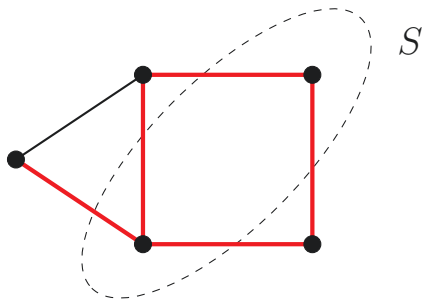
Henri Poincarré

Résumé

1 Introduction et notations

Soit $G = (V, E)$ un graphe simple non pondéré ayant pour ensemble de sommets $V = \{v_1, \cdots, v_n\}$ et pour ensemble d'arêtes E, on note δ_G (resp. Δ_G) le degré minimum (resp. maximum) des sommets de G. Pour un sous-ensemble de sommets $S \subseteq V$, la coupe définie par S est le sous-ensemble d'arêtes $\delta(S) = \{uv \in E : |\{u, v\} \cap S| = 1\}$ (e.g. Figure 1). Pour tout graphe ou sous-graphe G on note $V(G)$ et $E(G)$ respectivement l'ensemble des sommets et des arêtes de G.

Figure 1: Exemple d'une coupe (arêtes plus épaisses en rouge) definie par un sous-ensemble de sommets $S \subseteq V$ encerclé en pointillés



Une orientation Λ de G est l'affectation d'une direction à chacune de ses arêtes (non orientées) uv dans E, i.e. une fonction de E de la forme $\Lambda(uv) \in \{\overrightarrow{uv}, \overrightarrow{vu}\}$, $\forall uv \in E$. On appelle $\vec{O}(G)$ l'ensemble des orientations de G. Pour tout sommet v de G on note $d_G(v)$ ou $d(v)$ le degré de v dans G et $d^+_\Lambda(v)$ ou $d^+(v)$ (resp. $d^-_\Lambda(v)$ ou $d^-(v)$) le degré sortant (resp. entrant) de v dans G par rapport à Λ. Pour une orientation Λ de G et un sommet $v \in V$ on appelle $|d^+_\Lambda(v) - d^-_\Lambda(v)|$ (resp. $d^+_\Lambda(v) - d^-_\Lambda(v)$) le *déséquilibre* (resp. *déséquilibre signé*) de v dans G par rapport à Λ.

L'orientation de graphes est un domaine très étudié en théorie des graphes et en optimisation combinatoire, un grande variété de contraintes sur les orientations ainsi que de fonctions objectif ont déjà été considérées comme par exemple les problèmes

d'orientations avec des contraintes sur les degrés [43, 5, 6, 7]. D'autres problèmes ont été traités mettant en jeu d'autres critères tels que l'acyclicité, le diamètre [28] ou encore la connexité [83].

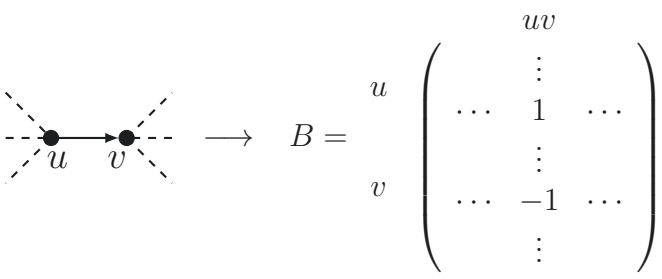
Nous étudions la famille de problèmes consistant à maximiser l'image du n-uplet $(d^+_\Lambda(v_1) - d^-_\Lambda(v_1), \cdots, d^+_\Lambda(v_n) - d^-_\Lambda(v_n))$ par une fonction $f \in \mathbb{R}^{\mathbb{R}^n}$ sur l'ensemble des orientations Λ d'un graphe G. En d'autres termes, le problème consiste à trouver une orientation optimisant les déséquilibres signés des sommets par rapport à la fonction objectif $f \in \mathbb{R}^{\mathbb{R}^n}$:

$$\max_{\Lambda \in \vec{O}(G)} f(d^+_\Lambda(v_1) - d^-_\Lambda(v_1), \cdots, d^+_\Lambda(v_n) - d^-_\Lambda(v_n)).$$

Considérons le graphe $G = (V, E)$ comme arbitrairement orienté et prenons $B \in \{-1, 0, 1\}^{|V| \times |E|}$ sa matrice d'incidence, i.e., la colonne correspondant à l'arc $\overrightarrow{uv}$ (ou, de façon équivalente, à l'arête uv orientée du sommet u au sommet v), a pour seules composantes non-nulles celles des lignes correspondant aux sommets u et v: $B_{u,uv} = 1$ et $B_{v,uv} = -1$, respectivement (cf Figure 2). Afin de décrire une orientation du graphe G, on prend une variable d'orientation $x \in \{-1, 1\}^{|E|}$ interprêtée de la façon suivante. Pour chaque arête $uv \in E$ originellement orientée du sommet u au sommet v: uv est orientée de u à v (i.e., l'orientation choisie est la même que l'originale) si $x_{uv} = 1$ et est orientée de v à u sinon (i.e., l'orientation choisie est "inversée" par rapport à l'originale). Si nous observons à présent le produit de B avec le vecteur d'orientation $x \in \{-1, 1\}^{|E|}$, nous obtenons $B_v x = d^+_x(v) - d^-_x(v),\ \forall v \in V$ où $d^+_x(v)$ (resp. $d^-_x(v)$) est le degré sortant (resp. entrant) de $v \in V$ dans G par rapport à l'orientation décrite par x et B_v est la ligne de la matrice B qui correspond au sommet v. En conséquence, le problème précédemment décrit pour une fonction objectif $f \in \mathbb{R}^{\mathbb{R}^n}$ peut être exprimé :

$$\max_{x \in \{-1,1\}^E} f(Bx).$$

Figure 2: Construction de la matrice d'incidence d'un graphe orienté



Cette famille de problèmes est liée à une grande variété de domaines en optimisation de graphes et en suggère de nouveaux. Par exemple, prendre pour fonction objectif $f = ||\cdot||$ (i.e. la norme euclidienne) mène à un majorant original du nombre isopérimétrique de G que nous mentionnons dans la section suivante.

2 Le nombre isopérimétrique

Définition 1. *Le nombre isopérimétrique (ou constante de Cheeger modifiée) [27] d'un graphe non-orienté $G = (V, E)$ est défini par*

$$h'_G = \inf_{\emptyset \neq S \subset V} \frac{|\delta(S)|}{\min(|S|, |V \setminus S|)}$$

Il peut être interprêté comme une mesure numérique de la "connectivité générale" d'un graphe. Une basse valeur indique la présence d'un goulot d'étranglement, i.e. l'existence de deux gros sous-ensembles de sommets connectés entre eux par peu d'arêtes. Au contraire, une haute valeur implique que tout sous-ensemble de sommets est connecté par de nombreuses arêtes au reste des sommets. Dans le cas où le graphe représente un réseau de communications, le nombre isopérimétrique peut être considéré comme une mesure de la "vulnérabilité" du réseau [72]. Il a de nombreuses applications aussi bien en mathématiques qu'en informatique (e.g., en analyse d'image [78]).

Le calcul du nombre isopérimétrique d'un graphe est un problème globalement NP-difficile [81]. Petr Golovach [49] a montré que le problème consistant à décider si $h'_G \leq \frac{p}{q}$ pour un graphe G tel que $\Delta_G \leq 3$ et deux entiers p et q. En revanche, Bojan Mohar [81] présente un algorithme en temps linéaire pour le cas particulier où le graphe est un arbre et James Park & Cynthia Phillips un algorithme en temps pseudo-polynomial pour le cas des graphes planaires [85].

Mohar [81] donne différents majorants et minorants du nombre isopérimétrique. D'une part un majorant général formulé en termes des nombres de sommets et d'arêtes, et d'autre part des majorants et minorants en termes de la seconde plus petite valeur propre du Laplacien du graphe.

Le nombre isopérimétrique peut être exprimé de la manière suivante :

$$\text{(Cheeger)} \quad h'_G = \min_{S \subset V \,:\, |S| \leq \frac{|V|}{2}} \frac{1}{|S|} \max_{\Lambda \in \vec{\mathcal{O}}(G)} \sum_{v \in S} |d^+_\Lambda(v) - d^-_\Lambda(v)|.$$

Cette expression mène au résultat suivant.

Proposition 2. *Pour tout graphe G, on a*

$$h'_G \leq \frac{1}{\sqrt{\lfloor n/2 \rfloor}} \max_{x \in [-1,1]^E} ||Bx|| \leq \sqrt{\frac{\lambda^{max}(B^t B)|E|}{\lfloor n/2 \rfloor}},$$

où on note pour toute matrice $M \in \mathbb{R}^{n \times n}$, sa valeur propre maximale $\lambda^{max}(M)$.

On retrouve ici l'expression générale de notre problème générique pour $f = \frac{1}{\sqrt{\lfloor n/2 \rfloor}} ||\cdot||$. Le plus grand majorant présenté dans la Proposition 18 est obtenu à partir du majorant intermédiaire par une ample majoration par la valeur propre maximale de B. Une étude du problème générique présenté dans la section précédente pourrait mener à un majorant de la constante de Cheeger de meilleure qualité.

Un autre exemple de fonction objectif f du problème générique est le cas $f(x_1, \cdots, x_n) = \min\{|x_1|, \cdots, |x_n|\}$ qui va nous mener à un nouveau problème d'optimisation de graphes.

3 Maximiser le déséquilibre d'une orientation

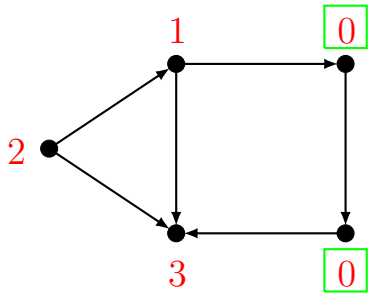
On appelle le *déséquilibre* d'une orientation $\Lambda \in \vec{O}(G)$ d'un graphe $G = (V, E)$ le minimum des déséquilibres des sommets de G: $\min_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)|$. La figure 3 montre un graphe orienté où le déséquilibres des sommets est indiqué en rouge et celui de l'orientation est encadré en vert. On considère maintenant le problème consistant à trouver une orientation avec un déséquilibre maximum:

$$(\textsc{MaxIm}) \quad \textsc{MaxIm}(G) = \max_{\Lambda \in \vec{O}(G)} \min_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)|$$

et on appelle $\textsc{MaxIm}(G)$ la valeur de MaxIm pour G. Le degré minimum δ_G de G est un majorant trivial pour $\textsc{MaxIm}(G)$. Étant donné que la valeur de MaxIm pour un graphe est le minimum des valeurs de MaxIm pour ses composantes connexes, nous considérerons dorénavant uniquement des graphes connexes.

Le problème d'orientation au déséquilibre maximum est un problème nouveau qui surgit naturellement de l'étude de problème d'orientations de graphes similaires. Par exemple, le problème qui consiste à décider si une suite d'entiers positifs peut être réalisé par un graphe orienté comme la suite de ses déséquilibres [82]. Un autre exemple, en 1962, Lester Ford & Delbert Fulkerson ont caractérisé les graphes mixtes (i.e. partiellement orientés) dont l'orientation partielle peut être complétée en une orientation eulérienne, c'est-à-dire une orientation dont le déséquilibre de chaque sommet est nul [42]. De nombreux autres problèmes d'orientation ont été proposés,

Figure 3: Exemple de graphe orienté: l'entier rouge à côté de chaque sommet indique son déséquilibre par rapport à l'orientation et ceux encadrés en vert sont égaux au déséquilibre de l'orientation



certains sont recensés dans [10]. Une partie des résultat qui suivent sont détaillés dans [14].

3.1 Complexité, Inapproximabilité et Approximabilité

Nous commençons par introduire une variante du problème de satisfaisabilité que nous réduisons à MaxIm : le problème "not-all-equal at most 3-SAT(3V)". Il s'agit d'une restriction de "not-all-equal at most 3-SAT" qui est lui-même une restriction de 3-SAT, problème NP-complet connu [91] où chaque clause contient au plus trois littéraux et dans chaque clause, tous les littéraux ne peuvent pas être vrais simultanément. La restriction additionnelle de "not-all-equal at most 3-SAT(3V)" est que chaque variable ne peut apparaître plus de trois fois dans une formule, le problème reste NP-complet. A travers la construction d'un graphe pour chaque instance du problème "not-all-equal at most 3-SAT(3V)" dont la valeur par rapport (MaxIm) permettra de renseigner la satisfaisabilité de φ, nous allons réduire ce problème au problème (MaxIm), ce qui nous permet de conclure le résultat suivant.

Théorème 1. MaxIm *est* NP*-complet et inapproximable à plus de* $\frac{1}{2} + \varepsilon$ *pour tout* $\varepsilon \in \mathbb{R}_+^*$*, à moins que* $\mathsf{P} = \mathsf{NP}$.

Nous considérons maintenant le cas des graphes bipartis: si $G = (V_1 \bigsqcup V_2, E)$ est un graphe biparti, l'orientation consistant à affecter à chaque arête dans E une direction de leur extrémité en V_1 à leur extrémité en V_2 a un déséquilibre égal à δ_G, i.e. optimal. Ce cas simple permet d'obtenir le minorant suivant :

Théorème 2. *Pour tout graphe* G*,*

$$\textsc{MaxIm}(G) \geq \left\lceil \frac{\delta_G}{2} \right\rceil - 1.$$

La preuve de ce théorème passe par la consruction d'une orientation dont le déséquilibre est minoré par $\frac{\left\lceil \frac{\delta_G}{2} \right\rceil - 1}{\delta_G} \cdot \textsc{MaxIm}(G)$, ce qui met en évidence un algorithme polynomial $(\frac{1}{2} - \frac{1}{\delta_G})$-approché en général et $(\frac{1}{2})$-approché lorsque $\delta_G \equiv 0 \pmod 4$ ou $\delta_G \equiv 1 \pmod 4$.

3.2 Caractériser les graphes tels que MaxIm$(G) = 0$

Nous définissons la classe de graphes $\mathcal{C}^{odd}$ de la façon suivante : un graphe simple G appartient à $\mathcal{C}^{odd}$ s'il existe $C_1, \cdots, C_n$ cycles impairs $(n \geq 1)$ tels que :

- $\cup_{i=1}^{n} C_i = G$,
- $|V(\cup_{k=1}^{i-1} C_k) \cap V(C_i)| = 1, \ \forall i \in [\![2, n]\!]$.

Alors nous montrons le résultat suivant.

Théorème 3. *Pour tout graphe simple G, $\textsc{MaxIm}(G) = 0$ si et seulement si $G \in \mathcal{C}^{odd}$.*

Nous donnons ensuite la caractérisation plus élégante suivante.

Corollary 3. *Pour tout graphe simple G,*

$$\textsc{MaxIm}(G) = 0 \Leftrightarrow G \text{ est Eulérien sans cycle pair}$$

3.3 Algorithme exact pour les cactus

La famille de graphes présentée dans la sous-section 3.1 est un cas particulier de cactus. Un *cactus* est un graphe simple connecté tel que chacune de ses arêtes est contenue dans au plus un cycle du graphe. De façon équivalente, un cactus est un graphe connecté tel que chaque bloc (ou sous-graphe biconnexe maximal), est une arête simple ou un cycle. En mettant en valeur la structure arborescente des blocs et points d'articulations d'un graphe connexe, nous présentons un algorithme polynomial résolvant le problème (MaxIm) pour les cactus dont les valeurs possibles sont 0,1 ou 2 car leur degré minimum est au plus 2.

Théorème 4. *Pour tout cactus G, $\textsc{MaxIm}(G)$ peut être calculé en temps polynomial.*

3.4 Formulations en programmation mixte

En prenant une variable d'orientation $x \in \{-1,1\}^E$, le problème (MaxIm) peut être exprimé de la façon non-linéaire suivante :

$$\begin{cases} \max h \\ \text{s.t.} \\ h \leq |B_v x|, \ \forall v \in V \\ x \in \{-1,1\}^{|E|}. \end{cases}$$

Si nous prenons maintenant une version $0-1$ de cette formulation, mettons la fonction objectif au carré et linéarisons par l'ajout de variables produits, nous obtenons la formulation linéaire en programmation mixte suivante pour (MaxIm) avec $O(m^2)$ variables, dont $O(m)$ sont des variables entières et $O(m^2)$ contraintes.

$$(\mathsf{MIP1}) \begin{cases} \max h \\ \text{s.t.} \\ h \leq d(v) + 2\sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv}B_{v,wv}(4z_{uv,wv} - 2x_{uv} - 2x_{wv} + 1), \ \forall v \in V \\ z_{uv,wp} \leq x_{uv} \\ z_{uv,wp} \leq x_{wp} \qquad\qquad , \ \forall uv, wp \in E, \ uv \neq wp \\ z_{uv,wp} \geq x_{uv} + x_{wp} - 1 \\ x \in \{0,1\}^{|E|}, \ z_{uv,wp} \geq 0, \ uv, wp \in E, \ uv \neq wp, \ h \in \mathbb{R}. \end{cases}$$

Sa relaxation linéaire donne le degré minimum du graphe entré, i.e. le majorant naturel de MaxIm(G). Nous présentons une seconde formulation en programmation mixte avec un nombre réduit de variables et de contraintes. Outre une variable d'orientation, cette seconde formulation met en jeu des variables indicatrices y_k^v avec $v \in V$ un sommet de G et $k \in [\![-d(v), d(v)]\!]$ interprétées comme suit: $y_k^v = 1$ si et seulement si $B_v x = d_x^+(v) - d_x^-(v) = k$.

$$(\mathsf{MIP2}) \begin{cases} \max h \\ \text{s.t.} \\ h \leq \sum_{\substack{k \in [\![-d(v),d(v)]\!] \\ k \equiv d(v)[2]}} |k| y_k^v, \ \forall v \in V \\ \sum_{\substack{k \in [\![-d(v),d(v)]\!] \\ k \equiv d(v)[2]}} y_k^v = 1, \ \forall v \in V \\ \sum_{\substack{k \in [\![-d(v),d(v)]\!] \\ k \equiv d(v)[2]}} k y_k^v = B_i x, \ \forall v \in V \\ x \in [-1;1]^{|E|}, \ y_k^v \in \{0,1\}, \ \forall (v,k) \in V \times [\![-d(v), d(v)]\!], \text{ s.t. } k \equiv d(v)[2], \ h \in \mathbb{R}. \end{cases}$$

Théorème 5. *Pour tout graphe G,*

$$\mathsf{MIP2}(G) = \textsc{MaxIm}(G),$$

où $\mathsf{MIP2}(G)$ *est la racine carrée de la valeur optimale de* $\mathsf{MIP2}$ *pour* G.

Cette formulation a $O(m+n)$ variables, dont $O(m)$ sont des variable entières et $O(n)$ contraintes et sa relaxation linéaire donne également le degré minimum du graphe entré.

3.5 Renforcer (MIP2)

Nous avons mentionné précédemment la mauvaise qualité des relaxation linéaires de (MIP1) et (MIP2) qui donnent toutes deux le degré minimum du graphe entré. Il est possible d'améliorer la qualité de cette relaxation en ajoutant une méthode de plans coupant à la formulation en programmation mixte par la génération de familles d'inégalités valides pour toutes les solutions de notre problème. Nous nous sommes concentrés sur (MIP2) étant la formulation la plus "prometteuse" selon des résultats préliminaires. Une étude polyédrale détaillée dans la Section 5 permet d'obtenir la famille d'inégalités suivantes valides pour l'ensemble des solutions faisables de (MIP2) .

$$h \leq u - \sum_{v=1}^{n} \sum_{\substack{k \in [\![0,u-1]\!] \\ k \equiv d(v) \pmod 2}} \lambda_k^v (y_k^v + y_{-k}^v), \ \forall \lambda \in \Lambda_u \tag{1}$$

où les sommets du graphe étudié G sont numérotés de 1 à $|V| = n$, et

$$\Lambda_u = \left\{ \lambda = (\lambda_k^v)_{(v,k) \in [\![1,n]\!] \times [\![0,u-1]\!]} \in \mathbb{N}^{nu} \;\middle|\; \begin{array}{l} \lambda_{k+1}^v \leq \lambda_k^v, \ \forall (v,k) \in [\![1,n]\!] \times [\![0,u-2]\!] \\ \sum_{v=1}^{n} \lambda_k^v = u-k, \ \forall k \in [\![0,u-1]\!] \end{array} \right\}.$$

On observe que les coefficients λ_k^v sont des entiers positifs décroissants en k. Pour chaque k, il existe un unique v tel que $\lambda_{k+1}^v = \lambda_k^v - 1$ tandis que $\lambda_{k+1}^w = \lambda_k^w$ pour tout $w \neq v$.

Une autre famille d'inégalités valides provenant de l'étude du comportement des variables d'orientation des arêtes incidentes à un sommet en en faisant varier le déséquilibre du sommet peut être extraite:

$$\sum_{\{vu_1, \cdots, vu_p\} \subseteq \delta(\{v\})} B_{v,vu_i} x_{vu_i} + \sum_{k \in [\![0,p-1]\!]} 2(p-k) y_{2k-d(v)}^v \leq p. \tag{2}$$

pour tout sommet $v \in V$ et $p \in [\![1, d(v)]\!]$.

Nous présentons une troisième famille d'inégalités valides basée le comportement

des variables correspondant aux sommets d'un cycle.

$$\sum_{v \in V(C)} (2y^v_{d(v)} + y^v_{d(v)-2}) \leq |V(C)|, \tag{3}$$

$$\sum_{v \in V(C)} (2y^v_{-d(v)} + y^v_{-d(v)+2}) \leq |V(C)|. \tag{4}$$

où C est un cycle du graphe.

Une dernière famille similaire basée sur les cliques du graphe entrée suit:

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1,d(v))} (p-k) y^v_{d(v)-2k} \leq \frac{p(p+1)}{2}, \ \forall p \in [\![1, |\mathcal{K}|]\!], \tag{5}$$

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1,d(v))} (p-k) y^v_{2k-d(v)} \leq \frac{p(p+1)}{2}, \ \forall p \in [\![1, |\mathcal{K}|]\!]. \tag{6}$$

où $\mathcal{K}$ est une clique du graphe entré.

Les quatre familles d'inégalités valides pour l'enveloppe convexe des solutions de (MIP2) ont une taille exponentielle en terme de la taille du graphe entré. La complexité de leur séparation est donc essentiel par rapport à notre objectif qui est de les incorporer à la résolution de (MIP2) par le biais d'une méthode de plans coupants. Nous montrons que les problèmes de séparation des familles d'inégalités (2.6), (2.7), (2.8) and (2.9) peuvent être résolus en temps polynomial. Nous complétons ce résultat en fournissant une méthode heuristique pour générer des inégalités de la famille (2.10) et (2.11) à partir d'une solution courante de (MIP2) , ce qui permet l'élaboration d'une méthode de plans coupants efficace performante pour renforcer la formulation (MIP2) .

3.6 Résultats numériques

Nous présentons les résultats de la résolution de nos deux formulations pour un ensemble de graphes dont la nature et la taille est très variées. Pour chaque instance, outre les temps de calcul pour les deux formulations, nous relevons pour (MIP2) le nombre d'inégalités générées pour chacune des familles d'inégalités valides exhibées dans la sous-section précédente. Nous pouvons y constater la meilleure performance de (MIP2) par rapport à (MIP1) de façon générale ainsi que l'amélioration de la qualité de la relaxation linéaire de (MIP2) par la méthode de plans coupants qu'on lui a intégré.

Un autre exemple de fonction objectif f du problème générique menant à un problème d'optimisation connu est le cas $f = \frac{1}{2}||\cdot||_1$ qui va permettre d'obtenir une

approche originale du célèbre problème de la coupe de cardinalité maximale.

4 Coupe de cardinalité maximale

On appelle le poids d'une coupe la somme des poids des arêtes de la coupe. Le problème de la coupe maximum consiste à trouver dans un graphe une coupe de poids maximum. Ce problème est un problème fondamental d'optimisation combinatoire émergeant dans de nombreuses disciplines scientifiques: intégration à très grande échelle [11], calcul de matrices creuses [8], programmation parallèlle [24], physique statistique [11], programmation quadratique [55], affectation de fréquences, etc. Le problème de la coupe maximum est globalement NP-complet [67], et inapproximable à plus de $\frac{16}{17} + \epsilon$ pour tout $\epsilon > 0$ à moins que P = NP [59]. En revanche, peut être résolu en temps polynomial dans certains cas [18].

Plusieurs approches de résolution de ce problème ont été développées à travers les années. Depuis le milieu des années 1990 et l'article novateur de Michel Goemans et David Williamson [47], il y a eu un intérêt croissant pour les algorithmes basées sur la programmation semi-définie positive. Leur travail présente un algorithme 0.87856-approché pour le problème de la coupe maximum. Leur méthode repose sur la formulation semi-définie positive du problème suivante pour un graphe $G = (V, E)$ avec $V = \{1, \cdots, n\}$.

$$(\mathsf{SDP}_{GW}) \begin{cases} Z^{\star}_{\mathsf{SDP}_{GW}} \max \frac{1}{2} \sum_{ij \in E} (1 - y_{ij}) \\ s.t. \\ y_{ii} = 1, \ \forall i \in [\![1, n]\!], \\ Y \succeq 0, \\ Y \in \mathbb{S}^n, \end{cases}$$

où Y représente la matrice dont la composante de la i-ème ligne et la $j$ème colonne est y_{ij}, $Y \succeq 0$ est la contrainte exigeant la semi-définie positivité de Y, et, pour tout entier n, on note $\mathbb{S}^n$ l'ensemble des matrices symmétriques d'ordre n. Cette approche semi-définie positive du problème a également mené à des solveurs efficaces tels que BiqMac [87] et BiqCrunch [70].

Nous considérons le problème consistant à trouver une coupe de cardinalité maximum , i.e. le problème de la coupe maximum où tous les poids sont égaux :

$$(\textsc{MaxCut}) \quad \textsc{MaxCut}(G) = \max_{S \subseteq V} |\delta(S)|.$$

et nous notons $\textsc{MaxCut}(G)$ la valeur de $\textsc{MaxCut}$ pour G.

Pour un graphe $G = (V, E)$, nous considérons le problème de trouver une orien-

tation maximisant $\sum_{v\in V} |d^+(v) - d^-(v)|$. Ce problème est principalement motivé par le résultat suivant.

Théorème 6. *Pour tout graphe G, le cardinal d'une coupe de cardinalité maximum est égal à la moitié du maximum de la somme des déséquilibres de tous les sommets sur toutes les orientations de G :*

$$(\textsc{MaxCut}) \quad \textsc{MaxCut}(G) = \max_{\Lambda\in\vec{O}(G)} \frac{1}{2}\sum_{v\in V} |d^+_\Lambda(v) - d^-_\Lambda(v)|.$$

Nous voyons ici apparaître notre problème générique pour $f = \frac{1}{2}||\cdot||_1$. Différentes approches d'amélioration du majorant fournie par la formulation $(SDP0)$ [47] ont été proposée dans la littérature: mise à profit de propriétés connues du polytopes des coupes et ajout d'inégalités linéaires [40, 61], utilisation de techniques de coupes disjonctives [3, 74].

4.1 Un nouveau majorant issu de la programmation semi-définie positive

Nous commençons par introduire une nouvelle formulation exact de (MaxCut) sous la forme d'un programme linéaire mixte en nombre entiers. Cette formulation met en jeu des variables indicatrices notées y^v_k, avec $v \in V$ et $k \in [\![-d(v), d(v)]\!]$ et sont interprétées de la façon suivante : $y^v_k = 1$ si et seulement si $B_v x = k$.

$$(\mathsf{MIP4}) \begin{cases} Z^\star_{\mathsf{MIP4}} = \max \frac{1}{2}\sum_{v\in V}\sum_{k\in I_v^-\cup I_v^+} |k| y^v_k \\ \text{s.t.} \\ \sum_{k\in I_v} y^v_k = 1, \ \forall v \in V, \\ \sum_{k\in I_v} k y^v_k = \sum_{uv\in E}(\sum_{k\in I_v^+} y^v_k - \sum_{k\in I_u^+} y^u_k), \ \forall v \in V, \\ y^v_k \in \{0,1\}, \ \forall v \in V, \ \forall k \in I_v. \end{cases}$$

où $I_v^- = \left[\!\left[-d(v), -\left\lceil \frac{d(v)}{2}\right\rceil\right]\!\right]$, $I_v^+ = \left[\!\left[\left\lceil \frac{d(v)}{2}\right\rceil, d(v)\right]\!\right]$ et $I_v = I_v^- \cup I_v^+$, pour tout $v \in V$.

La formulation (MIP4) met en jeu $O(|E|)$ variables et $O(|V|)$ contraintes. En partant de (MIP4), nous introduisons maintenant des variables binaires Y^{uv}_{kl}, avec $(u, v) \in V^2, (k, l) \in (I_u^- \cup I_u^+) \times (I_v^- \cup I_v^+)$, représentant le produit de variables $y^u_k y^v_l$. Ceci permet de renforcer (MIP4) avec des techniques reformulation et linéarisation consistant à prendre des contraintes de a formulation de base par certaines variables et en linéarisant grâce aux variables produits. Nous nous intéressons à la relaxation semi-définie positive de ce renforcement.

$$
(\mathsf{SDP5}) \begin{cases} Z^\star_{\mathsf{SDP5}} = \max \frac{1}{2} \sum_{v\in V} \sum_{k\in I_v^- \cup I_v^+} |k| Y_{kk}^{vv} \\ \text{s.t.} \\ \sum_{k\in I_v} Y_{kk}^{vv} = 1,\ \forall v \in V, \\ \sum_{k\in I_v} k Y_{kk}^{vv} = \sum_{uv\in E} \left(\sum_{k\in I_v^+} Y_{kk}^{vv} - \sum_{k\in I_u^+} Y_{kk}^{uu} \right),\ \forall v \in V, \\ Y_{kk}^{vv} = \sum_{l\in I_u} Y_{kl}^{vu},\ \forall u, v \in V,\ \forall k \in I_v, \\ (d(v)-k) Y_{kk}^{vv} = \sum_{uv\in E} \sum_{l\in I_u^+} Y_{kl}^{vu},\ \forall v \in V,\ \forall k \in I_v^+, \\ -k Y_{kk}^{vv} = \sum_{uv\in E} \sum_{l\in I_u^+} Y_{kl}^{vu},\ \forall v \in V,\ \forall k \in I_v^-, \\ \sum_{l\in I_v} l Y_{kl}^{vu} = \sum_{uw\in E} \left(\sum_{l\in I_w^+} Y_{kl}^{vw} - \sum_{l\in I_u^+} Y_{kl}^{vu} \right),\ \forall v \neq u \in V,\ \forall k \in I_v, \\ Y - Diag(Y) Diag(Y)^t \succeq 0 \\ Y_{kl}^{vu} \geq 0, \forall u, v \in V,\ \forall (k,l) \in I_v \times I_u, \end{cases}
$$

où $Diag(Y)$ est le vecteur correspondant à la diagonale de la matrice Y. Pour tout graphe G, $Z^\star_{\mathsf{SDP}_{GW}}$ et $Z^\star_{\mathsf{SDP5}}$ sont tous les deux des majorants de MaxCut(G).

Proposition 4. *Pour tout graphe G, $Z^\star_{\mathsf{SDP5}} \leq Z^\star_{\mathsf{SDP}_{GW}}$.*

Le nouveau majorant $Z^\star_{\mathsf{SDP5}}$ est exact (i.e., égal à $w^\star$) pour certaines familles de graphes. Nous savons que $Z^\star_{\mathsf{SDP}_{GW}}$ est exacte en ce qui concerne les graphes complets d'ordre pair mais pas pour ceux d'ordre impair. Parallèlement, $Z^\star_{\mathsf{SDP5}}$ est exacte pour tous les graphes complets. Bien que nous n'ayons encore pu le démontré, les résultats numériques que $Z^\star_{\mathsf{SDP5}}$ est exacte aussi pour les graphes roues.

4.2 Formulations en programmation mixte additionnelles

Nous présentons maintenant trois nouvelles formulations en programmation mixte pour le problème de la coupe de cardinalité maximum basées sur notre approche du problème comme un problème d'orientation de graphe. Ces formulations présentent des performances numériques intéressantes. La première découle de (MIP4) en utilisant le fait que $\sum_{k\in I_v^-} y_k^i = 1 - \sum_{k\in I_v^+} y_k^i$ pour tout $v \in V$, ce qui nous autorise à supprimer toutes les variables y_k^i avec $k \leq 0$. On obtient ainsi la formulation exacte suivante.

$$
(\mathsf{MIP5}) \begin{cases} \max \sum_{v\in V} \sum_{k\in I_v^+} k y_k^v \\ \text{s.t.} \\ \sum_{k\in I_v^+} y_k^v \leq 1,\ \forall v \in V, \\ \sum_{uv\in E} (\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u) \leq \sum_{k\in I_v^+} k y_k^v - \left\lceil \frac{d(v)}{2} \right\rceil (1 - \sum_{k\in I_v^+} y_k^v),\ \forall v \in V, \\ \sum_{uv\in E} (\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u) \geq \sum_{k\in I_v^+} k y_k^v - d(v)(1 - \sum_{k\in I_v^+} y_k^v),\ \forall v \in V, \\ y_k^v \in \{0,1\},\ \forall v \in V, \forall k \in I_v^+. \end{cases}
$$

Elle met en jeu à peu près deux fois moins de variables que (MIP4) et présente des performances globalement meilleures, des détails sur ces résultats se trouvent dans la Section 3.5. Afin de réduire encore le nombre de variables dans une formulation exacte, nous pouvons envisager d'aggréger les variables y_k^v avec $k \in I_v^+$ pour un sommet v dans une seule variable x^v égale à $\sum_{k\in I_v^+} y_k^v$. Ce faisant, il nous faut une autre variable z^v égale à $\sum_{k\in I_v^+} k y_k^v$ afin de conserver l'information du déséquilibre de v qui est nécessaire pour le calcul de la valeur de l'orientation dans la fonction objectif. On obtient donc la formulation suivante.

$$(\mathsf{MIP7}) \begin{cases} \max \sum_{v\in V} z^v \\ \text{s.t.} \\ \left\lceil \frac{d(v)}{2} \right\rceil + \left\lfloor \frac{d(v)}{2} \right\rfloor x^v - z^v \leq \sum_{uv\in E} x^u \leq d(v) - z^v, \ \forall v \in V, \\ \left\lfloor \frac{d(v)}{2} \right\rfloor x^v \leq z^v \leq d(v) x^v, \ \forall v \in V, \\ x \in \{0,1\}^V, \ z \in \mathbb{R}^V. \end{cases}$$

(MIP7) contient $2|V|$ variables, dont la moitié sont des variables entières et fait preuve d'une meilleure performance que (MIP6) sur de nombreuses instances (voir Section 3.5). Il y a au contraire des instances pour lesquelles (MIP7) se montre bien moins performante que (MIP6), suggérant que ce processus d'agrégation des variables est un peu brutal. Nous proposons par conséquent une troisième formulation pensée comme un compromis entre (MIP6), i.e. aucune aggrégation des variables (MIP7), i.e. aggrégation totale des variables pour chaque sommet. Pour ce faire, nous partitionnons l'interval I_v^+ pour chaque sommet $v \in V$: soit $\alpha > 1$, nous paramétrons notre partition par α au moyen des suites d'entiers suivantes pour chaque sommet $v \in V$

$$\begin{cases} a_1^v = \left\lceil \frac{d(v)}{2} \right\rceil, \\ a_i^v = 1 + b_{i-1}^v, \text{ pour } i > 1, \\ b_i^v = \min(\lfloor \alpha * a_i^v \rfloor, d(v)), \end{cases}$$

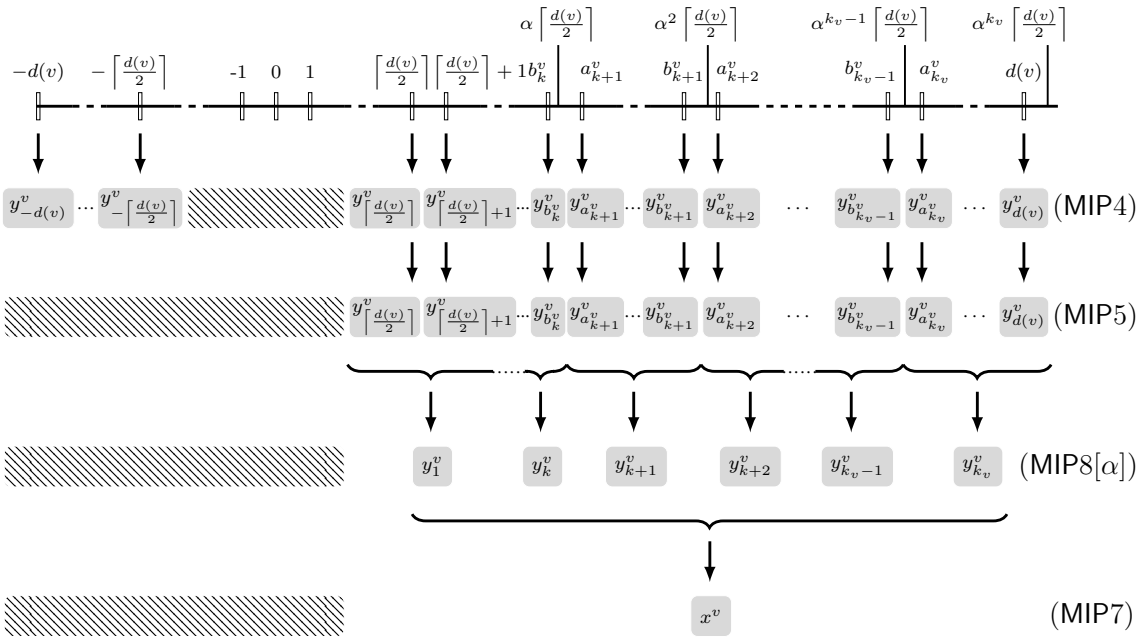
et prenons k_v, le plus petit entir tel que $b_{k_v}^v = d(v)$. Alors, comme dans les formulations (MIP4), (MIP5), (SDP5) et (MIP6), nous faisons appel à des variables indicatrices y_k^v pour chaque sommet $v \in V$ et chaque $k \in [\![1, k_v]\!]$ dont l'interprétation est la suivante: $y_k^v = 1$ si et seulement si $z^v \in [\![a_k^v, b_k^v]\!]$. On obtient par conséquent la

formulation suivante pour tout $\alpha > 1$.

$$(\mathsf{MIP8}[\alpha]) \begin{cases} \max \sum_{v \in V} z^v \\ \text{s.t.} \\ \left\lceil \frac{d(v)}{2} \right\rceil + \left\lfloor \frac{d(v)}{2} \right\rfloor x^v - z^v \leq \sum_{uv \in E} x^u \leq d(v) - z^v, \ \forall v \in V, \\ \sum_{k=1}^{k_v} y_k^v = x^v, \ \forall v \in V, \\ \sum_{k=1}^{k_v} a_k^v y_k^v \leq z^v \leq \sum_{k=1}^{k_v} b_k^v y_k^v, \ \forall v \in V, \\ x \in [0,1]^V, \ y^v \in \{0,1\}^{k_v}, \ \forall v \in V, \ z \in \mathbb{R}^V. \end{cases}$$

Ces trois formulation exactes que nous donnons pour le problème de la coupe de cardinalité maximum sont basées sur la même approche. La Figure 4 schématise les différences théoriques entre ces formulations ainsi qu'avec (MIP4). Pour chacune de ces formulations, en dessous d'un entier représenté sur le segment du haut, il y a soit une zone hachurée indiquant que cet entier n'est pas considéré comme une valeur possible de déséquilibre du sommet v dans cette formulation, soit une variable indicatrice qui sera présente dans la formulation pour renseigner si le déséquilibre de v prend cette valeur ou non, soit une accolade qui signifie que cette valeur fait partie d'un intervalle correspondant tout entier à une variable indicatrice pour cette formulation.

Figure 4: Distribution des variables indicatrices pour un sommet v sur le segment $[\![-d(v), d(v)]\!]$ pour les formulations (MIP4), (MIP5), (MIP7) and (MIP8[α])



Dans le but de mesurer les performances absolues de nos nouvelles formulations, nous mentionnons maintenant une dernière formulation exacte pour le problème de la coupe de cardinalité maximum basée sur les inégalités triangulaires. Elle met en

jeu une variable $x_{i,j}$ pour chaque paire non-ordonnée de sommets $\{i,j\} \subset V$ $(i \neq j)$. Elle a donc $O(n^2)$ variables et $O(n^3)$ contraintes.

$$(\mathsf{MIP9}) \begin{cases} \max \sum_{ij \in E} x_{i,j} \\ s.t. \\ x_{i,j} + x_{j,k} + x_{i,k} \leq 2, \ \forall \{i,j,k\} \subset V, \ |\{i,j,k\}| = 3, \\ x_{i,j} + x_{j,k} - x_{i,k} \geq 0, \ \forall (i,j,k) \in V^3, \ |\{i,j,k\}| = 3, \\ x_{i,j} \in \{0,1\}, \ \forall \{i,j\} \subset V, \ i \neq j. \end{cases}$$

4.3 Résultats numériques

Des calculs ont été lancés afin d'évaluer la qualité de notre nouveau majorant issu de la programmation semi-définie positive. Pour chaque instance, nous relevons les valeurs optimales des formulations (SDP_{GW}) et ($\mathsf{SDP5}$) que nous comparons à la valeur du problème MaxCut(G). Nous présentons également des résultats concernant les performances des différentes formulations exactes en programmation mixte en nombres entiers pécédemment présentées. Nous relevons les temps de calcul de chacune des formulations exactes que nous avons introduites ainsi que celle basée sur les inégalités triangulaires en tant que témoins de même que le temps de calcul du solveur BiqCrunch (BC) pour chaque instance. On observe bien que $Z^{\star}_{\mathsf{SDP5}} =$ MaxCut(G) pour les graphes complets, ce qui semble être valable également pour les roues (en se basant uniquement sur les résultats numériques). Les résultats numériques montrent un écart significatif entre notre majorant et celui issu de la relaxation de Goemans & Williamson. Regardons maintenant les temps de calcul des formulations exactes sur des instances plus grandes.

En ce qui concerne les formulations exactes, on observe en premier lieu que en dehors du cas des graphes aléatoires, nos nouvelles formulations ont des performances globalement meilleures que la formulation des classique des inégalités triangulaires. Plus spécifiquement, on voit que ($\mathsf{MIP6}$) est globalement plus performante que ($\mathsf{MIP4}$) et sur certaines instances, ($\mathsf{MIP6}$) est drastiquement meilleure que ($\mathsf{MIP7}$), alors que sur d'autres, on observe le phénomène inverse. Le plus remarquable est sans doute que ($\mathsf{MIP8}$) étant un compromis entre ($\mathsf{MIP6}$) et ($\mathsf{MIP7}$), il existe pour presque chaque instance une valeur de α pour laquelle ($\mathsf{MIP8}[\alpha]$) a le plus petit temps de calcul.

Nous étudions à présent une famille de polyèdres faisant naturellement surface dans la recherche d'inégalités valides pour l'ensemble des solutions de la formulation exacte ($\mathsf{MIP2}$) pour le problème de l'orientation de déséquilibre maximum. Cette famille de polyèdres apparaît également dans le contexte d'autres problèmes sujets à des méthodes de résolution en programmation linéaire.

5 Étude polyèdrale

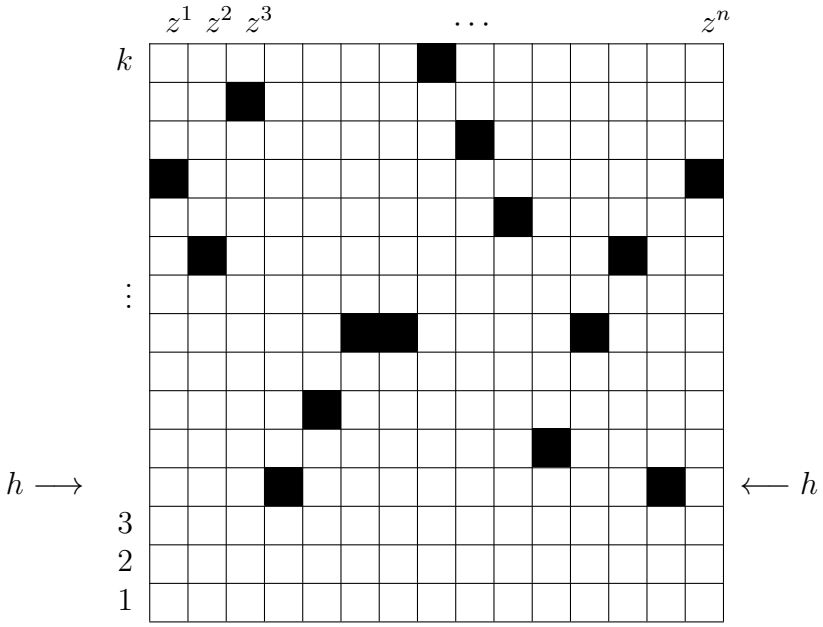


Figure 5: Sur cette grille représentant une matrice k-par-n, chaque colonne est une variable z^i et la hauteur de la cellule noire qu'elle contient correspond à sa valeur affectée. h correspond donc à la hauteur de la ligne non-vide la plus basse (dans ce cas, $h = 4$).

Nous considérons les problèmes d'optimisation combinatoire mettant en jeu $n \in \mathbb{N}^*$ variables z^i, chacune d'entre elles pouvant prendre pour valeur un entier dans $[\![1, k]\!]$ ainsi que $h \equiv \min_{i=1}^{n} z^i$. Un polytope naturellement lié à ces problèmes est le suivant.

$$P = Conv\left\{ \left(\begin{bmatrix} y_k^1 & y_k^2 & \cdots & y_k^n \\ \vdots & \vdots & \ddots & \vdots \\ y_2^1 & y_2^2 & \cdots & y_2^n \\ y_1^1 & y_1^2 & \cdots & y_1^n \end{bmatrix}, h \right) \in M_{k,n} \times \mathbb{N}^* \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1, \ \forall i \in [\![1, n]\!], \\ h = \min\limits_{i \in [\![1,n]\!]} \sum_{l=1}^{k} l y_l^i, \end{array} \right\},$$

où $M_{k,n}$ est l'ensemble des matrices k-par-n à coefficients dans $\{0, 1\}$ et les variables y_j^i sont interprétées de la façon suivante : $y_j^i = 1$ si et seulement si $z^i = j$. La matrice $(y_j^i)_{i,j}$ peut être vue comme une matrice binaire d'affectation où chaque colonne contient exactement un coefficient égal à 1 et h correspond à l'indice de sa plus basse ligne non-identiquement nulle (cf Fig. 4.1).

Une variante de P peut être obtenue en considérant l'indice de la plus haute ligne non-identiquement nulle d'une matrice binaire d'affectation. On obtient ainsi

le polytope suivant :

$$P' = Conv\left\{ \left(\begin{bmatrix} x_k^1 & x_k^2 & \cdots & x_k^n \\ \vdots & \vdots & \ddots & \vdots \\ x_2^1 & x_2^2 & \cdots & x_2^n \\ x_1^1 & x_1^2 & \cdots & x_1^n \end{bmatrix}, g \right) \in M_{k,n} \times \mathbb{N}^* \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} x_l^i = 1, \ \forall i \in [\![1,n]\!], \\ g = \max\limits_{i \in [\![1,n]\!]} \sum_{l=1}^{k} l x_l^i, \end{array} \right\}.$$

Les polytopes P et P' apparaissent naturellement dans le contexte de divers problèmes d'optimisation combinatoire tels que les problèmes d'affectation de fréquences [1], les problèmes d'ordonnancement des tâches [94], le problème de la clique maximum [80] ou encore le problème de l'orientation la plus déséquilibrée dans un graphe : (MaxIm). Ces problèmes peuvent donc grandement bénéficier d'une description complète de ces polytopes.

Théorème 7.

$$P = \left\{ \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1, \ \forall i \in [\![1,n]\!], \\ \sum_{l=2}^{k} \sum_{i=1}^{n} \lambda_l^i y_l^i \geq h - 1, \ \forall \lambda \in \Lambda, \\ \sum_{l=1}^{h_{\max}-1} \sum_{i=1}^{n} (l - h_{\max}) y_l^i + h_{\max} \leq h, \ \forall h_{\max} \in [\![1,k]\!], \\ y_l^i \geq 0, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!], \ h \in \mathbb{R}, \end{array} \right.$$

avec

$$\Lambda = \left\{ \lambda = (\lambda_l^i)_{(i,l) \in [\![1,n]\!] \times [\![1,k]\!]} \in \mathbb{N}^{nk} \;\middle|\; \begin{array}{l} \lambda_{l+1}^i \geq \lambda_l^i, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k-1]\!] \\ \sum_{i=1}^{n} \lambda_l^i = l - 1, \ \forall l \in [\![1,k]\!] \end{array} \right\}.$$

P est défini par n inégalités, kn contraintes de positivité, k contraintes du type $\sum_{l=1}^{h_{\max}-1} \sum_{i=1}^{n} (l - h_{\max}) y_l^i + h_{\max} \leq h$ et n^{k-1} du type $\sum_{l=2}^{k} \sum_{i=1}^{n} \lambda_l^i y_l^i \geq h - 1$. Le nombre total de contraintes est donc exponentiel. En revanche, ke problème de séparation inhérent $= P$ consistant à décider si un vecteur $(y,h) \in \mathbb{R}^{nk+1}$ est dans P, et si non, à retourner une contrainte de P violée par (y,h) peut être résolu en temps polynomial. Ceci est crucial dans le contexte d'algorithmes de plans coupants où seules les inégalités violées par la solution courante sont ajoutées au modèle, et non toutes les inégalités.

Les formulations en programmation linéaire visant à maximiser (resp. minimiser) l'indice de la plus basse (resp. haute) ligne non-identiquement nulle d'une matrice d'affectation sont également liés au polytope Q (resp. Q') décrit ci-après. Observons qu'il est simplement éxigé de h (resp. g) qu'il soit inférieur (resp. supérieur) ou égal à $\min_{i \in [\![1,n]\!]} \sum_{l=1}^{k} l y_l^i$ (resp. $\max_{i \in [\![1,n]\!]} \sum_{l=1}^{k} l x_l^i$).

$$Q = Conv\left\{ \left(\begin{bmatrix} y_k^1 & y_k^2 & \cdots & y_k^n \\ \vdots & \vdots & \ddots & \vdots \\ y_2^1 & y_2^2 & \cdots & y_2^n \\ y_1^1 & y_1^2 & \cdots & y_1^n \end{bmatrix}, h \right) \in M_{k,n} \times \mathbb{N}^* \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1,\ \forall i \in [\![1,n]\!], \\ h \leq \min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i, \end{array} \right\}$$

$$Q' = Conv\left\{ \left(\begin{bmatrix} x_k^1 & x_k^2 & \cdots & x_k^n \\ \vdots & \vdots & \ddots & \vdots \\ x_2^1 & x_2^2 & \cdots & x_2^n \\ x_1^1 & x_1^2 & \cdots & x_1^n \end{bmatrix}, g \right) \in M_{k,n} \times \mathbb{N}^* \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} x_l^i = 1,\ \forall i \in [\![1,n]\!], \\ g \geq \max_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l x_l^i, \end{array} \right\}$$

Théorème 8.

$$Q = \left\{ \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1,\ \forall i \in [\![1,n]\!], \\ \sum_{l=2}^{k} \sum_{i=1}^{n} \lambda_l^i y_l^i \geq h-1,\ \forall \lambda \in \Lambda, \\ y_l^i \geq 0,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ h \geq 1. \end{array} \right.$$

□

De façon similaire, nous pouvons déduire que le problème de séparation inhérent à Q peut être résolu en temps polynomial également et il en va de même pour P' et Q' dont nous donnons aussi les descriptions complètes.

Théorème 9.

$$P' = \left\{ \begin{array}{l} \sum_{l=1}^{k} x_l^i = 1,\ \forall i \in [\![1,n]\!], \\ \sum_{l=1}^{k-1} \sum_{i=1}^{n} \lambda_l^i x_l^i \leq g-k,\ \forall \lambda \in \tilde{\Lambda}, \\ \sum_{l=g_{\min}+1}^{k} \sum_{i=1}^{n} (l-g_{\min}) x_l^i + g_{\min} \geq g,\ \forall g_{\min} \in [\![1,k]\!], \\ x_l^i \geq 0,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ g \in \mathbb{R}, \end{array} \right.$$

$$Q' = \left\{ \begin{array}{l} \sum_{l=1}^{k} x_l^i = 1,\ \forall i \in [\![1,n]\!], \\ \sum_{l=1}^{k-1} \sum_{i=1}^{n} \lambda_l^i x_l^i \leq g-k,\ \forall \lambda \in \tilde{\Lambda}, \\ x_l^i \geq 0,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ g \leq k \end{array} \right.$$

avec

$$\tilde{\Lambda} = \left\{ \lambda = (\lambda_l^i)_{(i,l)\in[\![1,n]\!]\times[\![1,k]\!]} \in \mathbb{N}^{nk} \;\middle|\; \begin{array}{l} \lambda_{l+1}^i \leq \lambda_l^i,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k-1]\!] \\ \sum_{i=1}^{n} \lambda_l^i = k-l,\ \forall l \in [\![1,k]\!] \end{array} \right\}.$$

6 Recherche future

Le choix de notre problème générique $\max_{\Lambda \in \vec{O}(G)} f(d^+_\Lambda(v_1) - d^-_\Lambda(v_1), \cdots, d^+_\Lambda(v_n) - d^-_\Lambda(v_n))$ se trouve être très pertinent car l'étude de différentes fonctions objectif f possibles mène à une large variété de problèmes d'optimisation de graphes tels que le problème de la coupe maximum ou le calcul du nombre isopérimétrique. Nous avons obtenu des formulations performantes et des bornes de qualité grâce à notre approche de ces problèmes basée sur les orientations. Nous avons également amélioré les relaxations et donc la performance de certaines de ces formulations grâce à l'étude d'une famille de polyèdres émergeant dans le contexte de différents problèmes d'optimisation, étude conclue par un description complète de ces polytopes ainsi qu'une preuve de leur séparation polynomiale. Il serait intéressant d'étudier le comportement de ces problèmes avec un système de poids sur les arêtes.

Abstract

The *imbalance* of a vertex in a directed graph is the absolute value of the difference between its outdegree and indegree. In this thesis we study the problem of orienting the edges of a graph in such a way that the image of the vector which components are the imbalances of the vertices of the graph under an objective function f is maximized. We analyze the problem for several cases of objective functions.

We study among those cases the problem of maximizing the minimum imbalance of all the vertices over all the possible orientations of the input graph. We call this minimum *the imbalance* of the orientation. The higher it gets, the more imbalanced the orientation is. This case is denoted by MaxIm. We first characterize graphs for which the optimal objective value of MaxIm is zero. Next we show that MaxIm is generally NP-complete and cannot be approximated within a ratio of $\frac{1}{2} + \varepsilon$ for any constant $\varepsilon > 0$ in polynomial time unless P = NP even if the minimum degree of the graph δ equals 2. Then we describe a polynomial-time approximation algorithm whose ratio is almost equal to $\frac{1}{2}$. An exact polynomial-time algorithm is also derived for cacti. Finally, two mixed integer linear programming formulations are presented. Several valid inequalities are exhibited with the related separation algorithms. The performance of the strengthened formulations is assessed through several numerical experiments.

Next, we show that the case for $f = \frac{1}{2}||\cdot||_1$ is the famous unweighted maximum cut problem, denoted MaxCut. We introduce some new mixed integer linear programming formulations along with a new semidefinite relaxation shown to be tighter than Michel Goemans & David Williamson's semidefinite relaxation. Theoretical and computational results regarding bounds quality and performance are also reported.

Finally, in order to find families of valid inequalities to strengthen the linear relaxation of some mixed integer programming formulations of the studied problems, we study a specific class of polytopes which hyperplane description will consist in such targeted inequalities. Consider a $\{0, 1\}$ assignment matrix where each column contains exactly one coefficient equal to 1 and let h be the index of the lowest row that is not identically equal to the zero row. The polytope consisting in the

convex hull of all feasible assignments appended with the extra parameter h naturally appears in some formulations of MaxIm. We give a full description of this polytope and some of its variants which naturally appear in the context of several combinatorial optimization problems including, inter alia, frequency assignment, job scheduling, maximum clique. We also show that the underlying separation problems are solvable in polynomial time and thus linear optimization over those polytopes can be done in polynomial time.

Contents

Chapter 1

Introduction

1.1 Basic definitions and notation

1.1.1 Graphs, subgraphs and cuts

A *graph* is an ordered pair $G = (V, E)$ comprising a set $V \neq \emptyset$ of *vertices* (or *nodes*) together with a set E of *edges*, which are 2-elements subsets of V. For an edge $e = \{u, v\} \in E$ (i.e. $u, v \in V$ and $u \neq v$), we say that

- e is *incident* to u and v,
- u and v are the *endpoints* of e,
- u and v are *adjacent*,
- u and v are *connected*,
- u and v are *neighbours*,

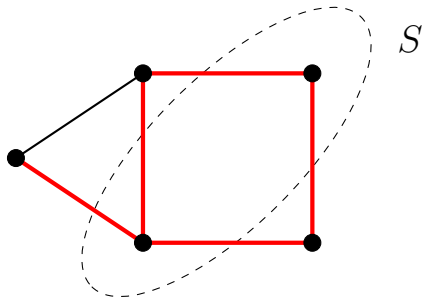
and e will simply be denoted as uv or vu. Typically, a graph is represented in diagrammatic form as dots symbolizing the vertices, connected by lines (which length of shape is irrelevant) symbolizing the edges. This definition of a graph is often given under the name of simple and/or unweighted graph to avoid ambiguity. Simple as opposed to multigraph which allows the presence of multiple edges connecting two vertices as well as loop edges, connecting a vertex to itself. Unweighted to stipulate the absences of weights on edges and/or vertices. We shall deal with no multigraph and seldom use weights, that is never without specifying it precisely beforehand. We therefore stick with our denomination of an unweighted simple graph simply as a graph.

A *subgraph* of G is another graph formed from a subset of the vertices and edges of G with the constraint that its vertex subset must contain all the endpoints of the

edges of the edge subset. An *induced* subgraph is one that includes all the edges whose endpoints belong to the vertex subset. For any graph or subgraph H we will use the notations $V(H)$ and $E(H)$ to refer to the set of vertices of H and the set of edges of H respectively. By extension, for a subset of vertices $S \subseteq V$, $G(S)$ will denote the subraph of G induced by S and $E(S)$ will denote its set of edges, in other words, all the edges having both endpoints in S.

For a vertex $v \in V$, we call the *neighbourhood* of v in G the set of neighbours of v in G and denote it $N_G(v)$ or $N(v)$. Then we call the *degree* of v in G the cardinality of the neighbourhood of v in G (i.e. the number of edges of G adjacent to v). The degree of v in G is denoted by $d_G(v)$ or $d(v)$ and the minimum (resp. maximum) degree of the vertices of G is denoted by δ_G (resp. Δ_G). Given an edge subset T, the *incidence vector* of T denoted x^T is the element of $\{0,1\}^{|E|}$ where $x^T_e = 1$ if and only if $e \in T$. Given a node subset $S \subseteq V$, the *cut* defined by S, denoted $\delta(S)$, is the set of edges having exactly one endpoint in S, i.e. $\delta(S) = \{uv \in E \colon |\{u,v\} \cap S| = 1\}$. An example is depicted in Figure 1.1.

Figure 1.1: Example of a cut (thicker edges in red) defined by a dashed-circled subset of vertices $S \subseteq V$



1.1.2 Paths, trees and connectivity

Let $(u,v) \in V^2$ be a pair of vertices of V, a *path* between (or connecting) u and v (or *uv-path*) is a finite sequence of vertices $p = (u_1, \cdots, u_k)$ such that $u_1 = u$, $u_k = v$ and $u_i u_{i+1} \in E$, $\forall i \in [\![1, k-1]\!]$; its *length* is then $k-1$. A path is therefore a subset of vertices, it can also be seen as a subgraph of G with $V(p) = \{u_1, \cdots, u_k\}$ and $E(p) = \{u_1u_2, \cdots, u_{k-1}u_k\}$. If no two vertices in $(u_1, \cdots, u_k)$ are equal (except for u_1 and u_k possibly), then p is called an *elementary path* and if $u = v$, then p is called a *cycle*. We talk of *even* (resp. *odd*) cycle for a cycle of even (resp. odd) length. A cycle of length 3 is called a *triangle* and an edge connecting two non-consecutive vertices of a cycle is called a *chord* of the cycle, it is therefore an edge of the subgraph induced by the cycle, but not an edge of the cycle as a subgraph. A

graph that contains no cycle is called a *forest*. The *distance* between two vertices u and v in G denoted $d(u, v)$ is the length of a shortest path (i.e. a path with minimal length) connecting u and v in G (by convention, $d(u, u) = 0$ for any vertex u and $d(u, v) = +\infty$ if there is no uv-chain). The *diameter* of G is the largest distance between any two distinct vertices of G, i.e. $\max_{(u,v)\in V^2} d(u, v)$. The *radius* of G is the minimum distance between a vertex in G and the vertex most distant from it in G, i.e. $\min_{u\in V} \max_{v\in V} d(u, v)$. If there is a path connecting any two distinct vertices in G, then G is *connected*. A graph is therefore connected if and only if its diameter is finite. A connected forest is called a *tree*. A *connected component* of G is a maximal connected subgraph of G and we will denote by $\pi(G)$ the number of connected components of G. A graph is therefore connected if and only if it has one connected component and a connected component of a forest is a tree. A vertex $v \in V$ of G is a *cut-vertex* if $\pi(G(V\backslash\{v\})) > \pi(G)$. Correspondingly, an edge of G is a *bridge* if its removal strictly increases the number of connected components in G. If G has no cut-vertex, then it is *biconnected* (or *2-connected*). A *biconnected component* (or *block*) of G is a maximal biconnected subgraph of G.

1.1.3 Orienting the edges

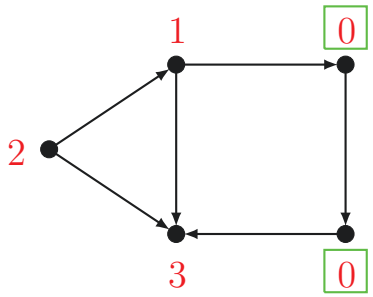
An edge $e = \{u, v\} \in E$ can be assigned an *orientation* (or *direction*), that is its endpoints are ordered, one of them is called the *origin* and the other the *destination* and e is then said to be *oriented* (or *directed*) *from* the origin *to* the destination. An oriented edge is called an *arc* and is therefore an oriented pair of distinct vertices. An arc $a = (u, v) \in V^2$ (i.e. $u \neq v$) is denoted by $\overrightarrow{uv}$ or $\overleftarrow{vu}$ and we say that a is an *outgoing* (resp. *incoming*) arc (or edge) of u (resp. v). By extension, we call an *orientation* of G an assignment of a direction to each (undirected) edge uv in E, i.e. any function on E such that $\Lambda(uv) \in \{\overrightarrow{uv}, \overrightarrow{vu}\}$, $\forall\{uv\} \in E$. Let $\vec{O}(G)$ denote the set of all the orientations of G.

If G is given an orientation $\Lambda \in \vec{O}(G)$ and a path $p = (u = u_1, \cdots, u_k = v)$ is such that $\Lambda(u_i u_{i+1}) = \overrightarrow{u_i u_{i+1}},\ \forall i \in [\![1, k-1]\!]$, then p is called a *directed* path from u to v, and if p is a cycle, then it is called a *circuit*. An orientation that contains no circuit is called *acyclic*. If for any two distinct vertices u and v of G there exists a directed path from u to v and a directed path from v to u w.r.t. Λ, then Λ is called a *strong* (or *strongly connected*) orientation and G oriented by Λ is said to be *strongly connected*. For two vertices u and v of G, the *directed distance* from u to v in G w.r.t. Λ denoted $d_\Lambda(u, v)$ is the length of the shortest directed path from u to v. In contrast with the (undirected) distance, $d_\Lambda(u, v)$ does not necessarily coincide with $d_\Lambda(v, u)$, it is even possible that only one of them is finite. The *diameter* of Λ is the largest distance from a vertex in G to another distinct vertex in G w.r.t. Λ,

i.e. $\max_{(u,v)\in V^2} d_\Lambda(u,v)$. The *radius* of Λ is the minimum distance from a vertex in G to the vertex most distant from it in G w.r.t Λ, i.e. $\min_{u\in V} \max_{v\in V} d_\Lambda(u,v)$. An orientation is strong if and only if its diameter is finite.

For an orientation $\Lambda \in \vec{O}(G)$, we call the number of outgoing (resp. incoming) arcs of a vertex $v \in V$ w.r.t. Λ the *outdegree* (resp. *indegree*) of v in G w.r.t. Λ, denoted by $d^+_\Lambda(v)$ or $d^+(v)$ (resp. $d^-_\Lambda(v)$ or $d^-(v)$). Let $v \in V$, We call $|d^+_\Lambda(v) - d^-_\Lambda(v)|$ the imbalance of v w.r.t. Λ and $d^+_\Lambda(v) - d^-_\Lambda(v)$ the signed imbalance of v w.r.t. Λ. Figure 1.2 shows an oriented graph where the imbalances of the vertices are shown in red and the imbalance of the orientation is squared in green.

Figure 1.2: Example of an oriented graph: the red integer next to each vertex is its imbalance w.r.t. the orientation and those which are inside a green square equal the imbalance of the orientation



1.1.4 Special graphs

G is said to be *bipartite* if V can be partitioned into two subsets V_1 and V_2 such that each edge in E has exactly one endpoint in V_1. In other words, it is a graph for which there is a subset of vertices $S \subseteq V$ such that $\delta(S) = E$ (namely V_1 or V_2). A bipartite graph can be denoted $G = (V_1, V_2, E)$. A *complete* graph is a graph for which every pair of distinct vertices is connected. The complete graph with n vertices is denoted K_n and has $\binom{n}{2}$ edges. A graph is said to be *planar* if it can be embedded in he plane, i.e. it can be drawn on the plane in such a way that its edges intersect only at their endpoints (that is to say no edges cross each other). In a graph, an *edge contraction* is an operation consisting in the removal of an edge from the graph followed by the merger of the two vertices it used to connect. A graph H is called a *minor* of the graph G if it can be formed from G by a sequence of edge deletion, vertex deletion and edge contraction; G is then said to be *contractible to* H.

1.2 Bases of theory of computations

1.2.1 Complexity

The *complexity* (or *time complexity* as opposed to space complexity) of an algorithm quantifies the number of elementary operations performed by the algorithm as a function of size n of the input. Since this quantity may vary with different inputs of the same size, the worst case is considered, that is the maximum number of elementary operations performed on any input of size n. The complexity of an algorithm is commonly expressed as an element of the set $O(g(n))$ defined for some function $g \in \mathbb{R}^{+\mathbb{N}}$ as follows. For a function $f \in \mathbb{R}^{+\mathbb{N}}$, $f \in O(g)$ (or $f(n) \in O(g(n))$) if

$$\exists (N, m)\mathbb{N} \times \mathbb{R}^{+} \text{ s.t. } \forall n \in [\![n, +\infty]\!],\ f(n) \leq mg(n).$$

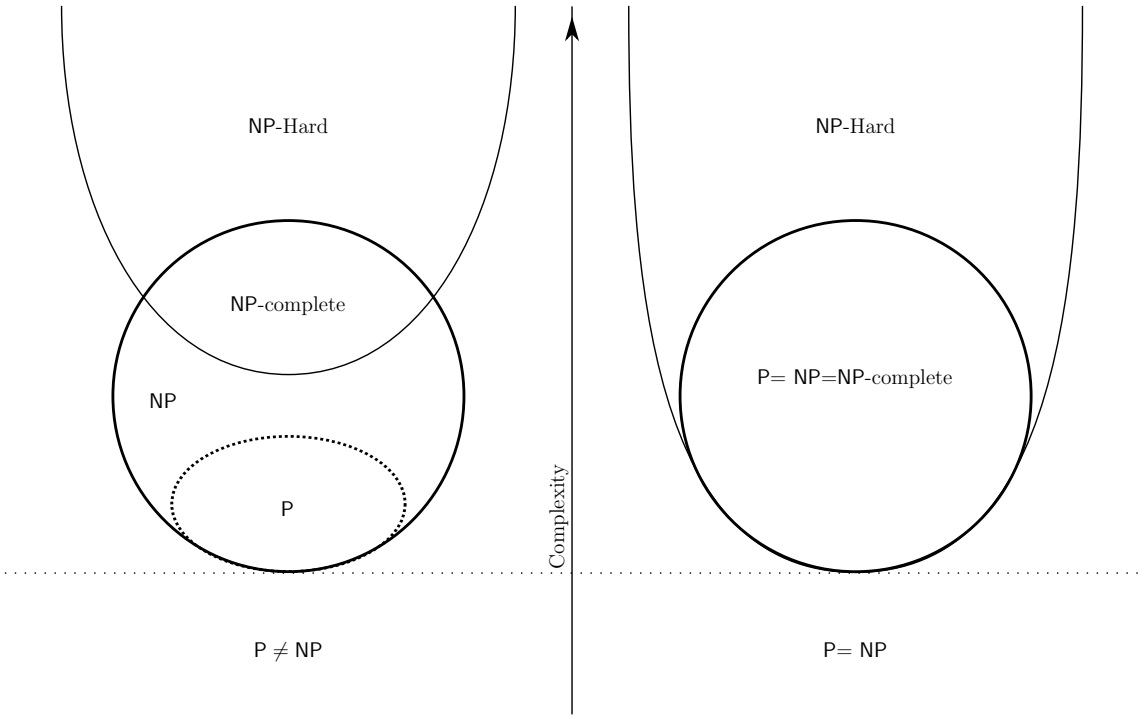
The use of those sets of functions to qualify the complexity of an algorithm excludes coefficients and lower order terms, describing it "asymptotically". As a common abuse of notations, if the complexity of an algorithm is $f(n) \in O(g(n))$, we say that the complexity of the algorithm is $O(g(n))$. If the complexity of an algorithm is a polynomial of the input size n, then it is called a *polynomial* (or *polynomial-time*) algorithm.

1.2.2 Decision problems

A *decision problem* is a problem that has only two possible outputs - "yes" or "no" - on any input. The complexity class P is the set of all the decision problems that can be solved using a polynomial-time algorithm. For example, in 2002, Agrawal et al. gave a polynomial-time algorithm to test if a positive integer is a prime number, thus showing that this problem belongs to P[2]. The complexity class NP is the set of all the decision problems for which the answer is "yes" can be verified using a polynomial-time algorithm. For example, let us consider the subset sum problem; given a set of integers, is there a non-empty subset whose sum is zero? There is no known polynomial time algorithm that can answer this question for any input. And yet if given such a subset, it is very easy to check whether its sum is zero, that is if a solution is given to the problem, answering "yes", there exists a polynomial-time algorithm that can verify if the proposed answer is in fact a solution answering "yes" to the decision problem. Although it is clear that $\mathsf{P} \subset \mathsf{NP}$ the opposite isn't quite clear, in fact the question of the coincidence of P and NP is one of the principal open problems in computer science today. We say that a problem A is *reducible* to a problem B if a polynomial algorithm solving B (whether it exists or not) could be used as an elementary operation in a polynomial algorithm solving A. If A is

reducible to B, the solving A cannot be "harder" than solving B ("harder" here means having a higher "asymptotic" complexity). A problem A belonging to NPis said to be NP-*complete* if any problem of NP is reducible to A. A problem A (not necessarily belonging to NP) is said to be NP-*hard* if any problem of NP is reducible to A. Therefore, A NP-hard problem is NP-complete if and only if it belongs to NP. The relationship between those classes of complexity is illustrated in Figure 1.3, taking into account the coincidence or difference of P and NP.

Figure 1.3: Euler diagram for P, NP, NP-complete and NP-hard sets of problems according to either assumption of the P versus NP problem. Retrieved from the Wikipedia entry "P versus NP" (https://en.wikipedia.org/wiki/P_versus_NP_problem)



1.2.3 Optimization problems

An *optimization problem* is the problem of maximizing (or minimizing) an objective value under certain constraints: the input defines the constraint (for a given input, the constraints are set and we speak of an *instance* of the problem) and the output is the maximum (or minimum) objective value or *optimal objective value*. An optimization problem A has an inherent parameterized decision problem $A(k)$ that consists in answering for a given input if the output w.r.t. A is larger than k. If there is a k for which $A(k)$ is NP-complete (resp. NP-hard), then as an abuse of

language, we say that A is NP-complete (resp. NP-hard). For a maximization (resp. minimization) problem, an *approximation algorithm of ratio* $\rho < 1$ (resp. $\rho > 1$), or *ρ-approximation algorithm*, is an algorithm which value $ALG(x)$ for an input x verifies

$$\begin{aligned} \rho OPT(x) \leq & ALG(x) \leq OPT(x), \\ \text{resp. } OPT(x) \leq & ALG(x) \leq \rho OPT(x), \end{aligned}$$

for any input x, where $OPT(x)$ is the optimal objective value of the problem for the input x.

1.3 Bases of mathematical programming

1.3.1 Convex polytopes

We denote $M_{k\times n}(\mathbb{R})$ the set of matrices with k rows, n columns and which components are elements of $\mathbb{R}$. A *convex polytope* P of dimensions $n \in \mathbb{N}$ is a convex set of points in $\mathbb{R}^n$ with a finite number of extreme points (its *vertices*). It may be defined in several ways, two of which will catch our attention:

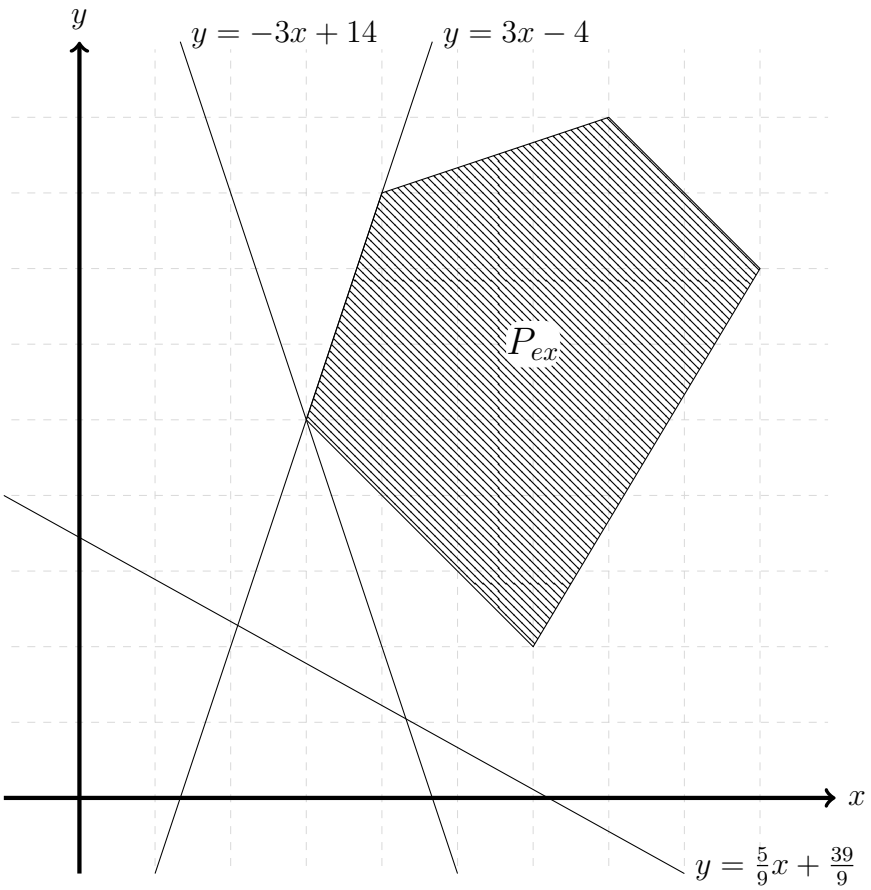
- as the convex hull of a finite set of points: $P = Conv\{x_1, ..., x_k\}$ for some $\{x_1, ..., x_k\} \subset \mathbb{R}^n$,
- as the intersection of a finite number of half-spaces: $P = \{x \in \mathbb{R}^n | Ax \leq b\}$ for some matrix $A \in M_{k\times n}(\mathbb{R})$ and column vector $b \in \mathbb{R}^k$.

The latter is called an *hyperplane representation* of P. Both definitions allow the empty set as a convex polytope, realized either as the convex hull of an empty set of points or the intersection of an empty set of half-spaces. The second definition also allows unbounded convex polytopes which are included in our definition of convex polytopes, even though they can't be realized as the convex hull of a finite set of points. A *face* of P is the intersection of P with a closed half-space whose boundary is disjoint from the interior of P. From this definition it follows that the set of faces of a polytope includes the polytope itself and the empty set. If n is the dimension of P, then a *facet* of P is a face of P which dimension is $n-1$. An hyperplane is said to be *facet-defining* for P if its intersection with P is a facet of P. Consider the 2-dimensional polytope P_{ex} presented in Figure 1.4, the hyperplane $\{y = \frac{5}{9}x + \frac{39}{9}\}$ defines two faces: the whole polytope P_{ex} (the intersection of P_{ex} with $\{y \geq \frac{5}{9}x + \frac{39}{9}\}$) and the empty-face (the intersection of P_{ex} with $\{y \leq \frac{5}{9}x + \frac{39}{9}\}$) with dimension -1

(by convention). The hyperplane $\{y = -3x + 14\}$ defines two faces: the whole polytope P_{ex} (the intersection of P_{ex} with $\{y \geq -3x + 14\}$) and the point $(3, 5)$ (the intersection of P_{ex} with $\{y \leq -3x + 14\}$) which is a face with dimension 0. The hyperplane $\{y = 3x - 4\}$ defines two faces: the whole polytope P_{ex} (the intersection of P_{ex} with $\{y \leq 3x - 4\}$) and the segment $[(3, 5), (4, 8)]$ (the intersection of P_{ex} with $\{y \geq 3x - 4\}$) which is a face with dimension 1, i.e. a facet of P_{ex}. The hyperplane $\{y = 3x - 4\}$ is therefore facet-defining for P_{ex}, and by extensions, we say the same for $y \leq 3x - 4$ which is a valid inequality for P_{ex} (that is verified by all points in P_{ex}) that defines a half-space which boundary intersects P_{ex} in a facet of P_{ex}.

The *separation problem* inherent to P defined by an hyperplane representation consists in deciding if a point $x \in \mathbb{R}^n$ belongs to P and if not, returning a constraint of P violated by x. By extension, one can talk of the separation problem inherent to a set of inequalities as well.

Figure 1.4: Example of half-spaces defining faces of a 2-dimensional polytope P



1.3.2 Linear optimization

A *linear program* (or LP) is an optimization problem consisting in maximizing (or minimizing) a linear objective function f of $\mathbb{R}^n$ over a convex polytope: $\max_{x \in P} f(x)$ or, given an hyperplane representation of P,

$$\begin{cases} \max f(x) \\ \text{s.t.} \\ Ax \leq b. \end{cases}$$

In 1979, Leonid Khachiyan presented the first polynomial time algorithm to solve linear programs: the ellipsoid method [68], but a larger theoretical and practical breakthrough was brought by Narendra Karmarkar in 1984 showing that interior point methods can solve linear optimization in polynomial time [66]. The fact that the ellipsoid method works in polynomial time means that the complexity of the algorithm is a polynomial of the size of the input convex polytope. More precisely, the ellipsoid method uses a separation algorithm on P as a subroutine. So its polynomial complexity is therefore dependent on the polynomial complexity of this subroutine. In other words, if the linear program is the formulation of an optimization problem from which input we derive a convex polytope, then the size of the polytope may not be a polynomial of the size of the problem's input, thus compromising the polynomial complexity of Khachiyan's algorithm for this problem. Martin Grötschel et al. therefore showed that, through the ellipsoid method, the optimization problem over a convex polytope has polynomial complexity if and only if the separation problem inherent to this convex polytope has polynomial complexity [51].

Before these theoretical considerations about linear optimization, over the course of 1946 and 1947, George Dantzig designed the Simplex Algorithm to solve linear programs [30]. The algorithm is based on the fact that if the objective function has a maximum value on a convex polytope with at least one vertex (otherwise it is a half-space and every point on its border is an optimal solution), then it has this value on (at least) one of its vertices. If a vertex is not a maximum point w.r.t. the objective function, then there is an edge containing the point so that the objective function is strictly increasing on the edge moving away from the point. If the edge leads to another vertex, it has a greater objective value than the precedent. The simplex algorithm uses this insight by walking along edges of the polytope to vertices with greater and greater objective value until the maximum value is reached. Even though we have no guarantee concerning the complexity of this algorithm, it is remarkably efficient in practice and was a great improvement over earlier methods and is still today the cornerstone of most linear optimization solvers used in many fields, our work included.

1.3.3 Mixed integer linear optimization

A *mixed integer linear program* (or MIP) is a linear program with the added constraint that some (or all) variables must be integer valued, they are then called *integer variables*. The minimum vertex cover problem consisting in finding a minimum subset of vertices in a graph such all the edges are incident with at least one vertex in the subset is a NP-complete problem [67] that can be reduced to mixed integer linear optimization, showing that it is a generally NP-hard problem. The most common method used to solve mixed integer linear programs and thus most NP-hard combinatorial optimization problems is the *branch & bound* algorithm first proposed by Alisa Land & Alison Doig in 1960 [71]. It consists in alternating two sorts of steps: splitting the set of solutions (i.e. integer valued points in the convex polytope) into smaller spaces (this step is called *branching*), and computing bounds on the objective value of the solutions in the solution subsets in order to eliminate those not containing optimal solutions (this step is called *bounding*). To improve the performance of a method solving mixed integer programs, one can put to use a known set of inequalities verified by any solution of the problem. It consists in solving the linear program inherent to the mixed integer linear program at hand and then solving the separation problem inherent to the afore-mentioned set of inequalities with the fractional optimal solution. If the separation returns a violated inequality, then it can be added to the linear program in order to obtain a fractional optimal solution which value is "closer" to that of the optimal solution. This method called *cutting-plane* introduced by Ralph Gomory in the 1950s highlights the appeal for the separation problem inherent to a set of inequalities to be solvable in polynomial time. Cutting planes have been studied and considered by many people, some papers aim at strengthening the use of various cuts such as Chvtal-Gomory cuts and Gomory fractional cuts (e.g. [76]). If this method is a step added in the process of a branch a bound algorithm, it is then called a *branch & cut* algorithm.

1.3.4 Semidefinite optimization

A matrix $M \in M_{n\times n}(\mathbb{R})$ is *positive semidefinite* if $x^\intercal Mx \geq 0$ for all $x \in \mathbb{R}^n$, and we denote $M \succeq 0$. For any matrix $A \in M_{n\times n}(\mathbb{R})$, the matrix $A^\intercal A$ is positive semidefinite and conversely, any positive semidefinite matrix M can be written as $M = A^\intercal A$ with $A \in M_{n\times n}(\mathbb{R})$ that can be obtained in polynomial time. A *semidefinite program*

(or SDP) is a problem that can be formulated as follows.

$$\begin{cases} \max f(X) \\ \text{s.t.} \\ A_i X \leq bi, \ \forall i \in [\![1, m]\!] \\ X \succeq 0. \end{cases}$$

In other words, semidefinite programs are a generalization of linear programs where instead of using real variables, we use a real-valued positive semidefinite matrix for variable and we are allowed to take the dot product of matrices in the constraints. Tractability-wise, most interior point methods used to solve linear programs have been generalized to solve semidefinite programs. As in linear optimization, these methods have polynomial complexity and perform well in practice with the difference that they output the value of the SDP up to an additive error $\varepsilon > 0$ and their complexity is therefore a polynomial of the input size and $\log(\frac{1}{\varepsilon})$. More about semidefinite programs can be found in [93].

1.4 Graph orientation

Graph orientation is a well studied area in graph theory and combinatorial optimization. A large variety of constrained orientations as well as objective functions have been considered so far.

1.4.1 Degree-constrained orientation

Among all the considered constraints arise the popular degree-constrained orientation problems: in 1976, András Frank & András Gyárfás [43] gave a simple characterization of the existence of an orientation such that the outdgree of each vertex is between a lower and an upper bound.

Theorem 5 (Frank & Gyárfás, 1976)**.** *Let $G = (V, E)$ be a graph and take $L : V \to \mathbb{Z}^+$ and $U : V \to \mathbb{Z}^+ \cup \{\infty\}$ two functions mapping each vertex in V to respectively a lower bound and an upper bound such that $L(v) \leq U(v), \ \forall v \in V$. Then we have*

- ▷ *There exists an orientation $\Lambda \in \vec{O}(G)$ such that $d^+_\Lambda(v) \geq L(v), \ \forall v \in V$ if and only if*

$$|E(S)| + |\delta(S)| \geq \sum_{v \in S} L(v), \ \forall S \subseteq V. \tag{1.1}$$

- ▷ *There exists an orientation $\Lambda \in \vec{O}(G)$ such that $d^+_\Lambda(v) \leq U(v), \ \forall v \in V$ if and*

only if

$$|E(S)| \leq \sum_{v \in S} U(v), \ \forall S \subseteq V. \tag{1.2}$$

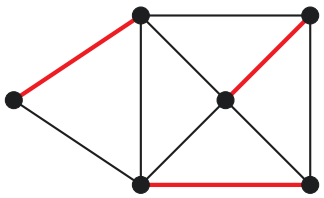
▷ *There exists an orientation* $\Lambda \in \vec{O}(G)$ *such that* $L(v) \leq d^+(v) \leq U(v), \ \forall v \in V$ *if and only if* G *satisfies* (1.1) *and* (1.2).

The proof can either be approached using results on flows or with straightforwards graph arguments. The latter leads to a proof from which a polynomial-time algorithm can be derived which finds a desired orientation or a proof of its non-existence. This result has many consequences, one of which is the renowned marriage theorem which characterizes the existence of a perfect matching (a subset of non incident edges which endpoints cover all V, e.g. Figure 1.5) in a bipartite graph. Note that a bipartite graph $G = (V_1, V_2, E)$ possesses a perfect matching if and only if there exists an orientation $\Lambda \in \vec{O}(G)$ such that

$$\begin{cases} d^+_\Lambda(v) = 1, \ \forall v \in V_1, \\ d^+_\Lambda(v) = d(v) - 1, \ \forall v \in V_2. \end{cases}$$

Then the marriage theorem is a natural result of Theorem 5.

Figure 1.5: Example of a perfect matching: thicker edges in red



Corollary 6 (Hall's marriage theorem, 1935)**.** *Let* $G = (V_1, V_2, E)$ *be a bipartite graph. There exists a perfect matching of* G *if and only if*

$$|S| \leq |N(S)|, \ \forall S \subseteq V.$$

Where $N(S)$ *is the union of all the neighbourhoods of the vertices in* S*.*

Closely related to Theorem 5, Marek Chrobak & David Eppstein give in [26] linear time algorithms to build a 3-bounded outdegree orientation (i.e. an orientation such that the outdegree of every vertices is lower than or equal to 3) and a 5-bounded outdegree acyclic orientation.

A similar result can be given concerning strong orientations, it is associated with Herbert Robbin's Theorem [88].

Theorem 7 (Robbins' Theorem, 1939)**.** *A connected graph G has a strong orientation if and only if it has no bridge.*

Fan Chung provided a linear time algorithm for checking whether a graph has such an orientation and finding one if it does [27]. The generalization of Robbins' Theorem given by Frank & Gyárfás follows.

Theorem 8 (Frank & Gyárfás, 1976)**.** *Let $G = (V, E)$ be a graph with no bridge and take $L : V \to \mathbb{Z}^+$ and $U : V \to \mathbb{Z}^+ \cup \{\infty\}$ two functions mapping each vertex in V to respectively a lower bound and an upper bound such that $L(v) \leq U(v),\ \forall v \in V$. Then we have*

- ▷ *There exists a strong orientation $\Lambda \in \vec{O}(G)$ such that $d^+_\Lambda(v) \geq L(v),\ \forall v \in V$ if and only if*

$$|E(S)| + |\delta(S)| \geq \sum_{v \in S} L(v) + \pi(G(V \backslash S)),\ \forall S \subseteq V,\ S \neq \emptyset. \tag{1.3}$$

- ▷ *There exists a strong orientation $\Lambda \in \vec{O}(G)$ such that $d^+_\Lambda(v) \leq U(v),\ \forall v \in V$ if and only if*

$$|E(S)| + \pi(G(V \backslash S)) \leq \sum_{v \in S} U(v),\ \forall S \subseteq V,\ S \neq \emptyset. \tag{1.4}$$

- ▷ *There exists a strong orientation $\Lambda \in \vec{O}(G)$ such that $L(v) \leq d^+(v) \leq U(v),\ \forall v \in V$ if and only if G satisfies (1.3) and (1.4).*

Theorems 5 and 8 remain valid replacing all occurrences of outdegree by indegree. Another generalization of Robbin's Theorem was given by Crispin Nash-Williams and deals with high arc-strength and edge-connectivity in a graph [83]. A connected graph is *k-edge-connected* if the removal of any $k - 1$ of its edges does not impair its connectivity and a strong orientation of a graph is *k-arc-strong* if its restriction on the graph after the removal of any $k - 1$ edges of the graph is strong. Therefore, a connected graph is 2-edge-connected if and only if it has no bridge.

Theorem 9 (Nash-Williams' orientation Theorem, 1960)**.** *A graph admits a k-arc-strong orientation if and only if it is $2k$-edge-connected.*

Details about the previous results and many others on orientations and directed graphs in general can be found in [10].

1.4.2 Minimizing the maximum outdegree

In 2004, Venkat Venkateswaran presented the problem consisting in finding an orientation of a graph such that the maximum outdegree is minimized [95]:

$$(\textsc{MinMaxOut}) \quad \min_{\Lambda \in \vec{O}(G)} \max_{v \in V} d^+_\Lambda(v).$$

A problem of this type arises in the design of restorable telecommunication networks. He gives a polynomial time algorithm with complexity $O(|E|^2)$. A few years later, Yuichi Asahiro et al. present their work on the MinMaxOut [4, 5, 6] which they motivate as a discretization of the art gallery problem and give an algorithm with better time complexity than Venkateswaran's. They also study the weighted version of the problem where the edges of the input graph are associated with a weight (i.e. a positive real number) and the problem then consists in finding an orientation minimizing the maximum weighted outdegree of a vertex, i.e. the sum of the weights of its outgoing edges. They give a polynomial time algorithm for trees and show that the MinMaxOut is generally NP-complete by a reduction from the partition problem (the problem of partitioning a set of numbers into two equal-sum subsets). They give several approximation algorithms for special cases of weight sets and show that the general problem is inapproximable within less than 3/2 unless P = NP by a reduction from 3-SAT (a NP-complete problem detailed in Chapter 2, Section 2.1). Various newer results about MinMaxOut and some variations can be found in [7].

1.4.3 Balanced vertex ordering

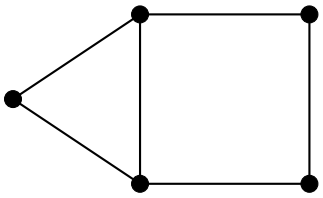
Therese Biedl et al. studied the problem of finding an acyclic orientation of unweighted graphs minimizing the sum of the imbalances of all the vertices:

$$(\textsc{MinIm}) \quad \min_{\substack{\Lambda \in \vec{O}(G) \\ \Lambda \text{ acyclic}}} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)|.$$

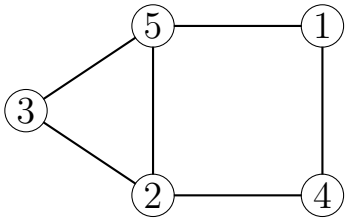
The problem is trivial without the acyclicity constraint on the orientation, the optimal objective value then is the number of odd-degree vertices in the input graph. An acyclic orientation can be simply built by giving an order to the vertices and then orient all the edges of the graph from "left to right", that is to say from its lowest endpoint w.r.t. said order to its greatest. Biedl et al. accordingly called the problem "Balanced vertex ordering" (cf Figure 1.6. They give polynomial time algorithms to solve MinIm for trees and graphs with maximum degree at most three and they show that it is NP-complete generally by a reduction from not-all-equal 3-SAT (a NP-complete problem detailed in Chapter 2, Section 2.1) and remains NP-complete

for bipartite graphs with maximum degree six. They also give a $\frac{13}{8}$-approximation algorithm for the general problem and show that the weighted version of the problem is NP-complete even if the input graph is a tree. Two years later, Jan Kára et al. closed the gap proving the NP-completeness of MinIm for graphs with maximum degree four. Furthermore, they show that the problem remains NP-complete for planar graphs with maximum degree four and for 5-regular graphs [65].

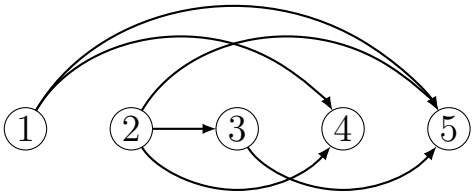
Figure 1.6: Example of a vertex ordering, i.e. acyclic orientation of the edges



(a) The input graph



(b) Assigning an order on the vertices



(c) Orientation of the edges derived from the order on the vertices

1.4.4 Graph realizing sequences of integers

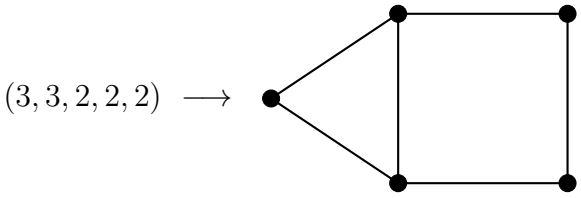
Graph orientation problems also appear in variations of the graph realization problem which is a decision problem consisting in deciding for a finite sequence $(d_1, \cdots, d_n) \in \mathbb{N}^n$ if there exists a graph such that $(d_1, \cdots, d_n)$ is the degree sequence of this graph. If this is the case, then the sequence is said to be *graphic*. The problem can be solved in polynomial time. This follows from Erdős-Gallai Theorem [37] stating a simple characterization of the graphic sequences.

Theorem 10 (Erdős-Gallai Theorem, 1960)**.** *A sequence* $(d_1, \cdots, d_n) \in \mathbb{N}^n$ *such that* $d_1 \geq \cdots \geq d_n$ *is a graphic sequence if and only if* $\sum_{i=1}^{n} d_i$ *is even and*

$$\sum_{i=1}^{k} d_i \leq k(k-1) + \sum_{i=k+1}^{n} \min(d_i, k), \ \forall k \in [\![1, n]\!].$$

An example of a graphic sequence realized by a graph can be found in Figure 1.7.

Figure 1.7: Example of a graphic sequence



Additionally, a solution can be quickly computed using the Havel-Hakimi algorithm[60, 54] which constructs a special solution with the use of a recursive algorithm. In parallel, Hyman Landau worked on a similar problem concerning *tournaments*, i.e. oriented complete graphs. The problem consists in deciding for a finite sequence $(d_1, \cdots, d_n) \in \mathbb{N}^n$ if there exists an orientation of the complete graph K_n such that $(d_1, \cdots, d_n)$ is the outdegree sequence (or *score sequence*) of this orientation. He gives the following simple characterization for score sequences.

Theorem 11 (Landau's Theorem, 1953)**.** *A sequence* $d_1, \cdots, d_n) \in \mathbb{N}^n$ *such that* $(d_1 \leq \cdots \leq d_n)$ *is a score sequence if and only if* $\sum_{i=1}^{n} d_i = \binom{n}{2}$ *and*

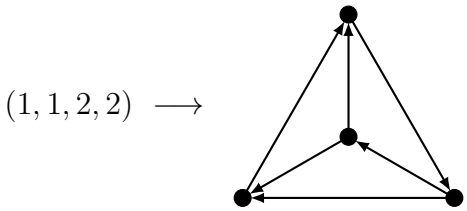
$$\sum_{i=1}^{k} d_i \geq \binom{k}{2}, \ \forall k \in [\![1, n-1]\!].$$

An example of a score sequence realized by a tournament can be found in Figure 1.8, notice that the orientation of its edges is strong. Later, Frank Harary & Leo Moser characterized score sequences of strongly connected tournaments [56].

Theorem 12 (Harary & Moser, 1966)**.** *A sequence* $(d_1, \cdots, d_n) \in \mathbb{N}^n$ *such that* $d_1 \leq \cdots \leq d_n$ *and* ≥ 3 *is the score sequence of a strongly connected tournament if and only if* $\sum_{i=1}^{n} d_i = \binom{n}{2}$ *and*

$$\sum_{i=1}^{k} d_i > \binom{k}{2}, \ \forall k \in [\![1, n-1]\!].$$

Figure 1.8: Example of a score sequence



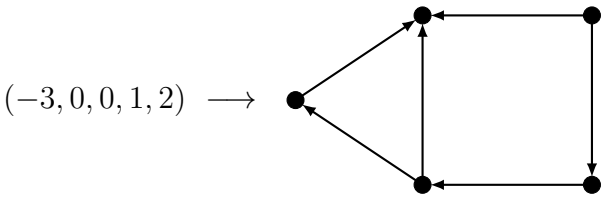
Analogous realizability results for the "signed imbalance sequences" of directed graphs are given by Dhruv Mubayi et al. [82]. They give a simple characterization for a sequence $(d_1, \cdots, d_n) \in \mathbb{Z}^n$ to be the signed imbalance sequence of an oriented graph.

Theorem 13 (Mubayi et al., 1998)**.** *A sequence $(d_1, \cdots, d_n) \in \mathbb{N}^n$ such that $d_1 \geq \cdots \geq d_n$ is the signed imbalance sequence of an oriented graph if and only if $\sum_{i=1}^{n} d_i = 0$ and*

$$\sum_{i=1}^{k} d_i \leq k(n-k), \ \forall k \in [\![1, n-1]\!].$$

An example of a signed imbalance sequence realized by an oriented graph can be found in Figure 1.9.

Figure 1.9: Example of a signed imbalance sequence



1.4.5 Minimizing the diameter and radius of a strong orientation

Another example of graph orientation problem is the search of strong orientations with minimum diameter or radius in a graph introduced in 1978 by Václav Chvátal & Carsten Thomassen [28]. They showed that it is NP-hard to decide whether a graph admits an orientation of diameter 2 and that it is NP-hard to decide whether a graph admits an orientation of radius 2. It was then proven to be NP-hard even if the graph is restricted to a subset of chordal graphs by Fedor Fomin et al. (2004) who gave also approximability and inapproximability results [41].

1.5 A generic problem

We have just given here a sample of the various studies on graph orientation problems, concentrating our concern on degree optimization and imbalance. Our work is focused on problems which consist in maximizing the image of the n-tuple $(d^+_\Lambda(v_1) - d^-_\Lambda(v_1), \cdots, d^+_\Lambda(v_n) - d^-_\Lambda(v_n))$ under a function $f : \mathbb{R}^n \to \mathbb{R}$ over all the orientations Λ of the graph G. In other words, the problems consist in finding an orientation that optimizes the imbalance of the vertices w.r.t. the objective function f:

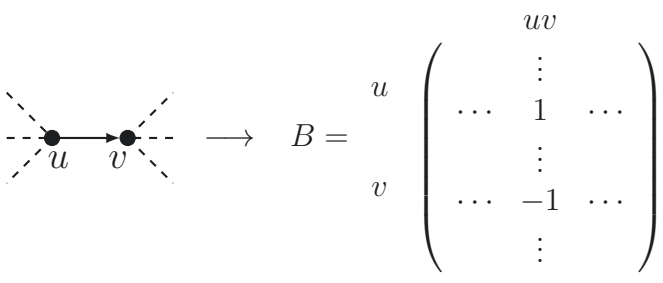
Generic Problem.

$$\max_{\Lambda \in \vec{O}(G)} f(d^+_\Lambda(v_1) - d^-_\Lambda(v_1), \cdots, d^+_\Lambda(v_n) - d^-_\Lambda(v_n)).$$

Now consider an arbitrary orientation of G and let $B \in M_{|V|\times|E|}(\{-1, 0, 1\})$ stand for the incidence matrix of G as a directed graph. i.e., the column corresponding to the arc $\overrightarrow{uv}$, has only nonzero entries in the rows corresponding to the nodes u and v: $B_{u,uv} = 1$ and $B_{v,uv} = -1$, respectively (cf Figure 1.10). In order to describe an orientation of the graph G, we take an orientation variable $x \in \{-1, 1\}^{|E|}$ interpreted as follows. For each edge $uv \in E$ which is originally directed from node u to node v: if $x_{uv} = 1$ then uv is directed from u to v (i.e., the orientation is the same as the original one) and is directed from v to u otherwise (i.e., the orientation of the edge is "reversed" with respect to the original orientation). Then if we look at the product of B with an orientation vector $x \in \{-1, 1\}^{|E|}$ we obtain $B_v x = d^+_x(v) - d^-_x(v),\ \forall v \in V$ where $d^+_x(v)$ (resp. $d^-_x(v)$) is the outdegree (resp. indegree) of $v \in V$ in G w.r.t. the orientation described by x and B_v denotes the row of the matrix B which corresponds to node v. Consequently, a problem of the type described in the previous paragraph with objective function $f : \mathbb{R} \to \mathbb{R}^n$ can be expressed as follows:

$$\max_{x \in \{-1,1\}^{|E|}} f(Bx).$$

Figure 1.10: Construction of the incidence matrix from an oriented graph



This family of problems is related to various graph optimization topics and suggests new ones. For example, taking $f = ||\cdot||$ (i.e. the Euclidean norm) leads to an original upper bound on the isoperimetric number of G that we consider just after.

1.6 The isoperimetric number

1.6.1 Definition, interpretation and complexity

Definition 14. *The* isoperimetric number *or* Cheeger constant *[27] of G is defined by*

$$h_G = \inf_{\emptyset \neq S \subset V} \frac{|\delta(S)|}{\min(|S|, |V \setminus S|)}$$

The Cheeger constant may be interpreted as a numerical measure of the "well connectedness" of a graph on the whole. A low value indicates the presence of a bottleneck, i.e. the presence of two large sets of vertices with few edges between them. On the contrary, a high value of the Cheeger constant means that any subset of vertices is connected to the rest of the graph with many edges. If the graph represents a communications network it may be considered as a measure of "graph vulnerability" [72]. It finds many applications in mathematics and computer science (e.g., image analysis [78]).

Computing the isoperimetric number of graphs with multiple edges is NP-hard in general [81] and Petr Golovach [49] showed the following problem is NP-complete: "Given a graph G with $\Delta_G \leq 3$ and two integers p and q, decide if $h'_G \leq \frac{p}{q}$". Bojan Mohar presents a linear-time algorithm for the particular case of trees [81] and James Park & Cynthia Phillips a pseudo-polynomial time algorithm for planar graphs [85].

1.6.2 Alternative formulations and existing bounds

We consider that $V = [\![1, n]\!]$ and build the diagonal matrix $D \in M_{n\times n}(\mathbb{N})$ as follows: $D_{i,i} = d(i)$ and $D_{i,j} = 0,\ \forall (i,j) \in [\![1,n]\!]^2$ such that $i \neq j$. Take $A \in \{0,1\}^{n\times n}$ such that $A_{i,j} = 1$ if $ij \in E$ the *adjacency matrix* of G, then the *Laplacian* of G is the matrix $L = D - A$. If we look at the product $z^\intercal Lz$ where $z \in \{0,1\}^n$ is the incidence vector of some subset of vertices $S \subseteq V$, we have

$$z^\intercal Lz = z^\intercal Dz - z^\intercal Az = \sum_{v\in V} z_v^2 d_v - \sum v \in V \left(\sum_{\substack{u\in V \\ uv\in E}} -x_u \right) = \sum_{v\in S} d_v - 2|E(S)| = |\delta(S)|.$$

Then, since $z^\intercal z = |S|$, the isoperimetric number may then be expressed as follows:

$$h_G = \min_{\substack{z\in\{0,1\}^n \\ 0\neq z^\intercal z\leq \lfloor \frac{n}{2}\rfloor}} \frac{z^\intercal L z}{z^\intercal z}.$$

Using this formulation and a spectral study of the Laplacian of the graph, Jeff Cheeger gave the following bounds on the isoperimetric number [25]:

Proposition 15 (Cheeger, 1970)**.**

$$\frac{\lambda_1}{2} \leq h_G \leq \sqrt{2\Delta_G \lambda_1}$$

with $\Delta_G = \max_{i\in V} d_i$*,*

where the eigenvalues of L are denoted by $0 = \lambda_0 \leq \lambda_1 \leq \ldots \leq \lambda_{n-1}$. Excluding some particular cases, Mohar [81] showed the following better upper bound, improving on the one of Proposition 15.

Proposition 16 (Mohar, 1989)**.** *If G is not equal to any of K_1, K_2, or K_3 then*

$$h_G \leq \sqrt{\lambda_1(2\Delta_G - \lambda_1)}$$

.

Another general bound given by Mohar [81] is as follows.

Theorem 17 (Mohar, 1989)**.**

$$h_G \leq \frac{2|E|\,\lceil |V|/2\rceil}{|V|(|V|-1)}.$$

1.6.3 New upper bounds

Now if we consider the incidence matrix B of G as defined in the previous section (i.e. w.r.t. a chosen arbitrary orientation), the isoperimetric number may be expressed as follows:

$$\begin{aligned} h_G &= \min_{S\subset V:\ |S|\leq \frac{n}{2}} \frac{1}{|S|} \max_{\Lambda\in\vec{O}(G)} \sum_{v\in S} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \\ &= \min_{S\subset V:\ |S|\leq \frac{n}{2}} \frac{1}{|S|} \max_{x\in\{-1,1\}^{|E|}} \sum_{v\in S} |B_v x|, \end{aligned}$$

It is therefore the optimal objective value of the following mixed integer programming formulation with nonlinear constraints.

$$\begin{cases} h_G = \min h \\ \text{s.t.} \\ 1 \leq \sum_{v \in V} z_v \leq \frac{n}{2}, \\ h \geq \frac{\sum_{v \in V} z_v |B_v x|}{\sum_{v \in V} z_v}, \ \forall x \in \{-1,1\}^{|E|}, \\ z \in \{0,1\}^n, h \in \mathbb{R}. \end{cases}$$

Using Cauchy's inequality, we have

$$\sum_{v \in V} z_v |B_v x| \leq \sqrt{\sum_{v \in V} z_v^2} \sqrt{\sum_{v \in V} (B_v x)^2} = \sqrt{\sum_{v \in V} z_v} ||Bx||.$$

Thus, looking at the nonlinear constraint of the previous formulation, it yields

$$h \geq \frac{\sum_{v \in V} z_v |B_v x|}{\sum_{v \in V} z_v}, \ \forall x \in \{-1,1\}^{|E|} \ \Leftarrow \ h^2 \sum_{v \in V} z_v \geq ||Bx||^2, \ \forall x \in \{-1,1\}^{|E|}.$$

We may then consider the following restriction of the previous formulation, thus leading to an upper bound on h_G:

$$\begin{cases} \min h \\ \text{s.t.} \\ 1 \leq \sum_{v \in V} z_v \leq \frac{n}{2}, \\ h^2 \sum_{v \in V} z_v \geq ||Bx||^2, \forall x \in \{-1,1\}^{|E|}, \\ z \in \{0,1\}^n, z \in \mathbb{R}. \end{cases}$$

This leads to the following new upper bound:

Proposition 18.

$$h_G \leq \frac{1}{\sqrt{\lfloor \frac{n}{2} \rfloor}} \max_{x \in [-1,1]^{|E|}} ||Bx|| \leq \sqrt{\frac{\lambda_{n-1}|E|}{\lfloor \frac{n}{2} \rfloor}},$$

Proof. The lowest upper bound $h_G \leq \frac{1}{\sqrt{\lfloor \frac{n}{2} \rfloor}} \max_{x \in [-1,1]^{|E|}} ||Bx||$ is a direct consequence of the restriction it follows. Now concerning the greatest upper bound, we have

$$||Bx|| = \sqrt{(Bx)^\intercal Bx} = \sqrt{x^\intercal B^\intercal Bx} \leq \sqrt{\lambda^{max}(B^\intercal B)||x||^2} \leq \sqrt{\lambda^{max}(B^\intercal B)|E|},$$

where given a matrix $M \in \mathbb{R}^{n \times n}$, $\lambda^{max}(M)$ stands for its maximum eigenvalue. Since a matrix and its transpose have the same spectra and we know that $BB^\intercal = L$, then

$\lambda^{max}(B^\intercal B) = \lambda_{n-1}$ and we obtain the aimed upper bound. □

We find here the expression of the general problem we defined in Chapter 1 with $f = \frac{1}{\sqrt{\lfloor \frac{n}{2} \rfloor}}||\cdot||$. The greatest upper bound of Proposition 18 is loosely obtained from the part preceding it through the maximum eigenvalue of the Laplacian. Study on the generic problem presented in the previous section may lead to a tighter upper bound for the Cheeger constant. Another example of objective function f leading to a known optimization problem is the case $f = \frac{1}{2}||\cdot||_1$ which will give an original approach of the famous maximum cardinality cut problem.

1.7 The maximum cut

Let $(w_e)_{e\in E}$ denote nonnegative weights on the edges of G. The *weight* of the cut defined by S, denoted by $w(\delta(S))$ is the sum of the weights of the edges belonging to the cut, i.e., $w(\delta(S)) = \sum_{e\in\delta(S)} w_e$. The maximum cut problem consists in finding a cut of maximum weight, denoted by $w^\star$, in the graph G:

$$\max_{S\subset V} w(\delta(S)).$$

It is one of Richard Karp's 21 NP-complete problems [67], he showed its NP-completeness by a reduction from the partition problem.

1.7.1 Spin glasses

The maximum cut problem is a fundamental combinatorial optimization problem that emerges in several scientific disciplines. One example stems from Physics and more precisely, the study of spin glasses. A spin glass is an alloy of non-magnetic metal containing local magnetic impurities. As opposed to a ferromagnetic solid (e.g. a standard magnet that can be found on many refrigerators) where the magnetic spins (the orientation of the north and south magnetic poles in three-dimensional space) are all aligned in the same direction, a spin glass is a disordered magnet where the magnetic spin of the component atoms are not aligned in a regular pattern. The magnetic disorder in a spin glass is analog to the positional disorder of a conventional, chemical glass, e.g., a window glass, hence the term "glass". The atomic bonds structure in window glass (or any amorphous solid) is highly irregular whereas, in contrast, it follows a uniform pattern in the case of a crystal.

An accurate simplified model of the orientation of the spin of an impurity consists in assigning a value 1 or -1 to the impurity, meaning magnetic north pole "up" or "down". Such a representation is called the Ising model and the 1/-1 value is called

the Ising spin of the impurity. Between each pair of magnetic atoms there is an interaction that depends on the nonmagnetic material and above all on the distance between the atoms. The most realistic model consider only interaction between "close" impurities and consider null interactions between distant impurities. Then we can consider the *interactions graph* G with the impurities of the graph as its node set where two interacting impurities will be connected by an edge with a weight equal to this interaction. A spin configuration of the glass then consists in an assignment of an Ising spin to each vertex of G. For a spin configuration, if we call S the set of impurities which Ising spin is equal to 1, then the energy of the whole system is inversely proportional to the weight of the cut defined by S. Consequently, Finding the lowest-energy spin configuration of the glass reached at very low temperatures is equivalent to finding a maximum cut in what is called the "interactions graph". The study of spin glasses has many applications from optimization in Economy to neuron modeling and learning in Biology.

1.7.2 VLSI design

Another application of the maximum cut problem comes up in Very-Large-Scale Integration (VLSI) design and printed circuit boars design. The design of a chip consists in several steps, one of which will catch our attention: layer assignment. After placing all the cells and nets, they are routed together with wires that can be horizontal or vertical. Each segment of wire must be assigned a layer in such a way that crossing wires are assigned different layers. Physically, a change of layers is achieved by placing a *via*: a drilled hole in a printed circuit board causing additional costly work and contributing to failure of the board due to cracking. In the context of VLSI design, a via is a special contact treated in the production process that needs additional space which is an obstacle in compaction and decreases the yield in the fabrication process. The number of vias in the designing of the chip must therefore be preferably minimized. It is usually impossible to place all the components on a single layer without crossing wire segments. If we consider the case where only two layers are available (corresponding to several real applications such as chip cards), then we can build a *conflict (weighted) graph* G such that finding an assignment of layers minimizing the number of vias is equivalent to solving the maximum cut problem in G. This equivalence also allows to take into account several practical constraints on the original problem such as the length of wire segments.

Details about the the two applications of the maximum cut problem described above can be found in [11].

1.7.3 Frequency assignment

A less known application of the maximum cut problem concerns a special case of the frequency assignment problem which arises in many telecommunication contexts such as wireless networks or cellular phone communication systems. The goal is to assign frequencies to a set of transmitters subject to possible interferences. The constraints as well as the objective may vary along with the context, the most common objective being the minimization of the number of assigned frequencies with the constraint that no two interfering transmitters are assigned the same frequency. Such a plain case is equivalent to the graph colouring problem (consisting in assigning a colour to each vertex of the graph such the endpoints of each edge have different colours and using as few colours as possible) on the interference graph (i.e. the graph where the transmitters are the vertices and two transmitters are connected if and only if their signal may interfere with one another).

We consider the frequency assignment problem where the weight of each edge represents the interference level between two nodes and only two frequencies are available. The whole interference of an assignment is the sum of the weights of the edges which endpoints use the same assigned frequency. It is equal to the the sum of the weights of all the edges minus the sum of the weights of the edges which endpoints have different frequencies, i.e. the weight of the cut defined by the set of vertices using one specific frequency (it can be either of the two). Then properly assigning a frequency to each node whilst minimizing the whole interference is equivalent to finding a maximum cut problem in this weighted interference graph.

The maximum cut problem has many other famous applications such as sparse matrix computation [8], parallel programming [24], quadratic programming [55], etc.

1.7.4 Special cases

On the other hand, the maximum cut problem becomes simple for many special cases. Orlova & Dorfman [84] and Frank Hadlock [53] independently showed that the problem can be solved in polynomial time if the graph. They based their approach on the duality of planar graphs and gave an equivalence between the maximum cut problem and the maximum weighted matching problem for which there exists a polynomial bounded algorithm. Francisco Barahona extended this result showing the polynomial time complexity of the problem for the graphs not contractible to K_5 [12]. Martin Grötschel & William Pulleyblanck took this result further by defining a family of graphs called *weakly bipartite*, containing the planar graphs, the bipartite graphs and the graphs not contractible to K_5, and showing that the maximum cut problem remains easy for this larger class of graphs [50] which was then characterized by Bertrand Guenin [52]. Other polynomial cases are reviewed, e.g., in [18].

1.7.5 (In)approximibilty and the cut polytope

Concerning the approximability of the maximum cut problem, Johan Håstad showed it is not approximable within a ratio $\frac{16}{17} + \epsilon$ for any $\epsilon > 0$ unless $\mathsf{P} = \mathsf{NP}$ [59]. One line of research to approach the optimal objective value of this problem has consisted in the development of (meta)heuristics. In 1976, Sartaj Sahni & Teofilo Gonzalez proposed a (greedy) $\frac{1}{2}$-approximation algorithm [90]. Several randomized heuristics derived from a greedy randomized adaptive search procedure are proposed, implemented and tested in [39] and a randomized heuristic consisting in finding an approximate solution of a formulation of the maximum cut problem as an unconstrained non-convex optimization problem is given in [21]. Another important line of research relies on linear programming formulations of the problem. This has namely led to deep investigations on the polyhedral structure of the *cut polytope*, namely the convex hull of the incidence vectors of all the cuts of the graph:

$$P_c(G) = Conv\{x^{\delta(S)} : S \subseteq V\} \subset \{0,1\}^{|E|}.$$

The cut polytope has been extensively used in the context of Branch and Cut algorithms with cutting-plane methods. Francisco Barahona & Ali Ridha Mahjoub [13] introduced the following valid family of inequalities for the cut polytope.

$$\sum_{e \in F} x_e - \sum_{e \in E(C) \setminus F} x_e \leq |F| - 1, \ \forall C \text{ cycle of } G, \ \forall F \subseteq C \text{ s.t. } |F| \equiv 1 \pmod 2, \quad (1.5)$$

$$0 \leq x_e \leq 1, \ \forall e \in E. \quad (1.6)$$

It is easy to see that any integer-valued vector satisfying the above constraints describes a cut of G. Consequently, they induce an integer programming formulation for the maximum cut problem. If the input graph is complete, then the following constraints induce an integer programming formulation for the maximum cut problem as well.

$$\begin{cases} x_{uv} + x_{vw} + x_{uw} \leq 2, \\ -x_{uv} + x_{vw} + x_{uw} \leq 0, \\ x_{uv} - x_{vw} + x_{uw} \leq 0, \\ x_{uv} + x_{vw} - x_{uw} \leq 0, \end{cases} \forall \{u, v, w\} \subseteq V, \quad (1.7)$$

The constraints (1.5) are called the *cycle inequalities*, the constraints (1.6) are called the *trivial inequalities* and the constraints (1.7) are called the triangle inequalities. Barahona et al. give characterizations of the constraints (1.5) and (1.6) that define a facet of $P_c(G)$ [13].

Theorem 19 (Barahona et al., 1986)**.**

- *An inequality (1.5) defines a facet of $P_c(G)$ if and only if the cycle C corresponding to the inequality is chordless.*
- *An inequality (1.6) defines a facet of $P_c(G)$ if and only if the edge e corresponding to the inequality does not belong to a triangle.*

It is easy to see that the separation problem inherent to (1.6) has polynomial complexity, Barahona et al. showed that it is true for (1.5) too [13]

Theorem 20 (Barahona et al., 1986)**.** *The separation problem inherent to (1.5) can be solved in polynomial time.*

Other valid and separable families of inequalities and results on the cut polytope can be found in [13, 18, 33].

1.7.6 Goemans & Williamson's semidefinite breakthrough

More recently, essentially since the mid-1990's and the breakthrough paper by Goemans & Williamson [47], there has been a growing interest in semidefinite optimization based algorithms. The following formulations concern complete graphs without loss of generality for one can set $w_{uv} = 0$ for $(u, v) \in V^2$ such that $uv \notin E$. Moreover, we consider that $V = [\![1, n]\!]$. The maximum cut problem can be formulated as the following quadratic program:

$$\begin{cases} \max \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} x_i (1 - x_j) \\ \text{s.t.} \\ x \in \{0, 1\}^n \end{cases}$$

Changing variables with $z = 1 - 2x$ leads to the following -1/1 formulation:

$$\begin{cases} \max \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (1 - z_i z_j) \\ \text{s.t.} \\ z \in \{-1, 1\}^n \end{cases} \tag{1.8}$$

Introducing a variable Y_{ij} representing the product of variables $z_i z_j$ for all $(i, j) \in V^2$, we obtain the following:

$$\begin{cases} \max \frac{1}{4} \sum_{i=1}^{n} \sum_{j=1}^{n} w_{ij} (1 - Y_{ij}) \\ \text{s.t.} \\ Y_{ii} = 1, \ \forall i \in V, \\ Y = z z^{\intercal}, \\ Y \in M_{n \times n}(\mathbb{R}), \ z \in \mathbb{R}^n \end{cases}$$

In this formulation, the matrix is necessarily positive semidefinite, we can then obtain from the exact formulation for the maximum cut problem above a relaxed SDP formulation which is the Goemans & Williamson based their algorithm on:

$$(\mathsf{SDP}_{GW}) \begin{cases} \max \frac{1}{2} \sum_{i=1}^{n} \sum_{j=i+1}^{n} w_{ij}(1 - Y_{ij}) \\ \text{s.t.} \\ Y_{ii} = 1, \ \forall i \in V, \\ Y \succeq 0, \\ Y \in M_{n \times n}(\mathbb{R}), \end{cases}$$

where $\mathbb{S}^n$ denotes the set of symmetric matrices with order n. The interest in such a relaxed SDP formulation resides in the quality of the approximate solution obtained. Related to the maximum cut problem, Goemans & Williamson's work presents a 0.87856-approximation algorithm [47]. It consists in first solving the formulation (SDP_{GW}) which can be done in polynomial time with a set precision $\varepsilon > 0$, we obtain as solution the positive semidefinite matrix Y. let $H \in \mathbb{R}^{m \times n}$ (for some $m \leq n$) denote a matrix such that $Y = H^\intercal H$ (H can be obtained in $O(n^3)$ time using an incomplete Cholesky decomposition) and let r denote a vector which is randomly generated according to a uniform distribution on the unit sphere in $\mathbb{R}^m$. The cut returned by the algorithm is then $\delta(S)$ with $S := \{v \in V : r^\intercal h_v \geq 0\}$, where h_v stands for the column of H corresponding to node $v \in V$. For a uniformly generated vector r from the unit sphere, Goemans and Williamson have shown that the expectation of the cardinality of the cut $E(|\delta(S)|)$ can be bounded as follows.

$$\alpha w^\star \leq E(|\delta(S)|) \leq w^\star,$$

Where $\alpha = \min_{\theta \in [0,\pi]} \frac{2\theta}{\pi(1-\cos\theta)} \simeq 0.87856$.

Several variations of the algorithm have been proposed in order to improve its quality, one of which consists in directly improving the quality of the bound $Z^\star_{\mathsf{SDP}_{GW}}$ given by the formulation (SDP_{GW}). To do so, different approaches have been proposed in the literature: namely by making use of polyhedral knowledge on the cut polytope and adding linear inequalities [40, 61], or by means of lift-and-project techniques [3, 74]. Another way to improve the upper bound given by the semidefinite relaxation is described in [19, 20] where some spectral techniques are used leading to polynomial-time algorithms for some low rank weight matrices. This semidefinite approach of the problem also led to efficient solvers such as BiqMac [87] and BiqCrunch [70].

The reader can find in [18, 33] and the references therein further results about the maximum cut problem including applications, polynomial cases, approximation algorithms, relationships with other combinatorial problems, polyhedral studies etc.

1.8 Outline

In Chapter 2, we study the problem of finding a graph orientation that maximizes the minimum imbalance over all the vertices. We examine the complexity and some properties of special graph instances and give several mixed integer programming formulations for this problem. In Chapter 3, we give a new approach of the maximum cardinality cut problem through the study of our problem with $f = \frac{1}{2}||\cdot||_1$. We introduce new mixed integer programming and semidefinite programming formulations along with their analysis of complexity and performance. In Chapter 4, we give the full description of polytopes related to the index of the lowest nonzero row of an assignment matrix, polytopes that derive from a formulation of the problem studied in Chapter 2. And we conclude in Chapter 5.

Chapter 2

Maximizing the imbalance of an orientation

We consider the problem of finding an orientation with maximized imbalance:

$$(\textsc{MaxIm}) \quad \textsc{MaxIm}(G) = \max_{\Lambda \in \vec{O}(G)} \min_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)|$$

and we call $\textsc{MaxIm}(G)$ the value of MaxIm for G. We find here our generic problem introduced in Section 1.5 for the function

$$f : \begin{array}{l} \mathbb{R}^n \to \mathbb{R} \\ \quad x \mapsto \min_{i \in [\![1,n]\!]} |x_i| \end{array}.$$

The minimum degree δ_G of a graph G is a trivial upper bound for $\textsc{MaxIm}(G)$. Since the value of MaxIm for a graph is the minimum of the values of MaxIm on its connected components, all the graphs we consider in this chapter are assumed to be connected.

The rest of the chapter is organized as follows. In Section 2.2, we give several characterizations of the the graphs verifying $\textsc{MaxIm}(G) = 0$. In Section 2.1, we show that MaxIm is generally NP-complete even for graphs with minimum degree 2 and inapproximable within a ratio $\frac{1}{2} + \varepsilon$ for any constant $\varepsilon > 0$ and then give an approximation algorithm whose ratio is almost equal to $\frac{1}{2}$. In Section 2.3, we present a polynomial-time exact algorithm for cactus. Section 2.4 is devoted to mixed integer linear programming formulations of MaxIm where some families of valid inequalities are presented. These formulations have been implemented and the computational results are reported in Section 2.6.

2.1 Complexity and (in)approximability

The MaxIm problem being new, one of the first questions to be raised is what kind of difficulty are we dealing with when trying to maximize the imbalance of an orientation. And more precisely, can we highlight a relationship between MaxIm and classic combinatorial problems in order to excerpt some of the knowledge we have about them?

2.1.1 NP-completeness

It is clear that the decision problem inherent to MaxIm is in NP: given a solution for an instance of MaxIm, i.e. an orientation, it is easy (namely linear) to check whether its imbalance is larger than a given integer. Hence the problem is in NP. Now the best way to prove that a problem in NP is NP-complete is to reduce a known NP-complete problem to it. In order to show the NP-completeness of MinMaxOut, Asahiro et al. [4] take an instance of the partition problem: S a finite set of integers such that $\sum_{s\in S} s$ is even and build a graph G_S such that the following assertions are equivalent.

- There exists a subset $S' \subset S$ such that $\sum_{s\in S'} s = k$.
- There exists an orientation $\Lambda \in \vec{O}(G)$ such that $\max_{v\in V(G_S)} d^+_\Lambda(v) \leq k$.

Solving the latter is therefore equivalent to solving the former for $k = \frac{1}{2}\sum_{s\in S} s$ which is an instance of a $\mathsf{NP} - \textit{complete}$ problem. Concerning our problem, will show that MaxIm is NP-complete by highlighting that answering if $\textsc{MaxIm}(G)$ equals 2 for a graph G such that $\delta_G = 2$ is equivalent to solving an instance of a particular NP-complete problem. For that purpose, we introduce a variant of the satisfiability problem: the not-all-equal at most 3-SAT(3V).

Not-all-equal at most 3-SAT(3V) is a restriction of not-all-equal at most 3-SAT which is itself a restriction of 3-SAT known to be NP-complete [91]. 3-SAT is a satisfiability problem considering only formulas respecting the following constraint.

▷ Each clause contains at most three literals.

The added restriction of not-all-equal at most 3-SAT is:

▷ In each clause, not all the literals can be true. In other words, for the formula to be satisfiable, there must exist an assignment of the variables such that in each clause, at least one of the literals is true and at least one of the literal is false.

And the added restrictions of not-all-equal at most 3-SAT(3V) are:

▷ Each variable (not literal) appears at most three times in a formula.

▷ Each variable occurs in both its possible versions: as a positive literal and as a negative literal.

The resulting problem is still NP-complete.

Lemma 21. *The not-all-equal at most 3-SAT(3V) problem is NP-complete.*

Proof. See A. □

Now we associate to a not-all-equal at most 3-SAT(3V) instance φ with n variables $\{x_1, \cdots, x_n\}$ and m clauses $\{c_1, \cdots, c_m\}$ a graph G_φ for which the value w.r.t. MaxIm will give the answer to whether φ is satisfiable or not. The graph G_φ consists of gadgets that mimic the variables and the clauses of φ and additional edges that connect them together:

- the gadget corresponding to a variable x_i consists of two vertices labeled x_i and $\neg x_i$ and one edge connecting them (see 2.1(a));

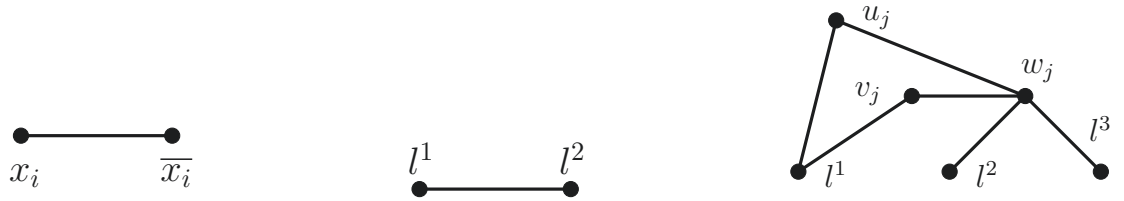
- the gadget corresponding to a two-literals clause $c_j = (l^1 \vee l^2)$, where l^1 and l^2 are its literals, consists in two vertices labeled $a^j_{l^1}$ and $b^j_{l^2}$ corresponding to l^1 and l^2 respectively (the index "l^k" of the vertices labels stands for the literal it represents, i.e. x_i if l^k is the variable x_i and $\neg x_i$ if l^k is the negation of the variable x_i) and one edge connecting them (see 2.1(b));

- the gadget corresponding to a three-literals clause gadget consists in six vertices and six edges. For a clause $c_j = (l^1 \vee l^2 \vee l^3)$, where l^1, l^2 and l^3 are its literals (the order is arbitrary), three vertices labeled $a^j_{l^1}$, $b^j_{l^2}$ and $b'^j_{l^3}$ correspond to l^1, l^2 and l^3 respectively. Three additional vertices are labeled u^j, v^j and w^j and the edges of the gadget are $a^j_{l^1}u_j$, $a^j_{l^1}v_j$, u_jw_j, v_jw_j, $w_jb^j_{l^2}$ and $w_jb'^j_{l^3}$ (see 2.1(c));

- $\forall i \in [\![1, n]\!]$, the vertex labeled x_i (resp. $\neg x_i$) is connected to all the vertices labeled $a^j_{x_i}$, $b^j_{x_i}$ or $b'^j_{x_i}$ (resp. $a^j_{\neg x_i}$, $b^j_{\neg x_i}$ or $b'^j_{\neg x_i}$), $\forall j \in [\![1, m]\!]$.

As an example, for a formula

Figure 2.1: Gadgets of G_φ



(a) Variable gadget

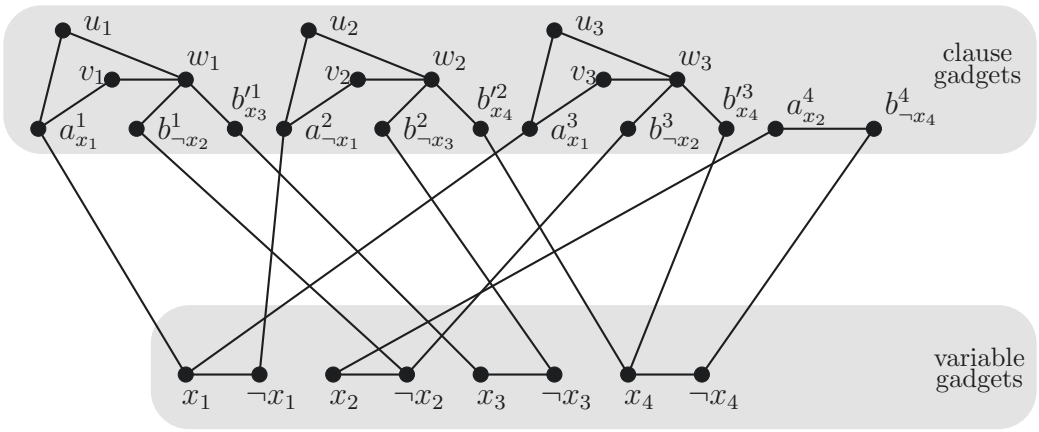
(b) Two-literals clause gadget

(c) Three-literals clause gadget

$$\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee x_4) \wedge (x_2 \vee \neg x_4), \qquad (2.1)$$

the corresponding graph G_φ is represented in Fig. 2.2.

Figure 2.2: G_φ for $\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee x_4) \wedge (x_2 \vee \neg x_4)$



Theorem 22. *A not-all-equal at most 3-SAT(3V) formula φ is satisfiable if and only if*
$\textsc{MaxIm}(G_\varphi) = 2$.

Proof. • $\Rightarrow$ Suppose φ is satisfiable and let $v : \{x_1, \cdots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ be a satisfying assignment of $x_1, \cdots, x_n$. We know that $\delta_{G_\varphi} = 2$ which yields $\textsc{MaxIm}(G_\varphi) \leq 2$. So let us build an orientation $\Lambda \in \vec{O}(G_\varphi)$ for which the imbalance is larger than or equal to 2. First, we assign an orientation to the

edges of the variable gadgets:

$$\Lambda(x_i \neg x_i) = \begin{cases} \overrightarrow{x_i \neg x_i} & \text{if } v(x_i) = \text{TRUE;} \\ \overrightarrow{\neg x_i x_i} & \text{otherwise.} \end{cases}$$

For example, for the formula

$$\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee x_4) \wedge (x_2 \vee \neg x_4)$$

satisfied by the assignment

$$v(x_1, x_2, x_3, x_4) = (\text{FALSE}, \text{TRUE}, \text{TRUE}, \text{TRUE}),$$

the edges of the variable gadgets of graph G_φ are oriented as in figure 2.3(a). Since each variable x_i occurs at least once as a positive literal and at least once as a negative literal, $2 \leq d_{G_\varphi}(x_i) \leq 3$ and $2 \leq d_{G_\varphi}(\neg x_i) \leq 3$, $\forall i \in [\![1, n]\!]$. Then to ensure our objective on the imbalance of Λ, the orientation of the edges connecting variable gadgets and clause gadgets must be such that $\forall i \in [\![1, n]\!]$, $|d^+_\Lambda(x_i) - d^-_\Lambda(x_i)| = d_{G_\varphi}(x_i)$ and $|d^+_\Lambda(\neg x_i) - d^-_\Lambda(\neg x_i)| = d_{G_\varphi}(\neg x_i)$. In other words, for $i \in [\![1, n]\!]$, if $v(x_i) = \text{TRUE}$ (resp. $v(x_i) = \text{FALSE}$), then the edges adjacent to the vertex x_i are oriented from x_i (resp. to x_i) and the edges adjacent to the vertex $\neg x_i$ are oriented to $\neg x_i$ (resp. from $\neg x_i$), e.g. Fig. 2.3(b).

So far, all the edges in the variable gadgets and the edges connecting the variable gadgets and the clause gadgets have been oriented and the vertices in the variables gadgets have imbalance larger than or equal to 2. In order to complete our orientation Λ we have to orient the edges in the clause gadgets.

Let $c_j = (l^1 \vee l^2)$ be a two-literals clause. Since v satisfies φ, we know that exactly one of the two literals is true w.r.t. v. Which, according to the way we oriented edges so far, means that exactly one of $a^j_{l^1}$ and $b^j_{l^2}$ has one incoming arc from a variable gadget and the other has one outgoing arc to a variable gadget. If $a^j_{l^1}$ is the one with the incoming arc from a variable gadget (meaning that $v(l^1) = \text{TRUE}$), then we assign $\Lambda(a^j_{l^1} b^j_{l^2}) = \overrightarrow{b^j_{l^2} a^j_{l^1}}$, otherwise the opposite. We obtain

$$|d^+_\Lambda(a^j_{l^1}) - d^-_\Lambda(a^j_{l^1})| = |d^+_\Lambda(b^j_{l^2}) - d^-_\Lambda(b^j_{l^2})| = 2.$$

Let $c_j = (l^1 \vee l^2 \vee l^3)$ (the order is identical to the one chosen to build the clause gadget, i.e. $d_{G_\varphi}(a^j_{l^1}) = 3$ and $d_{G_\varphi}(b^j_{l^2}) = d_{G_\varphi}(b'^j_{l^3}) = 2$) be at three-literals clause. If the edge connecting $a^j_{l^1}$ to a variable gadget is oriented to $a^j_{l^1}$ (meaning that $v(l^1) = \text{TRUE}$), then we assign $\Lambda(a^j_{l^1}u_j) = \overrightarrow{u_j a^j_{l^1}}$, $\Lambda(a^j_{l^1}v_j) = \overrightarrow{v_j a^j_{l^1}}$, $\Lambda(u_j w_j) = \overrightarrow{u_j w_j}$ and $\Lambda(v_j w_j) = \overrightarrow{v_j w_j}$. Since $v(l^1) = \text{TRUE}$, either both $v(l^2)$ and $v(l^3)$ are FALSE or exactly one of $v(l^2)$ and $v(l^3)$ is TRUE and one is FALSE. If both are FALSE then $b^j_{l^2}$ and $b'^j_{l^3}$ have an outgoing arc to a variable gadget. In that case, we orient $w_j b^j_{l^2}$ and $w_j b'^j_{l^3}$ to w_j and we obtain

- $|d^+_\Lambda(a^j_{l^1}) - d^-_\Lambda(a^j_{l^1})| = 3$,
- $|d^+_\Lambda(b^j_{l^2}) - d^-_\Lambda(b^j_{l^2})| = 2$;
- $|d^+_\Lambda(b'^j_{l^3}) - d^-_\Lambda(b'^j_{l^3})| = 2$;
- $|d^+_\Lambda(u_j) - d^-_\Lambda(u_j)| = 2$;
- $|d^+_\Lambda(v_j) - d^-_\Lambda(v_j)| = 2$;
- $|d^+_\Lambda(w_j) - d^-_\Lambda(w_j)| = 4$.

If exactly one of $v(l^2)$ and $v(l^3)$ is TRUE and one is FALSE, then exactly one of $b^j_{l^2}$ and $b'^j_{l^3}$ has an incoming arc from a variable gadget and the other an outgoing arc to a variable gadget. If $b^j_{l^2}$ is the one with the incoming arc from a variable gadget (meanings that $v(l^2) = \text{TRUE}$ and $v(l^3) = \text{FALSE}$), then we assign $\Lambda(w_j b^j_{l^2}) = \overrightarrow{w_j b^j_{l^2}}$ and $\Lambda(w_j b'^j_{l^3}) = \overrightarrow{b'^j_{l^3} w_j}$, otherwise the opposite. We obtain

- $|d^+_\Lambda(a^j_{l^1}) - d^-_\Lambda(a^j_{l^1})| = 3$;
- $|d^+_\Lambda(b^j_{l^2}) - d^-_\Lambda(b^j_{l^2})| = 2$;
- $|d^+_\Lambda(b'^j_{l^3}) - d^-_\Lambda(b'^j_{l^3})| = 2$;
- $|d^+_\Lambda(u_j) - d^-_\Lambda(u_j)| = 2$;
- $|d^+_\Lambda(v_j) - d^-_\Lambda(v_j)| = 2$;
- $|d^+_\Lambda(w_j) - d^-_\Lambda(w_j)| = 2$.

If, on the other hand, the edge connecting $a^j_{l^1}$ to a variable gadget is oriented from $a^j_{l^1}$ (meanings that $v(l^1) = \text{FALSE}$), then we assign $\Lambda(a^j_{l^1}u_j) = \overrightarrow{a^j_{l^1} u_j}$, $\Lambda(a^j_{l^1}v_j) = \overrightarrow{a^j_{l^1} v_j}$, $\Lambda(u_j w_j) = \overrightarrow{w_j u_j}$ and $\Lambda(v_j w_j) = \overrightarrow{w_j v_j}$. By symmetry, we conclude in the same way that

- $|d^+_\Lambda(a^j_{l^1}) - d^-_\Lambda(a^j_{l^1})| = 3$;
- $|d^+_\Lambda(b^j_{l^2}) - d^-_\Lambda(b^j_{l^2})| = 2$;
- $|d^+_\Lambda(b'^j_{l^3}) - d^-_\Lambda(b'^j_{l^3})| = 2$;
- $|d^+_\Lambda(u_j) - d^-_\Lambda(u_j)| = 2$;
- $|d^+_\Lambda(v_j) - d^-_\Lambda(v_j)| = 2$;
- $|d^+_\Lambda(w_j) - d^-_\Lambda(w_j)| = 2$.

Consequently, the imbalance of the resulting orientation Λ is larger than or equal to 2, e.g. Fig. 2.3(c).

- $\Leftarrow$ Now we assume that $\textsc{MaxIm}(G_\varphi) = 2$, let $\Lambda \in \vec{O}(G_\varphi)$ with optimal imbalance. Since all the vertices in the variable gadgets have degree at most 3, each vertex x_i (or $\neg x_i$) is necessarily adjacent to only incoming arcs or only outgoing arcs w.r.t. Λ. We will show that the assignment $v : \{x_1, \cdots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ of $x_1, \cdots, x_n$ defined by

$$v(x_i) = \begin{cases} \text{TRUE} & \text{if } d^+_\Lambda(x_i) > d^-_\Lambda(x_i); \\ \text{FALSE} & \text{otherwise;} \end{cases}$$

satisfies φ. Suppose φ does not satisfy a clause c_j, $j \in [\![1, m]\!]$. If c_j is a two-literals clause $(l^1 \vee l^2)$ then either $v(l^1) = v(l^2) = \text{TRUE}$ or $v(l^1) = v(l^2) = \text{FALSE}$, i.e. either both $a^j_{l^1}$ and $b^j_{l^2}$ have an incoming arc from a variable gadget or both have an outgoing arc to a variable gadget and in both cases, whichever is the orientation assigned to $a^j_{l^1}b^j_{l^2}$ by Λ, either $a^j_{l^1}$ or $b^j_{l^2}$ has a zero imbalance which contradicts our assumption. So c_j is a three-literals clause $(l^1 \vee l^2 \vee l^3)$ (the order is identical to the one chosen to build the clause gadget, i.e. $d_{G_\varphi}(a^j_{l^1}) = 3$ and $d_{G_\varphi}(b^j_{l^2}) = d_{G_\varphi}(b'^j_{l^3}) = 2$). Then either $v(l^1) = v(l^2) = v(l^3) = \text{TRUE}$ or $v(l^1) = v(l^2) = v(l^3) = \text{FALSE}$, i.e. either all $a^j_{l^1}$, $b^j_{l^2}$ and $b'^j_{l^3}$ have an incoming arc from a variable gadget or they all have an outgoing arc to a variable gadget. In the first case, it implies

- $\Lambda(a^j_{l^1}u_j) = \overrightarrow{u_j a^j_{l^1}}$,
- $\Lambda(a^j_{l^1}v_j) = \overrightarrow{v_j a^j_{l^1}}$,
- $\Lambda(u_j w_j) = \overrightarrow{u_j w_j}$,

- ▷ $\Lambda(v_j w_j) = \overrightarrow{v_j w_j}$,
- ▷ $\Lambda(w_j b^j_{l^2}) = \overrightarrow{w_j b^j_{l^2}}$,
- ▷ $\Lambda(w_j b'^j_{l^3}) = \overrightarrow{w_j b'^j_{l^3}}$,

and we obtain $|d^+_\Lambda(w_j) - d^-_\Lambda(w_j)| = 0$ which contradicts the optimality of Λ. Similarly, in the second case it implies that the orientations assigned to the edges of the clause gadgets are the opposite from the previous ones and we obtain the same contradiction. So we can conclude that v does satisfy φ.

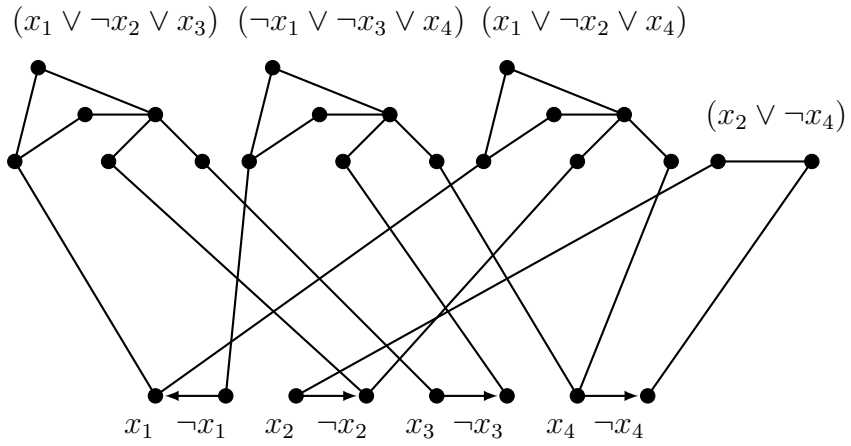
□

We can now safely conclude about the complexity of MaxIm.

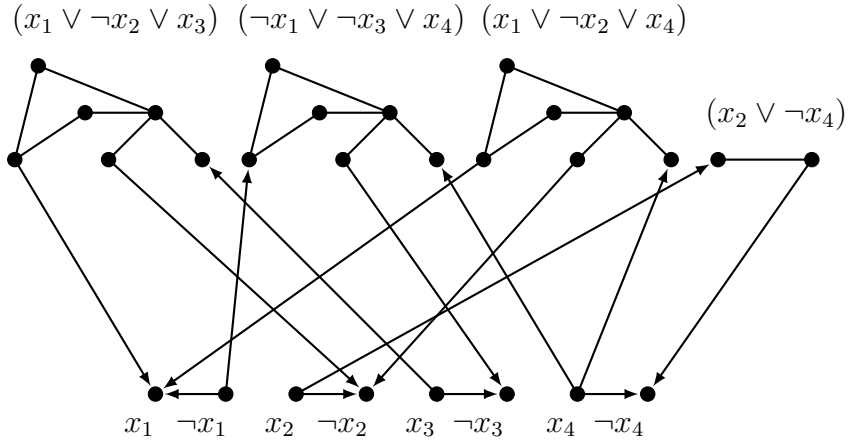
Corollary 23. *The decision problem consisting in asserting for a graph G and an integer $k \in \mathbb{N}$ if $\textsc{MaxIm}(G) \geq k$ is* NP*-complete .*

Proof. For a graph G and an integer $k \in \mathbb{N}$, if given a solution for MaxIm, that is to say an orientation of G, one can check if its value is larger than k in polynomial time: one only has to compute the imbalance of each vertex w.r.t. said orientation and then sum them, obtaining the value of the solution to compare with k. Hence the decision problem inherent to MaxIm is in NP. The NP-completeness of the problem comes naturally from Theorem 22. □

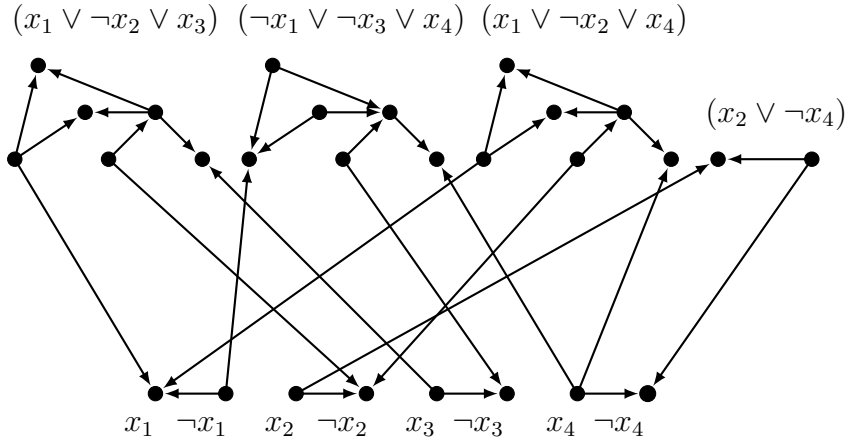
Figure 2.3: G_φ corresponding to $\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee \neg x_2 \vee x_4) \wedge (x_2 \vee \neg x_4)$ satisfied by $v(x_1, x_2, x_3, x_4) = (\text{FALSE}, \text{TRUE}, \text{TRUE}, \text{TRUE})$.



(a) orientation of the edges in the variable gadgets



(b) orientation of the edges between the variable gadgets and the clause gadgets



(c) orientation of the edges in the clause gadgets

2.1.2 Inapproximability

The reduction of such a famous NP-complete problem as 3-SAT to MaxIm allows to further derive results about its difficulty by simply transposing results about 3-SAT.

Corollary 24. MaxIm *is inapproximable within* $\frac{1}{2}+\varepsilon$ *where* $\varepsilon \in \mathbb{R}_+^*$, *unless* $\mathsf{P} = \mathsf{NP}$.

Proof. Let $\varepsilon \in \mathbb{R}_+^*$ and suppose that there exists a polynomial approximation algorithm giving $val \geq (\frac{1}{2}+\varepsilon)\,\textsc{MaxIm}(G)$ for an input graph G. Let φ be a not-all-equal at most 3-SAT(3V) formula and G_φ its associated graph. We have $\delta_{G_\varphi} = 2$, hence $\textsc{MaxIm}(G_\varphi) \in \{0, 1, 2\}$. Since $(\frac{1}{2} + \varepsilon)\,\textsc{MaxIm}(G_\varphi) \leq val \leq \textsc{MaxIm}(G_\varphi)$, if the polynomial approximation algorithm returns a value lower than or equal to 1 then

$$(\frac{1}{2} + \varepsilon)\,\textsc{MaxIm}(G_\varphi) \leq 1 \Rightarrow \textsc{MaxIm}(G_\varphi) < 2 \Rightarrow \textsc{MaxIm}(G_\varphi) \leq 1.$$

On the other hand, if it returns a value larger than 1, then $\textsc{MaxIm}(G_\varphi)$ is larger than 1 hence equal to 2. In other words the polynomial approximation algorithm output answers, through Theorem 22, whether φ is satisfiable or not which implies $\mathsf{P} = \mathsf{NP}$. □

2.1.3 Lower bound and approximation algorithm

Let us now define a nifty little family of graphs. A path in G is *Eulerian* if it visits each edge of G exactly once. An *Eulerian cycle* is a Eulerian path which starts and ends on the same vertex. A connected graph is said to be *Eulerian* if it admits a Eulerian cycle. They were first discussed by Leonhard Euler while solving the famous "seven bridges of Königsberg problem" in 1736 and thus laying the foundations of graph theory and prefiguring the idea of topology [38]. Euler proved that a necessary condition for the existence of Eulerian cycles is that all vertices in the graph have an even degree and stated its sufficiency without proof. It was proved sufficient in 1873 by Carl Hierholzer [62].

Notice that Eulerian graphs are interesting fellows vis-à-vis imbalance: they admit an orientation such that the imbalance of each of their vertices equals zero. It is obtained by orienting a Eulerian cycle of the graph as a circuit. This trick may come in handy if we manage to partially orient a graph guaranteeing a certain lower bound on the imbalance of all the vertices. If the subgraph corresponding to the unoriented edges is Eulerian, we can then mantain the lower bound on the imbalance of all the vertices whilst completing the orientation.

If the input graph is dense, more precisely, if δ_G is high, then we can think of a way to orient the edges in such a way that for each vertex, a guaranteed proportion of the edges incident to it are ingoing (or outgoing). To do that, one must bear in

mind that in the end the set of vertices will be partitioned into the vertices with a positive signed imbalance and those with a negative signed imbalance so when we choose the orientation of the edges, we must try to minimize the edges connecting two vertices inside the same partition for it will necessarily lower the imbalance of one of its endpoint. If we consider the case of bipartite graphs: if $G = (V1 \uplus V_2, E)$ is a bipartite graph, the orientation that consists in assigning to each edge in E the orientation from its endpoint in V_1 to its endpoint in V_2 has an imbalance equal to δ_G, i.e. optimal. A good idea would then be to search for a bipartite subgraph of the input graph with a set of edges containing a guaranteed proportion of the edges incident to each vertex.

All of these considerations will permit us to obtain the following lower bound:

Theorem 25. *For every graph G,*

$$\textsc{MaxIm}(G) \geq \left\lceil \frac{\delta_G}{2} \right\rceil - 1.$$

Proof. Let $S \subseteq V$ be a a subset of vertices of G defining a *locally maximum cut*, that is a cut verifying $|\delta(\{v\}) \cap \delta(S)| \geq \left\lceil \frac{d(v)}{2} \right\rceil$, $\forall v \in V$ obtained via a simple greedy algorithm. Such a cut exists: for example a maximum cardinality cut verifies this property, otherwise we could find a higher cardinality cut by switching a vertex $v \in V$ s.t. $|\delta(\{v\}) \cap \delta(S)| < \left\lceil \frac{d(v)}{2} \right\rceil$ from S to $V \backslash S$ (or the contrary). Moreover, if we iterated this process starting from a random cut, we would converge in polynomial time to such a cut. Now we define $\Lambda \in \vec{O}(G)$ as follows, process illustrated in Figure 2.4. We begin by removing all edges in $\delta(S)$ and orient the edges of the subgraph H, union of the induced subgraphs $G(S)$ and $G(V \backslash S)$. If it is not Eulerian, we add a new vertex v_0 and an edge between v_0 and each vertex with an odd degree in H, the resulting graph is therefore Eulerian (see Figure 2.4(b)). We therefore consider a Eulerian cycle -or one on each connected component to be more accurate- (see Figure 2.4(c)) and orient it as a circuit (see Figure 2.4(d)). Removing v_0 if necessary and putting back the edges of $\delta(S)$ oriented from their endpoint in S to their endpoint to $V \backslash S$ (see Figure 2.4(e)). The imbalance of each vertex in H is now in $\{0, 1\}$ which implies that $\forall v \in V$ we have $|d^+_\Lambda(v) - d^-_\Lambda(v)| \geq \left\lceil \frac{d(v)}{2} \right\rceil - 1$, hence, $\textsc{MaxIm}(G) \geq \left\lceil \frac{\delta_G}{2} \right\rceil - 1$. □

Observation. It is easy to see that for a graph G, the orientation Λ obtained in polynomial time following the previous proof has an imbalance $val(\Lambda)$ bounded in the following conditioned manner.

- $val(\Lambda) \geq \frac{\delta_G}{2} \geq \frac{1}{2}\,\textsc{MaxIm}(G)$ if $\delta_G \equiv 0 \pmod 4$;

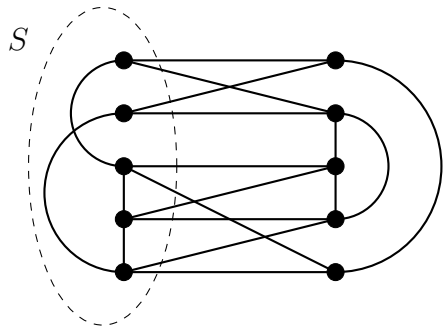
- $val(\Lambda) \geq \frac{\delta_G - 1}{2} \geq (\frac{1}{2} - \frac{1}{2\delta_G})\,\textsc{MaxIm}(G)$ if δ_G is odd;

- $val(\Lambda) \geq \frac{\delta_G}{2} - 1 \geq (\frac{1}{2} - \frac{1}{\delta_G}\,\textsc{MaxIm}(G)$ if $\delta_G \equiv 2 \pmod 4$.

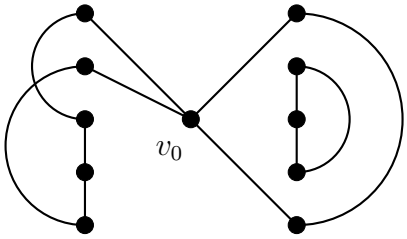
This leads to a polynomial time approximation algorithm for MaxIm whose ratio is $\frac{1}{2} - \frac{1}{\delta_G}$ in general and more precisely $\frac{1}{2}$ if $\delta_G \equiv 0 \pmod 4$ and $\frac{1}{2} - \frac{1}{2\delta}$ if δ_G is odd.

Now to gather more insight concerning our orienting graphs minimizing the imbalance, we can look for a way to divide our problem in sub-problems.

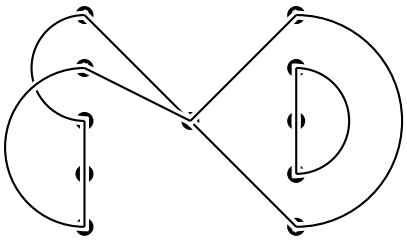
Figure 2.4: Orientation derived from a bipartite subgraph



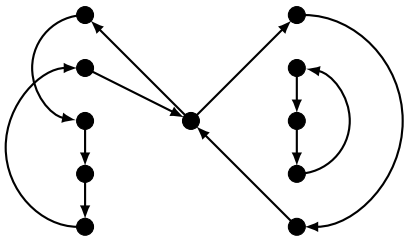
(a) The input graph with its locally maximum cut defined by S



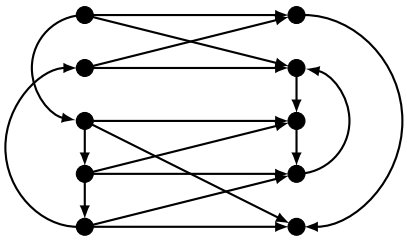
(b) Removing the edges of the cut and adding a temporary vertex v_0 connected to the odd-degree vertices of the remaining subgraph (unless it is already Eulerian)



(c) Finding a Eulerian cycle (or one in each connected component)



(d) Orienting the Eulerian cycle(s) as a circuit



(e) Removing v_0 and completing the orientation by by orienting the edges of $\delta(s)$ from S to $V \backslash S$

2.1.4 Block-cut-vertex tree

A good way to divide our problem is to orient its blocks and then somehow assemble these orientations in such a way that the imbalance of the resulting orientation is at least the minimum of the imbalances of the orientations of the blocks. In similar manner, consider the problem called *graph colouring* consisting in assigning a colour to each vertex of a graph in such a way that no two connected vertices bear the same colour. In 1941, Leonard Brooks stated that the vertices of a graph G can be coloured with only Δ_G colours except for two cases: complete graphs and odd cycle graphs which require at least $\Delta_G + 1$ [23]. In 1975, László Lovász gave a simplified proof consisting in first colouring the blocks of the input graph separately and then combining them in order to obtain a colouring of the whole graph [77]. Whether it is for an orientation or a colouring, it is not obvious why it should be easy to assemble sub-colourings or sub-orientations into a whole without loss of crucial properties. Let us mention a structure introduced by Frank Harary -one of the "fathers" of graph theory- in his seminal book *Graph Theory* in 1969 [58]. This structure comes from the fact that a connected graph with many cut-vertices bears a resemblance to a tree. The idea is then to associate to every connected graph a tree displaying this resemblance, focusing on the structure followed by the blocks of the graph.

Definition 26. *Let $G = (V, E)$ be a graph, V_B the set of its blocks and V_C the set of its cut-vertices. The* block-cut-vertex tree *of G is the graph $T = (V_B \uplus V_C, E')$ where*

$$E' = \{Zu | Z \in V_B,\ u \in V_C \text{ and } u \in V(Z)\}.$$

Observe that the block-cut-vertex tree of a connected graph is a tree [34]. In Figure 2.5 we have an example of a graph, its cut-vertices and its blocks (Figure 2.5(a) and its block-cut-vertex tree 2.5(b).

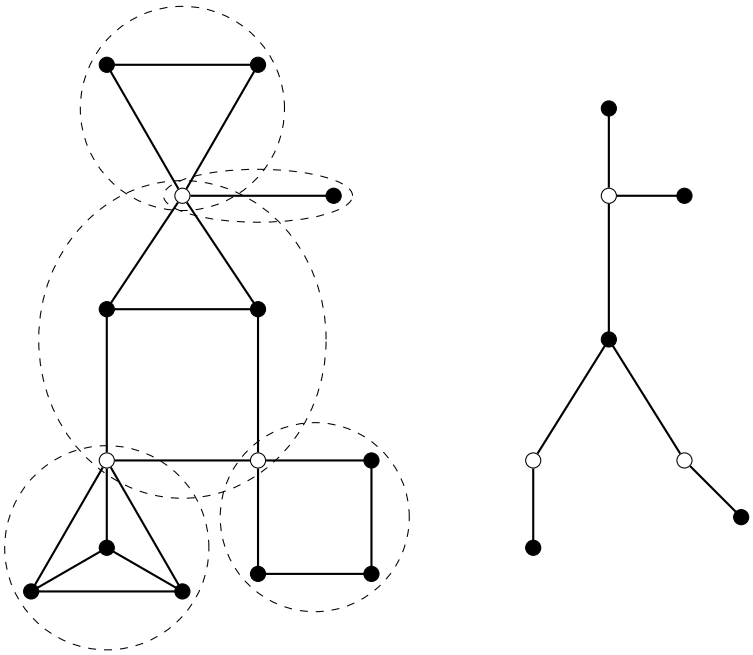
This tree-like structure of the blocks of a connected graph is the reason why we can assemble the orientations of the blocks into an orientation where the imbalance of each vertex is at least its the sum of its imbalances in each block it belongs to if it's a cut vertex and a least its imbalance inside the block it belongs to otherwise.

Proposition 27. *Let G be a graph and $B_1, \cdots, B_n$ its blocks. Let $(\Lambda_1, \cdots, \Lambda_n) \in \vec{O}(B_1) \times \cdots \times \vec{O}(B_n)$ be optimal orientations of respectively $B_1, \cdots, B_n$ for* MaxIm. *Then we have*

$$\textsc{MaxIm}(G) \geq \min\Big(\min_{c \in V_C} \sum_{i | c \in V(B_i)} |d^+_{\Lambda_i}(c) - d^-_{\Lambda_i}(c)|,\ \min_{\substack{v \in V \setminus V_C \\ i \in [\![1,n]\!] \\ s.t.\ v \in V(B_i)}} |d^+_{\Lambda_i}(v) - d^-_{\Lambda_i}(v)|\Big), \quad (2.2)$$

$$\geq \min_{i \in [\![1,n]\!]} \textsc{MaxIm}(B_i). \quad (2.3)$$

Figure 2.5: Example of a block-cut-vertex tree



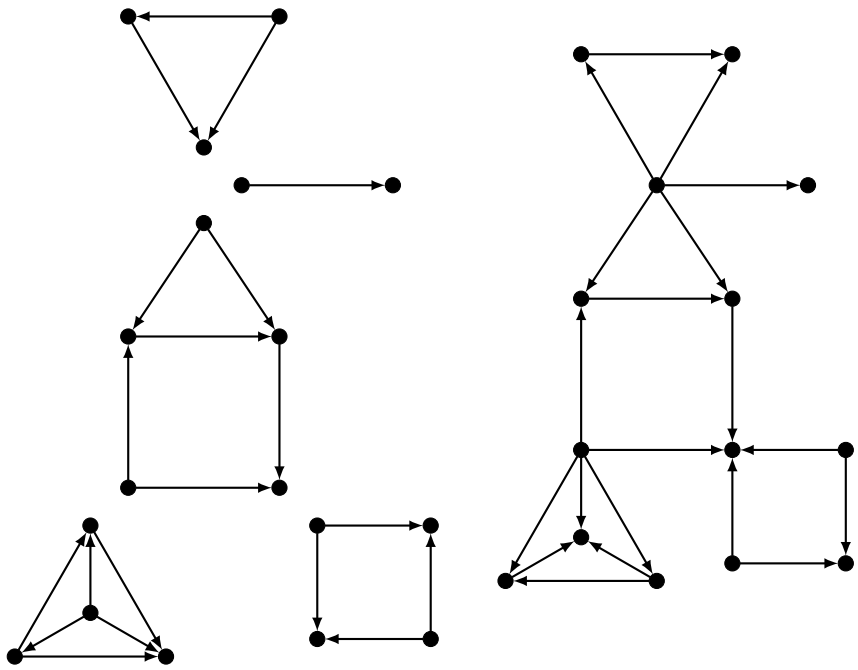
(a) A connected graph with its blocks circled and its cut-vertices in white

(b) Its block-cut-vertex tree with blocks as black vertices and cut-vertices as white vertices

Proof. The idea is to optimally orient the blocks of G separately, and then reassembling them one at a time. When "gluing" two pieces that share a cut-vertex c back together, two cases appear. Either the signed imbalances of c in each block have the same size in which case we assemble it as is and the imbalance of c becomes the sum of its imbalances in each block ; or they have opposite signs in which case we can reverse the orientation inside all of one of the two blocks (along with all the blocks eventually already connected to it). The imbalance of the vertices are left unchanged, only the signed imbalances have changed and we are now back to the first case. The imbalance or the resulting orientation is the left hand side of (2.2). This process is illustrated in Figure 2.6 where we start by optimally orienting each block individually (Figure 2.6(a)) and then assembling these orientation, adjusting them as explained above (Figure 2.6(b)). In this example, we have that (2.2) is an equality and (2.3) is a strict inequality, showing the possible gap between these two lower bounds for MaxIm(G). □

We just used the fact that reversing an orientation leaves imbalances unchanged, the same way that Lovász used the fact that permuting the colours of a colouring leads to a equally proper colouring with the same number of colours.

Figure 2.6: Orienting the graph from Figure 2.5 from optimal orientations of its blocks



(a) Each block optimally oriented (individually)

(b) Reassembling the blocks adjusting the orientations

2.2 Characterizing the graphs for which $\mathbf{MaxIm}(G) = 0$

In order to have a better grasp of the mechanics of maximum imbalance orientations, we choose to consider the graphs verifying $\textsc{MaxIm}(G) = 0$. This class of graphs can be designated as follows: the graphs in which there is at least one vertex with imbalance equal to zero w.r.t. whichever orientation of the edges. The first candidates that come to mind are the odd cycles. Their counterparts the even cycles can elegantly be oriented alternating the direction of the edges to obtain a 2-imbalanced orientation, which cannot be done for the odd cycles. Then one can start thinking of appending subgraphs to cycles and one realizes pretty fast that the problem is not that clear.

First, Theorem 25 implies that $\delta_G \leq 2$. then, in light of Proposition 27, we know that at least one of the block of such a graph must admit only zero imbalance orientations. Let's start by unveiling several additional necessary conditions and properties of such graphs.

2.2.1 Choosing the balanced vertex

First we can show that concerning such a graph, we can find an orientation with positive imbalance on all but one vertex.

Proposition 28. *Let G be a graph such that* $\textsc{MaxIm}(G) = 0$ *and $u \in V$. Then there exists an orientation $\Lambda \in \vec{O}(G)$ such that u is the only vertex of G with imbalance equal to zero w.r.t. Λ.*

Proof. Let $\Lambda \in \vec{O}(G)$ be an orientation minimizing $|\{v \in V / |d^+_\Lambda(v) - d^-_\Lambda(v)| = 0\}|$. We suppose that $|\{v \in V / |d^+_\Lambda(v) - d^-_\Lambda(v)| = 0\}| \geq 2$. We choose two distinct vertices v and w in $\{v \in V / |d^+_\Lambda(v) - d^-_\Lambda(v)| = 0\}$ and an undirected path $p = (v = u_0, \cdots, u_n = w)$ between v and w. If we switch the orientation of the edge u_0u_1, then the imbalance of u_0 becomes positive and necessarily the imbalance of u_1 becomes zero otherwise the resulting orientation would contradict the minimality of Λ. Using the same reasoning, if we switch the orientation of all the edges $u_0u_1, \cdots, u_{n-2}u_{n-1}$, we obtain an orientation where both u_{n-1} and u_n have an imbalance equal to zero while the imbalance is positive on all the vertices $u_0, \cdots, u_{n-2}$ and unchanged on all other vertices. So now if we switch the orientation of the edge $u_{n-1}u_n$ as well, then the resulting orientation contradicts the minimality of Λ. Hence, $|\{v \in V / |d^+_\Lambda(v) - d^-_\Lambda(v)| = 0\}| = 1$.

Now let v be this unique vertex of G such that $|d^+_\Lambda(v) - d^-_\Lambda(v)| = 0$. Let $u \neq v$ be an arbitrary vertex and let $p = (v = u_0, \cdots, u_n = u)$ be a path between v and u. If we switch the orientation of the edge u_0u_1, then the imbalance of u_0 becomes positive and necessarily the imbalance of u_1 becomes zero otherwise the resulting orientation would contradict $\textsc{MaxIm}(G) = 0$. Using the same reasoning, if we switch the orientation of all the edges $u_0u_1, \cdots, u_{n-1}u_n$, we obtain an orientation Λ' where u has an imbalance equal to zero while the imbalance is positive for all the other vertices. □

This yields the following necessary condition: if G is a graph such that $\textsc{MaxIm}(G) = 0$, then G is Eulerian. For suppose $\textsc{MaxIm}(G) = 0$ and let $u \in V$, we know there exists $\Lambda \in \vec{O}(G)$ such that $\{v \in V / |d^+_\Lambda(v) - d^-_\Lambda(v)| = 0\} = \{u\}$. Then $d^+_\Lambda(u) = d^-_\Lambda(u)$, hence $d(u) = d^+_\Lambda(u) + d^-_\Lambda(u) = 2d^+_\Lambda(u)$ is even. Since we can get this conclusion for any vertex in the graph G, we then know that it is Eulerian.

2.2.2 Orienting the blocks

Propositions 27 and 28 lead us to a strong characteristic for graphs such that $\textsc{MaxIm}(G) = 0$: all of their blocks must follow the same constraint.

Proposition 29. *Let G be a graph such that* $\textsc{MaxIm}(G) = 0$ *and* $B_1, \cdots, B_n$ *its blocks. Then we have*

$$\textsc{MaxIm}(B_i) = 0, \ \forall i \in [\![1, n]\!].$$

Proof. Suppose that G has a block B_{i_0} such that $\textsc{MaxIm}(B_{i_0}) > 0$, Let us build an orientation of G with a strictly positive imbalance. Consider the block-cut-vertex tree T of G rooted in B_0, we orient the blocks of G running through T. We start by taking an optimal orientation of B_{i_0}, thus with a strictly positive imbalance. Then we run down the tree T as follows. Let $B \in V_B \backslash \{B_{i_0}\}$ a block of G, $c \in V_C$ the cut-vertex of G that is the parent node of B in T and $B' \in V_B$ the block of G that is the parent node of c in T (c is then a cut-vertex of G belonging to both B and B'. We have two cases concerning the maximum imbalance of B:

- if $\textsc{MaxIm}(B) > 0$, then we orient its edge in an optimal way such that the signed imbalance of c in B shares the sign of the signed imbalance of C in B';
- if $\textsc{MaxIm}(B) = 0$, then we orient it in such a way that its only zero imbalance vertex is c.

The resulting orientation has a strictly positive orientation for the imbalance of each non-cut-vertex in each block is strictly positive and the imbalance of each cut-vertex being the sum ot its imbalances in each block is strictly positive because its imbalance in the block that is its parent node in T is strictly positive. Therefore $\textsc{MaxIm}(G) > 0$. □

With this proposition, we now know that a good candidate for our sought family could be an assembling of biconnected graphs from the same family following a tree structure. Thus when we think about biconnected graphs with a minimum degree lower than or equal to 2, we first think of cycles or single edges. Among those, only the odd cycles are part of our sought family. With the various properties being respected by our graphs we are able now to give, we can present a first constructive characterization.

2.2.3 A first characterization

The following lemma about Eulerian graphs will be useful for the proof of our characterization. It asserts that in a Eulerian graph, there is a cycle that one can qualify as "peripheral", that is, the removal of its edges will not increase the number of connected components (not counting isolated vertices):

Lemma 30. *If G is an Eulerian graph, then there exists an elementary cycle (*hereafter just called cycle*) C of G such that $G - E(C)$ has at most one connected component that is not an isolated vertex.*

Proof. Since G is Eulerian and connected, it can be decomposed into edge-disjoint cycles that we can order $C_1, \cdots, C_n$ according to the following condition: $\cup_{k=1}^{i} C_i$ is connected, $\forall i \in [\![1, n]\!]$. It boils down to assembling them to form G in such a way that we have a connected subgraph of G at each step. Then C_n is the cycle we are looking for. □

We call $\mathcal{C}^{odd}$ the family of connected graphs for which every block is an odd cycle. In order to manipulate them easily, we give the following characterization for the graphs in $\mathcal{C}^{odd}$.

Lemma 31. *A graph G is in $\mathcal{C}^{odd}$ if and only if there exists a finite sequence of odd cycles $C_1, \cdots, C_n$ ($n \geq 1$) such that:*

$$\begin{aligned} &\bullet\ \cup_{i=1}^{n} C_i = G, \\ &\bullet\ |V(\cup_{k=1}^{i-1} C_k) \cap V(C_i)| = 1,\ \forall i \in [\![2, n]\!]. \end{aligned} \tag{2.4}$$

Proof. To obtain this sequence from a graph in $\mathcal{C}^{odd}$, consider T the block-cut-vertex tree of G rooted in one vertex corresponding to a block of G (which will be C_1) and n the number of blocks in G. We do a depth-first-search (or a breadth-first-search) in T, labeling only the vertices of T representing blocks of G (i.e. the vertices in V_B) from 1 to n. Then C_i is the block of G (that is an odd cycle, by definition of $\mathcal{C}^{odd}$) labeled i. □

As one might have expected, the previously defined and characterized family of graphs $\mathcal{C}^{odd}$ is the one we were looking for.

Theorem 32. *For any graph G, $\textsc{MaxIm}(G) = 0$ if and only if $G \in \mathcal{C}^{odd}$.*

Proof.

- $\Leftarrow$ We work by induction on the number n of cycles contained in the graph. Nothing is required for these cycles except that they must be elementary. If $n = 1$, then our graph is an odd cycle which implies $\textsc{MaxIm}(G) = 0$. Let $n \geq 2$, we assume that all graphs of $\mathcal{C}^{odd}$ with $k \leq n-1$ cycles verify $\textsc{MaxIm}(G) = 0$. Let $G \in \mathcal{C}^{odd}$ with n cycles $C_1, \cdots, C_n$ as in (2.4). Suppose there exists $\Lambda \in \vec{O}(G)$ with strictly positive imbalance. Let us call $G' = \cup_{i=1}^{n-1} C_i$ the graph obtained from G after removing C_n and let us take a look at $\Lambda_{|G'}$: the orientation of the edges of G' obtained from Λ restricted to $E(G')$. As G' is a graph of $n-1$ cycles in $\mathcal{C}^{odd}$, our inductive hypothesis implies that we have a vertex $u \in V(G')$ such that $|d^+_{\Lambda_{|G'}}(u) - d^-_{\Lambda_{|G'}}(u)| = 0$. Necessarily, $u = V(G') \cap V(C_n)$. Thus $|d^+_{\Lambda}(u) - d^-_{\Lambda}(u)| = |d^+_{\Lambda_{|C_n}}(u) - d^-_{\Lambda_{|C_n}}(u)| > 0$ implying that $\textsc{MaxIm}(C_n) > 0$, a contradiction because C_n is an odd cycle.

- $\Rightarrow$ Since $\textsc{MaxIm}(G) = 0$, we know that G is Eulerian. We work again by induction on the number of elementary cycles n. If $n = 1$, then our graph is Eulerian with a unique cycle, hence it is a cycle. Now as $\textsc{MaxIm}(G) = 0$, necessarily it is an odd cycle and is therefore in $\mathcal{C}^{odd}$. Let $n \geq 2$, we assume that all graphs with $k \leq n-1$ cycles verifying $\textsc{MaxIm}(G) = 0$ are in $\mathcal{C}^{odd}$. Let G be a graph with n cycles such that $\textsc{MaxIm}(G) = 0$. Thanks to Lemma 30, there exists a cycle C of G such that $G - E(C)$ has at most one connected component G' that is not an isolated vertex.
  Suppose that $\textsc{MaxIm}(G') > 0$, let $\Lambda \in \vec{O}(G')$ with strictly positive imbalance. Let $u_0 \in V(G') \cap V(C)$, we name the vertices of C as follows: $u_0, u_1, \cdots, u_k = u_0$. Without loss of generality, we can assume that $d^+_\Lambda(u_0) - d^-_\Lambda(u_0) > 0$; if it was not the case, replace Λ by its reverse. We complete Λ in an orientation of G by orienting the edges of C: we orient u_0u_1 from u_0 to u_1 and go on as follows:

$$\forall i \in [\![1, k-1]\!], \begin{cases} \text{if } u_i \in V(G'), & \text{we orient } u_iu_{i+1} \text{ as } u_{i-1}u_i, \\ \text{otherwise,} & \text{we orient } u_iu_{i+1} \text{ as } u_iu_{i-1}. \end{cases}$$

  Where orienting an edge ab as another edge cd means orienting it from a to b if cd was oriented from c to d and from b to a otherwise. Let us have a look at the resulting orientation Λ' (cf Fig. 2.7): when completing Λ in Λ', the imbalance of the vertices in $V(G')\backslash\{u_0\}$ was left unchanged, the imbalance of the vertices in $V(C)\backslash V(G')$ equals 2 and the imbalance of u_0 was either left unchanged or augmented by two. Hence Λ' has strictly positive imbalance which contradicts $\textsc{MaxIm}(G) = 0$, therefore, $\textsc{MaxIm}(G') = 0$.
  Suppose $|V(G') \cap V(C)| \geq 2$ and let u and v be 2 distinct vertices in $V(G') \cap V(C)$ such that $u \neq v$. Thanks to Proposition 28, we know that there exists an orientation $\Lambda \in \vec{O}(G')$ such that $\{w \in V / |d^+_\Lambda(w) - d^-_\Lambda(w)| = 0\} = \{v\}$ and without loss of generality, $d^+_\Lambda(u) - d^-_\Lambda(u) > 0$. We name the vertices of C as follows: $u = u_0u_1 \cdots u_k = u_0$, $v = u_l$ and we complete Λ in an orientation of G by orienting the edges of C: we orient u_0u_1 from u_0 and u_1 and go on as follows:

$$\forall i \in [\![1, k-1]\!] \setminus \{l\}, \begin{cases} \text{if } u_i \in V(G'), & \text{we orient } u_iu_{i+1} \text{ as } u_{i-1}u_i, \\ \text{otherwise,} & \text{we orient } u_iu_{i+1} \text{ as } u_iu_{i-1}. \end{cases}$$

  And we orient u_lu_{l+1} as u_lu_{l-1}. In the resulting orientation Λ', the imbalance of the vertices in $V(G')\backslash\{u, v\}$ was left unchanged, the imbalance of the vertices in $V(C)\backslash V(G')$ equals 2, the imbalance of v was augmented by two and the

imbalance of u was either left unchanged or augmented by two. Hence Λ' contradicts $\textsc{MaxIm}(G) = 0$, therefore, $|V(G') \cap V(C)| = 1$.
Suppose C is even. We call $u \in V(G')$ such that $V(G') \cap V(C) = \{u\}$, and $\Lambda \in \vec{O}(G')$ such that $\{v \in V / |d^+_\Lambda(v) - d^-_\Lambda(v)| = 0\} = \{u\}$. We name the vertices of C as follows: $u = u_0 u_1 \cdots u_k = u_0$ and we complete Λ in an orientation of G by orienting the edges of C: we orient u_0u_1 from u_0 to u_1 and u_iu_{i+1} as $u_iu_{i-1},\ \forall i \in [\![1, k-1]\!]$. In the resulting orientation Λ', the imbalance of the vertices in $V(G')\backslash\{u\}$ was left unchanged, the imbalance of the vertices in $V(C)\backslash V(G')$ equals 2 and, C being even, the imbalance of u was augmented by two. Hence Λ' contradicts $\textsc{MaxIm}(G) = 0$, therefore, C is odd.
As G' is a graph with at most $n-1$ cycles verifying $\textsc{MaxIm}(G) = 0$, by the induction hypothesis, there exist $C_1, \cdots, C_{n-1}$ odd cycles such that:

- $\cup_{i=1}^{n-1} C_i = G'$,
- $|V(\cup_{k=1}^{i-1} C_k) \cap V(C_i)| = 1,\ \forall i \in [\![2, n-1]\!]$.

Adding the odd cycle $C_n = C$, we obtain that $G \in \mathcal{C}^{odd}$.

□

Figure 2.7: The vertices of C in G' are left unchanged imbalance-wise, the other vertices of C are set to 2 and in the end $|d^+_{\Lambda'}(u_0) - d^-_{\Lambda'}(u_0)| \geq |d^+_\Lambda(u_0) - d^-_\Lambda(u_0)| > 0$.

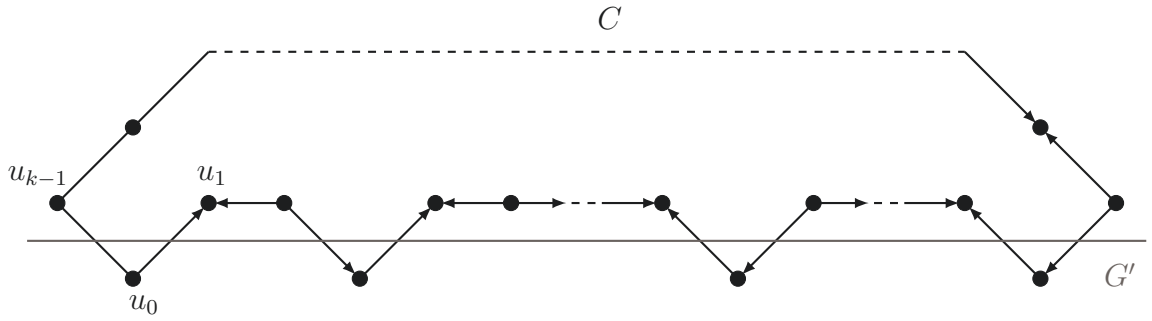
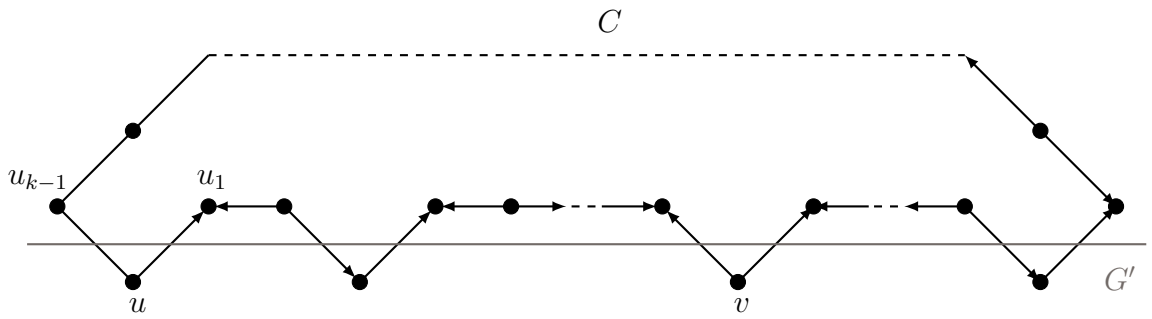


Figure 2.8: The vertices of C in G' are left unchanged imbalance-wise except for v which is set to 2, like the other vertices of C and in the end $|d^+_{\Lambda'}(u_0) - d^-_{\Lambda'}(u_0)| \geq |d^+_{\Lambda}(u_0) - d^-_{\Lambda}(u_0)| > 0$.



2.2.4 A more elegant characterization

Now in order to widen our perception of those graphs, we want to give a simpler characterization for the graphs in $\mathcal{C}^{odd}$. Recall Proposition 27, it is clear that if a block of a graph is an even cycle, then we can build an orientation of its edges with a strictly positive imbalance. Therefore the presence of an even cycle as a block is prohibitive w.r.t. $\mathcal{C}^{odd}$. We can go further in this direction, leaving the concept of blocks behind.

Theorem 33. *For every graph G,*

$$G \in \mathcal{C}^{odd} \Leftrightarrow G \text{ is Eulerian with no even cycle}$$

Proof.

- $\Rightarrow$ By construction, every graph in $\mathcal{C}^{odd}$ is Eulerian with no even cycle.

- $\Leftarrow$ We will once again work by induction on the number of cycles n.
  If $n = 1$, then our graph is Eulerian with a unique odd cycle, hence it is an odd cycle and is therefore in $\mathcal{C}^{odd}$.
  Let $n \geq 2$, we assume that all Eulerian graphs with no even cycle and $k \leq n-1$ odd cycles are in $\mathcal{C}^{odd}$. Let G be a graph with no even cycle and n odd cycles. Thanks to Lemma 30, there exists an odd cycle C of G such that $G - E(C)$ has only one connected component G' that is not an isolated vertex. As G' is Eulerian and even-cycle-free with $n-1$ odd cycles, by induction hypothesis, $G' \in \mathcal{C}^{odd}$, hence there exist $C_1, \cdots, C_{n-1}$ odd cycles such that:

  - $\cup_{i=1}^{n-1} C_i = G'$,

- $|V(\cup_{k=1}^{i-1} C_k) \cap V(C_i)| = 1, \ \forall i \in [\![2, n-1]\!]$.

Suppose there exist u and v ($u \neq v$) belonging to $V(\cup_{k=1}^{n-1} C_k) \cap V(C)$. Since G' is connected, let p be an elementary path in G' between u and v. We can assume that u and v are the only vertices of C contained in p, otherwise we could replace v by the first vertex of C encountered when traveling on p from u. The cycle C defines two other vertex-disjoint paths between u and v: one even that we will call p_{even} and one odd that we will call p_{odd}. The path p being vertex disjoint with either p_{even} or p_{odd}, by concatenating it with the one corresponding to its parity, we obtain an even cycle of G, contradicting our hypothesis on G (cf. Fig. 2.9(a) and 2.9(b)). This yields that $|V(C) \cap V(G')| = 1$. From that we can conclude

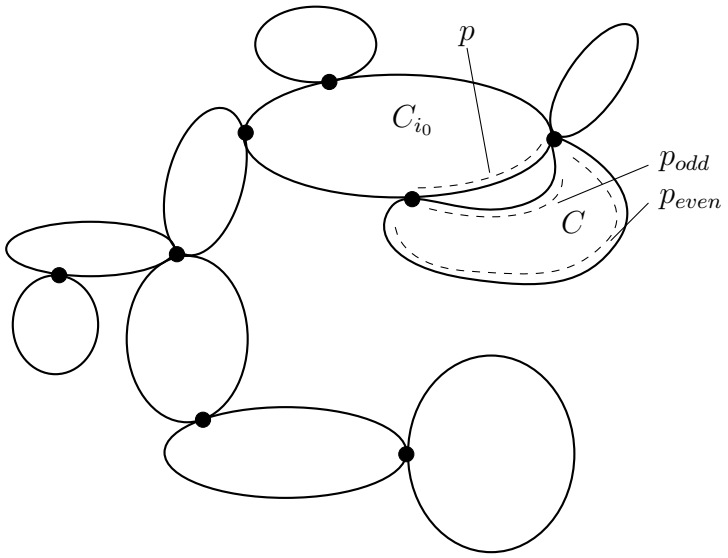
- $\cup_{i=1}^{n} C_i = G$,
- $|V(\cup_{k=1}^{i-1} C_k) \cap V(C_i)| = 1, \ \forall i \in [\![2, n]\!]$.

Hence $G \in \mathcal{C}^{odd}$.

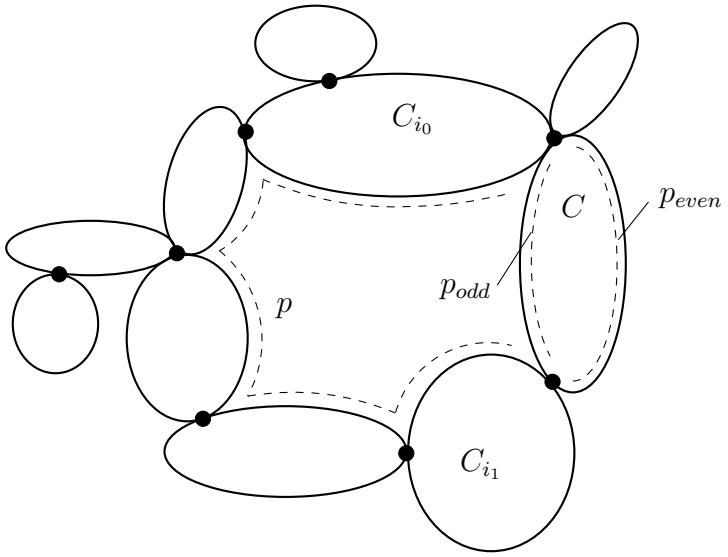
□

This last characterization is also equivalent to Eulerian graphs with no elementary even cycle.

Figure 2.9: In both cases, concatenating p with p_{even} or p_{odd} yields an even cycle in G (closed shapes represent cycles, only vertices belonging to more than one cycle are represented as a lack dot; dashed segments are used to higlight certain chains of vertices



(a) C intersects at least twice another cycle C_{i_0}



(b) C intersects at least two cycles C_{i_0} and C_{i_1}

2.3 Exact algorithm for cacti

The class of graphs defined in the previous section is a special case of cacti. A *cactus* (formerly known as Husimi trees) is a connected graph for which every edge belongs to at most one elementary cycle. Equivalently, a cactus is a connected graph for which every block is a single edge or a cycle. Notice that any cycle in a cactus is chordless.

The graph presented in Figure 2.10 in which we isolate its blocks (b) and give its block-cut-vertex tree T (c) is a cactus. For now we don't know much about the value of MaxIm for cacti except that it can be 0 for the family $\mathcal{C}^{odd}$ introduced in the previous section is a family of cacti (those which blocks are all odd cycles).

2.3.1 A lower bound for cacti

The next lemma is related to the minimum degree of a cactus.

Lemma 34. *Let G be a cactus graph,*

$$\delta_G \leq 2.$$

Proof. Let G be a cactus graph and T its block-cut-vertex tree. Let $u \in V(T)$ be a leaf of T. By the definition of a cut-vertex, u corresponds necessarily to a block of G. If u corresponds to a bridge of G, then one of its endpoint (the one that is not a cut-vertex) is a leaf of G which implies δ_G=1. If u corresponds to a cycle of G, being a leaf of T, it contains only one cut-vertex of G and therefore the degree of its other vertices equals 2. □

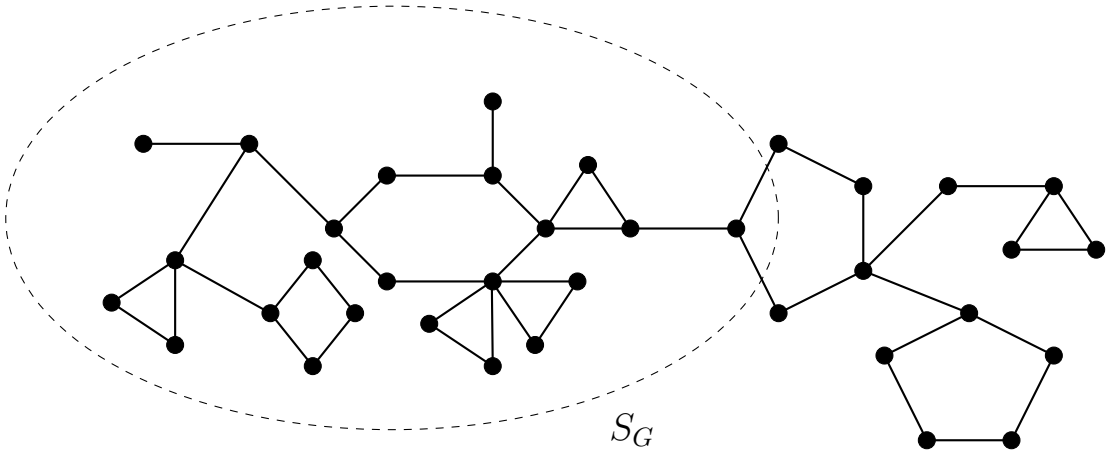
The previous lemma automatically provides an upper bound for MaxIm(G).

Proposition 35. *Let G be a cactus,*

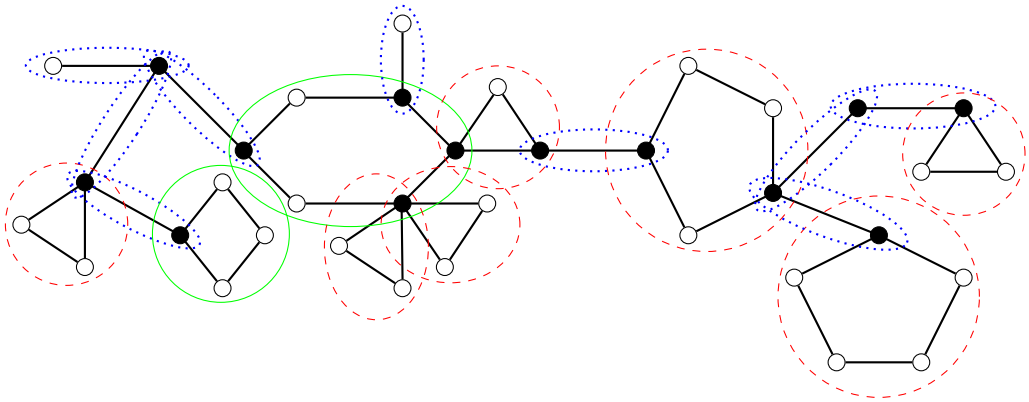
$$\text{MaxIm}(G) \leq 2.$$

And this bound allows to conlude that for a cactus G, we have MaxIm(G) $\in \{0, 1, 2\}$. We have already characterized which of the cacti verify MaxIm(G) $= 0$. The next question is can we find similar characterizations corresponding to the other possible values of MaxIm for a cactus.

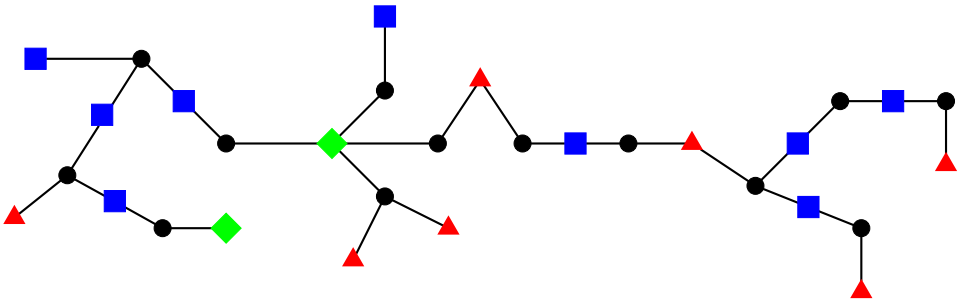
Figure 2.10: Building the block-cut-vertex tree



(a) G a cactus and S_G.



(b) Its blocks: even cycles (⬭), odd cycles (⬭) and bridges (⬭).



(c) And the block-cut-vertex tree T: cut vertices (●), even cycles (◆), odd cycles (▲) and bridges (■)

2.3.2 Characterizing the cacti for which MaxIm$(G) = 2$

Let us now characterize the cacti such that MaxIm$(G) = 2$. The following definition of the subset S_G for each cactus G will be the essence of our characterization.

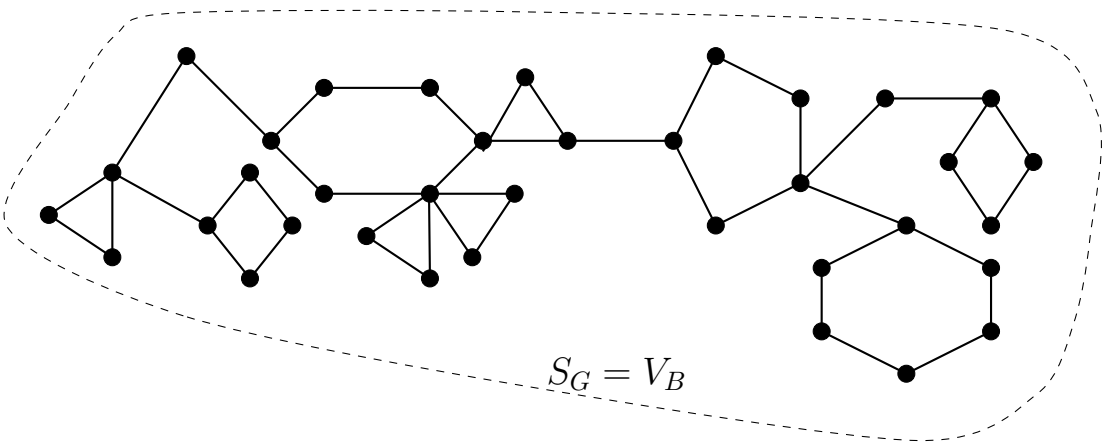
Let $G = (V, E)$ be a cactus and $T = (V_B \cup V_C, E')$ its block-cut-vertex tree. A subset $A \in V_B$ will hereafter refer to a subset of vertices of T as well as the subgraph consisting in the union of the blocks of G corresponding to these vertices. We denote $S_0 \subseteq V_B$ the set containing all the even cycles og G. We define for a cactus G the set $S_G \subseteq V_B$ as the smallest subset of blocks of G containing S_0 and satisfying the following conditions:

- If $Y \in V_B$ corresponds to an odd cycle of G that shares a cut-vertex with a cycle or with two bridges contained in S_G, then $Y \in S_G$,

- if $Y \in V_B$ corresponds to a bridge of G that shares a cut-vertex with a block contained in S_G, then $Y \in S_G$.

In Figures 2.10(a) and 2.11 are examples of two cacti along with their subsets S_G.

Figure 2.11: A graph G such that $\delta_G = 2$ and $S_G = V_B$.



Theorem 36. *Let G be a cactus,*

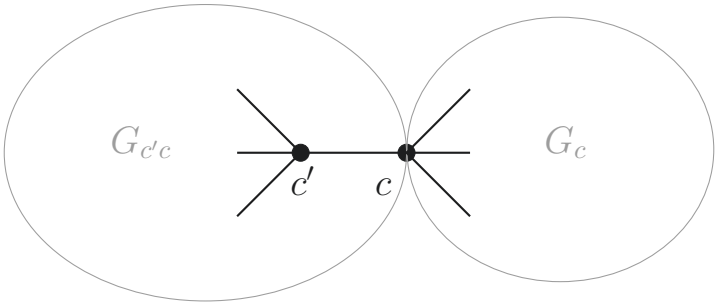
$$\text{MaxIm}(G) = 2 \iff S_G = V_B \text{ and } \delta_G = 2.$$

The idea of this result and its proof is that if we have a cactus G, we consider the subgraph S_0 which is bipartite and therefore can be oriented with imbalance at least 2. Then we can extend the orientation of S_0 into an orientation of S_G, keeping an imbalance of at least two for each vertex incident to at least two oriented edges.

Now if $\delta_G = 2$ and $S_G = V_B$, we obtain an orientation of G with imbalance at least 2. Moreover, for a cactus G, the subset S_G can be computed in polynomial time by a basic search on the block-cut-vertex tree of G.

In order to prove Theorem 36, we first show two lemmas that will be useful in the proof of our theorem.

Figure 2.12: A graph G and a bridge cc'.



Lemma 37. *Let $G = (V, E)$ be a graph, $cc' \in E$ a bridge of G. Let us call G_c the connected component of $G \backslash cc'$ containing c and $G_{c'c}$ the graph obtained by adding c and cc' to the connected component of $G \backslash cc'$ containing c' (see Fig. 2.12). Then*

$$\textsc{MaxIm}(G) \leq \max_{\Lambda \in \vec{O}(G_c)} \min(\min_{v \in V(G_c) \setminus \{c\}} |d^+_\Lambda(v) - d^-_\Lambda(v)|, |d^+_\Lambda(c) - d^-_\Lambda(c)| + 1).$$

Proof. Let $\Lambda \in \vec{O}(G)$ be an optimal orientation of G w.r.t. MaxIm. If $d^+_{\Lambda_{|G_{c'c}}}(c) - d^-_{\Lambda_{|G_{c'c}}}(c)$ and $d^+_{\Lambda_{|G_c}}(c) - d^-_{\Lambda_{|G_c}}(c)$ do not have the same sign, then we switch the assignment of Λ of the edges of $G_{c'c}$. Doing so, the imbalance of all the vertices of G is unchanged except for that of c which got risen by 2 hence Λ is still optimal and $|d^+_\Lambda(c) - d^-_\Lambda(c)| \leq |d^+_{\Lambda_{|G_c}}(c) - d^-_{\Lambda_{|G_c}}(c)| + 1$. Moreover,

$$\textsc{MaxIm}(G) = \min_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| \leq \min_{v \in V(G_c) \setminus \{c\}} |d^+_{\Lambda_{|G_c}}(v) - d^-_{\Lambda_{|G_c}}(v)|$$

which yields

$$\begin{aligned} \textsc{MaxIm}(G) &\leq \min(\min_{v \in V(G_c) \setminus \{c\}} |d^+_{\Lambda_{|G_c}}(v) - d^-_{\Lambda_{|G_c}}(v)|, |d^+_{\Lambda_{|G_c}}(c) - d^-_{\Lambda_{|G_c}}(c)| + 1) \\ &\leq \max_{\Lambda \in \vec{O}(G_c)} \min(\min_{v \in V(G_c) \setminus \{c\}} |d^+_\Lambda(v) - d^-_\Lambda(v)|, |d^+_\Lambda(c) - d^-_\Lambda(c)| + 1). \end{aligned}$$

□

Lemma 38. *Let G be a cactus such that $\delta_G = 2$. Then there exists a cycle of G with at most one gate (vertex adjacent to any vertex not belonging to the cycle).*

Proof. Let T be the block-cut-vertex tree of G. If T has no leaf, then it is a vertex graph, hence G is a cycle which necessarily has no gate. If T has a leaf l, it necessarily corresponds to a block of G. If l was a bridge, the degree of its endpoint in G which is not a cut-vertex would be 1, hence l is a cycle of G. And being a leaf in T, it has at most one gate in G. □

Proof of Theorem 36.

- $\Leftarrow$ We assume that $S_G = V_B$ and $\delta_G \geq 2$. From Proposition 35, we know that $\textsc{MaxIm}(G) \leq 2$. We will build an orientation $\Lambda \in \vec{O}(G)$ such that $\min_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| = 2$. We start by orienting the edges of the subgraph S_0 consisting in the union of all the even cycles of G. The subgraph S_0 having no odd cycles, it is bipartite and we can therefore choose an orientation of the edges of S_0 such that

$$|d^+_{\Lambda_{|S_0}}(v) - d^-_{\Lambda_{|S_0}}(v)| = d_{S_0}(v) \geq 2, \ \forall v \in V(S_0).$$

We now recursively extend Λ to the rest of the blocks in S_G ensuring an imbalance of at least 2 for each vertex adjacent to at least two oriented edges. Let $Z \in V_B$ be an unoriented block of G that is either:

- an odd cycle of G sharing a cut-vertex with an oriented cycle or two oriented bridges,
- a bridge sharing a cut-vertex with an oriented block.

If there was no such block and the graph G was not totally oriented, the set of oriented blocks of G denoted by $\vec{S}$ would contradict the minimality of S_G.

If Z is an odd cycle, we choose a cut-vertex c of Z adjacent to oriented edges and name the vertices of B $c = v_1 v_2 \cdots v_k = c$. We assign:

- $\Lambda(v_i v_{i+1}) = \begin{cases} \overrightarrow{v_i v_{i+1}} & \text{if } i \text{ is odd;} \\ \overrightarrow{v_{i+1} v_i} & \text{otherwise} \end{cases}, \ \forall i \in [\![1, k-1]\!].$

Let us now consider the imbalance of the vertices of Z. Since c is adjacent to either a cycle or two bridges in $\vec{S}$, it is adjacent to at least two edges in $\vec{S}$ and therefore $|d^+_{\Lambda_{|\vec{S}}}(c) - d^-_{\Lambda_{|\vec{S}}}(c)| \geq 2$ according to our inductive hypothesis. Since $|d^+_{\Lambda_{|Z}}(c) - d^-_{\Lambda_{|Z}}(c)| = 0$, $|d^+_\Lambda(c) - d^-_\Lambda(c)| = |d^+_{\Lambda_{|\vec{S}}}(c) - d^-_{\Lambda_{|\vec{S}}}(c)| \geq 2$. If there is another cut-vertex c' of Z that is adjacent to a block in $\vec{S}$ such that $d^+_{\Lambda_{|\vec{S}}}(c') - d^-_{\Lambda_{|\vec{S}}}(c')$ and $d^+_{\Lambda_{|Z}}(c') - d^-_{\Lambda_{|Z}}(c')$ do not have the same sign, then we switch the assignment of Λ of the edges of the whole connected component of $\vec{S}$ containing c' for its opposite. Necessarily, c' is the only vertex this connected component shares with Z otherwise Z would be contained in a bigger block

of G. Then doing so, the imbalance of all vertices is left unchanged except for that of c' which is now equal to

$$|d^+_{\Lambda_{|\vec{S}}}(c') - d^-_{\Lambda_{|\vec{S}}}(c')| + |d^+_{\Lambda_{|Z}}(c') - d^-_{\Lambda_{|Z}}(c')| = |d^+_{\Lambda_{|\vec{S}}}(c') - d^-_{\Lambda_{|\vec{S}}}(c')| + 2.$$

This process is repeated for all the cut-vertices c' of Z different from c adjacent to a block in $\vec{S}$ to be such that $|d^+_\Lambda(c') - d^-_\Lambda(c')| \geq 2$. If $u \in V(Z)$ is not a cut-vertex, then $|d^+_\Lambda(u) - d^-_\Lambda(u)| = |d^+_{\Lambda_{|Z]}}(u) - d^-_{\Lambda_{|Z]}}(u)| = 2$ and Λ becomes an orientation of $\vec{S} \cup \{Z\}$ with imbalance at least two for all the vertices of G adjacent to at least two oriented edges.

If Z is a bridge, we take c one of its endpoints adjacent to an oriented edge, we call u the other one and assign

$$\Lambda(cu) = \begin{cases} \overrightarrow{cu} & \text{if } d^+_{\Lambda_{|\vec{S}}}(c) > d^-_{\Lambda_{|\vec{S}}}(c); \\ \overrightarrow{uc} & \text{otherwise.} \end{cases}$$

Concerning the imbalance of c, by inductive hypothesis, $|d^+_{\Lambda_{\vec{S}}}(c) - d^-_{\Lambda_{\vec{S}}}(c)| \geq 1$, then

$$|d^+_\Lambda(c) - d^-_\Lambda(c)| = |d^+_{\Lambda_{|\vec{S}}}(c) - d^-_{\Lambda_{|\vec{S}}}(c)| + |d^+_{\Lambda_{|Z}}(c) - d^-_{\Lambda_{|Z}}(c)| \geq 2.$$

If u is adjacent to a block in $\vec{S}$ as well and $d^+_{\Lambda_{|\vec{S}}}(u) - d^-_{\Lambda_{|\vec{S}}}(u)$ and $d^+_{\Lambda_{|Z}}(u) - d^-_{\Lambda_{|Z}}(u)$ do not have the same sign then we switch the assignment of Λ of the edges of the whole connected component of $\vec{S}$ containing u for its opposite. This connected component necessarily does not contain c otherwise z would be contained in a bigger block of G. Then doing so, the imbalance of all vertices is left unchanged except for that of u which is now equal to

$$|d^+_{\Lambda_{|\vec{S}}}(u) - d^-_{\Lambda_{|\vec{S}}}(u)| + |d^+_{\Lambda_{|Z}}(u) - d^-_{\Lambda_{|Z}}(u)| = |d^+_{\Lambda_{|\vec{S}}}(u) - d^-_{\Lambda_{|\vec{S}}}(u)| + 1 \geq 2.$$

Λ thus becomes an orientation of $\vec{S} \cup \{Z\}$ with imbalance at least two for all the vertices of G incident with at least two oriented edges.

We now add Z to $\vec{S}$ and proceed like this for all blocks Z in $S_G \backslash \vec{S}$ until $\vec{S} = S_G$. Now since $V_B = S_G$ and $\delta_G = 2$, we conclude that Λ is an orientation of all the edges of G with an imbalance equal to 2.

In Fig. 2.13 can be found the orienting process of the cactus presented in

Fig. 2.11. First, the even cycles are oriented (Fig. 2.13(a)), then the blocks adjacent to the even cycles (Fig. 2.13(b)) and then iteratively the unoriented blocks adjacent to oriented blocks (Fig. 2.13(c), (d), (e) and (f)). In Fig. 2.13(c) and (e), the vertices with imbalance zero are circled, and in the next step, the orientation is reversed on one of the subtree where the circled vertex is the root in order to ensure an orientation imbalance of at least 2.

- $\Rightarrow$ If $\delta_G < 2$ then $\textsc{MaxIm}(G) < 2$. So we assume that $\delta_G = 2$ and $S_G \subsetneq V_B$. We denote $V_B \backslash S_G$ by $\overline{S_G}$ and we take $c \in V(\overline{S_G}) \cap V(S_G)$. Belonging to both $V(S_G)$ and $V(\overline{S_G})$, c must belong to at least two different blocks of G and is therefore a cut-vertex of G. Let Z be a block in S_G adjacent to c, Z is necessarily a bridge of G otherwise all the blocks adjacent to Z would be in S_G thus contradicting $c \in V(\overline{S_G})$, and for the same reason, Z is the only bridge in S_G adjacent to c. Calling c' the endpoint of Z that is not c and G_c the subgraph of G obtained by taking the connected component of $G \backslash cc'$ containing c, we are in the conditions of lemma 37 and thus we obtain

$$\textsc{MaxIm}(G) \leq \max_{\Lambda \in \vec{O}(G_c)} \min\Big(\min_{v \in V(G_c) \backslash \{c\}} |d^+_\Lambda(v) - d^-_\Lambda(v)|, |d^+_\Lambda(c) - d^-_\Lambda(c)| + 1 \Big).$$

Proceeding like this for all $c \in V(\overline{S_G}) \cap V(S_G)$ yields that $\textsc{MaxIm}(G)$ is at most

$$\max_{\Lambda \in \vec{O}(\overline{S_G})} \min\Big(\min_{v \in V(\overline{S_G}) \backslash V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \min_{v \in V(\overline{S_G}) \cap V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)| + 1 \Big).$$

Now if we choose a connected component $\tilde{S}$ of $\overline{S_G}$ we get that $\textsc{MaxIm}(G)$ is at most

$$\max_{\Lambda \in \vec{O}(\tilde{S})} \min\Big(\min_{v \in V(\tilde{S}) \backslash V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \min_{v \in V(\tilde{S}) \cap V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)| + 1 \Big).$$

Now we show that the previous quantity is lower than or equal to 1. Suppose it equals 2 and let $\Lambda \in \vec{O}(\tilde{S})$ satisfying

$$\min\Big(\min_{v \in V(\tilde{S}) \backslash V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \min_{v \in V(\tilde{S}) \cap V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)| + 1 \Big) \geq 2. \quad (2.5)$$

Thus we have $d_{\tilde{S}}(v) \geq 2, \ \forall v \in V(\tilde{S}) \backslash V(S_G)$ and we know that for any $v \in V(\tilde{S}) \cap V(S_G)$, all the blocks in $\tilde{S}$ adjacent to v are odd cycles, hence $d_{\tilde{S}}(v) \geq 2$. So $\delta_{\tilde{S}} \geq 2$ and according to lemma 38, there is a cycle C of $\tilde{S}$ with at most one

gate. If C has no gate then it means that $\tilde{S}$ consists only of C and since we know that all the cycles of $\tilde{S}$ are odd, it directly contradicts the existence of Λ. Hence C has exactly one gate g. So for all $v \in V(C)\backslash\{g\}$, $|d^+_\Lambda(v) - d^-_\Lambda(v)| = |d^+_{\Lambda_{|C}}(v) - d^-_{\Lambda_{|C}}(v)|$ and if we name the vertices of C $g = v_1v_2\cdots v_k = g$ the assignment of Λ must be

$$\Lambda(v_iv_{i+1}) = \begin{cases} \overrightarrow{v_iv_{i+1}} & \text{if } i \text{ is odd;} \\ \overrightarrow{v_{i+1}v_i} & \text{otherwise} \end{cases}, \ \forall i \in [\![1, k-1]\!];$$

or its reverse so that Λ satisfies our assumption. Thus we have

$$\begin{aligned} &\max_{\Lambda\in\vec{\mathcal{O}}(\tilde{S})} \min\Big(\min_{v\in V(\tilde{S})\backslash V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \min_{v\in V(\tilde{S})\cap V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)| + 1\Big) \\ = &\max_{\Lambda\in\vec{\mathcal{O}}(\tilde{S}\backslash C)} \min\Big(\min_{v\in V(\tilde{S}\backslash C)\backslash V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \min_{v\in V(\tilde{S}\backslash C)\cap V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)| + 1\Big). \end{aligned}$$

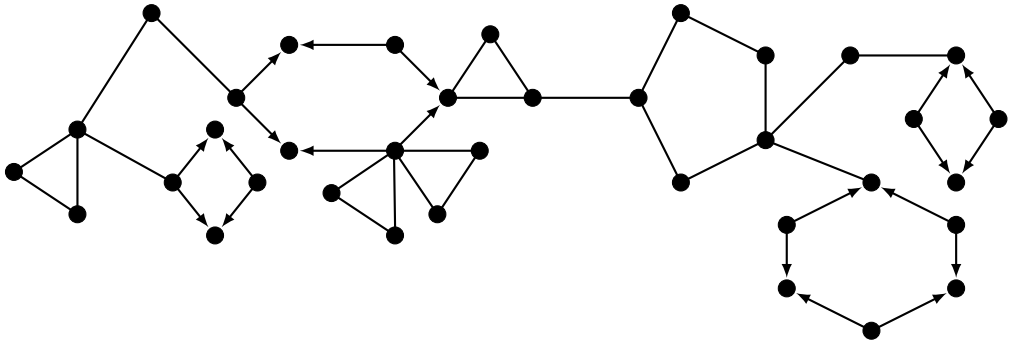
If there exists a vertex of degree one in $\tilde{S}\backslash C$ then it is adjacent to at bridge in $\tilde{S}$ and is therefore in $V(\overline{S_G}\backslash C)\backslash V(S_G)$, thus contradicting (2.5). So we assume that $\delta_{\tilde{S}\backslash C} = 2$ and we can reiterate the same process with another odd cycle with exactly one gate until we are left with an odd cycle C_{end} with no gates and conclude that

$$\max_{\Lambda\in\vec{\mathcal{O}}(C_{end})} \min\Big(\min_{v\in V(C_{end})\backslash V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)|, \min_{v\in V(C_{end})\cap V(S_G)} |d^+_\Lambda(v) - d^-_\Lambda(v)| + 1\Big) \geq 2;$$

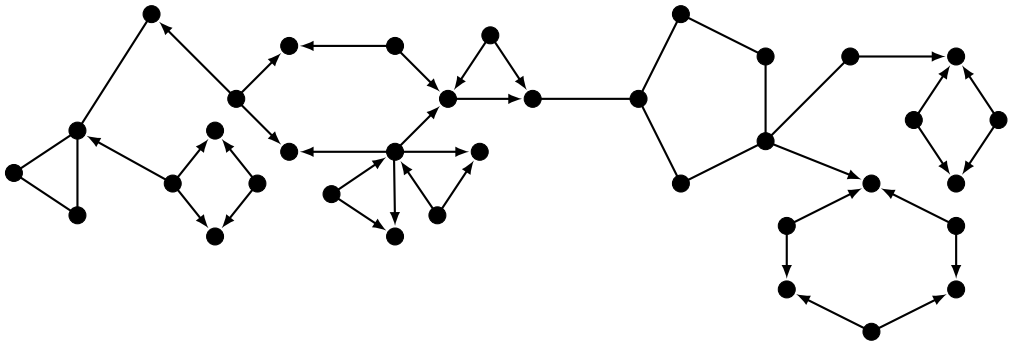
Since this is impossible for an odd cycle, we deduce that (2.5) does not hold. Hence, $\textsc{MaxIm}(G) \leq 1$.

□

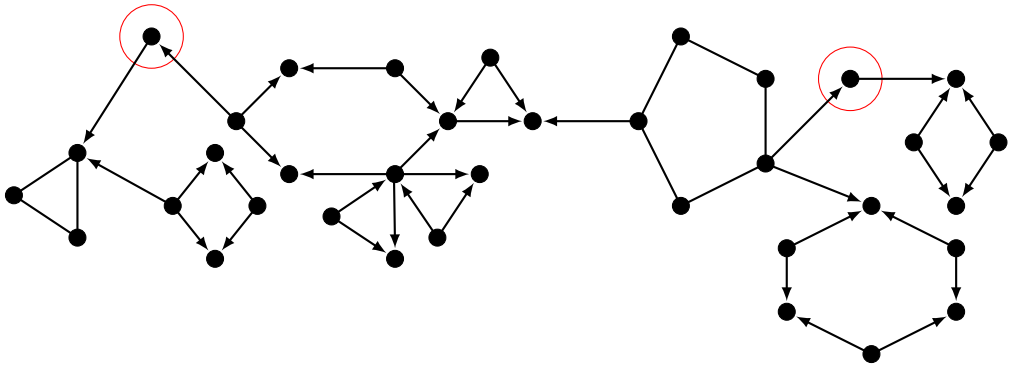
Figure 2.13: Orienting the edges of G verifying $\delta_G = 2$ and $S_G = V_B$ in such a way that $\textsc{MaxIm}(G) = 2$.



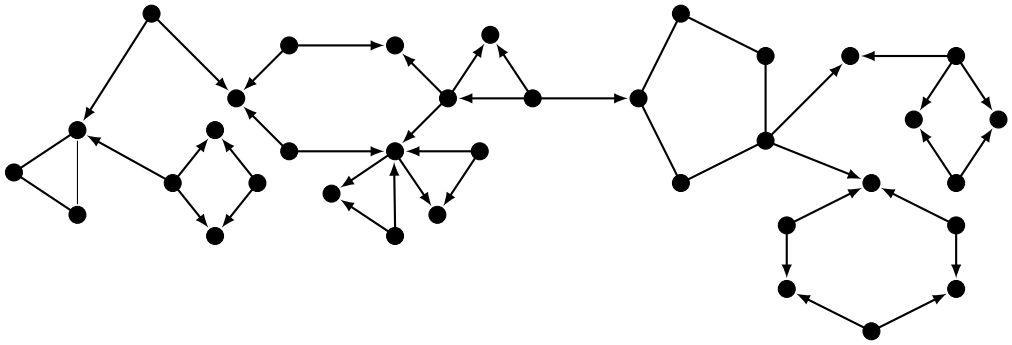
(a)



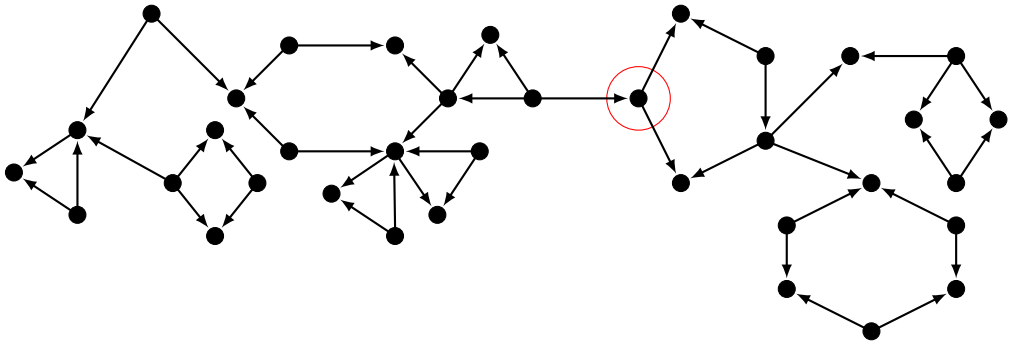
(b)



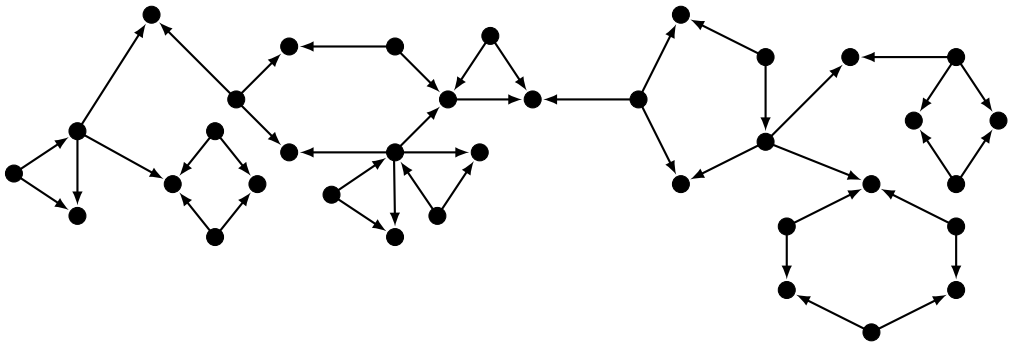
(c)



(d)



(e)



(f)

2.3.3 Exact polynomial-time algorithm for cacti

The polynomial-time characterization of the cacti such that $\textsc{MaxIm}(G) = 0$ in Section 2.2 put together with the polynomial-time characterization of the cacti such that $\textsc{MaxIm}(G) = 2$ entail a polynomial-time algorithm solving MaxIm for cacti. An optimal orientation can easily be derived since either $\textsc{MaxIm}(G) = 0$, in which case any orientation is optimal, or $\textsc{MaxIm}(G) > 0$ in which case the characterization allowing to decide between the values 1 and 2 entails an optimal orientation the value of which permits to answer the problem.

Asahiro et al. gave polynomial-time algorithm for cacti w.r.t. MaxMinOut in [6]. More about cacti can be found in [34, 58]. The problem MaxIm however remains generally NP-complete as shown in Section 2.1. In the same Section, we have shown that it cannot be approximated in polynomial tim within more than $\frac{1}{2}$. The optimal value may nonetheless be computed using non-polynomial mixed integer linear optimization.

2.4 Mixed integer linear programming formulations

Even if the complexity of mixed integer programming is exponential, it remains important when designing a MIP to limit the number of integer variables and to ensure the quality of its linear relaxation in order for the objective values of fractional solutions to be as close as can be from the optimal objective value.

In this section, we gradually introduce two formulations for the MaxIm problem. In order to describe an orientation of the graph G, we take the incidence matrix of the graph B and orientation variables $x \in \{-1, 1\}^{|E|}$ as introduced in Section 1.5 of the Introduction. And we obtain from the given generic formumation from the same Section the following for MaxIm.

$$\max_{x \in \{-1,1\}^{|E|}} \min_{v \in V} |B_v x|.$$

This expression of MaxIm is non-linear. We need to modify its non-linear elements in order to obtain a computable mixed integer linear program.

2.4.1 A first MIP

First we address the minimum in the objective function and introduce a variable h that will assume the role of this minimum.

$$\begin{cases} \max h \\ \text{s.t.} \\ h \leq |B_v x|, \ \forall v \in V \\ x \in \{-1, 1\}^{|E|}. \end{cases}$$

Now in order to remove the absolute values of the constraint, we square it : $h^2 \leq (B_v x)^2, \ \forall v \in V$. The variables x have only two possible values (-1 and 1), they are thus called *binary variables*. Regarding the computability of a formulation, it makes more sense for a binary variable to be 0/1-valued. Therefore we change the variables in the following way: $x \to 2x - 1$. Developing our constraint, we obtain:

$$\begin{aligned} & (B_v(2x - \mathbf{1}))^2 \\ =& \left(\sum_{uv \in E} B_{v,uv}(2x_{uv} - 1) \right)^2 \\ =& \sum_{uv \in E} B_{v,uv}^2 (2x_{uv} - 1)^2 + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv} B_{v,wv} (2x_{uv} - 1)(2x_{wv} - 1) \\ =& \sum_{uv \in E} (4x_{uv}^2 - 4x_{uv} + 1) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv} B_{v,wv} (4x_{uv}x_{wv} - 2x_{uv} - 2x_{wv} + 1) \\ =& d(v) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv} B_{v,wv} (4x_{uv}x_{wv} - 2x_{uv} - 2x_{wv} + 1). \end{aligned}$$

Since it still is not linear, we have to consider variables representing the product of two variables and for that we substitute the x variables by their 0-1 version. So developing, we obtain Regarding h, maximizing h is equivalent to maximizing its square root. Hence, substituting h^2 by h, we then get the following formulation.

$$\begin{cases} \max h \\ \text{s.t.} \\ h \leq d(v) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv} B_{v,wv} (4x_{uv}x_{wv} - 2x_{uv} - 2x_{wv} + 1), \ \forall v \in V \\ x \in \{0, 1\}^{|E|}. \end{cases}$$

In order to linearize it, we introduce product variables $z_{uv,wp} \in \{0, 1\}, \ uv, wp \in E, \ uv \neq wp$ representing the variables product $x_{uv}x_{wp}$. For a pair of edges $(uv, wp) \in$

E^2, $uv \neq wp$, we add the constraints $z_{uv,wp} \leq x_{uv}$, $z_{uv,wp} \leq x_{wp}$ and $z_{uv,wp} \geq x_{uv} + x_{wp} - 1$ so as to force $z_{uv,wp} = x_{uv}x_{wp}$ for any integer solution. With these additional constraints, we can relax the integer constraint on the z variables, its values will still be integers for any solution where the values of x are integers. We obtain the following mixed integer linear programming formulation.

$$(\mathsf{MIP1}) \begin{cases} \max h \\ \text{s.t.} \\ h \leq d(v) + 2\sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv}B_{v,wv}(4z_{uv,wv} - 2x_{uv} - 2x_{wv} + 1), \ \forall v \in V \\ z_{uv,wp} \leq x_{uv} \\ z_{uv,wp} \leq x_{wp} \qquad\qquad , \ \forall uv, wp \in E, \ uv \neq wp \\ z_{uv,wp} \geq x_{uv} + x_{wp} - 1 \\ x \in \{0,1\}^{|E|}, \ z_{uv,wp} \geq 0, \ \forall uv, wp \in E, \ uv \neq wp, \ h \in \mathbb{R}. \end{cases}$$

The following result asserts that solving the mixed integer linear program (MIP1) is equivalent to solving MaxIm for any graph G.

Theorem 39. *For any graph G,*

$$\mathsf{MIP1}(G) = \textsc{MaxIm}(G),$$

Where $\mathsf{MIP1}(G)$ *is the square root of the optimal objective value of (*MIP1*) for G.*

Proof. $x \in \{0,1\}^{|E|}$ covers all the possible orientations of G and for every vertex $v \in V$, the first constraint is equivalent to $h \leq (d_x^+(v) - d_x^-(v))^2$. □

Thus we have a first mixed integer linear programming formulation for our problems with $O(m^2)$ variables, $O(m)$ of which are integer variables and $O(m^2)$ constraints.

Let us take a look at the linear program obtained by relaxing the integer constraint of (MIP1) (i.e. the *linear relaxation* of MIP1) on an input graph $G = (V, E)$, and let us consider the triplet (x^{lp}, z^{lp}, h^{lp}) where

- ▷ $x^{lp}_{uv} = \frac{1}{2}, \ \forall uv \in E$;
- ▷ $z^{lp}_{uv,wp} = 0$ for all pairs of edges $uv, wp \in E$ that share no endpoint;
- ▷ $z^{lp}_{uv,wv} = \frac{1+B_{v,uv}B_{v,wv}}{4}$ for all pairs of edges $uv, wv \in E$ (i.e. all pairs of edges in E that share an endpoint);
- ▷ $h^{lp} = \delta^2_G$.

Lemma 40. *(x^{lp}, z^{lp}, h^{lp}) is a feasible solution of the linear relaxation of (MIP1) with objective value δ_G^2.*

Proof. Observe that $\forall uv, wp \in E$,

$$0 = x_{uv}^{lp} + x_{wp}^{lp} - 1 \leq z_{uv,wp}^{lp} \leq x_{uv}^{lp} = x_{wv}^{lp} = \frac{1}{2}.$$

Moreover, $\forall v \in V$,

$$\begin{aligned}
&d(v) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv} B_{v,wv} (4 z_{uv,wv}^{lp} - 2 x_{uv}^{lp} - 2 x_{wv}^{lp} + 1) \\
=&d(v) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} B_{v,uv} B_{v,wv} (1 + B_{v,uv} B_{v,wv} - 1 - 1 + 1) \\
=&d(v) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} (B_{v,uv} B_{v,wv})^2 \\
=&d(v) + 2 \sum_{\substack{uv,wv \in E \\ uv \neq wv}} 1 \\
=&d(v) + (d(v) - 1) d(v) \\
=&d(v)^2 \geq \delta_G^2.
\end{aligned}$$

□

The previous lemma proves that the optimal value of the linear relaxation of (MIP1) is at least the minimum degree of the input graph squared, which corresponds to the trivial upper bound of MaxIm(G). This implies that the value of an optimal solution for the linear relaxation of MIP1 can be quite distant from its optimal objective value, which can significantly impair the general performance of the formulation.

2.4.2 A more elaborated MIP

We present a second formulation with reduced number of variables and constraints. This second formulation involves the same orientation variables $x \in \{-1, 1\}^{|E|}$ and a second type of variables that are binary: indicator variables y_k^v with $v \in V$ a vertex of G and $k \in [\![-d(v), d(v)]\!]$. They have the following interpretation: $y_k^v = 1$ if and only if $B_v x = d_x^+(v) - d_x^-(v) = k$, so that the following equation trivially holds

$$\sum\nolimits_{k \in [\![-d(v), d(v)]\!]} k y_k^v = B_v x, \forall v \in V.$$

Given the interpretation for the variables y, among those of the form y_k^v, for some fixed node $v \in V$, exactly one of them has value 1. Thus, the following constraints are satisfied

$$\sum\nolimits_{k \in [\![-d(v), d(v)]\!]} y_k^v = 1, \ \forall v \in V.$$

Notice that for a vertex $v \in V$, the difference between its oudegree and indegree w.r.t. any orientation and its degree have the same parity. Thus instead of running k through $[\![-d(v), d(v)]\!]$, we can limit to $k \in [\![-d(v), d(v)]\!]$, s.t. $k \equiv d(v) \pmod 2$, i.e. the only possible values of $d_x^+(v) - d_x^-(v)$ when x runs through $\{-1, 1\}^{|E|}$. Then we can show the MaxIm problem may be formulated as the mixed integer program

$$(\mathsf{MIP2}) \begin{cases} \max h \\ \text{s.t.} \\ h \leq \sum\limits_{\substack{k \in [\![-d(v), d(v)]\!] \\ k \equiv d(v) \pmod 2}} \min(|k|, \delta_G) y_k^v, \ \forall v \in V \\ \sum\limits_{\substack{k \in [\![-d(v), d(v)]\!] \\ k \equiv d(v) \pmod 2}} y_k^v = 1, \ \forall v \in V \\ \sum\limits_{\substack{k \in [\![-d(v), d(v)]\!] \\ k \equiv d(v) \pmod 2}} k y_k^v = B_v x, \ \forall v \in V \\ x \in [-1; 1]^{|E|}, y_k^v \in \{0, 1\}, \forall (v, k) \in V \times [\![-d(v), d(v)]\!], \text{ s.t. } k \equiv d(v) \pmod 2, h \in \mathbb{R}. \end{cases}$$

The following result asserts that solving the mixed integer linear program (MIP2) is equivalent to solving MaxIm for any graph G.

Theorem 41. *For any graph G,*

$$\mathsf{MIP2}(G) = \textsc{MaxIm}(G),$$

Where $\mathsf{MIP2}(G)$ *is the optimal objective value of (*MIP*2) for G.*

Proof. First, we consider the stronger formulation where the orientation variables x are constrained to take value in $\{-1, 1\}$. Since $y_k^v = 1$ if and only if $B_v x = d_x^+(v) - d_x^-(v) = k$ and $x \in \{-1, 1\}^{|E|}$ covers all the possible orientations of G, for every vertex $v \in V$, the first and third constraints lead to $h \leq |d_x^+(v) - d_x^-(v)|$ and the optimal objective value of the resulting formulation would equal $\textsc{MaxIm}(G)$.

Now we know that the incidence matrix B is totally unimodular. Hence, there exists an integer optimal solution $(x^\star, y^\star, h^\star)$ of (MIP2). If $x^\star \in \{-1, 1\}^{|E|}$ then $(x^\star, y^\star, h^\star)$ is solution of the stronger version of (MIP2) mentioned above and therefore optimal, i.e. its objective value is $\textsc{MaxIm}(G)$. Otherwise $x^\star$ describes a partial orientation of G, i.e. for an edge $uv \in E$, if $x_{uv}^\star = 1$ then the original orientation of the edge is preserved, if $x_{uv}^\star = -1$ then it is reversed and if $x_{uv}^\star = 0$ then the edge is left unoriented. We know that for each vertex $v \in V$, $B_v x^\star \equiv d(v) \pmod 2$. So the

number of edges adjacent to v on which $x^\star$ is non-zero must have the same parity as the degree of v. Subsequently, the number of edges adjacent to v on which $x^\star$ is zero must be even. In other words, let $E' = \{uv \in E | x^\star_{uv} = 0\}$ and $G' = (V, E')$, $d_{G'}(v) \equiv 0 \pmod 2$ for each vertex $v \in V$. Since G' is Eulerian, we can take a cycle of G', orient it in a way that does not change the imbalance of any vertex, remove it and proceed like this until there are no more edges in G'. The resulting (complete) orientation can be described by an orientation vector $x' \in \{-1, 1\}^{|E|}$ and the triplet $(x', y^\star, h^\star)$ is a solution of the all-integer version of (MIP2) having the same objective value as $(x^\star, y^\star, h^\star)$, hence it is optimal and therefore equal to MaxIm(G). □

We now have a second mixed integer linear programming formulation of MaxIm with $O(m+n)$ variables, $O(m)$ of which are integer variable and $O(n)$ constraints. These asymptotic sizes are significantly smaller than those of the (MIP1), which can be significant for the performance of the formulation. It means that for an instance of the same size, the solver will have a much smaller volume of memory to process and much less variables on which to branch and/or bound. Let us now interest ourselves with the linear relaxation of (MIP2).

Let us consider the triplet (x^{lp}, y^{lp}, h^{lp}) where

- ▷ $x^{lp}_{uv} = 0, \ \forall uv \in E$;
- ▷ $y^{v,lp}_k = 0, \ \forall (v,k) \in V \times [\![-d(v)+1, d(v)-1]\!]$, s.t. $k \equiv d(v) \pmod 2$;
- ▷ $y^{v,lp}_{-d(v)} = y^{v,lp}_{d(v)} = \frac{1}{2}, \ \forall v \in V$.

Observe that (x^{lp}, y^{lp}, h^{lp}) is a feasible solution of the linear relaxation of (MIP2) with objective value δ_G. In other words, the linear relaxation (MIP2) is generally weak (possibly as weak as (MIP1)). Let us then try to find some valid inequalities in order to strengthen this linear relaxation.

2.5 Strengthening (MIP2)

There are several possible way to improve the general performance of our mixed integer linear program. One simple way is to try and reduce its size by eliminating redundant variables.

Remember from Section 2.2 that it is easy to check whether MaxIm$(G) = 0$. One can then set variables y^v_0 to 0 when MaxIm$(G) > 0$. We also know from the discussion at the end of Section 2.1 that MaxIm(G) can not be less than $\lceil \frac{\delta_G}{2} \rceil - 1$. More precisely, if $\delta_G \equiv 0 \pmod 4$ then MaxIm$(G) \geq \frac{\delta_G}{2}$ while MaxIm$(G) \geq \frac{\delta_G - 1}{2}$ when δ_G is odd and MaxIm$(G) \geq \frac{\delta_G}{2} - 1$ when $\delta_G \equiv 2 \pmod 4$.

More generally, if l is a known lower bound for MaxIm(G), then all variables y_k^v for $|k| < l$ can be fixed to 0, i.e. deleted from the formulation. To find such a lower bound, we use a standard greedy algorithm to find a locally maximum cut and orient edges as described in the proof of Theorem 25.

Another way to improve the performance of (MIP2) is to add constraints to its linear relaxation in order for it to be stronger, in other words, for the gap between the optimal objective value of (MIP2) and that of its relaxation to be smaller. These constraints may be extracted from valid families of inequalities for the set of solutions of ($MIP2$) by the mean of a cutting plane method, particularly in the case of families which inherent separation problem can be solved in polynomial time.

2.5.1 A family of valid inequalities obtained from a polyhedral study

Let u be an upper bound for MaxIm(G). We already know that MaxIm(G) $\leq \delta_G$, so we can assume that $u \leq \delta_G$. Consider the following inequality

$$h \leq u - \sum_{v=1}^{n} \sum_{\substack{k \in [\![0,u-1]\!] \\ k \equiv d(v) \pmod 2}} \lambda_k^v (y_k^v + y_{-k}^v), \ \forall \lambda \in \Lambda_u \tag{2.6}$$

where the vertices of G are numbered from 1 to $|V| = n$, and

$$\Lambda_u = \left\{ \lambda = (\lambda_k^v)_{(v,k) \in [\![1,n]\!] \times [\![0,u-1]\!]} \in \mathbb{N}^{nu} \;\middle|\; \begin{array}{l} \lambda_{k+1}^v \leq \lambda_k^v, \ \forall (v,k) \in [\![1,n]\!] \times [\![0,u-2]\!] \\ \sum_{v=1}^n \lambda_k^v = u-k, \ \forall k \in [\![0,u-1]\!] \end{array} \right\}.$$

Observe that coefficients λ_k^v are non-negative integer numbers and are non increasing in k. For each k, there exists only one v such that $\lambda_{k+1}^v = \lambda_k^v - 1$ while $\lambda_{k+1}^w = \lambda_k^w$ for any $w \neq v$.

The family of inequalities (2.6) derive from the study of the polyhedron consisting in the convex hull of an assignment matrix appended with the smallest assigned value. The study of this family of polyhedra detailed in Chapter 4 originates in the research of valid inequalities intended to strengthen formulation (MIP2). Let us prove that these inequalities are valid inequalities for the convex hull of feasible solutions of (MIP2) denoted by $\mathcal{P}_2$.

Proposition 42. *Inequalities (2.6) are valid for $\mathcal{P}_2$.*

Proof. Consider a feasible solution (x, y, h) of $\mathcal{P}_2$. If $y_k^v = 0$ and $y_{-k}^v = 0$ for any $v \in [\![1,n]\!]$ and any $k \in [\![0,u-1]\!]$, then $\sum_{v=1}^n \sum_{\substack{k \in [\![0,u-1]\!] \\ k \equiv d(v) \pmod 2}} \lambda_k^v (y_k^v + y_{-k}^v) = 0$ and inequality (2.6) becomes $h \leq u$. The last is valid since u is an upper bound for

MaxIm(G).

Let us now assume that $h \leq u-1$, then there exist $v \in [\![1,n]\!]$ and $k \leq u-1$ such that either y_k^v or y_{-k}^v is equal to 1 while $y_{k'}^w = 0$ for any $w \in [\![1,n]\!]$ and k' such that $|k'| < k$.

Using the fact that y_k^w is non increasing, we deduce that

$$\sum_{\substack{k' \in [\![0,u-1]\!] \\ k' \equiv d(w) \pmod 2}} \lambda_{k'}^w (y_{k'}^w + y_{-k'}^w) \leq \lambda_k^w \sum_{\substack{k' \in [\![k,u-1]\!] \\ k' \equiv d(w) \pmod 2}} (y_{k'}^w + y_{-k'}^w) \leq \lambda_k^w.$$

Summing up these inequalities for all w and using the fact that $\sum_{v=1}^n \lambda_k^v = u-k$ leads to $h \leq k$ which is valid by the definition of k. □

Observe that the first family of inequalities included in (MIP2) can be seen as a special case of inequalities (2.6).

Let us now study the separation problem of inequalities (2.6).

Proposition 43. *Inequalities (2.6) can be separated in polynomial time.*

Proof. Given a fractional solution (x, y, h), one can check whether an inequality of type (2.6) is violated by looking for coefficients $\lambda \in \Lambda_u$ maximizing $\sum_{v=1}^n \sum_{\substack{k \in [\![0,u-1]\!] \\ k \equiv d(v) \pmod 2}} \lambda_k^v (y_k^v + y_{-k}^v)$. Remember that for each k, there exists only one v such that $\lambda_{k+1}^v = \lambda_k^v - 1$ while $\lambda_{k+1}^w = \lambda_k^w$ for any $w \neq v$. Let v_k be such v. Then $\sum_{v=1}^n \sum_{\substack{k \in [\![0,u-1]\!] \\ k \equiv d(v) \pmod 2}} \lambda_k^v (y_k^v + y_{-k}^v)$ can be written as $\sum_{k \in [\![0,u-1]\!]} \sum_{k' \in [\![0,k]\!]} (y_{k'}^{v_k} + y_{-k'}^{v_k})$. This immediately leads to the following algorithm. First, all λ_k^v are initially set to 0. Then, we select v_{u-1} maximizing $\sum_{\substack{k \in [\![0,u-1]\!] \\ k \equiv d(v) \pmod 2}} (y_k^v + y_{-k}^v)$ and we increment by 1 all $\lambda_k^{v_{u-1}}$: $\lambda_k^{v_{u-1}} = \lambda_k^{v_{u-1}} + 1$ for any $k \leq u-1$. More generally, for each $w \in [\![0, u-1]\!]$, we select v_w maximizing $\sum_{\substack{k \in [\![0,w]\!] \\ k \equiv d(v) \pmod 2}} (y_k^v + y_{-k}^v)$ and we increment by 1 all $\lambda_k^{v_w}$ for $k \leq w$. The algorithm has clearly a polynomial-time complexity. □

2.5.2 Valid inequalities extracted from the orientation of the edges incident to one vertex

A second family of valid inequalities is defined for each vertex v, each integer number $p \in [\![1, d(v)]\!]$ and each subset of p edges incident to v.

$$\sum_{\{vu_1, \cdots, vu_p\} \subseteq \delta(\{v\})} B_{v,vu_i} x_{vu_i} + \sum_{k \in [\![0,p-1]\!]} 2(p-k) y_{2k-d(v)}^v \leq p. \tag{2.7}$$

These inequalities come from the study of the behavior of the indicator variables corresponding to a vertex and the orientation variables of the edges incident to it when varying the value of its imbalance.

Proposition 44. *Inequalities (2.7) are valid for $\mathcal{P}_2$.*

Proof. If all variables $y^v_{2k-d(v)}$ are equal to 0 for $k \in [\![0, p-1]\!]$, then inequality (2.7) is implied by the fact that $x \in [-1;1]^{|E|}$.

Assume that $y^v_{2k_0-d(v)} = 1$ for some $k_0 \in [\![0, p-1]\!]$. This requires that among all edges of $\delta(\{v\})$ there are exactly k_0 (resp. $d(v)-k_0$) edges vu such that $B_{v,vu}x_{vu} = 1$ (resp. $B_{v,vu}x_{vu} = -1$). Consequently $\sum_{\{vu_1,\cdots,vu_p\}\subseteq\delta(\{v\})} B_{v,vu_i}x_{vu_i}$ is upper bounded by $k_0-(p-k_0) = 2k_0-p$. Moreover, we have $\sum_{k\in[\![0,p-1]\!]} 2(p-k)y^v_{2k-d(v)} = 2(p-k_0)$. Inequality (2.7) immediately follows. □

The separation problem of inequalities (2.7) is also easy to solve.

Proposition 45. *Inequalities (2.7) can be separated in polynomial time.*

Proof. Given a fractional solution (x, y, h), for each vertex v and any $p \in [\![1, d(v)]\!]$, we order the edges of $\delta(\{v\})$ in descending order according to $B_{v,vu}x_{vu}$ and we select the first p edges. Then a violated inequality can be detected by comparison of $\sum_{\{vu_1,\cdots,vu_p\}\subseteq\delta(\{v\})} B_{v,vu_i}x_{vu_i} + \sum_{k\in[\![0,p-1]\!]} 2(p-k)y^v_{2k-d(v)}$ with p. □

2.5.3 Valid inequalities extracted from cycle orientation

In a cycle, the number of vertices which imbalance equals their degree cannot exceed half the size of the cycle. Similarly, the number of vertices which imbalance is larger than their degree minus 2 cannot exceed the size of the cycle. Using these considerations, we can design valid inequalities for the set of solutions of (MIP2). Let us now consider any cycle C of the input graph and the two following inequalities.

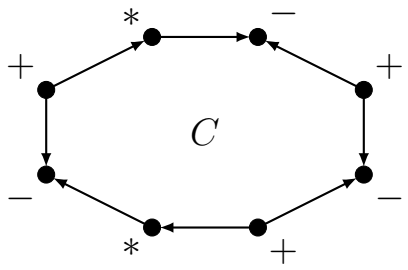
$$\sum_{v\in V(C)} (2y^v_{d(v)} + y^v_{d(v)-2}) \leq |V(C)|, \tag{2.8}$$

$$\sum_{v\in V(C)} (2y^v_{-d(v)} + y^v_{-d(v)+2}) \leq |V(C)|. \tag{2.9}$$

Proposition 46. *Inequalities (2.8) and (2.9) are valid for $\mathcal{P}_2$.*

Proof. Let us focus on the validity of inequalities (2.8) (inequalities (2.9) being provable in the same way). After orientation of the edges of the cycle C, let C^+ be the set of vertices of C for which their two incident edges are oriented outward from these vertices to their neighbours. We also define C^- in the same way by

Figure 2.14: A vertex of a cycle C is respectively in C^+, C^- or C^* if it has two, zero or one outgoing incident edge(s) in C.



considering vertices having two edges of the cycle oriented from their neighbours to them. The remaining set of vertices of the cycle is denoted by C^* (they have an incoming incident edge and an outgoing incident edge). These definitions are illustrated on Fig. 2.14. It is now easy to see that we always have $|C^+| = |C^-|$. Suppose that $y^v_{d(v)} = 1$, then $v \in C^+$. Observe also that $y^v_{d(v)-2} = 1$ requires v to be in $C^+ \cup C^*$. Consequently,

$$\sum_{v \in V(C)} (y^v_{d(v)} + y^v_{d(v)-2}) \leq |C^+| + |C^*|$$

and

$$\sum_{v \in V(C)} y^v_{d(v)} \leq |C^+|.$$

Summing up both inequalities leads to

$$\sum_{v \in V(C)} (2y^v_{d(v)} + y^v_{d(v)-2}) \leq 2|C^+| + |C^*| = |C^+| + |C^-| + |C^*| = |V(C)|.$$

□

Proposition 47. *Inequalities (2.8) and (2.9) can be separated in polynomial time.*

Proof. Let us again focus on inequalities (2.8) since (2.9) can be separated in a similar way. Given a fractional solution (x, y, h), we need an algorithm to either compute a cycle C such that $|V(C)| - \sum_{v \in V(C)} (2y^v_{d(v)} + y^v_{d(v)-2}) < 0$ or to certify that such a cycle does not exist. For each edge uv, let us consider a weight

$$c_{uv} = 1 - (y^v_{d(v)} + \frac{1}{2} y^v_{d(v)-2}) - (y^u_{d(u)} + \frac{1}{2} y^u_{d(u)-2}).$$

Then we clearly have

$$|V(C)| - \sum_{v \in V(C)} (2y^v_{d(v)} + y^v_{d(v)-2}) = \sum_{\{uv\} \in E(C)} c_{uv}.$$

We are then looking for an undirected cycle having a negative total weight. This can be done, for example, by computing a minimum weight $\emptyset$-join, i.e. a set of edge disjoint cycles. This can be done using the algorithm of [36]. Notice that this algorithm can provide many negative weight cycles with some 0 weight cycles. If the total weight of the $\emptyset$-join is strictly negative, a negative weight cycle can be easily extracted. A different algorithm can also be found in [17]. □

2.5.4 Valid inequalities extracted from clique orientation

Given any clique $\mathcal{K}$ and any number $p \in [\![1, |\mathcal{K}|]\!]$, we consider the two following inequalities.

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1, d(v))} (p-k) y^v_{d(v)-2k} \leq \frac{p(p+1)}{2}, \ \forall p \in [\![1, |\mathcal{K}|]\!], \tag{2.10}$$

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1, d(v))} (p-k) y^v_{2k-d(v)} \leq \frac{p(p+1)}{2}, \ \forall p \in [\![1, |\mathcal{K}|]\!]. \tag{2.11}$$

These inequalities come from the study of the behavior of the indicator variables corresponding to the vertices of a clique when varying the orientation of the edges of its edges. This study is based on considerations similar to the ones used to derive inequalities (2.8) and (2.9) concerning the maximum possible number of vertices which imbalance equal their degree in a clique and so on. Inequalities (2.10) can be seen as a generalization of the obvious inequalities $\sum_{v \in V(\mathcal{K})} y^v_{d(v)} \leq 1$ obtained when $p = 1$.

Proposition 48. *Inequalities (2.10) and (2.11) are valid for $\mathcal{P}_2$.*

Proof. We can focus on inequalities (2.10). To maximize the left hand side of (2.10), we can assume that all edges in $\delta(V(\mathcal{K}))$ are oriented from $V(\mathcal{K})$ to $V \backslash V(\mathcal{K})$ (this is due to the fact that the coefficient $p - k$ increases when k decreases). Then, $y^v_{d(v)-2k} = 1$ is equivalent to say that exactly k edges, inside $\mathcal{K}$ and incident to v, are oriented from v.
Observe that inequality (2.10) is equivalent to

$$p \sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1, d(v))} y^v_{d(v)-2k} \leq \sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1, d(v))} k y^v_{d(v)-2k} + \frac{p(p+1)}{2}. \tag{2.12}$$

Let q be the number of vertices of $\mathcal{K}$ whose outdegree (in the graph induced by $\mathcal{K}$) is at most $p-1$ after orientation. The left hand side of (2.12) is given by pq. The sum of the q lowest outdegrees is exactly given by

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1,d(v))} k y^v_{d(v)-2k}.$$

Using Landau's theorem for tournaments [73], we have

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1,d(v))} k y^v_{d(v)-2k} \geq \frac{q(q-1)}{2}.$$

Adding $\frac{p(p+1)}{2}$ to both sides, we get

$$\sum_{v \in V(\mathcal{K})} \sum_{k=0}^{\min(p-1,d(v))} k y^v_{d(v)-2k} + \frac{p(p+1)}{2} \geq \frac{p^2+q^2+p-q}{2}.$$

Moreover, from $(q-p)^2 \geq q-p$, we get that $\frac{p^2+q^2+p-q}{2} \geq pq$ proving (2.12). □

To separate inequalities (2.10), the following heuristic is used. Given a fractional solution (x, y, h), for each vertex v and any $p \in [\![1, d(v)+1]\!]$, we use a greedy approach to find a locally maximum-weight clique where the weight of any vertex $u \in V$ is $\sum_{k=0}^{\min(p-1,d(u))}(p-k)y^u_{d(u)-2k}$. We start with $V(\mathcal{K}) := \{v\}$ and then recursively find $u = \arg\max\{\sum_{k=0}^{\min(p-1,d(w))}(p-k)y^w_{d(w)-2k} | w \in \cap_{v' \in \mathcal{K}} N_G(v')\}$, where $N_G(v)$ denotes the set of the neighbours of v in G, and add it to $V(\mathcal{K})$ until $\cap_{v' \in V(\mathcal{K})} N_G(v') = \emptyset$. Then if $|V(\mathcal{K})| \geq p$ we can derive the inequality (2.10) corresponding to $\mathcal{K}$.

2.6 Computational results

In order to assess the performance of formulations (MIP1) and (MIP2), we present some computational results related to a wide variety of graphs. All algorithms were written in C++ calling IBM's ILOG CPLEX optimizer© and experiments have been performed using a processor 1.9GHzx4, 15.6GB RAM with four cores.

CPLEX solves integer programming problems, very large linear optimization problems using either primal or dual variants of the simplex methods or the barrier method.

2.6.1 Implementation scheme

While the implementation of (MIP1) is pretty straightforward: the model is created with all inequalities described in Section 2.4.1 and then the mixed integer linear programming solver of CPLEX is run with default parameters, the implementation of (MIP2) needs to be detailed.

First of all, an initial solution is determined by computing a locally maximum cut using the standard greedy algorithm described in the proof of Theorem 25. A starting integer solution is then known for both formulations (MIP1) and (MIP2).

One can observe that the solutions for the linear relaxations of (MIP1) and (MIP2) respectively given in Lemma 40 and at the end of Section 2.4 abuse the symmetry of the formulations, thus having a high objective value (equal to the minimum degree). In order to break this symmetry, a constraint on the sign of the imbalance of one arbitrarily chosen maximum degree vertex is added according to its sign in the initial integer solution mentioned above. Practically, this means that for example for (MIP2), let $v_0 \in V$ such that $d(v_0) = \Delta_G$ and (x_0, y_0) the starting integer solution. Then if

$$\sum_{\substack{k \in [\![0, d(v_0)]\!] \\ k \equiv d(v_0) \pmod 2}} {y_0}_k^{v_0} = 1,$$

then we add the constraints

$$y_k^{v_0} = 0, \ \forall k \in [\![-d(v_0), 0]\!] k \equiv d(v_0) \pmod 2$$

to the formulation, thus forcing the signed imbalance of v_0 to be positive on every solution. On the other hand, if

$$\sum_{\substack{k \in [\![0, d(v_0)]\!] \\ k \equiv d(v_0) \pmod 2}} {y_0}_k^{v_0} = 0,$$

then we add the constraints

$$y_k^{v_0} = 0, \ \forall k \in [\![0, d(v_0)]\!] k \equiv d(v_0) \pmod 2$$

to the formulation, thus forcing the signed imbalance of v_0 to be negative on every solution.

For both formulations and each instance, we set a limited total processing time of 15 minutes (900 seconds). If this limit is reached, then the process is interrupted and returns both the objective value of the current best integer solution L_{MIP1} or L_{MIP2} and the current best upper bound U_{MIP1} or U_{MIP2}. Those two values are equal if the instance is solved to optimality, i.e. under 900s.

For formulation (MIP1), the time spent in the solver is denoted by (MIP1), if it reached 900s and was therefore interrupted, then it will show "$_{>}$900".

For formulation (MIP2), a cutting-plane algorithm is implemented based on the inequalities of Section 2.5. Inequalities are generated in the following order:

- We look for a violated inequality of the type (2.6) according to the proof of Proposition 43.
- We generate cliques using the heuristic described in the end of section 2.5 and check for a violated inequality of the types (2.10) and (2.11).
- We search for a cycle with minimum weight with a simple flow formulation solved as mixed integer program where each edge $vw \in E$ has weight $\frac{1}{2}(2y^v_{d(v)} + y^v_{d(v)-2} + y^w_{d(w)} + y^w_{d(w)-2})$. If we find a negative weighted cycle, then corresponding inequality of the type (2.8) is violated. We do the same with the weight $\frac{1}{2}(2y^v_{-d(v)} + y^v_{2-d(v)} + y^w_{-d(w)} + y^w_{2-d(w)})$ for an edge vw to find a violated inequality of the type (2.9). For the sake of simplicity, inequalities (2.8) and (2.9) are not separated using the algorithms of [36] and [14] but using a simple integer linear program computing a minimum-weight cycle.

After various experimentations, we chose not to put the inequalities of type (2.7) in the cutting-plane phase because when included, while the number of generated inequalities increases excessively with the size of the graph, the optimal objective value of the linear program is left unimproved.

After the addition of violated inequalities, the linear relaxation is solved to get a fractional (x, y, h) solution and the process is repeated until no more violated inequalities can be found. The optimal objective value of the last LP is denoted by $v_{\mathsf{LP}\ 2}$. Notice that the cutting-plane phase is intentionally limited to less than 10 minutes (600 seconds). Thus, if either no more valid inequalities can be generated or the time spent in the cutting-plane phase reaches 10 minutes, we switch to the Branch&Bound solving phase. The time spent in the cutting-plane phase us denoted by $t_{\mathsf{LP}\ 2}$.

Notice that some automatic internal cuts are generated by the solver to accelerate the branch&bound. As already mentioned, the total running time is limited to 15 minutes (900 seconds). The time spent in the branch&bound phase is denoted by t_{MIP2}. If the 900s have passed before the solver found an optimal solution, then the table will show "$_{>}$900".

2.6.2 Guinea-pig graphs

For each of the instances we report L_{MIP1}, U_{MIP1}, t_{MIP1}, $v_{\mathsf{LP}\ 2}$, $t_{\mathsf{LP}\ 2}$, L_{MIP2}, U_{MIP2}, and t_{MIP2}, where $t_{\mathsf{LP}\ 2}$, t_{MIP1} and t_{MIP2} are expressed in seconds. We also report $\mathrm{nb}_{(2.6)}$,

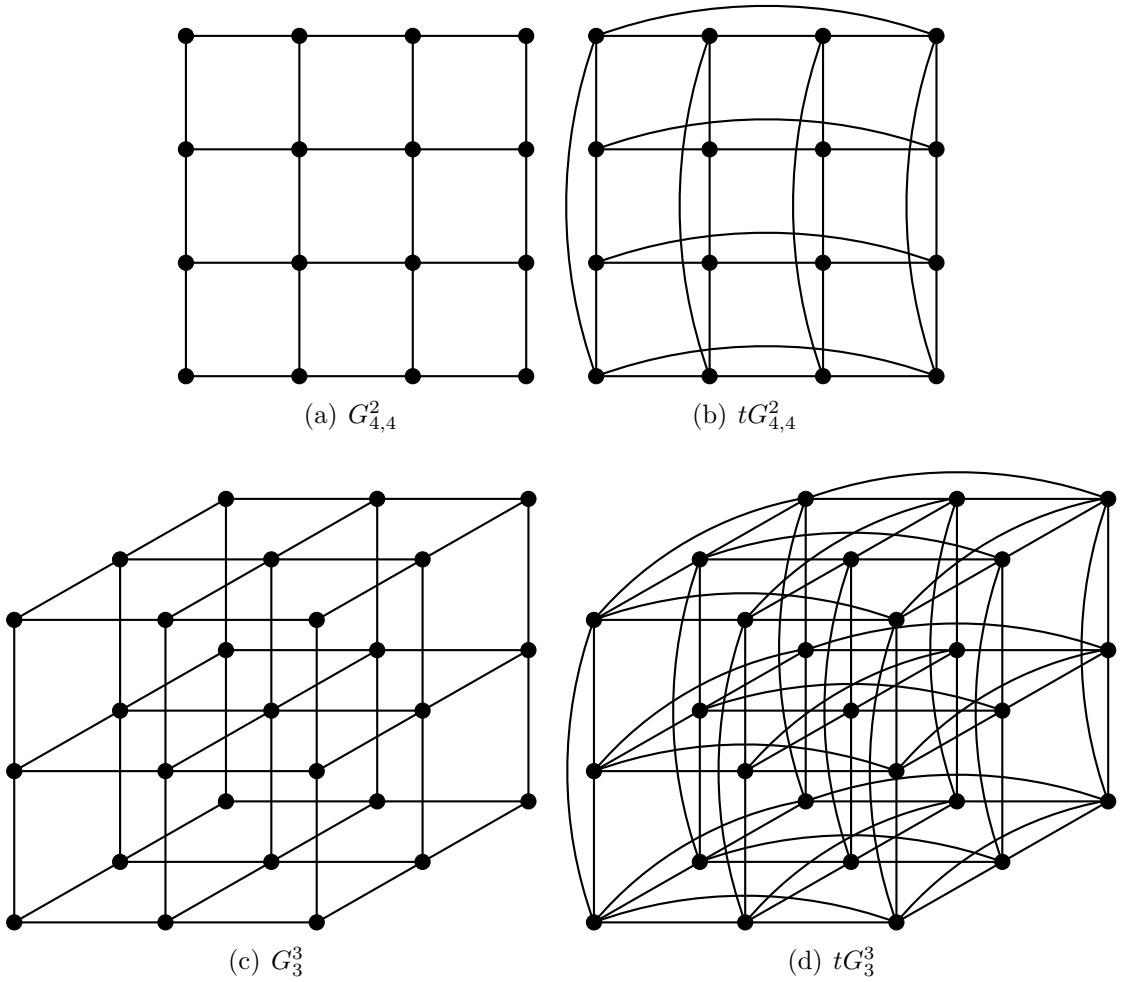
$\text{nb}_{(2.10,2.11)}$ and $\text{nb}_{(2.8,2.9)}$, respectively the number of inequalities of the type (2.6), (2.10) & (2.11) and (2.8) & (2.9) generated for each instance $G = (V, E)$ along with $|V|$,$|E|$ and δ_G.

The graph instances used for the computations are denoted as follows:

- K_n: the complete graph with n vertices,
- G_k^n: the n-dimensional grid of length k, i.e. the cartesian product of n paths graphs of length k: $\bullet_{i=1}^{n} L_k$, where L_k is a path graph on k vertices (e.g. Figure 2.15(c)),
- G_{n_1,n_2}^2: the 2-dimensional grid, i.e. the cartesian product of two path graphs of length $n_1 - 1$ and $n_2 - 1$: $L_{n_1} \bullet L_{n_2}$ (e.g. Figure 2.15(a)),
- tG_k^n: the n-dimensional toroidal grid of length k, i.e. the cartesian product of n cycless of length k: $\bullet_{i=1}^{n} C_k$, where C_k is a cycle of length k (e.g. Figure 2.15(d)),
- tG_{n_1,n_2}^2: the 2-dimensional toroidal grid, i.e. the cartesian product of cycles of length n_1 and n_2: $C_{n_1} \bullet C_{n_2}$ (e.g. Figure 2.15(b)),
- $P_{n,m}$: a randomly generated planar graph with n vertices and m edges,
- $R_{n,m}$: a randomly generated graph with n vertices and m edges.

The Cartesian product used above to define grid graphs is defined as follows. If G and H are two graphs, their *cartesian product* is the graph $G \bullet H$ with set of vertices $V(G) \times V(H)$ and set of edges $\{(u, u')(vv')|u = v \text{ and } u'v' \in E(H) \text{ or } u' = v' \text{ and } uv \in E(G)\}$. More about the Cartesian product can be found in [34] and about the toroidal grids in [31].

Figure 2.15: Example of grid graphs



(a) $G^2_{4,4}$ (b) $tG^2_{4,4}$

(c) G^3_3 (d) tG^3_3

2.6.3 Results

Table 2.1 shows that formulation (MIP2) has a significantly better performance than (MIP1). On many instances we can observe that the cutting-plane algorithm of (MIP2) drastically improved the upper bound δ_G, sometimes so far as to the optimal objective value as can be seen for example on the complete graphs, the planar graphs or most of the n-dimensional toroidal grids. While the number and nature of the generated inequalities varies a lot, it seems to never grow excessively. We can see that as soon as the size of the instance becomes substantial, formulation (MIP1) needs too much time and/or memory to be processed, while (MIP2) allows us to handle much larger graphs.

Table 2.1: Numerical results

graph	L_{MIP1}	U_{MIP1}	t_{MIP1}	$v_{\text{LP 2}}$	$t_{\text{LP 2}}$	L_{MIP2}	U_{MIP2}	t_{MIP2}	nb$_{(2,6)}$	nb$_{(2,10,2,11)}$	nb$_{(2,8,2,9)}$	$\lvert V\rvert$	$\lvert E\rvert$	δ_G
K_{10}	5	5	25.2	5	0	5	5	0	4	2	0	10	45	9
K_{50}	full memory			25	1.6	25	25	4.9	24	12	0	50	1225	49
K_{150}				75	143.4	75	75	431.8	74	52	0	150	11175	149
K_{200}				99	427.5	99	99	859.4	100	75	0	200	19900	199
K_{225}				150	603.8	112	147	>900	74	46	0	225	25200	224
G_4^4	4	4	361	4	0	4	4	0.1	0	0	0	256	768	4
G_3^5	2	5	>900	5	17.5	5	5	54.7	0	0	22	243	810	5
G_5^4	full memory			4	0.1	4	4	0.5	0	0	0	625	2000	4
$G_{5,10}^2$	2	2	68.3	2	0.3	2	2	0.9	0	0	3	50	85	2
$G_{5,15}^2$	2	2	330.9	2	0.8	2	2	2.4	0	0	6	75	130	2
$G_{10,10}^2$	1	2	>900	2	0.8	2	2	2.6	0	0	5	100	180	2
$G_{10,100}^2$	full memory			2	226.8	2	2	682.2	0	0	21	1000	1890	2
$G_{20,50}^2$				2	136	2	2	410	0	0	13	1000	1930	2
tG_4^3	6	6	10.5	6	0.1	6	6	0.2	0	0	0	64	192	6
tG_5^3	2	6	>900	4	83.8	4	4	288.3	2	0	190	125	375	6
tG_4^4	full memory			8	0.7	8	8	3	0	0	0	256	1024	8
tG_4^5				10	17	10	10	484	0	0	0	1024	5120	10
tG_5^5				9	600	4	7	>900	1	0	0	3125	15625	10
$tG_{5,10}^2$	2	2	44.6	2	0	2	2	0.1	2	0	0	50	100	4
$tG_{5,15}^2$	2	2	608.5	2	0	2	2	0.1	2	0	0	75	150	4
$tG_{10,10}^2$	4	4	130.5	4	0.1	4	4	0.3	0	0	0	100	200	4
$tG_{25,50}^2$	full memory			2	14.2	2	2	42.9	2	0	0	1250	2500	4
$tG_{25,100}^2$				2	50.1	2	2	150.9	2	0	0	2500	5000	4
$tG_{50,75}^2$				2	215	2	2	645.5	2	0	0	3750	7500	4
$P_{24,51}$	2	2	0.4	2	0.4	2	2	1.2	0	7	6	24	51	2
$P_{20,54}$	3	3	2.9	3	0.2	3	3	0.7	0	24	4	20	54	3
$P_{25,69}$	3	3	14.7	3	0.3	3	3	1	0	14	3	25	69	3
$R_{10,33}$	3	3	6.9	4	0.2	3	3	0.8	1	11	3	10	33	5
$R_{10,36}$	3	3	277.5	4	0	3	3	0.3	0	11	0	10	36	4
$R_{10,40}$	4	4	449.6	5	0	4	4	0.3	2	35	0	10	40	7
$R_{15,52}$	4	4	25.6	4	0.1	4	4	0.4	1	7	1	15	52	5
$R_{40,156}$	3	3	374.3	3	0.1	3	3	0.6	0	0	0	40	156	3
$R_{15,78}$	5	7	>900	7	0.1	5	5	35.5	1	60	0	15	78	8
$R_{20,95}$	5	6	>900	6	0.6	5	5	12.3	1	40	9	20	95	7
$R_{40,234}$	4	7	>900	7	0.4	7	7	71.5	0	6	2	40	234	7
$R_{30,217}$	6	10	>900	10	0.4	8	9	>900	0	31	4	30	217	10
$R_{40,390}$	7	14	>900	14	1.2	9	13	>900	0	73	6	40	390	14
$R_{75,693}$	5	9	>900	9	0.5	9	9	6.9	0	5	1	75	693	9
$R_{50,918}$	full memory			29	0.5	17	28	>900	0	588	0	50	918	29
$R_{75,1387}$				26	0.6	16	26	>900	0	152	0	75	1387	26

Chapter 3

The maximum cardinality cut

We consider the problem of finding a maximum cardinality cut, i.e. a maximum cut problem for the case where all the edge-weights are equal to 1: $w_e = 1, \forall e \in E$:

$$(\textsc{MaxCut}) \quad \textsc{MaxCut}(G) = \max_{S \subseteq V} |\delta(S)|.$$

and we call $\textsc{MaxCut}(G)$ the value of MaxCut for G.

The chapter is organized as follows. We present the unweighted maximum cut problem seen as a graph orientation optimization problem in Section, 3.1. Next, in light of this new point of view, we introduce some new mixed integer linear programming formulations in Section 3.2. A semidefinite programming relaxation is then proposed. We show that the bound provided by this new SDP relaxation is stronger than the bound given by the relaxation (SDP_{GW}) introduced by Goemans & Williamson in Section 3.3. We also prove that the new bound is tight for complete graphs. We then introduce further Mixed Integer Programming formulations in Section 3.4. Several numerical experiments have been conducted showing the relevance of the SDP formulation and the performances of the new Mixed Integer Programming formulations in Section 3.5.

3.1 Maximum cardinality cut and orientation

In order to build a cut, one has to select a part of the vertices of the graph, in other words to divide the set of vertices into two parts. For example, the exact formulation for the maximum cut problem on which Goemans & Williamson's SDP relaxation (SDP_{GW}) is based can be seen as "labeling" the vertices with either 1 or -1, the derived cut being the edges whose extremities have different labels.

3.1.1 Orienting the edges partitions the vertices

When picking an orientation for a graph, one also separates the set of vertices into two parts: the vertices which signed imbalance is positive and those which signed imbalance is negative. Furthermore, the cardinality of the cut derived from this partition of the vertices equals the sum of the imbalances of the vertices which signed imbalance is positive, which also equals the sum of the imbalances of the vertices which signe imbalance is negative. Thus searching for a maximum cardinality cut is equivalent to searching for an orientation maximizing the sum of the imbalances of all the vertices in G:

Theorem 49. *For any graph G, the maximum cardinality of a cut equals half of the maximum sum of the imbalances of all the vertices over all the orientations of G:*

$$(\textsc{MaxCut}) \quad \textsc{MaxCut}(G) = \max_{\Lambda \in \vec{O}(G)} \frac{1}{2} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)|.$$

And here appears our generic problem for $f = \frac{1}{2}|| \cdot ||_1$.

Proof. Considering our generic formulation in Section 1.5, we have

$$\max_{\Lambda \in \vec{O}(G)} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| = \max_{x \in \{-1,1\}^{|E|}} \sum_{v \in v} |B_v x|.$$

Adding a sign variable $z \in \{-1,1\}^{|V|}$, we have

$$\max_{\Lambda \in \vec{O}(G)} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| = \max_{\substack{x \in \{-1,1\}^{|E|} \\ z \in \{-1,1\}^{|V|}}} z^\intercal B x = \max_{\substack{x \in \{-1,1\}^{|E|} \\ z \in \{-1,1\}^{|V|}}} x^\intercal B^\intercal z.$$

By construction of B, the product of the row of $B^\intercal$ corresponding to an arc $\overrightarrow{uv}$ with z is $B^\intercal_{uv} z = z_u - z_v$. Thus,

$$\begin{aligned} \max_{\Lambda \in \vec{O}(G)} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| &= \max_{\substack{x \in \{-1,1\}^{|E|} \\ z \in \{-1,1\}^{|V|}}} \sum_{uv \in E} x_{uv}(z_u - z_v) \\ &= \max_{z \in \{-1,1\}^{|V|}} \sum_{uv \in E} |z_u - z_v|. \end{aligned}$$

Now, for $S \subseteq V$, if we consider the vector $z \in \{-1,1\}^n$ such that for each $v \in V$, $z_v = 1$ if and only if $v \in S$, then for an edge $uv \in E$, the quantity $|z_u - z_v|$ equals 2 if and only if $uv \in \delta(S)$ and equals 0 otherwise. Consequently, the maximum cardinality cut can be expressed as

$$\frac{1}{2} \max_{z \in \{-1,1\}^{|V|}} \sum_{uv \in E} |z_u - z_v|. \qquad \square$$

3.1.2 Vice versa

We have shown that given an orientation with maximized sum of the imbalances, it is easy to derive a maximum cardinality cut: $S = \{v \in V | d^+_\Lambda(v) \geq d^-_\Lambda(v)\}$. Moreover, the cardinality of S equals half of the sum of the imbalances w.r.t. said orientation, see Fig. 3.1.

Conversely, one can easily derive from a maximum cardinality cut an orientation with maximized sum of the imbalances.

Proposition 50. *For any graph G, if $S \subseteq V$ is a maximum cardinality cut, then any $\Lambda \in \vec{O}(G)$ such that*

$$\Lambda(uv) = \overrightarrow{uv}, \ \forall uv \in E \ s.t. \ (u,v) \in S \times V \backslash S \tag{3.1}$$

verifies

$$\frac{1}{2} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| = \textsc{MaxCut}(G).$$

In other words, any such Λ is optimal w.r.t. MaxCut.

Proof. Let H be the subgraphs of G with edge set $\delta(S)$. Since S is a maximum cardinality cut, then we have $|\delta(\{v\}) \cap \delta(S)| \geq \left\lceil \frac{d(v)}{2} \right\rceil$ for all vertex $v \in V$ as explained in the proof of Theorem 25. Therefore, since Λ verifies (3.1), then we know that $d^+_\Lambda(v) - d^-_\Lambda(v) > 0$ for any vertex $v \in S$ and $d^+_\Lambda(v) - d^-_\Lambda(v) < 0$ for any vertex $v \in V \backslash S$. We now develop the value of Λ w.r.t. MaxCut:

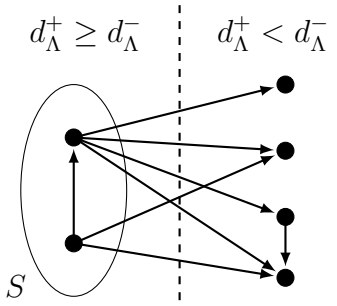
$$\begin{aligned}
&\frac{1}{2} \sum_{v \in V} |d^+_\Lambda(v) - d^-_\Lambda(v)| \\
=&\frac{1}{2} \left(\sum_{v \in S} \left(d^+_\Lambda(v) - d^-_\Lambda(v) \right) + \sum_{v \in V \backslash S} \left(d^-_\Lambda(v) - d^+_\Lambda(v) \right) \right) \\
=&\frac{1}{2} \left(|\delta(S)| + |\delta(S)| \right) \\
=&|\delta(S)|.
\end{aligned}$$

□

The special link we have between cut and orientation in this problem will turn out to be very fruitful. An optimal orientation carries the information of an optimal cut, but not only, for an optimal cut may be obtained via many orientations. This correlation will lead to exact formulations with higher quality relaxations and consequently to tighter relaxed approximation formulations.

In light of this new approach of the maximum cardinality cut problem, we derive new mixed integer linear prgramming formulations for MaxCut.

Figure 3.1: Example of the relationship between an optimal cut $\delta(S)$ and an optimal orientation Λ.



3.2 Mixed integer linear programming formulations

In the preceding chapter, we have seen several proper ways to give exact formulations for one of our orientation optimization problem, their core coming from the generic formulation given in Section 1.5. The additional work came from the fact that the objective function of MaxIm is not linear and we therefore had to make use of various linearization techniques to obtain a mixed integer linear formulation. In this case, the objective function $f = \frac{1}{2}||\cdot||_1$ is not linear either, we therefore will have to find the adequate linearizations again.

3.2.1 A first naïve formulation

The generic formulation from Section 1.5 here gives

$$\max_{x\in\{-1,1\}^{|E|}} \sum_{v\in V} |B_v x|,$$

involving orientations variables x. We therefore have once again the absolute values to deal with. Let us then summon binary indicators variables as we did in the design of (MIP2) in Section 2.4: y^v_k, with $v \in V$ and $k \in [\![-d(v), d(v)]\!]$. Once again, they have the following interpretation: $y^v_k = 1$ if and only if $B_v x = k$, so that the following equations trivially hold

$$\sum_{k=-d(v)}^{d(v)} k y^v_k = B_v x, \forall v \in V. \tag{3.2}$$

Also, given the interpretation for the variables y, among those of the form y^v_k, for some fixed node $v \in V$, exactly one of them has value 1. Thus, the following

constraints are satisfied

$$\sum_{k=-d(v)}^{d(v)} y_k^v = 1, \ \forall v \in V. \tag{3.3}$$

Then we can show that the maximum cardinality cut problem may be formulated as the mixed integer program

$$(\mathsf{MIP3}) \begin{cases} Z^\star_{\mathsf{MIP3}} = \frac{1}{2} \max \sum_{v \in V} \sum_{k=-d(v)}^{d(v)} |k| y_k^v \\ \text{s.t. } (3.2), (3.3), \\ x \in [-1;1]^{|E|}, \\ y_k^v \in \{0,1\}, \ \forall v \in V, \ \forall k \in [\![-d(v), d(v)]\!]. \end{cases}$$

The following result asserts that solving the mixed integer linear program (MIP3) returns the maximum cardinality of a cut in the input graph.

Proposition 51. *The optimal objective value of* (MIP3) *equals the maximum cardinality of a cut in the graph* G*:* $Z^\star_{\mathsf{MIP3}} = w^\star$.

Proof. If we show that for any graph, there is an optimal solution $(x, y) \in [-1, 1]^m \times \{0, 1\}^{4|E|}$ of (MIP3) such that $x \in \{-1, 1\}^{|E|}$, then we can conclude with Theorem 49. If $(x, y) \in [-1, 1]^{|E|} \times \{0, 1\}^{4|E|}$ is a solution of (MIP3), then since y is integer-valued and B is totally unimodular, then x is an integer-valued as well. Hence x describes a partial orientation of the edges of the input graph G, i.e. there may be some unoriented edges of G w.r.t. x: $\{e \in E | x_e = 0\}$. We take $S = \{v \in V | d_x^+(v) \geq d_x^-(v)\}$. If an edge $e \in E$ left unoriented w.r.t. x belonged to $\delta(S)$, then orienting it from S to $V \backslash S$ would increase by 2 the value of (x, y) thus contradicting its optimality. Then $e \notin \delta(S)$ and x can therefore be arbitratily completed into an orientation of G (i.e. an element of $\{-1, 1\}^{|E|}$) whilst remaining optimal, according to Proposition 50.

□

3.2.2 The orientation variables become redundant

Now we will see that with an observation of the behavior of the indicator variables of the endpoints of an edge in relation to its orientation, we can remove orientation variables from our formulation altogether.

Given an optimal solution $(x, y) \in [-1, 1]^{|E|} \times \{0, 1\}^{4|E|}$ of (MIP3), for any edge $uv \in E$ which has the original orientation from u to v (i.e. $b_{v,uv} = -1$), the following equation holds: $x_{uv} = \sum_{k=1}^{d(u)} y_k^u - \sum_{k=1}^{d(v)} y_k^v$. Reversly, if the original orientation of uv is from v to u (i.e. $B_{v,uv} = 1$), then $x_{uv} = \sum_{k=1}^{d(v)} y_k^v - \sum_{k=1}^{d(u)} y_k^u$, which yields, whichever the original orientation of uv,

$$B_{v,uv}x_{uv} = \sum_{k=1}^{d(v)} y_k^v - \sum_{k=1}^{d(u)} y_k^u.$$

From the latter we deduce (developing the expression $B_v x$):

$$\sum_{k=-d(v)}^{d(v)} k y_k^v = B_v x = \sum_{uv\in E} \left(\sum_{k=1}^{d(v)} y_k^v - \sum_{k=1}^{d(u)} y_k^u \right). \tag{3.4}$$

Observe also that for a maximum cardinality cut $\delta(S)$, $S \subseteq V$, each vertex v is incident to at least $\left\lceil \frac{d(v)}{2} \right\rceil$ edges in $\delta(S)$, as explained in the proof of Theorem 25. Therefore, since Λ verifies (3.1). This implies that the variables y_k^v with $k \in \left\{1 - \left\lceil \frac{d(v)}{2} \right\rceil, \ldots, \left\lceil \frac{d(v)}{2} \right\rceil - 1\right\}$ may be removed from formulation (MIP3), while Proposition 51 remains valid.

We therefore denote for all $v \in V$ the sets

- $I_v^- = \left[\!\left[-d(v), -\left\lceil \frac{d(v)}{2} \right\rceil \right]\!\right]$,
- $I_v^+ = \left[\!\left[\left\lceil \frac{d(v)}{2} \right\rceil, d(v) \right]\!\right]$,
- $I_v = I_v^- \cup I_v^+$.

It follows that, in place of (MIP3), we may consider a formulation involving exclusively variables of the form y_k^v by replacing equations (3.2) with (3.4). For each vertex v, we only consider y_k^v variables for $k \in I_v$.

$$\text{(MIP4)} \begin{cases} Z^\star_{\text{MIP4}} = \frac{1}{2} \max \sum_{v\in V} \sum_{k\in I_v} |k| y_k^v \\ \text{s.t.} \\ \sum_{k\in I_v} y_k^v = 1, \ \forall v \in V, & \text{(3.5a)} \\ \sum_{k\in I_v} k y_k^v = \sum_{uv\in E} (\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u), \ \forall v \in V, & \text{(3.5b)} \\ y_k^v \in \{0,1\}, \ \forall v \in V, \ \forall k \in I_v. \end{cases}$$

We show that solving the mixed integer linear program (MIP3) returns the maximum cardinality of a cut in the input graph.

Proposition 52. *The optimal objective value of* (MIP4) *equals the maximum cardinality of a cut in the graph G: $Z^\star_{\text{MIP4}} = w^\star$.*

Proof. If $(x, y) \in [-1, 1]^{|E|} \times \{0, 1\}^{4|E|}$ is a solution of (MIP3) then the y_v^k with $v \in V$ and $\in I_v$ form a solution of (MIP4) with the same value. Conversely, we

take $(y_k^v)_{v\in V,\ k\in I_v}$ a solution of (MIP4). We build the following partial orientation. For each edge $uv \in E$, if $\sum_{k\in I_u} k y_k^u$ and $\sum_{k\in I_v} k y_k^v$ have different sign, then uv is oriented from its "non-negative" endpoint to its "negative" endpoint. We take $\tilde{x} \in \{-1,0,1\}^{|E|}$ that describes this partial orientation and $\tilde{y} \in \{0,1\}^{4|E|}$ where for each vertex $v \in V$, $\tilde{y}_k^v = y_k^v$ if $k \in I_v$ and $\tilde{y}_k^v = 0$ if $k \in [\![1 - \left\lceil \frac{d(v)}{2} \right\rceil, \left\lceil \frac{d(v)}{2} \right\rceil - 1]\!]$. Then $(\tilde{x}, \tilde{y})$ is a solution of (MIP3) sharing its value with y w.r.t. (MIP4). (MIP3) and (MIP4) are therefore equivalent.

□

Formulation (MIP4) involves $O(|E|)$ (yet, roughly half as mush as (MIP3)) variables and $O(|V|)$ constraints. If we take following fractional solution of (MIP4) : $y^v_{-d(v)} = y^v_{d(v)} = \frac{1}{2}$, $\forall v \in V$ and $y_k^v = 0$, $\forall v \in V$, $\forall k \in I_v \backslash \{-d(v), d(v)\}$, we see that its value is m, i.e. a trivial upper bound for the maximum cardinality cut. Hence, the linear relaxation of (MIP4) has yet to be improved.

3.2.3 A stronger MIP

We will consider a strengthening of the linear relaxation of (MIP4) through reformulation-linearization techniques. The latter is obtained by multiplying constraints of (MIP4) and then linearizing. If we take (3.5a) for $u \in V$ and multiply it by y_k^v for $v \in V$ and $k \in I_v$, we get

$$y_k^v = \sum_{l\in I_u} y_l^u y_k^v. \tag{3.6}$$

Take (3.5b) for $v \in V$ an multiply it by y_k^v for $k \in I_v^+$, we obtain

$$\begin{aligned}
\sum_{l\in I_v} l y_l^v y_k^v &= \sum_{uv\in E} \left(\sum_{l\in I_v^+} y_l^v y_k^v - \sum_{l\in I_u^+} y_l^u y_k^v \right), \\
\sum_{l\in I_v} l y_l^v y_k^v &= d(v) \sum_{l\in I_v^+} y_l^v y_k^v - \sum_{uv\in E} \sum_{l\in I_u^+} y_l^u y_k^v, \\
\sum_{l\in I_v} (d(v) - l)\, y_l^v y_k^v &= \sum_{uv\in E} \sum_{l\in I_u^+} y_l^u y_k^v.
\end{aligned}$$

Notice that since only one indicator per variable per vertex can be not equal to zero, we have that for any vertex $v \in V$, $y_k^v y_l^v = 0$ for any $(k,l) \in I_v r$ such that $k \neq l$. Which yields

$$(d(v) - k)\, y_k^v y_k^v = \sum_{uv\in E} \sum_{l\in I_u^+} y_l^u y_k^v. \tag{3.7}$$

Using the very same reasoning on (3.5b) for $v \in V$ an multiply it by y_k^v for $k \in I_v^-$, we get

$$-k y_k^v y_k^v = \sum_{uv \in E} \sum_{l \in I_u^+} y_l^u y_k^v. \tag{3.8}$$

Now if we take (3.5b) for $u \in V$ and this time multiply it by y_k^v for $v \in V \backslash \{u\}$ and $k \in I_v$, then we have

$$\sum_{l \in I_u} l y_l^u y_k^v = \sum_{uw \in E} \left(\sum_{l \in I_u^+} y_l^u y_k^v - \sum_{l \in I_w^+} y_l^w y_k^v \right). \tag{3.9}$$

The equalities (3.6), (3.7), (3.8) and (3.9) are not linear in terms of the indicator variables because they all bear product of two of these variables. One way to make use of them in a linear formulation is to switch to product variables: let Y_{kl}^{vu}, with $(v,u) \in V^2$ and $(k,l) \in I_v \times I_u$, denote a binary variable representing the product $y_k^v y_l^u$. Then Y denotes a symmetric matrix with rows and columns indexed by all pairs (v,k) with $u \in V$ and $k \in I_v$. The entry in the row indexed by (v,k) and column indexed by (u,l) corresponds to the variable Y_{kl}^{vu} which is binary since it represents the product of two binary variables. According to their interpretation, we have for $Y_{kk}^{vv} = y_k^v$ for any $v \in V$ and $k \in I_v$ and $Y_{kl}^{vu} = Y_{lk}^{uv}$ for any $(v,u) \in V^2$ and $(k,l) \in I_v \times I_u$. Then we can deduce from (MIP4) the following exact formulation for MaxCut.

$$(\mathsf{MIP5}) \begin{cases} Z^\star_{\mathsf{MIP5}} = \dfrac{1}{2} \max \displaystyle\sum_{v \in V} \sum_{k \in I_v} |k| Y_{kk}^{vv} \\ \text{s.t.} \\ \displaystyle\sum_{k \in I_v} Y_{kk}^{vv} = 1, \ \forall v \in V, & (3.10a) \\ \displaystyle\sum_{k \in I_v} k Y_{kk}^{vv} = \sum_{uv \in E} \Big(\sum_{k \in I_v^+} Y_{kk}^{vv} - \sum_{k \in I_u^+} Y_{kk}^{uu} \Big), \ \forall v \in V, & (3.10b) \\ Y_{kk}^{vv} = \displaystyle\sum_{l \in I_u} Y_{kl}^{vu}, \ \forall u, v \in V, \ \forall k \in I_v, & (3.10c) \\ (d(v) - k) Y_{kk}^{vv} = \displaystyle\sum_{uv \in E} \sum_{l \in I_u^+} Y_{kl}^{vu}, \ \forall v \in V, \ \forall k \in I_v^+, & (3.10d) \\ -k Y_{kk}^{vv} = \displaystyle\sum_{uv \in E} \sum_{l \in I_u^+} Y_{kl}^{vu}, \ \forall v \in V, \ \forall k \in I_v^-, & (3.10e) \\ \displaystyle\sum_{l \in I_u} l Y_{kl}^{vu} = \sum_{uw \in E} \Big(\sum_{l \in I_u^+} Y_{kl}^{vu} - \sum_{l \in I_w^+} Y_{kl}^{vw} \Big), \ \forall v \neq u \in V, \ \forall k \in I_v, & (3.10f) \\ Y_{kl}^{vu} = Y_{lk}^{uv}, \ \forall u, v \in V, \ \forall (k,l) \in I_v \times I_u, & (3.10g) \\ Y_{kl}^{vu} \in \{0,1\}, \ \forall u, v \in V, \ \forall (k,l) \in I_v \times I_u. & (3.10h) \end{cases}$$

The constraints (3.10a) and (3.10b) directly come from (3.5a) and (3.5b); (3.10c), (3.10d), (3.10e) and (3.10f) respectively stem from (3.6), (3.7), (3.8) and (3.9).

Proposition 53. *The optimal objective value of* (MIP5) *equals the maximum cardinality of a cut in the graph* G*:* $Z^\star_{\mathsf{MIP5}} = w^\star$.

Proof. Given the interpretation of the matrix variable Y in (MIP5), the diagonal of a solution of (MIP5) is a solution for (MIP4) with the same value and, conversely, for a solution of (MIP4) y, $yy^\intercal$ is a solution for (MIP5) with the same value. Hence (MIP4) and (MIP5) are equivalent. □

Observe that if we take $u = v$ in constraints (3.10c), we get $Y^{vv}_{kl} = 0$ if $k \neq l$. Exactly as we should since for any vertex, only one indicator variable is not equal to zero. We thus know now which variables are irrelevant and can be removed from the formulation in order to reduce its size. (MIP5) was derived from (MIP4) using lifting. Other formulations could be obtained using some other well-known lifting techniques such as the one of Lassere, the Sherali-Adams technique or the lifting of Lovász-Schrijver [89, 74, 96]. We do not elaborate more on this topic.

This lifting technique does not guarantee any improvement in the performance of our formulations. However, it is highly probable that it improves the quality of the relaxations inherent to our formulations and/or the bounds the bounds that might be derived from such relaxations, whether it is a linear program or a semidefinite one.

3.3 A semidefinite programming bound

We now build a semidefinite programming formulation from (MIP5). Let us first mention several ways to summon SDP constraints depending on the nature of the problem to be relaxed.

3.3.1 Handling SDP constraints

Consider an optimization problem on a variable $x \in \{-1, 1\}^n$ with an objective function and constraints all linear on the products of two components of x. In other words, an optimization problem that can be formulated in the following manner:

$$\begin{cases} \max tr(C^\intercal xx^\intercal) \\ \text{s.t.} \\ tr(A_1^\intercal xx^\intercal) \leq b_1 \\ \quad \vdots \\ tr(A_m^\intercal xx^\intercal) \leq b_m \\ x \in \{-1;1\}^n, \end{cases}$$

where $C, A_1, \cdots, A_m$ belong to $\mathbb{S}_n$ and $tr(M)$ denotes the trace of the matrix M, i.e. the sum of the components on its diagonal. Now let $X = xx^\intercal$, then the integrity constraint $x \in \{-1;1\}^n$ is equivalent to $X_{i,i} = 1,\ \forall i \in [\![1,n]\!]$ and the problem can thus be relaxed into the following semidefinite program:

$$\begin{cases} \max tr(C^\intercal X) \\ \text{s.t.} \\ tr(A_1^\intercal X) \leq b_1 \\ \quad \vdots \\ tr(A_m^\intercal X) \leq b_m \\ X_{i,i} = 1,\ \forall i \in [\![1,n]\!] \\ X \succeq 0 \\ X \in M_{n\times n}(\mathbb{R}), \end{cases}$$

thus providing an upper bound on the considered optimization problem.

For example, if for a weighted graph with set of vertices $[\![1,n]\!]$ we call W the matrix such that the component on row i and column j is w_{ij}, the weight of the edge connecting i to j (equal to zero if they are not neighbours), then the non-linear formulation for the maximum cut problem (1.8) given in Section 1.7 of the Introduction can be rewritten with $Y = zz^\intercal$ as

$$\begin{cases} \max \frac{1}{4}\sum_{i=1}^n \sum_{j=1}^n w_{ij} - tr(W^\intercal Y) \\ \text{s.t.} \\ Y_{i,i} = 1,\ \forall i \in [\![1,n]\!] \\ Y = zz^\intercal \\ Y \in M_{n\times n}(\mathbb{R}),\ z \in \mathbb{R}^n, \end{cases}$$

The semidefinite relaxation of $Y = zz^\intercal$ directly leads to the SDP formulation (SDP_{GW}) introduced in the introduction on which Goemans & Williamson based their ground-breaking approximation algorithm.

Having explained how to obtain the SDP relaxation of a -1/1 optimization problem, let us see how do we do it with a 0/1 optimization problem that can be formulated as follows:

$$\begin{cases} \max tr(C^\intercal xx^\intercal) \\ \text{s.t.} \\ tr(A_1^\intercal xx^\intercal) \leq b_1 \\ \qquad \vdots \\ tr(A_m^\intercal xx^\intercal) \leq b_m \\ x \in \{0;1\}^n, \end{cases}$$

where $C, A_1, \cdots, A_m$ belong to $\mathbb{S}_n$. Let $X = xx^\intercal$, then the integrity constraint $x \in \{0;1\}^n$ is equivalent to $X_{i,i} = x_i, \ \forall i \in [\![1,n]\!]$ and the problem can thus be relaxed, leading to the following semidefinite program:

$$\begin{cases} \max tr(C^\intercal X) \\ \text{s.t.} \\ tr(A_1^\intercal X) \leq b_1 \\ \qquad \vdots \\ tr(A_m^\intercal X) \leq b_m \\ X - Diag(X)Diag(X)^\intercal \succeq 0 \\ X \in M_{n\times n}(\mathbb{R}). \end{cases}$$

To follow up our example on relaxing formulations of the maximum cut problem, let us turn formulation (1.8) into a 0/1 formulation.

$$\begin{cases} \max \frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n w_{ij}(X_{i,i} + X_{j,j} - 2X_{i,j}) \\ \text{s.t.} \\ X_{i,i} = x_i, \ \forall i \in [\![1,n]\!] \\ X = xx^\intercal \\ X \in M_{n\times n}(\mathbb{R}), \ x \in \mathbb{R}^n, \end{cases}$$

The semidefinite relaxation of $X = xx^\intercal$ leads to the following SDP formulation.

$$(\mathsf{SDP}^b_{GW}) \begin{cases} Z^\star_{\mathsf{SDP}^b_{GW}} = \max \frac{1}{2}\sum_{i=1}^n \sum_{j=1}^n w_{ij}(X_{i,i} + X_{j,j} - 2X_{i,j}) \\ \text{s.t.} \\ X - Diag(X)Diag(X)^\intercal \succeq 0, \\ X \in M_{n\times n}(\mathbb{R}), \end{cases}$$

Its optimal objective value coincides with $Z^\star_{\mathsf{SDP}_{GW}}$, it simply is the result of a change of variable applied to (SDP_{GW}).

The following result derived from Schur's Lemma (cf. [74]) is quite handy in the case of implementing such a relaxation derived from a 0/1 optimization problem

Proposition 54 (Lovász & Schrijver, 1991)**.** *For any matrix $X \in M_{n\times n}(\mathbb{R})$, the following assertions are equivalent:*

- $X - Diag(X)Diag(X)^\intercal \succeq 0$,
- $\begin{pmatrix} 1 & Diag(X)^\intercal \\ Diag(X) & X \end{pmatrix} \succeq 0.$

The reader may find details concerning semidefinite optimization and more precisely SDP relaxations in [44, 48, 74, 75]. In light of these considerations, we may now build a SDP relaxation of (MIP5).

3.3.2 Relaxing MIP5 into SDP

Relaxing the integrity constraint (3.10h) from (MIP5) and replacing it with a semidefinite positivity constraint on $Y - Diag(Y)Diag(Y)^\intercal$ lead us to the following formulation.

$$
(\mathsf{SDP5}) \begin{cases}
Z^\star_{\mathsf{MIP4}} = \dfrac{1}{2} \max \displaystyle\sum_{v \in V} \sum_{k \in I_v} |k| Y^{vv}_{kk} \\
\text{s.t.} \\
\displaystyle\sum_{k \in I_v} Y^{vv}_{kk} = 1, \ \forall v \in V, & (3.11a) \\
\displaystyle\sum_{k \in I_v} k Y^{vv}_{kk} = \sum_{uv \in E} \Big(\sum_{k \in I^+_v} Y^{vv}_{kk} - \sum_{k \in I^+_u} Y^{uu}_{kk} \Big), \ \forall v \in V, & (3.11b) \\
Y^{vv}_{kk} = \displaystyle\sum_{l \in I_u} Y^{vu}_{kl}, \ \forall u, v \in V, \ \forall k \in I_v, & (3.11c) \\
(d(v) - k) Y^{vv}_{kk} = \displaystyle\sum_{uv \in E} \sum_{l \in I^+_u} Y^{vu}_{kl}, \ \forall v \in V, \ \forall k \in I^+_v, & (3.11d) \\
- k Y^{vv}_{kk} = \displaystyle\sum_{uv \in E} \sum_{l \in I^+_u} Y^{vu}_{kl}, \ \forall v \in V, \ \forall k \in I^-_v, & (3.11e) \\
\displaystyle\sum_{l \in I_u} l Y^{vu}_{kl} = \sum_{uw \in E} \Big(\sum_{l \in I^+_u} Y^{vu}_{kl} - \sum_{l \in I^+_w} Y^{vw}_{kl} \Big), \ \forall v \neq u \in V, \ \forall k \in I_v, & (3.11f) \\
Y - Diag(Y)Diag(Y)^\intercal \succeq 0, & (3.11g) \\
Y^{vu}_{kl} \geq 0, \forall u, v \in V, \forall k \in I^-_v \cup I^+_v, l \in I^-_u \cup I^+_u. & (3.11h)
\end{cases}
$$

The semidefinite positivity constraint rendered constraint (3.10g) obsolete.

Relaxing constraints in a maximization formulation leads to formulations that provide upper bounds on the optimal objective value originally sought. The loss of exactness that a relaxation leads to is generally compensated by the gain in complexity and more practically in efficiency. Our SDP relaxation bearing many resemblances with Goemans & Williamson's is more cumbersome in terms of size. The question that follows is whether the quality of the bound provided by (SDP5) compensates its augmented size.

3.3.3 Domination of the new upper bound

We are going to prove that this semidefinite relaxation provides a generally better upper bound for the maximum cardinality cut problem than that from Goemans & Williamson's relaxation. In this context, generally better means at least as good as (i.e. lower than) Goemans & Williamson's upper bound for any graph.

Proposition 55. *The following inequality holds*

$$Z^{\star}_{\mathsf{SDP5}} \leq Z^{\star}_{\mathsf{SDP}^{b}_{GW}} (= Z^{\star}_{\mathsf{SDP}_{GW}}).$$

Proof. Let Y denote a feasible solution for the formulation ($\mathsf{SDP5}$) and let $(X, x) \in \mathbb{R}^{n\times n} \times \mathbb{R}^n$ be defined as follows:

- $X_{uv} := \sum_{k\in I_v^-} \sum_{l\in I_u^-} Y^{vu}_{kl}$, $\forall u, v \in V$ with $u \neq v$,
- $X_{vv} = x_v := \sum_{k\in I_v^-} Y^{vv}_{kk}, \forall v \in V$.

We now show $X - xx^{\mathsf{T}} \succeq 0$. Let $z \in \mathbb{R}^n$ and define $\overline{z}$ of convenient dimension as follows: $\overline{z}^v_k = z_v$ if $k < 0$ and 0 otherwise. Then, we have

$$\begin{aligned} \overline{z}^{\mathsf{T}} Y \overline{z} &= \sum_{v\in V} \sum_{k\in I_v^-} \sum_{u\in V} \sum_{l\in I_u^-} \overline{z}^v_k \overline{z}^u_l Y^{vu}_{kl} \\ &= \sum_{v\in V} \sum_{u\in V} z_v z_u (\sum_{k\in I_v^-} \sum_{l\in I_u^-} Y^{vu}_{kl}) \\ &= \sum_{v\in V} \sum_{u\in V} z_v z_u X_{uv} = z^{\mathsf{T}} X z. \end{aligned}$$

Also,

$$\begin{aligned} \overline{z}^{\mathsf{T}} Diag(Y) &= \sum_{v\in V} \sum_{k\in I_v^-} \overline{z}^v_k Y^{vv}_{kk} \\ &= \sum_{v\in V} (z_v \sum_{k\in I_v^-} Y^{vv}_{kk}) \\ &= \sum_{v\in V} z_v x_v = z^{\mathsf{T}} x. \end{aligned}$$

It follows that

$$z^{\mathsf{T}}(X - xx^{\mathsf{T}})z = \overline{z}^{\mathsf{T}}(Y - Diag(Y)Diag(Y)^{\mathsf{T}})\overline{z} \geq 0,$$

where the last inequality follows from the feasibility of Y w.r.t. ($\mathsf{SDP5}$), and thus $X - xx^{\mathsf{T}} \succeq 0$. So, we have shown (X, x) is a feasible solution for (SDP^{b}_{GW}).

Since Y is symmetric, the same holds for X. We now show the objective value

of (X, x) w.r.t. (SDP^b_{GW}), denoted Z_X equals that of Y w.r.t. $(\mathsf{SDP5})$.

$$\begin{aligned}
Z_X &= \tfrac{1}{2}(\textstyle\sum_{v\in V}\sum_{u:uv\in E}(x_u + x_v - 2X_{uv})) \\
&= \textstyle\sum_{v\in V} d(v)x_v - \sum_{v\in V}\sum_{u:uv\in E} X_{uv} \\
&= \textstyle\sum_{v\in V} d(v)\sum_{k\in I_v^-} Y^{vv}_{kk} - \sum_{v\in V}\sum_{u:uv\in E}\sum_{k\in I_v^-}\sum_{l\in I_u^-} Y^{vu}_{kl} \\
&= \textstyle\sum_{v\in V} d(v)\sum_{k\in I_v^-} Y^{vv}_{kk} - \sum_{v\in V}\sum_{k\in I_v^-}(\sum_{u:uv\in E}\sum_{l\in I_u^-} Y^{vu}_{kl}) \\
&= \textstyle\sum_{v\in V} d(v)\sum_{k\in I_v^-} Y^{vv}_{kk} - \sum_{v\in V}\sum_{k\in I_v^-}(\sum_{u:uv\in E}(Y^{vv}_{kk} - \sum_{l\in I_u^+} Y^{vu}_{kl})) \\
&= \textstyle\sum_{v\in V} d(v)\sum_{k\in I_v^-} Y^{vv}_{kk} - \sum_{v\in V}\sum_{k\in I_v^-}(d(v)Y^{vv}_{kk} - \sum_{u:uv\in E}\sum_{l\in I_u^+} Y^{vu}_{kl}) \\
&= \textstyle\sum_{v\in V} d(v)\sum_{k\in I_v^-} Y^{vv}_{kk} - \sum_{v\in V}\sum_{k\in I_v^-}(d(v)Y^{vv}_{kk} + kY^{vv}_{kk}) \\
&= \textstyle\sum_{v\in V} d(v)\sum_{k\in I_v^-} Y^{vv}_{kk} - \sum_{v\in V}(k + d(v))\sum_{k\in I_v^-} Y^{vv}_{kk} \\
&= -\textstyle\sum_{v\in V}\sum_{k\in I_v^-} kY^{vv}_{kk} \\
&= \tfrac{1}{2}\textstyle\sum_{v\in V}\sum_{k\in I_v} |k| Y^{vv}_{kk}.
\end{aligned}$$

□

We now have the theoretic result of domination of the bound given by $(\mathsf{SDP5})$ over that of (SDP_{GW}). Which doesn't mean it really is strictly lower, significantly or even at all. We will see in Section 3.5 that it is indeed practically strictly lower, and significantly for many instances.

The resemblance we mentioned above between (SDP_{GW}) and $(\mathsf{SDP5})$ will allow us to derive from $(\mathsf{SDP5})$ the same randomized method to extract an approximation algorithm that Goemans& Williamson used to build their with the same approximation ratio. The idea again is to extract a solution for (SDP_{GW}) from a solution of $(\mathsf{SDP5})$ by aggregating the variables adequately.

Corollary 56. *There exists a polynomial time α-approximation algorithm for the maximum cardinality cut problem based on formulation* $(\mathsf{SDP5})$*, where* $\alpha \simeq 0.87856$.

Proof. Take Y a solution of $(\mathsf{SDP5})$ and let $Z \in \mathbb{R}^{n\times n}$ denote the matrix with entries $Z_{uv} = 4(X_{uv} - x_u x_v) + (2x_u - 1)(2x_v - 1)$, where

- ▷ $X_{uv} \coloneqq \sum_{k\in I_v^-}\sum_{l\in I_u^-} Y^{vu}_{kl}$, $\forall u, v \in V$ with $u \neq v$,
- ▷ $X_{vv} = x_v \coloneqq \sum_{k\in I_v^-} Y^{vv}_{kk}$, $\forall v \in V$.

Proposition 55 gives us that Z is a feasible solution for the formulation (SDP_{GW}) with the same objective value as Y. Then, following Goemans & Williamson's randomized approximation algorithm (cf.[47]) summarized in Section 1.7, we get a cut $\delta(S)$ which expectation of the cardinality is be bounded as follows.

$$\alpha w^\star \leq E(|\delta(S)|) \leq w^\star,$$

Where $\alpha = \min_{\theta \in [0,\pi]} \frac{2\theta}{\pi(1-\cos\theta)} \simeq 0.87856$.

□

We therefore note that the choice of sacrificing the lightness of (SDP_{GW}) is relevant for it may pay off in improvement of the quality of the bound, we see that such expectation is theoretically viable. Other aspects might benefit from this choice.

3.3.4 Exactness for complete graphs

The new bound given by relaxation ($\mathsf{SDP5}$) is exact (i.e., equal to the maximum cardinality of a cut) for some graph classes. For example, We already know that the bound provided by relaxation (SDP_{GW}) is exact for even complete graphs (K_n with n even) [32]. This does not hold for odd complete graphs. We prove that the bound given by ($\mathsf{SDP5}$) is exact for all complete graphs.

Proposition 57. *For a complete graph, the optimal objective value of* ($\mathsf{SDP5}$) *is exact:* $Z^\star_{\mathsf{SDP5}} = w^\star$.

Proof. Since $Z^\star_{\mathsf{SDP}_{GW}}$ is exact for even complete graphs, $Z^\star_{\mathsf{SDP5}}$ is also exact by Proposition 55.
Let us now consider odd complete graphs. We take $n = |V|$ and we have for all $v \in V$,

- $d(v) = n - 1$,
- $I_v^+ = [\![\frac{n-1}{2}, n-1]\!] =: I^+$,
- $I_v^- = [\![1-n, \frac{1-n}{2}]\!] =: I^-$,
- $I_v = I_v^- \cup I_v^+ = I^- \cup I^+ =: I$.

We will build an optimal solution inheriting strong symmetries from the problem as well as from the input graph. One can observe that formulation ($\mathsf{SDP5}$) respects the following symmetry: if Y is an optimal solution of ($\mathsf{SDP5}$), then then Y' defined by $Y'^{vu}_{kl} = Y^{vu}_{(-k)(-l)}$ for all $(u,v,k,l) \in V^2 \times I^2$ is also an optimal solution of ($\mathsf{SDP5}$). This symmetry can be linked to our original orientation problem where if we have an optimal orientation for (MaxCut), then the reversed orientation is optimal as well. We therefore take Z', an optimal solution of ($\mathsf{SDP5}$) and Z'' defined by $Z''^{vu}_{kl} = Z'^{vu}_{(-k)(-l)}$ for all $(u,v,k,l) \in V^2 \times I^2$, which is also an optimal solution of ($\mathsf{SDP5}$). By linearity, we have that $Z = \frac{1}{2}(Z' + Z'')$ is yet another optimal solution verifying $Z^{vu}_{kl} = Z^{vu}_{(-k)(-l)}$ for all $(u,v,k,l) \in V^2 \times I^2$.

Let σ be any permutation of the set of vertices. By symmetry of the complete graph, the solution defined by $Y_{kl}^{\sigma(v)\sigma(u)} = Y_{kl}^{vu}$ for all $(u, v, k, l) \in V^2 \times I^2$ is still an optimal solution of (SDP5). By considering the set of all permutations S_n, and combining all solutions we still get an optimal solution Y by linearity where for all $(u, v, k, l) \in V^2 \times I^2$,

$$Y_{kl}^{vu} = \frac{1}{|S_n|} \sum_{\sigma \in S_n} Z_{kl}^{\sigma(v)\sigma(u)}.$$

We can notice that Y inherited $Y_{kl}^{vu} = Y_{(-k)(-l)}^{vu}$, $\forall (u, v, k, l) \in V^2 \times I^2$, and since we considered all permutations, for $(k, l) \in I^2$, Y_{kl}^{vu} does not depend on u and v. In other words, there are numbers $f(k, l)$ and $g(k)$ such that

- $Y_{kl}^{vu} = f(k, l)$, $\forall (u, v, k, l) \in V^2 \times I^2$ such that $u \neq v$,
- $Y_{kk}^{vv} = g(k)$, $\forall (v, k) \in V \times I$.

Furthermore, we know that

- $f(k, l) = f(-k, -l)$, $\forall (k, l) \in I^2$,
- $g(k) = g(-k)$, $\forall k \in I$.

Constraints (3.11a) lead to

$$\sum_{k \in I} g(k) = 1 = 2 \sum_{k \in I^+} g(k). \tag{3.12}$$

From (3.11c), we deduce that

$$\sum_{l \in I} f(k, l) = g(k), \ \forall k \in I. \tag{3.13}$$

Using equalities (3.11d), we can write that for $k \in I^+$

$$\begin{aligned} ((n-1) - k)g(k) &= (n-1) \sum_{k \in I^+} f(k, l), \\ &= (n-1) \left(g(k) - \sum_{k \in I^-} f(k, l) \right). \end{aligned}$$

which leads to

$$kg(k) = (n-1) \sum_{l \in I^-} f(k, l), \ \forall k \in I^+., \ \forall k \in I^+. \tag{3.14}$$

Considering (3.11f) for $k \in I^+$, we obtain

$$\begin{aligned}\sum_{l\in I} lf(k,l) &= (n-1)\sum_{l\in I^+} f(k,l) - (n-2)\sum_{l\in I^+} f(k,l) - g(k),\\ &= \sum_{l\in I^+} f(k,l) - g(k),\\ &= -\sum_{l\in I^-} f(k,l).\end{aligned}$$

which is equivalent to

$$\sum_{l\in I^-} -(l+1)f(k,l) = \sum_{l\in I^+} lf(k,l),\ \forall k \in I^+. \tag{3.15}$$

Combining (3.13) and (3.14), we deduce that

$$(n-1-k)\sum_{l\in I^-} f(k,l) = k\sum_{l\in I^+} f(k,l),\quad \forall k \in I^+. \tag{3.16}$$

Observe that (3.16) implies that when $k = n-1$ then $f(n-1,l) = 0$ for $l \in I^+$. By (3.15), we get that $f(n-1,l) = 0$ for $l \in I^-$. We can then assume in the rest of the proof that $k < n-1$.

Observe that the left side of (3.15) satisfies

$$\sum_{l\in I^-} -(l+1)f(k,l) \geq \frac{n-3}{2}\sum_{l\in I^-} f(k,l) = \frac{n-3}{2}\frac{k}{n-1-k}\sum_{l\in I^+} f(k,l),$$

where the last equality is induced by (3.16). Let k^{max} be the largest k such that $f(k,l) \neq 0$ for some l. We already know that $k^{max} \leq n-2$. The right side of (3.15) necessarily satisfies

$$\sum_{l\in I^+} lf(k,l) \leq k^{max}\sum_{l\in I^+} f(k,l).$$

Combining the two previous inequalities together with (3.15), we obtain

$$(k^{max} - \frac{n-3}{2}\frac{k}{n-1-k})\sum_{l\in I^+} f(k,l) \geq 0,\ \forall k \in [\![\frac{n-1}{2}, k^{max}]\!]. \tag{3.17}$$

By considering the case $k = k^{max}$ in (3.17), the sum $\sum_{l\in I^+} f(k^{max},l)$ is strictly positive, leading to $k^{max} - \frac{n-3}{2}\frac{k^{max}}{n-1-k^{max}} \geq 0$. In other words, we necessarily have $k^{max} \leq \frac{n+1}{2}$. This implies that $g(k) = 0$ and $f(k,l) = 0$ if either $k > \frac{n+1}{2}$ or $k < -\frac{n+1}{2}$ (we use here the fact that $g(k) = g(-k)$ and (3.13)).

Writing (3.15) and (3.16) for $k = \frac{n-1}{2}$ and $k = \frac{n+1}{2}$, we get the next 4 equations.

$$(n-3)\left[f\left(\frac{n+1}{2}, \frac{-1-n}{2}\right) + f\left(\frac{n+1}{2}, \frac{1-n}{2}\right)\right] = (n+1)\left[f\left(\frac{n+1}{2}, \frac{n+1}{2}\right) + f\left(\frac{n+1}{2}, \frac{n-1}{2}\right)\right] \tag{3.18}$$

$$(n-1)\, f\left(\frac{n+1}{2}, \frac{-1-n}{2}\right) + (n-3)\, f\left(\frac{n+1}{2}, \frac{1-n}{2}\right) = (n+1)\, f\left(\frac{n+1}{2}, \frac{n+1}{2}\right) + (n-1)\, f\left(\frac{n+1}{2}, \frac{n-1}{2}\right) \tag{3.19}$$

$$f\left(\frac{n-1}{2}, \frac{-1-n}{2}\right) + f\left(\frac{n-1}{2}, \frac{1-n}{2}\right) = f\left(\frac{n-1}{2}, \frac{n+1}{2}\right) + f\left(\frac{n-1}{2}, \frac{n-1}{2}\right) \tag{3.20}$$

$$(n-1)\, f\left(\frac{n-1}{2}, \frac{-1-n}{2}\right) + (n-3)\, f\left(\frac{n-1}{2}, \frac{1-n}{2}\right) = (n+1)\, f\left(\frac{n-1}{2}, \frac{n+1}{2}\right) + (n-1)\, f\left(\frac{n-1}{2}, \frac{n-1}{2}\right) \tag{3.21}$$

Substracting (3.18) from (3.19) leads to

$$f(\frac{n+1}{2}, \frac{n-1}{2}) = -f(\frac{n+1}{2}, \frac{-1-n}{2}).$$

By non-negativity of the f values, we deduce that

$$f(\frac{n+1}{2}, \frac{n-1}{2}) = f(\frac{n+1}{2}, \frac{-1-n}{2}) = 0 \text{ and } f(\frac{n+1}{2}, \frac{1-n}{2}) = \frac{n+1}{n-3} f(\frac{n+1}{2}, \frac{n+1}{2}).$$

Substracting $(n-1)\times$(3.20) from (3.21) leads in a similar way to

$$f(\frac{n-1}{2}, \frac{1-n}{2}) = f(\frac{n-1}{2}, \frac{n+1}{2}) = 0 \text{ and } f(\frac{n-1}{2}, \frac{-1-n}{2}) = f(\frac{n-1}{2}, \frac{n-1}{2}).$$

Using (3.14) and the previous observations we get that:

$$g(\frac{n+1}{2}) = 2\frac{n-1}{n+1} f(\frac{n+1}{2}, \frac{1-n}{2}) \text{ and } g(\frac{n-1}{2}) = 2f(\frac{n-1}{2}, \frac{-n-1}{2}).$$

Using the fact that

$$f(\frac{n+1}{2}, \frac{1-n}{2}) = f(\frac{n-1}{2}, \frac{-n-1}{2}) \text{ and } g(\frac{n+1}{2}) + g(\frac{n-1}{2}) = \frac{1}{2},$$

one can deduce that

$$f(\frac{n-1}{2}, \frac{-n-1}{2}) = \frac{n+1}{8n}.$$

Consequently, $g(\frac{n+1}{2}) = \frac{n-1}{4n}$ and $g(\frac{n-1}{2}) = \frac{n+1}{4n}$.

Remember that

$$\begin{aligned} Z^{\star}_{\mathsf{SDP5}} &= \frac{1}{2}\sum\nolimits_{v\in V}\sum\nolimits_{k\in I_v}|k|Y^{vv}_{kk} \\ &= \sum\nolimits_{v\in V}\sum\nolimits_{k\in I^+_v}kY^{vv}_{kk} \\ &= n\left(\frac{n+1}{2}g(\frac{n+1}{2}) + \frac{n-1}{2}g(\frac{n-1}{2})\right) \\ &= \frac{n^2-1}{4}, \end{aligned}$$

and we are done. □

The proof of Proposition 57 did not use the semidefinite positivity of solution Y, which implies that the linear relaxation of (MIP5) is exact. Some other graph families may share this property with the complete graphs. We will see in Section 3.5 that the wheel graphs numerically seem to be one of these families, it has not been shown so far. Bounds and approximated solutions for optimization problems are extremely relevant, especially in the case of NP-complete problems. However it is important to work on efficient exact formulations as well because the optimal objective value of the problem is sometimes needed. Furthermore, asymptotic complexity has its limits as we saw for example when analyzing the complexity of the simplex algorithm which remains state of the art notwithstanding its exponential worst-case complexity. Therefore the NP-completeness of MaxCut must not prevent us from trying to design efficient new exact formulations that may reveal competitive for example in the context of medium-sized instances.

3.4 Further mixed integer linear programming formulations

Goemans & Williamson's semidefinite approach of the problem led to efficient SDP based Branch & Bound solvers such as BiqMac [87] and BiqCrunch [70]. They use a branch-and-bound method featuring an improved semidefinite bounding procedure, mixed with a polyhedral approach.

In this section, we present three new exact formulations for the unweighted maximum cardinality cut problem with interesting computational performances.

3.4.1 A cleaned-up all-indicator-variables formulation

The first one stems from (MIP4) using the fact that for all $v \in V$, $\sum_{k\in I_v^-} y_k^v = 1-\sum_{k\in I_v^+} y_k^v$. Then we have $-d(v)(1-\sum_{k\in I_v^+} y_k^v) \leq \sum_{k\in I_v^-} \leq -\left\lceil \frac{d(v)}{2} \right\rceil (1-\sum_{k\in I_v^+} y_k^v)$ and we can delete all the variables y_k^v where k is negative. We obtain the following exact formulation.

$$
(\mathsf{MIP6}) \begin{cases}
Z^\star_{\mathsf{MIP6}} = \max \sum_{v\in V} \sum_{k\in I_v^+} k y_k^v \\
\text{s.t.} \\
\sum_{k\in I_v^+} y_k^v \leq 1, \ \forall v \in V, & (3.22a) \\
\sum_{uv\in E} \left(\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u \right) \leq \sum_{k\in I_v^+} k y_k^v - \left\lceil \frac{d(v)}{2} \right\rceil \left(1 - \sum_{k\in I_v^+} y_k^v\right), \ \forall v \in V, & (3.22b) \\
\sum_{uv\in E} \left(\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u \right) \geq \sum_{k\in I_v^+} k y_k^v - d(v)\left(1 - \sum_{k\in I_v^+} y_k^v\right), \ \forall v \in V, & (3.22c) \\
y_k^v \in \{0,1\}, \ \forall v \in V, \forall k \in I_v^+.
\end{cases}
$$

Proposition 58. *The optimal objective value of* (MIP6) *equals the maximum cardinality of a cut in the graph* G*:* $Z^\star_{\mathsf{MIP6}} = w^\star$.

Proof. A solution of (MIP4) naturally yields a solution of (MIP6) with the same optimal objective value, it suffices to omit the variables y_k^v such that $k > 0$. Therefore $Z^\star_{\mathsf{MIP4}} \leq Z^\star_{\mathsf{MIP6}}$.

Now take y be a solution of (MIP6), let us complete it into a solution for (MIP4). Take $v \in v$,

- if $\sum_{k\in I_v^+} = 1$, then we set $y_k^v \coloneqq 0, \ \forall k \in I_v^-$.

- If $\sum_{k\in I_v^+} = 0$ then we want y to satisfy

$$
\sum_{k\in I_v^-} k y_k^v = \sum_{uv\in E} \left(\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u \right) - \sum_{k\in I_v^+} k y_k^v.
$$

  Since y is a solution of (MIP6), we know that the right-hand side of the previous equation belongs to $[\![-d(v), -\left\lceil \frac{d(v)}{2} \right\rceil]\!] = I_v^-$ and we call it k_v. We then set $y_k^v \coloneqq 0, \ \forall k \in I_v^- \backslash \{k_v\}$ and $y_{k_v}^v \coloneqq 1$.

The resulting y satisfies (3.5a) and (3.5b) for v. Now that we have our feasible solution of (MIP4) y, we compute its objective value w.r.t. (MIP4):

$$
\begin{aligned}
\frac{1}{2}\sum_{v\in V}\sum_{k\in I_v}|k|y_k^v &= \frac{1}{2}\left(\sum_{\substack{v\in V\\ \sum_{k\in I_v^+} y_k^v=1}}\sum_{k\in I_v}|k|y_k^v + \sum_{\substack{v\in V\\ \sum_{k\in I_v^+} y_k^v=0}}\sum k\in I_v|k|y_k^v\right)\\
&= \frac{1}{2}\left(\sum_{\substack{v\in V\\ \sum_{k\in I_v^+} y_k^v=1}}\sum_{k\in I_v^+} ky_k^v - \sum_{\substack{v\in V\\ \sum_{k\in I_v^+} y_k^v=0}}\sum_{k\in I_v^-} ky_k^v\right)\\
&= \frac{1}{2}\sum_{v\in V}\left(\sum_{k\in I_v^+} ky_k^v - \sum_{k\in I_v^-} ky_k^v\right)\\
&= \frac{1}{2}\sum_{v\in V}\left(\sum_{k\in I_v^+} ky_k^v - \left(\sum_{uv\in E}\left(\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u\right) - \sum_{k\in I_v^+} ky_k^v\right)\right)\\
&= \frac{1}{2}\sum_{v\in V}\left(2\sum_{k\in I_v^+} ky_k^v - \sum_{uv\in E}\left(\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u\right)\right)\\
&= \sum_{v\in V}\sum_{k\in I_v^+} ky_k^v - \frac{1}{2}\sum_{v\in V}\sum_{uv\in E}\left(\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u\right)\\
&= \sum_{v\in V}\sum_{k\in I_v^+} ky_k^v.
\end{aligned}
$$

Since its coincides with its objective value w.r.t. (MIP6), we conclude that $Z^\star_{\mathsf{MIP6}} \leq Z^\star_{\mathsf{MIP4}}$ □

(MIP6) involves about half as many variables as (MIP4) and has generally better performance, detailed results can be found in Section 3.5. Expanding the idea of improving the performance, we can try to further reduce the number of variables for an exact formulation.

3.4.2 Aggregation of the variables

we can aggregate the variables y_k^v with $k \in I_v^+$ for a vertex v to form a variable x^v equal to $\sum_{k\in I_v^+} y_k^v$. For doing so, we need another variable z^v equal to $\sum_{k\in I_v^+} ky_k^v$ in order to keep the information about the signed imbalance of v important for the

objective function. We thus obtain the following exact formulation.

$$(\mathsf{MIP7})\begin{cases} Z^\star_{\mathsf{MIP7}} = \max \sum_{v\in V} z^v \\ \text{s.t.} \\ \left\lceil \frac{d(v)}{2} \right\rceil + \left\lfloor \frac{d(v)}{2} \right\rfloor x^v - z^v \leq \sum_{uv\in E} x^u \leq d(v) - z^v, \ \forall v \in V, & (3.23a) \\ \left\lceil \frac{d(v)}{2} \right\rceil x^v \leq z^v \leq d(v)x^v, \ \forall v \in V, & (3.23b) \\ x \in \{0,1\}^{|V|}, \ z \in \mathbb{R}^{|V|}. \end{cases}$$

Proposition 59. *The optimal objective value of* (MIP7) *equals the maximum cardinality of a cut in the graph* G*:* $Z^\star_{\mathsf{MIP7}} = w^\star$.

Proof. We take y a solution of (MIP6) and for each $v \in V$, we set $x^v \coloneqq \sum_{k\in I_v^+} y_k^v$ and $z^v \coloneqq \sum_{k\in I_v^+} k y_k^v$. Observe that (3.22a) yields that $x \in \{0,1\}^{|V|}$ and for $v \in V$, if $x_v = 0$, then $\left\lceil \frac{d(v)}{2} \right\rceil x^v = z^v = d(v)x^v = 0$ and if otherwise $x^v = 1$, then $z^v = \sum_{k\in I_v^+} k y_k^v \in I_v^+ = [\![\left\lceil \frac{d(v)}{2} \right\rceil, d(v)]\!]$. Thus, (x, z) satisfies (3.23b). Moreover for $v \in V$, (3.22b) and (3.22c) yields

$$\sum_{k\in I_v^+} k y_k^v - d(v)(1 - \sum_{k\in I_v^+} y_k^v) \leq \sum_{uv\in E} (\sum_{k\in I_v^+} y_k^v - \sum_{k\in I_u^+} y_k^u) \leq \sum_{k\in I_v^+} k y_k^v - \left\lceil \frac{d(v)}{2} \right\rceil (1 - \sum_{k\in I_v^+} y_k^v)$$

$$z^v - d(v)(1 - x^v) \leq \quad d(v)x^v - \sum_{uv\in E} x^u \quad \leq z^v - \left\lceil \frac{d(v)}{2} \right\rceil (1 - x^v)$$

$$\left\lceil \frac{d(v)}{2} \right\rceil - \left\lceil \frac{d(v)}{2} \right\rceil x^v - z^v \leq \quad \sum_{uv\in E} x^u - d(v)x^v \quad \leq d(v) - d(v)x^v - z^v$$

$$\left\lceil \frac{d(v)}{2} \right\rceil + \left\lfloor \frac{d(v)}{2} \right\rfloor x^v - z^v \leq \quad \sum_{uv\in E} x^u \quad \leq d(v) - z^v.$$

We obtain here (3.23a), consequently, (x, z) is a solution of (MIP7) with an objective value equal to that of y. Hence $Z^\star_{\mathsf{MIP6}} \leq Z^\star_{\mathsf{MIP7}}$.

Now take $(x, z) \in \{0,1\}^{|V|} \times \mathbb{R}^{|V|}$ be a solution of (MIP7), let us extract from it a solution y for MIP6 with the same objective value. Take $v \in V$,

- if $x^v = 1$, then we have $z^v \in [\![\left\lceil \frac{d(v)}{2} \right\rceil, d(v)]\!] = I_v^+$ and we set $y^v_{z_v} \coloneqq 1$ and $y_k^v \coloneqq 0, \ \forall k \in I_v^+ \backslash \{z_v\}$.
- If $x^v = 0$, then we have $z^v = 0$ and we set $y_k^v \coloneqq 0, \ \forall k \in I_v^+$.

Then y is a solution of MIP6 with the same objective value than that of (x, z). Thus $Z^\star_{\mathsf{MIP7}} \leq Z^\star_{\mathsf{MIP6}}$. □

(MIP7) involves $2n$ variables, half of which are integer variables and its performance is better than that of (MIP6) for many instances (see Section 3.5). This formulation can also be obtained using the linearization technique of Fred Glover [46] applied to the standard quadratic program modeling the maximum cut problem. This concept of aggregating all the variables concerning a vertex into a single one may seem a bit drastic: a compromise may be drawn.

3.4.3 Partial aggregation

One can also propose a third formulation somewhat in between (MIP6), i.e. no aggregation of variables, and (MIP7), i.e. total aggregation of the variables for each vertex. To do so, we partition the interval I_v^+ for each vertex $v \in V$. Let $\alpha > 1$, we parametrize such a partition with α defining the following sequences for each $v \in V$

$$\begin{cases} a_1^v = \left\lceil \frac{d(v)}{2} \right\rceil, \\ a_i^v = 1 + b_{i-1}^v, \text{ for } i > 1, \\ b_i^v = \min(\lfloor \alpha * a_i^v \rfloor, d(v)), \end{cases}$$

and compute k_v, the smallest integer such that $b_{k_v}^v = d(v)$. Then similarly to the formulations (MIP3), (MIP4) and (MIP6), we take a variable y_k^v for each vertex $v \in V$ and each $k \in [\![1, k_v]\!]$ whose interpretation is the following: $y_k^v = 1$ if and only if $z^v \in [\![a_k^v, b_k^v]\!]$. We therefore obtain the following exact formulation for all $\alpha > 1$.

$$\text{MIP8}[\alpha] \begin{cases} Z^\star_{\text{MIP8}} = \max \sum_{v \in V} z^v \\ \text{s.t.} \\ \left\lceil \frac{d(v)}{2} \right\rceil + \left\lfloor \frac{d(v)}{2} \right\rfloor x^v - z^v \leq \sum_{uv \in E} x^u \leq d(v) - z^v, \ \forall v \in V, & \text{(3.24a)} \\ \sum_{k=1}^{k_v} y_k^v = x^v, \ \forall v \in V, & \text{(3.24b)} \\ \sum_{k=1}^{k_v} a_k^v y_k^v \leq z^v \leq \sum_{k=1}^{k_v} b_k^v y_k^v, \ \forall v \in V, & \text{(3.24c)} \\ x \in [0,1]^V, \ y^v \in \{0,1\}^{k_v}, \ \forall v \in V, \ z \in \mathbb{R}^V. \end{cases}$$

Proposition 60. *The optimal objective value of* (MIP8[α]) *equals the maximum cardinality of a cut in the graph* G*:* $Z^\star_{\text{MIP8}[\alpha]} = w^\star$.

Proof. Let $\alpha > 1$, we take y a solution of (MIP6) and for each $v \in V$, we set

- $x^v \coloneqq \sum_{l \in I_v^+} y_l^v$,
- $z^v \coloneqq \sum_{l \in I_v^+} l y_l^v$,

$\triangleright$ $y'^v_k \coloneqq \sum_{l=a^v_k}^{b^v_k} y^v_l,\ \forall k \in [\![1, k_v]\!]$.

Thus, (x, y', z) naturally satisfies (3.24b) and satisfies (3.24a) for x and z were built exactly like a solution for (MIP7) from one of (MIP6) as detailed at the beginning of the proof of Proposition 59. Moreover we know by construction of the $(a^v_k)_{v,k}$ and $(b^v_k)_{v,k}$ that for any $v \in V$, $k \in [\![1, k_v]\!]$ and $l \in [\![a^v_k, b^v_k]\!]$, we have $a^v_k y^v_l \leq l y^v_l \leq b^v_k y^v_l$. Summing on all $k \in [\![1, k_v]\!]$ and all $l \in [\![a^v_k, b^v_k]\!]$, we obtain (3.24c). Consequently, (x, y', z) is a solution of (MIP8[α]) with an objective value equal to that of y. Hence $Z^\star_{\mathsf{MIP6}} \leq Z^\star_{\mathsf{MIP8}[\alpha]}$.

Now take (x, y, z) be a solution of (MIP8[α]), let us extract from it a solution y' for MIP6 with the same objective value. Take $v \in V$, since $(a_k)_k$ and $(b^v_k)_k$ both are finite increasing sequences, (3.24c) yields $\left\lceil \frac{d(v)}{2} \right\rceil x^v = a^v_1 \sum_{k=1}^{k_v} y^v_k \leq z^v \leq b^v_{k_v} \sum_{k=1}^{k_v} y^v_k = d(v) x^v$. Hence,

- if $x^v = 1$, then we have $z^v \in [\![\left\lceil \frac{d(v)}{2} \right\rceil, d(v)]\!] = I^+_v$ and we set $y'^v_{z_v} \coloneqq 1$ and $y'^v_l \coloneqq 0,\ \forall l \in I^+_v \backslash \{z_v\}$.
- If $x^v = 0$, then we have $z^v = 0$ and we set $y'^v_k \coloneqq 0,\ \forall l \in I^+_v$.

Then y' is a solution of MIP6 with the same objective value than that of (x, y, z). Thus $Z^\star_{\mathsf{MIP8}[\alpha]} \leq Z^\star_{\mathsf{MIP7}}$. □

Observation. Let us take a look at the number of indicator variables per variables w.r.t. to the partition of the sets I^+_v parametrized by α and used in formulation (MIP$_{[}\alpha$]). We defined for a vertex $v \in V$, k_v as the smallest integer such that such that $b^v_{k_v} = d(v)$, which implies that

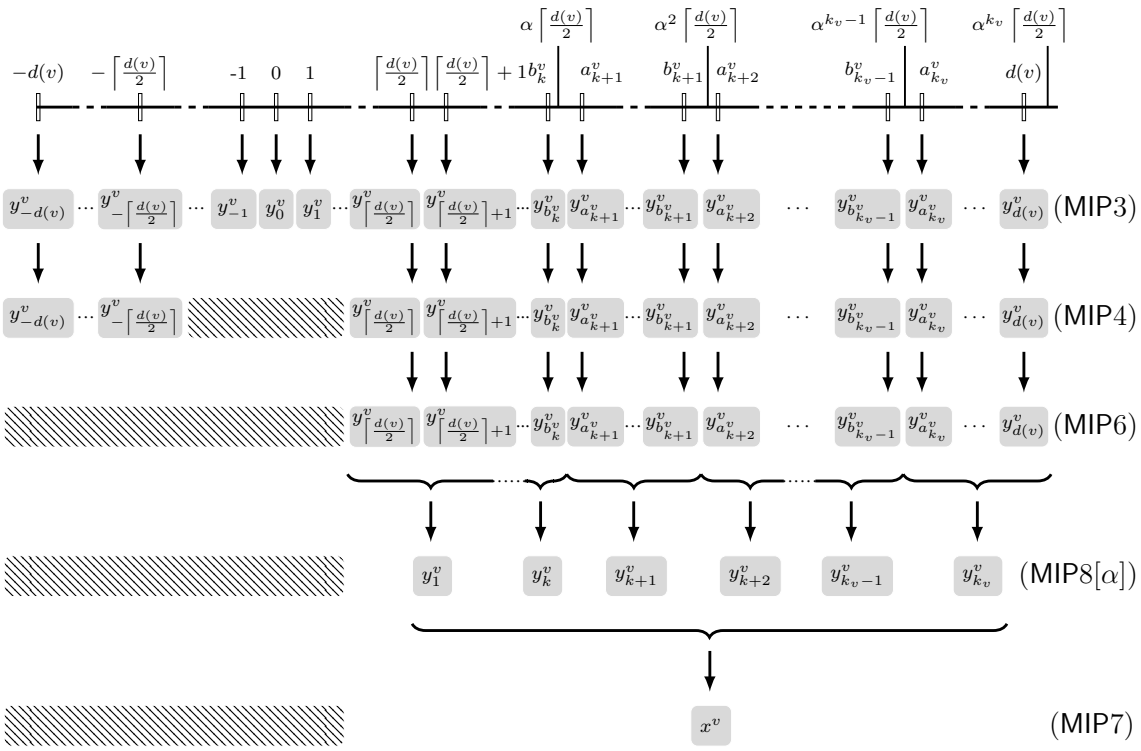
$$\begin{aligned} \alpha^{k_v - 1} \frac{d(v)}{2} &\leq d(v) \\ \alpha^{k_v - 1} &\leq 2 \\ (k_v - 1) \ln(\alpha) &\leq \ln(2) \\ k_v &\leq \frac{\ln(2)}{\ln(\alpha)} + 1 = \log_\alpha(2) + 1. \end{aligned}$$

Hence (MIP8[α]) has $O(n/\ln(\alpha))$ variables and $O(n)$ constraints. The closer α gets from 1, the higher the number of indicator variables in (MIP$_{[}\alpha$]) gets. For $\alpha = 1$, formulation (MIP8) is equivalent to formulation (MIP6), and for any graph G, there is a threshold for α (namely $\Delta_G / \left\lceil \frac{\Delta_G}{2} \right\rceil$) such that for any value of α beyond that threshold, formulation (MIP8) is equivalent to formulation (MIP7).

3.4.4 Weighing the exact formulations

We successively gave three mixed integer formulation all based on the same approach of MaxCut, before comparing their numerical performances, which shall be done in the next Section, let us analyze and compare them theoretically. The reader can find in figure 3.2 a schematized summary of the distribution of the indicator variables used to convey the information about the value of the signed imbalance of one vertex v in formulations (MIP4), (MIP6), (MIP7) and (MIP8[α]). In this figure, for each of the afore mentioned formulation, under an integer represented in the segment at the top is either a hashed region, an indicator variable or a brace, respectively meaning that the possible value for the signed imbalance of the vertex v is not taken into account, indicated by said variable or part of a subset of possible values indicated by a variable. The indicator variable under a brace will represent the sum of the variables over the brace. As soon as we start aggregating variables, although we reduce significantly the number of indicator variables in the formulation, it becomes necessary to add variables to store the value of the signed imbalance in order to be able to get the objective value of the solution, e.g. (MIP8[α]) and (MIP7).

Figure 3.2: Distribution of the indicator variables for one vertex on the segment $[\![-d(v), d(v)]\!]$ in formulations (MIP4), (MIP6), (MIP7) and (MIP8[α])



For the purpose of comparing performances, we now give a basic exact formulation for the unweighted maximum cut problem based on the triangle inequalities.

It involves one variable $x_{i,j}$ for each unordered pair of vertices $\{i,j\} \subset V$ $(i \neq j)$. Hence $O(n^2)$ variables and $O(n^3)$ constraints.

$$(\mathsf{MIP9}) \begin{cases} \max \sum_{ij \in E} x_{i,j} \\ \text{s.t.} \\ x_{i,j} + x_{j,k} + x_{i,k} \leq 2, \ \forall \{i,j,k\} \subset V, \ |\{i,j,k\}| = 3, \\ x_{i,j} + x_{j,k} - x_{i,k} \geq 0, \ \forall (i,j,k) \in V^3, \ |\{i,j,k\}| = 3, \\ x_{i,j} \in \{0,1\}, \ \forall \{i,j\} \subset V, \ i \neq j. \end{cases}$$

3.5 Computational experiments

Some numerical experiments have been conducted to evaluate the quality of the new SDP bound as long as the time performances of the new exact formulations presented in the previous Section.

3.5.1 Configuration and instances

The algorithms used for these computations were written in C/C++ calling COIN-OR's CSDP library to solve the semidefinite programs and IBM's ILOG CPLEX optimizer© for the linear and mixed integer programs; all have been performed with a processor 1.9GHzx4, 15.6GB RAM. In order to further the relevance of our comparison, we also give for each instance the running time (BC) of the semidefinite based solver BiqCrunch [70] compiled in Python and run on the same machine as the mixed integer programs.

Some of the graphs used for the computations are defined in Section 2.6 (complete graphs, grid graphs and randomly generated graphs). The others are denoted as follows:

- W_n: the wheel graph with n vertices, i.e. $n-1$ spokes. A wheel graph is a cycle to wich has been added one vertex connected to all the others: it is the "center" of the wheel and the edges incident to it are the "spokes" (e.g. Figure 3.3).
- Pe, Co, Oc, Do and Ic: the Petersen graph, the Coxeter graph, the octahedron, the dodecahedron and the icosahedron respectively (cf Figures 3.4, 3.5 and 3.6). Information about the platonic graphs can be found in [86],
- C_n: the cycle graph with n vertices,

Figure 3.3: W_9

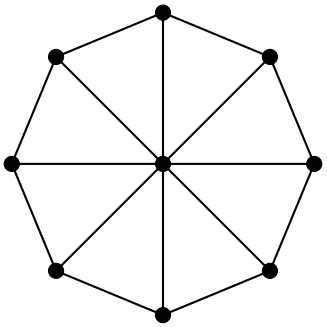


Figure 3.4: Petersen graph

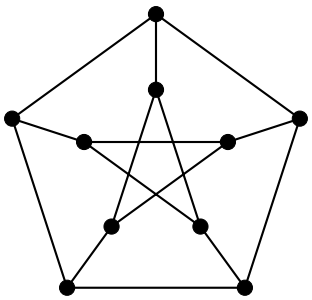


Figure 3.5: Coxeter graph

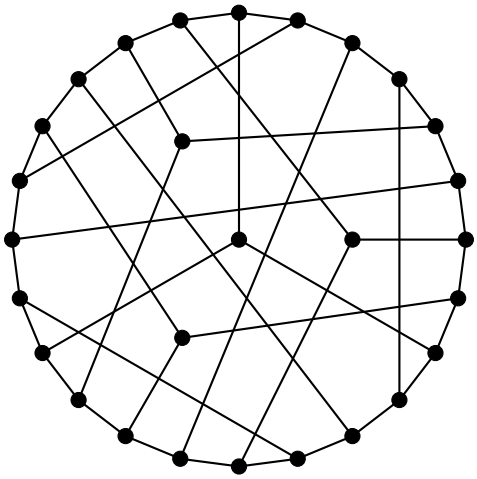
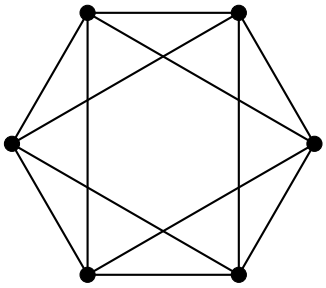
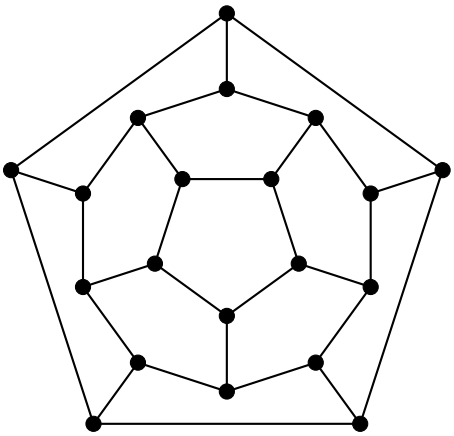


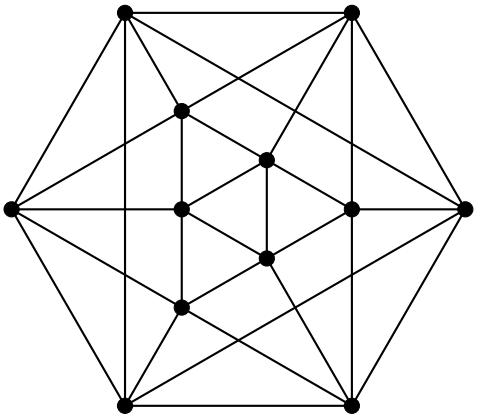
Figure 3.6: Platonic graphs



(a) Octahedron graph



(b) Dodecahedron graph



(c) Icosahedron graph

3.5.2 Results of the SDP formulations

Let us start with the results related to the new SDP bound. For each problem instance, we report $w^\star$ (the maximum cardinality of a cut), $Z^\star_{\mathsf{SDP}_{GW}}$ (the optimal objective value of (SDP_{GW})) and $Z^\star_{\mathsf{SDP5}}$ (the optimal objective value of (SDP5)). The first set of instances (Table 3.1) considered consists of two basic graph classes : odd complete graphs and wheel graphs. One can see that $Z^\star_{\mathsf{SDP5}} = w^\star$ for complete graphs as shown in Proposition 57. The bound seems to be exact for wheels (according to numerical experiments). We report in Table 3.2 results obtained on some well-known graphs: the Petersen graph, the Coxeter graph, the octahedron, the dodecahedron and the icosahedron, along with some toroidal grid graphs. The results reported in Table 3.3 were computed from randomly generated graphs.

The computational results from Tables 1-3 not only confirm the inequality proved in Proposition 55, but clearly point out that the quality of the new bound presented in the previous Sections is often significatively better than that of Goemans & Williamsons relaxation.

Table 3.1: Computational results of (SDP5) for complete graphs and wheel graphs

Instance	$w^\star$	$Z^\star_{SDP5}$	$Z^\star_{SDP_{GW}}$
K_5	6	6	6.25
K_7	12	12	12.25
K_{11}	30	30	30.25
W_5	6	6	6.25
W_8	10	10	10.614
W_{10}	13	13	13.809
W_{12}	16	16	16.979
W_{15}	21	21	21.875
W_{17}	24	24	25
W_{20}	28	28	29.566
W_{22}	31	31	32.703
W_{25}	36	36	37.5

Table 3.2: Computational results of (SDP5) for special graph classes

Instance	$w^\star$	$Z^\star_{SDP5}$	$Z^\star_{SDP_{GW}}$
Pe	12	12	12.5
Co	36	36.167	37.9
Oc	8	9	9
Do	24	25	26.18
Ic	20	21	21.708
C_3	2	2	2.25
C_5	4	4	4.523
C_7	6	6.125	6.653
C_9	8	8.25	8.729
C_{11}	10	10.383	10.777
C_{13}	12	12.463	12.811
C_{15}	14	14.523	14.836
C_{17}	16	16.58	16.855
C_{19}	18	18.621	18.87
C_{21}	20	20.653	20.883
C_{23}	22	22.685	22.893
C_{25}	24	24.709	24.901
$tG^2_{3,3}$	12	13.5	13.5
$tG^2_{3,4}$	20	20	21
$tG^2_{3,5}$	22	23.639	24.818
$tG^2_{4,5}$	36	36	38.09
$tG^2_{5,5}$	40	44.168	45.225
tG^3_3	54	60	60.75

Table 3.3: Computational results of (SDP5) for randomly generated graphs

Instance	$w^\star$	$Z^\star_{SDP5}$	$Z^\star_{SDP_{GW}}$
$R_{5,8}$	6	6	6.25
$R_{10,9}$	8	8	8.25
$R_{10,14}$	12	12	12.585
$R_{10,18}$	14	14	14.399
$R_{10,23}$	17	17	17.603
$R_{10,27}$	19	19	19.962
$R_{10,34}$	22	22	22.676
$R_{10,36}$	23	23	23.346
$R_{15,21}$	17	17	18.006
$R_{15,32}$	24	24.236	25.357
$R_{15,42}$	30	30.381	
$R_{15,53}$	36	36.567	37.39
$R_{20,19}$	16	16	16.679
$R_{20,38}$	29	29.202	30.682
$R_{20,57}$	43	43	44.757
$R_{30,44}$	37	37.31	39.005
$P_{5,7}$	5	5	5.432
$P_{5,9}$	6	6	6.25
$P_{10,10}$	8	8	8.409
$P_{10,12}$	10	10	10.715
$P_{10,18}$	13	13	13.932
$P_{10,24}$	16	16	16.992
$P_{20,11}$	9	9	9.25
$P_{20,16}$	15	15	15.25
$P_{20,27}$	21	21	22.495
$P_{20,41}$	30	30	31.289
$P_{20,54}$	36	36.207	38.131
$P_{25,35}$	28	28.091	29.705
$P_{25,52}$	39	39	40.614
$P_{25,69}$	46	46.446	48.468
$P_{30,8}$	7	7	7.25
$P_{30,17}$	15	15	15.25
$P_{30,42}$	33	33.037	34.412

3.5.3 Results of the MIP formulations

Let us now look at the running times of the Formulations (MIP9), (MIP4), (MIP6), (MIP7), (MIP8[1.5]), (MIP8[1.3]) and (MIP8[1.1]) and of the solver (BC) on several bigger instances found in Table 3.4 and Table 3.5. For the entries marked "$>$900", the running time exceeded 900s and the process was therefore interrupted and for the entries marked "-", the memory of the machine was full and the process was therefore interrupted.

First, one can see that the new formulations introduced in Section 3.4 perform much better than the classical triangular formulation (MIP9) for all the studied graph families except the general random graphs, and on all of these instances, there is one of our formulation that performs better than BiqCrunch does. More specifically, we can see that (MIP6) has generally better performance than (MIP4) and that for some instances, (MIP6) is drastically better than (MIP7), and for others, it is the other way around. Interestingly, we observe that being (MIP8) somewhat in between (MIP6) and (MIP7), there exists for almost each graph instance a value of α for which (MIP8[α]) has the shortest computing time. Practically, (MIP8[1.1]) seems to be the most robust of them.

In the next chapter, we take an interest in a precise family of polyhedra which arose in the research of valid inequalities for the set of solutions of formulation (MIP2) for the (MaxIm) problem. Computing the full description of the set of a part of the components of the feasible solutions for small instances lead to the isolation of the generic full description of a family of polyhedra.

Table 3.4: Running time for the MIP formulations (in seconds) for randomly generated graphs

graph	$w^\star$	MIP9	MIP4	MIP6	MIP7	MIP8[1.5]	MIP8[1.3]	MIP8[1.1]	BC	$\lvert V\rvert$	$\lvert E\rvert$
$R_{20,90}$	99	3	1	1	3	1	1	1	0	20	171
$R_{25,50}$	97	7	63	9	1	2	5	5	1	25	150
$R_{25,90}$	152	71	12	3	11	20	16	4	0	25	270
$R_{30,50}$	141	30	341	71	13	21	26	39	1	30	217
$R_{30,90}$	219	>900	44	21	144	111	99	23	0	30	391
$R_{40,25}$	136	29	>900	748	33	49	47	114	1	40	195
$P_{50,50}$	373	2	1	0	0	0	0	0	3	46	72
$P_{50,90}$	598	3	17	10	2	2	1	1	4	50	129
$P_{75,40}$	75	89	0	0	1	0	1	0	5	66	87
$P_{75,50}$	90	77	2	1	1	1	1	0	6	70	109
$P_{75,70}$	120	22	7	5	1	1	1	2	2	75	153
$P_{75,100}$	146	>900	>900	>900	172	>900	62	38	47	75	219
$P_{100,40}$	100	73	0	1	0	0	1	0	8	87	117
$P_{100,90}$	190	168	>900	>900	47	102	35	34	40	100	264
$P_{200,50}$	246	>900	>900	272	75	47	31	26	304	188	297
$P_{300,50}$	376	-	>900	>900	340	203	64	53	>900	282	447

Table 3.5: Running time for the MIP formulations (in seconds) for special graphs

graph	$w^\star$	MIP9	MIP4	MIP6	MIP7	MIP8[1.5]	MIP8[1.3]	MIP8[1.1]	BC	$\|V\|$	$\|E\|$
K_{50}	625	>900	2	1	0	0	0	0	0	50	1225
K_{100}	2500	>900	17	7	4	1	1	1	64	100	4950
K_{150}	5625	>900	179	42	29	4	3	3	166	150	11175
K_{175}	7656	>900	322	77	>900	>900	>900	3	5	175	15225
K_{200}	10000	>900	704	145	4	7	4	6	368	200	19900
K_{225}	12656	>900	>900	223	>900	>900	>900	8	11	225	25200
K_{300}	22500	-	>900	875	>900	>900	106	208	177	300	44850
tG_4^4	1024	>900	0	1	0	0	0	1	19	256	1024
$tG_{10,10}^2$	200	121	0	0	0	0	0	0	2	100	200
$tG_{8,15}^2$	232	>900	2	1	1	0	1	0	6	120	240
$tG_{11,12}^2$	252	>900	40	8	2	2	1	2	5	132	264
$tG_{15,20}^2$	580	>900	>900	>900	86	75	32	32	150	300	600
$tG_{20,20}^2$	800	>900	0	0	0	0	1	0	109	400	800
$tG_{100,100}^2$	20000	-	7	29	43	45	17	17	>900	10000	20000
W_{75}	111	26	2	3	1	1	0	1	4	75	148
W_{100}	148	58	2	7	5	3	0	1	11	100	198
W_{175}	261	>900	9	333	22	14	1	1	44	175	348
W_{250}	373	>900	61	>900	26	26	2	2	301	250	498
W_{400}	598	>900	62	>900	>900	>900	4	3	>900	400	798
W_{550}	823	-	>900	>900	>900	>900	5	5	>900	550	1098
W_{775}	1161	-	>900	>900	>900	>900	11	10	-	775	1548
W_{925}	1386	-	>900	>900	>900	626	15	15	-	925	1848
W_{1250}	1873	-	>900	>900	>900	>900	34	39	-	1250	2548
C_{100}	100	105	0	0	0	0	0	0	5	100	100
C_{175}	174	889	0	0	0	0	0	0	31	175	175
C_{250}	250	>900	0	0	0	0	0	0	71	250	250
C_{550}	550	-	1	0	0	0	0	0	>900	550	550
C_{1250}	1250	-	0	0	0	0	1	0	-	1250	1250
C_{1750}	1750	-	1	0	1	1	1	0	-	1750	1750
C_{3000}	3000	-	1	1	2	1	1	2	-	3000	3000
C_{5725}	5725	-	13	12	37	13	11	13	-	5725	5725
C_{9000}	9000	-	2	7	10	8	7	7	-	9000	9000

Chapter 4

Study of the polytopes related to the index of the lowest nonzero row of an assignment matrix

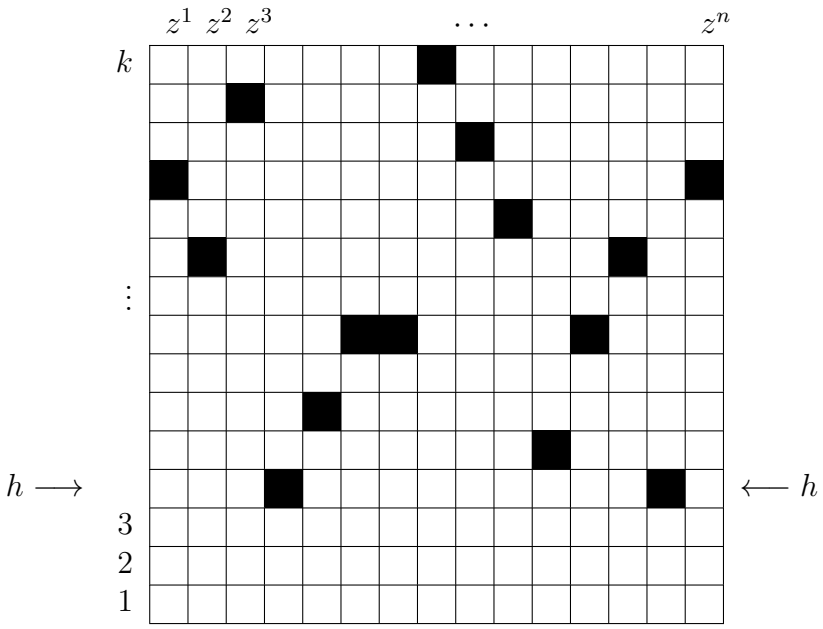


Figure 4.1: On this grid representing a k-by-n matrix, each column is a z^i variable and the height of its black cell is the value assigned to it. h thus corresponds to the first non-empty row of the grid starting from the bottom (in this case, $h = 4$).

When designing linear-based formulations for optimization prolems in the preceding chapters, we have seen the crucial role that valid inequalities for the set of solutions can play. A usually fruitful source of inequalities is an hyperplane representation of the convex hull of the solutions of a formulation.

4.0.1 Definition of the polytopes

Researching for new valid inequalities for the convex hull of the solutions of formulation (MIP2) presented in Chapter 2 lead to the study of the following family of polyhedra: for $(k,n) \in \mathbb{N}\backslash\{0\}^2$,

$$P = Conv\left\{ \left(\begin{bmatrix} y_k^1 & y_k^2 & \cdots & y_k^n \\ \vdots & \vdots & \ddots & \vdots \\ y_2^1 & y_2^2 & \cdots & y_2^n \\ y_1^1 & y_1^2 & \cdots & y_1^n \end{bmatrix}, h \right) \in M_{k\times n}(\{0,1\}) \times \mathbb{N}\backslash\{0\} \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1, \ \forall i \in [\![1,n]\!], \\ h = \min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i, \end{array} \right\}.$$

The polytope P is best interpreted in the context of a combinatorial optimization problem involving $n \in \mathbb{N}\backslash\{0\}$ variables z^i each of which can be assigned an integer number in $[\![1,k]\!]$. Then the interpretation given to the variables y_l^i is the following: $y_l^i = 1$ if and only if $z^i = l$ and then $h \equiv \min_{i=1}^{n} z^i$. The polytope P is consequently naturally related to these problems. The matrix (y_j^i) can be seen as a $\{0,1\}$ assignment matrix where each column contains exactly one coefficient equal to 1 while h denotes the index of the lowest row that is not identically equal to the zero row (cf Figure 4.1).

Another variant of P is obtained by considering the index of the highest row that is not identically equal to the zero row (cf Figure 4.2). In this case we get the polytope

$$P' = Conv\left\{ \left(\begin{bmatrix} x_k^1 & x_k^2 & \cdots & x_k^n \\ \vdots & \vdots & \ddots & \vdots \\ x_2^1 & x_2^2 & \cdots & x_2^n \\ x_1^1 & x_1^2 & \cdots & x_1^n \end{bmatrix}, g \right) \in M_{k\times n}(\{0,1\}) \times \mathbb{N}\backslash\{0\} \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} x_l^i = 1, \ \forall i \in [\![1,n]\!], \\ g = \max_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l x_l^i, \end{array} \right\}.$$

Polytopes P and P' naturally appear in the context of several other combinatorial optimization.

The rest of this chapter is organized as follows. First we present some examples of problems where polytope P or its variants arises in Section 4.1. Then we give a complete linear description of P in Section 4.2. Then we show that the separation problem with respect to P can be solved in polynomial time in Section 4.3. Finally we give similar results for some polyhedra which ar variants of P in Section 4.4.

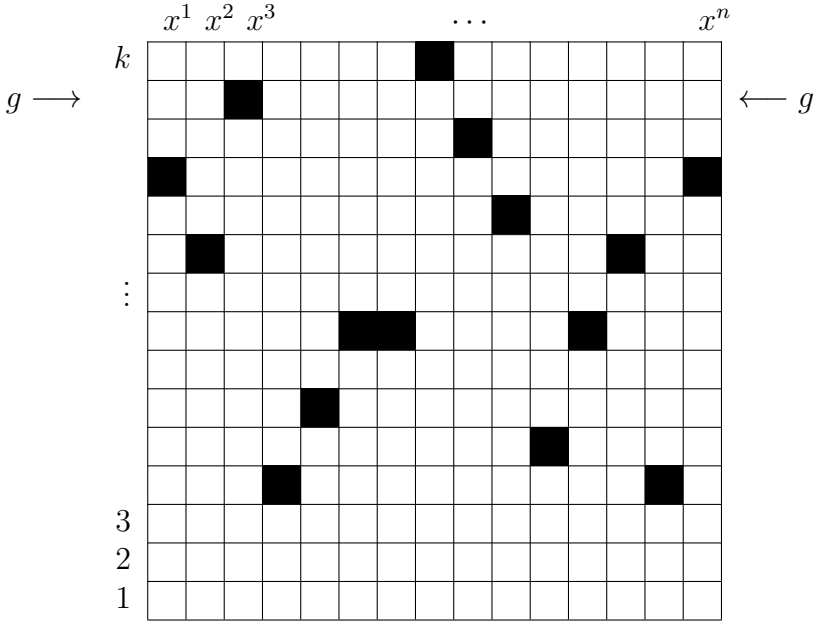


Figure 4.2: On this grid representing a k-by-n matrix, each column is a x^i variable and the height of its black cell is the value assigned to it. The value of g thus corresponds to the first non-empty row of the grid starting from the bottom (in this case, $g = k - 1$).

4.1 Motivations

4.1.1 MaxIm

As previously mentioned, polytope P arises in the study of (MaxIm) if we take a closer look at formulation (MIP2). If we extend the set of possible values for the signed imbalance of a vertex to $[\![-\Delta_G, \Delta_G]\!]$ and rename the indicator variables, then (MIP2) can be rewritten as follows:

$$\begin{cases} \max\ h \\ \quad \text{s.t. } h \leq \sum_{l=-k}^{k} |l| t_l^v,\ \forall v \in V, \\ \qquad \sum_{l=-k}^{k} t_l^v = 1,\ \forall v \in V, \\ \qquad \sum_{l=-k}^{k} l t_l^v = B_v x,\ \forall v \in V, \\ \qquad x \in [-1;1]^{|E|},\ t \in M_{n \times 2k+1}(\{0,1\}),\ h \in \mathbb{R}. \end{cases}$$

Where $k \coloneqq \Delta_G$. Introducing variables $y_l^v = t_{-l}^v + t_l^v,\ \forall (v,l) \in V \times [\![1,k]\!]$, it becomes

$$\begin{cases} \max\ h \\ \text{s.t. } \sum_{l=-k}^{k} l t_l^v = B_v x,\ \forall v \in V, \\ \qquad y_l^v = t_{-l}^v + t_l^v,\ \forall (v,l) \in V \times [\![1,k]\!], \\ \qquad x \in [-1;1]^{|E|},\ (y,h) \in P,\ t \in M_{n \times 2k+1}(\{0,1\}). \end{cases}$$

Considering this last formulation, the polytope P clearly appears and is therefore closely related to the problem (MaxIm) through formulation (MIP2). The polyhedral analysis of P detailed in this Chapter lead to the unveiling of the family of inequalities (2.6) which participates int the strengthening of (MIP2)'s linear relaxation within the framework of a cutting-plane algorithm (see, e.g. [29] and [97]).

4.1.2 Minimum-span frequency assignment

Let us consider the *minimum-span frequency-assignment problem* which is a variant of the NP-hard *frequency-assignment problem* [79]. The input is a graph $G = (V, E)$ that is generally called the *interference graph* in which the nodes that are antennas are connected if their signals can interfere with one another. The frequency assignment problem consists in assigning a frequency f from a set of available frequencies F to each vertex $v \in V$ in such a way that each pair of antennas $uv \in E$ that may interfere with one another are assigned different frequencies. Frequencies can be seen as ordered integer numbers. To reduce interferences, one might impose stronger constraints: a minimum separation between the frequencies assigned to u and v is required. If frequency i is assigned to u and j is assigned to v, then $|i-j| \geq s_{uv}$ where s_{uv} is a given number. This constraint differentiate it from a classic graph colouring problem. The minimum-span frequency-assignment problem (or MS-FAP) consists in assigning frequencies to nodes taking into account the separation requirements and minimizing the difference between the largest assigned number (frequency) and the smallest assigned number (see, e.g., [69]).

If we consider that $V = \{v_1, \cdots, v_n\}$, $F = [\![1,k]\!]$ where k is an upper bound of the minimum-span, then we obtain the following formulation for MS-FAP

$$\begin{cases} \min\ g \\ \text{s.t. } x_l^i + x_{l'}^j \leq 1,\ \forall (i,j,l,l') \in [\![1,n]\!]^2 \times [\![1,k]\!]^2 \text{ such that } v_i v_j \in E, |l-l'| < s_{v_i v_j} \\ \qquad (x,g) \in P',\ x \in M_{k \times n}(\{0,1\}). \end{cases}$$

where the interpretation of the x variable is the following: $x_l^i = 1$ if and only if the

frequency l is assigned to the antenna v_i.

We find again our polytope, we start to grasp the wide variety of optimization problems related to it.

4.1.3 Minimum makespan scheduling

Another example is the *minimum makespan scheduling*, which is a central problem in the scheduling area (see [94]). Given a set J of jobs, a set M of machines that can all process at most one job at a time, and the time $t^{i,j} \in \mathbb{N}$ taken to process job $j \in J$ on machine $i \in M$, the goal of the minimum makespan scheduling problem is to assign a machine $p \in M$ for each job $j \in J$ so as to minimize the *makespan*, i.e. the maximum processing time of any machine. Several approximation schemes have been developed to deal with this NP-hard problem [45], e.g. [63] and [64]. Since the processing times are integers, the timeline is discretized in identical units of time, namely days. We consider here the variant where all the machines in M are identical (or IM-MMS) and preemptions (i.e. interruption of a task being processed) are not allowed. In other words, for any job $j \in J$, $t^{i,j} = t^j$, $\forall i \in M$. In this case, assigning a machine to each job is equivalent to assigning a day d' to be the last day of processing this job, which also determines the first day d of processing and will therefore be processed by a machine free during the period $[d, d']$. Now to make a formulation for IM-MMS with the set of jobs $J = [\![1, n]\!]$ and $m \in \mathbb{N}\backslash\{0\}$ identical machines, we take $k = \sum_{i=1}^{n} t^i$ and the variable $x \in M_{k\times n}(\{0, 1\})$ whose interpretation is the following: $x_l^i = 1$ if and only if the processing of the job i ends on the day l. Then we have the following formulation for IM-MMS

$$\begin{cases} \min g \\ \quad \text{s.t. } \sum_{l=1}^{k} l x_l^i \geq t^i, \ \forall i \in [\![1, n]\!], \\ \qquad \sum_{i=1}^{n} \sum_{l'=l}^{\min(l+t^i-1,k)} x_{l'}^i \leq m, \ \forall l \in [\![1, k]\!], \\ \qquad (x, g) \in P', \ x \in M_{k\times n}(\{0, 1\}). \end{cases}$$

For a job $i \in [\![1, n]\!]$, $\sum_{l=1}^{k} l x_l^i \geq t^i$ ensures that its processing ends after enough time has passed for i to be processed, and for a day $l \in [\![1, k]\!]$, $\sum_{i=1}^{n} \sum_{l'=l}^{\min(l+t^i-1,k)} x_{l'}^i \leq m$ ensures that no more than m jobs are being processed. Some additional constraints can be added to this formulation such as a necessary precedence or release time. If we want a job $i \in [\![1, n]\!]$ to be processed before another job $j \in [\![1, n]\!]\backslash\{i\}$ starts processing, we add the constraint $\sum_{l=1}^{k} l x_l^i \leq \sum_{l=1}^{k} l x_l^j - t^j$. If we want a job $i \in [\![1, n]\!]$ to be processed before (resp. on, after) a day $d \in [\![1, k]\!]$, we add the constraint $\sum_{l=1}^{d-1} x_l^i = 1$ (resp. $x_d^i = 1$, $\sum_{l=d+1}^{k} x_l^i = 1$). The objective function can also be any linear function depending on the variable x and gs.

Polytope P also appears in the context of the maximum clique problem. A discretized formulation is proposed in [80] where a variable w_q^i indicates whether the vertex i belongs to a clique of size q. These variables are of course linked to standard vertex variables ($x_i = 1$ if i belongs to the maximum clique). The problem is then equivalent to maximizing q such that $w_q^i = 1$ for some i. This is again related to polytope P.

More generally, various combinatorial optimization problems where discretization techniques are used can benefit from a description of either P or some of its variants.

4.2 A full description of P

4.2.1 Definition of the hyperplanes

Let us define a set of inequalities that will prove to be an hyperplane representation of P.

$$(\tilde{P}) \begin{cases} \sum_{l=1}^{k} y_l^i = 1, \ \forall i \in [\![1,n]\!], & (4.1a) \\ \sum_{l=2}^{k} \sum_{i=1}^{n} \lambda_l^i y_l^i \geq h-1, \ \forall \lambda \in \Lambda, & (4.1b) \\ \sum_{l=1}^{h_{\max}-1} \sum_{i=1}^{n} (l-h_{\max}) y_l^i + h_{\max} \leq h, \ \forall h_{\max} \in [\![1,k]\!], & (4.1c) \\ y_l^i \geq 0, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!], \ h \in \mathbb{R}, \end{cases}$$

where

$$\Lambda = \left\{ \lambda = (\lambda_l^i)_{(i,l)\in[\![1,n]\!]\times[\![1,k]\!]} \in \mathbb{N}^{nk} \;\middle|\; \begin{array}{l} \lambda_{l+1}^i \geq \lambda_l^i, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k-1]\!] \\ \sum_{i=1}^{n} \lambda_l^i = l-1, \ \forall l \in [\![1,k]\!] \end{array} \right\}.$$

We clearly recognize the equalities (4.1a), they are essential to the concept of indicator variables. The inequalities (4.1c) form a family of constraints which cardinality is k, hence linear in k and n, which is not the case of the family formed by (4.1b).

Any element λ of Λ can be constructed as follows: we start with $\lambda_1^i = 0, \ \forall i \in [\![1,n]\!]$, choose an index $i_2 \in [\![1,n]\!]$ and set $\lambda_2^{i_2} = 1$ and $\lambda_2^i = 0, \ \forall i \in [\![1,n]\!] \backslash \{i_2\}$. And we proceed like this for $l = 2, \cdots, k$, we choose an index $i_l \in [\![1,n]\!]$ and set $\lambda_l^{i_l} = \lambda_{l-1}^{i_l} + 1$ and $\lambda_l^i = \lambda_{l-1}^i, \ \forall i \in [\![1,n]\!] \backslash \{i_l\}$. Schematically, it can be seen as creating n piles of coins. Starting with an empty table, at each step (a step being $l \leftarrow l+1$), we add one coin on one pile and we read each λ_l^i as the height of the corresponding pile.

4.2.2 Proof of the hyperplane representation

In order to show that $\tilde{P}$ is indeed a proper hyperplane representation of P, we are first going to prove that $\tilde{P}$ contains P. Then to show the opposite inclusion, we will prove that any facet-defining inequality of P is equivalent to one of the inequality listed in the definition of $\tilde{P}$. For the proof to be comfortably readable, those steps are presented as separate and thus independent lemmas. We start with the first inclusion.

Lemma 61.

$$P \subseteq \tilde{P}$$

Proof. Since P is the convex hull of integer points, it suffices to show that each of those points satisfies all the inequalities in $\tilde{P}$. Let $(y, h) \in P$ be one of those points, that is to say $(y, h) \in M_{k\times n}(\{0,1\}) \times \mathbb{N}\backslash\{0\}$, $\sum_{l=1}^{k} y_l^i = 1,\ \forall i \in [\![1, n]\!]$ and $h = \min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i$. We firstly show that $\sum_{l=2}^{k}\sum_{i=1}^{n} \lambda_l^i y_l^i \geq h - 1,\ \forall \lambda \in \Lambda$. For all $i \in [\![1, n]\!]$, there exists $l_i \in [\![h, k]\!]$ such that $y_{l_i}^i = 1$. Now since for a given $i \in [\![1, n]\!]$, λ_l^i increases with l, we have

$$\sum_{l=2}^{k}\sum_{i=1}^{n} \lambda_l^i y_l^i = \sum_{i=1}^{n} \lambda_{l_i}^i \geq \sum_{i=1}^{n} \lambda_h^i = h - 1.$$

Now we take $h_{\max} \in [\![1, k]\!]$, and we show that $h_{\max} - \sum_{l=1}^{h_{\max}-1}(h_{\max} - l)\sum_{i=1}^{n} y_l^i \leq h$. We have

$$\begin{aligned} h_{\max} - \sum_{l=1}^{h_{\max}-1} (h_{\max} - l)\sum_{i=1}^{n} y_l^i &= h_{\max} + \sum_{l=1}^{k} \min(l - h_{\max}, 0)\sum_{i=1}^{n} y_l^i \\ &= h_{\max} + \sum_{l=1}^{k} \min(l - 1, h_{\max} - 1)\sum_{i=1}^{n} y_l^i - n(h_{\max} - 1) \\ &= \sum_{i=1}^{n}\sum_{l=2}^{k} \min(l - 1, h_{\max} - 1) y_l^i + 1 - (n - 1)(h_{\max} - 1) \end{aligned}$$

There exists $i^* \in [\![1,n]\!]$ such that $y_h^{i^*} = 1$, then

$$\begin{aligned}
\sum_{i=1}^{n}\sum_{l=2}^{k} \min(l-1, h_{\max}-1) y_l^i &= \sum_{\substack{i=1\\ i\neq i^*}}^{n}\sum_{l=2}^{k} \min(l-1, h_{\max}-1) y_l^i &&+ \sum_{l=2}^{k} \min(l-1, h_{\max}-1) y_l^{i^*} \\
&\leq \sum_{\substack{i=1\\ i\neq i^*}}^{n}\sum_{l=2}^{k} (h_{\max}-1) y_l^i &&+ \sum_{l=2}^{k} (l-1) y_l^{i^*} \\
&\leq \sum_{\substack{i=1\\ i\neq i^*}}^{n} (h_{\max}-1) &&+ h - 1 \\
&= (n-1)(h_{\max}-1) &&+ h - 1.
\end{aligned}$$

□

Now to prove that P coincides with $\tilde{P}$, we show that all facet-defining inequalities for P are among those defining $\tilde{P}$. To that purpose, we shall often use the fact that a facet of P cannot be contained by another distinct facet of P otherwise it would be contained by the intersection of two distinct hyperplanes of $\mathbb{R}^{dim(P)}$, being therefore of dimension at most $dim(P)-2$. Consequently, for any two distinct valid inequalities for P, if one is facet-defining, then there exists a point of P saturating it and not the other. For example, in Figure 1.4, we know that $y \geq -3x + 14$ does not define a facet of P_{ex} because all the points of P_{ex} saturating it (that is $(3,5)$) saturates $y \leq 3x - 4$ which is a valid inequality for P_{ex}. Two inequalities are said to be *equivalent* if one can be obtained from the other by multiplying it by a non-zero scalar and adding a combination of equations of the type $\sum_{l=1}^{k} y_l^i = 1$. The idea is that any half-space of $\mathbb{R}^{kn+1}$ can be defined by an inequality written without loss of generality as $\sum_{l=1}^{k}\sum_{i=1}^{n} \beta_l^i y_l^i + \gamma \geq \alpha h$, where $\beta_l^i \in \mathbb{R},\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ (\gamma, \alpha) \in \mathbb{R} \times \{-1,0,1\}$. We can go further and say that any two inequalities so written and defining the same half-space share their α coefficient. We can then partition the set of all half-spaces into three cases: $\alpha = -1$, $\alpha = 0$ and $\alpha = 1$. Those three cases shall then be dealt with by the three following lemmas.

Lemma 62. *Let*

$$\sum_{l=1}^{k}\sum_{i=1}^{n} \beta_l^i y_l^i + \gamma \geq 0, \tag{4.2}$$

be a facet-defining inequality of P, with $\beta_l^i \in \mathbb{R},\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ \gamma \in \mathbb{R}$. Then there exists $(i,l) \in [\![1,n]\!] \times [\![1,k]\!]$ such that (4.2) is equivalent to $y_l^i \geq 0$.

For an extreme point (y,h) of P and $(\tilde{i}, \tilde{l}, \tilde{l}') \in [\![1,n]\!] \times [\![1,k]\!]^2$, such that $y_{\tilde{l}}^{\tilde{i}} = 1$ and $\tilde{l} \neq \tilde{l}'$, we denote by $(y_{\tilde{l} \xrightarrow{\tilde{i}} \tilde{l}'}, h_{\tilde{l} \xrightarrow{\tilde{i}} \tilde{l}'})$ the extreme point (y', h') of P such that $y'^i_l =$

$y^i_l,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!] \backslash \{(\tilde{i},\tilde{l}),(\tilde{i},\tilde{l}')\},\ y^{\tilde{i}}_{\tilde{l}} = 0,\ y^{\tilde{i}}_{\tilde{l}'} = 1$ and $h' = \min_{i \in [\![1,n]\!]} \sum_{l=1}^{k} l y'^i_l$. For $\tilde{l} \in [\![1,k]\!]$, we denote by $(y_{\to \tilde{l}}, \tilde{l})$ the point of P such that $y^i_{\tilde{l}} = 1,\ \forall i \in [\![1,n]\!]$ and $y^i_l = 0,\ \forall (i,l) \in [\![1,n]\!] \times ([\![1,k]\!] \backslash \{\tilde{l}\})$.

Proof. First, since for each $i \in [\![1,n]\!]$, we have $\sum_{l=1}^{k} y^i_l = 1$, we can replace y^i_1 by $1 - \sum_{l=2}^{k} y^i_l$ and get new coefficients $\tilde{\beta}^i_1 = 0$ and $\tilde{\beta}^i_l = \beta^i_l - \beta^i_1,\ \forall l \in [\![2,k]\!]$ and a new $\tilde{\gamma} = \gamma + \sum_{i=1}^{n} \beta^i_1$. Hence, without loss of generality, we can assume that $\beta^i_1 = 0$ for all $i \in [\![1,n]\!]$. Suppose that the facet defined by (4.2) is not equivalent to a facet defined by an inequality of the type $y^i_l \geq 0$. If we take $(\tilde{i},\tilde{l}) \in [\![1,n]\!] \times [\![1,k-1]\!]$, we know that there exists an extreme point (y,h) of P saturating (4.2) and such that $y^{\tilde{i}}_{\tilde{l}} = 1$, otherwise all the extreme points saturating (4.2) would saturate $y^{\tilde{i}}_{\tilde{l}} \geq 0$ thus contradicting the fact that (4.2) is facet-defining and not equivalent to $y^i_l \geq 0$ for some $(i,l) \in [\![1,n]\!] \times [\![1,k]\!]$. Since $(y',h') = (y_{\tilde{l} \xrightarrow{\tilde{i}} \tilde{l}+1}, h_{\tilde{l} \xrightarrow{\tilde{i}} \tilde{l}+1}) \in P$, we have $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y^i_l + \gamma = 0$ and $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y'^i_l + \gamma \geq 0$ which yields $\beta^{\tilde{i}}_{\tilde{l}+1} \geq \beta^{\tilde{i}}_{\tilde{l}}$. Similarly, taking an extreme point (y,h) of P saturating (4.2) and such that $y^{\tilde{i}}_{\tilde{l}+1} = 1$ and $(y',h') = (y_{\tilde{l}+1 \xrightarrow{\tilde{i}} \tilde{l}}, h_{\tilde{l}+1 \xrightarrow{\tilde{i}} \tilde{l}}) \in P$, we have $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y^i_l + \gamma = 0$ and $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y'^i_l + \gamma \geq 0$, yielding $\beta^{\tilde{i}}_{\tilde{l}} \geq \beta^{\tilde{i}}_{\tilde{l}+1}$. Hence, for all $(i,l) \in [\![1,n]\!] \times [\![1,k-1]\!]$, we have $\beta^i_l = \beta^i_{l+1}$, in other words, $\beta^i_l = 0,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!]$, and (4.2) is not facet-defining. □

Lemma 63. *Let*

$$\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y^i_l + \gamma \geq h, \tag{4.3}$$

be a facet-defining inequality of P, with $\beta^i_l \in \mathbb{R},\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ \gamma \in \mathbb{R}$. Then there exists $\lambda \in \Lambda$ such that (4.3) is equivalent to $\sum_{l=2}^{k} \sum_{i=1}^{n} \lambda^i_l y^i_l \geq h - 1$.

Proof. Again, without loss of generality, we assume that $\beta^i_1 = 0$ for all $i \in [\![1,n]\!]$. For an $(\tilde{i},\tilde{l}) \in [\![1,n]\!] \times [\![1,k-1]\!]$, there exists an extreme point (y,h) of P saturating (4.3) and such that $y^{\tilde{i}}_{\tilde{l}} = 1$. Since $(y',h') = (y_{\tilde{l} \xrightarrow{\tilde{i}} \tilde{l}+1}, h_{\tilde{l} \xrightarrow{\tilde{i}} \tilde{l}+1}) \in P$, we have $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y^i_l + \gamma = h$ and $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y'^i_l + \gamma \geq h' \geq h$ which yields $\beta^{\tilde{i}}_{\tilde{l}+1} \geq \beta^{\tilde{i}}_{\tilde{l}}$. Hence for all $i \in [\![1,n]\!]$, $(\beta^i_l)_l$ is increasing with l and therefore non-negative since $\beta^i_1 = 0,\ \forall i \in [\![1,n]\!]$.
If we consider the point $(y,h) = (y_{\to 1}, 1)$, we obtain that $\gamma \geq 1$. If we now consider an extreme point (y,h) of P saturating (4.3) and such that $y^{\tilde{i}}_1 = 1$ for an $\tilde{i} \in [\![1,n]\!]$, we get $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y'^i_l + \gamma = h = 1$. Since both the β^i_l and the y^i_l are non-negative, then so is $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta^i_l y'^i_l$. Hence $\gamma \leq 1$, yielding $\gamma = 1$.

Considering $(y_{\to l}, l) \in P$ for $l \in [\![1,k]\!]$, we obtain $\sum_{i=1}^{n} \beta^i_l \geq l - 1$. Let us show by induction on l that $\sum_{i=1}^{n} \beta^i_l = l - 1,\ \forall l \in [\![1,k]\!]$. Our induction is already initialized by $\sum_{i=1}^{n} \beta^i_1 = 0$. We suppose that for a $\tilde{l} \in [\![1,k-1]\!]$, we have

$\sum_{i=1}^{n} \beta_{\tilde{l}}^{i} = \tilde{l} - 1$ and show that $\sum_{i=1}^{n} \beta_{\tilde{l}+1}^{i} = \tilde{l}$. Suppose that all the extreme points (y, h) of P saturating (4.3) and such that $y_{\tilde{l}+1}^{i} = 1$ for some $i \in [\![1, n]\!]$ verify $h \leq \tilde{l}$. Then for each $\tilde{i} \in [\![1, n]\!]$, take one of those extreme saturating points such that $y_{\tilde{l}+1}^{\tilde{i}} = 1$ and let $(y', h') = (y_{\tilde{l}+1 \xrightarrow{\tilde{i}} \tilde{l}}, h_{\tilde{l}+1 \xrightarrow{\tilde{i}} \tilde{l}}) \in P$. Since $\beta_{\tilde{l}+1}^{\tilde{i}} \geq \beta_{\tilde{l}}^{\tilde{i}}$, we have $h' - 1 \leq \sum_{l=1}^{k} \sum_{i=1}^{n} \beta_l^i y_l'^i \leq \sum_{l=1}^{k} \sum_{i=1}^{n} \beta_l^i y_l^i = h - 1$, and since $h \leq \tilde{l}$, we have $h = h'$ and therefore, $\sum_{l=1}^{k} \sum_{i=1}^{n} \beta_l^i y_l'^i = \sum_{l=1}^{k} \sum_{i=1}^{n} \beta_l^i y_l^i = h - 1$, yielding $\beta_{\tilde{l}+1}^{\tilde{i}} = \beta_{\tilde{l}}^{\tilde{i}}$. Thus $\sum_{i=1}^{n} \beta_{\tilde{l}+1}^{i} = \sum_{i=1}^{n} \beta_{\tilde{l}}^{i} = \tilde{l} - 1$ which contradicts $\sum_{i=1}^{n} \beta_{\tilde{l}+1}^{i} \geq \tilde{l}$. So there exists $\tilde{i} \in [\![1, n]\!]$ and (y, h) an extreme point of P saturating (4.3) such that $y_{\tilde{l}+1}^{\tilde{i}} = 1$ and $h = \tilde{l} + 1$. We have $\tilde{l} \leq \sum_{i=1}^{n} \beta_{\tilde{l}+1}^{i} \leq \sum_{l=1}^{k} \sum_{i=1}^{n} \beta_l^i y_l^i = \tilde{l}$, hence $\sum_{i=1}^{n} \beta_{\tilde{l}+1}^{i} = \tilde{l}$, which concludes our induction.

Now let us show by induction on l that for all $(i, l) \in [\![1, n]\!] \times [\![1, k]\!]$, β_l^i is an integer. This induction is trivially initialized for $\beta_1^i = 0,\ \forall i \in [\![1, n]\!]$. We suppose that for a $\tilde{l} \in [\![1, k-1]\!]$ we have that the $\beta_{\tilde{l}}^i$ for $i \in [\![1, n]\!]$ are integers and we show that the same holds for the $\beta_{\tilde{l}+1}^i$ for $i \in [\![1, n]\!]$. We note $\alpha^i = \beta_{\tilde{l}+1}^i - \beta_{\tilde{l}}^i,\ \forall i \in [\![1, n]\!]$ and for each $i \in [\![1, n]\!]$ we build a new set of inequality coefficients : $\beta_l^{(i),j} = \beta_l^j - \alpha^j + \delta_{i,j},\ \forall (j, l) \in [\![1, n]\!] \times [\![1, k]\!]$, where $\delta_{i,j}$ equals 1 if $i = j$ and 0 otherwise. Let (y, h) be an extreme point of P and for all $j \in [\![1, n]\!]$, let $l_j \in [\![h, k]\!]$ be such that $y_{l_j}^j = 1$. Then, since $\sum_{i=1}^{n} \alpha^i = 1$, we have for $i \in [\![1, n]\!]$

$$\sum_{l=1}^{k} \sum_{j=1}^{n} \beta_l^{(i),j} y_l^j = \sum_{l=1}^{k} \sum_{j=1}^{n} (\beta_l^j - \alpha^j + \delta_{i,j}) y_l^j = \sum_{j=1}^{n} (\beta_{l_j}^j - \alpha^j + \delta_{i,j}) = \sum_{j=1}^{n} \beta_{l_j}^j \geq \sum_{i=1}^{n} \beta_h^i = h - 1,$$

which means that for all $i \in [\![1, n]\!]$, $\sum_{l=1}^{k} \sum_{j=1}^{n} \beta_l^{(i),j} y_l^j + 1 \geq h$ is valid for P. Now since for $(j, l) \in [\![1, n]\!] \times [\![1, k]\!]$,

$$\left(\sum_{i=1}^{n} \alpha^i \beta^{(i)} \right)_l^j = \sum_{i=1}^{n} \alpha^i \left(\beta_l^j - \alpha^j + \delta_{i,j} \right) = \left(\sum_{i=1}^{n} \alpha^i \right) \beta_l^j - \left(\sum_{i=1}^{n} \alpha^i \right) \alpha^j + \sum_{i=1}^{n} \alpha^i \delta_{i,j} = \beta_l^j$$

and $\alpha^i \geq 0,\ \forall i \in [\![1, n]\!]$, (4.3) is a convex combination of these inequalities. Moreover, if any of the $\beta_{l+1}^i,\ i \in [\![1, n]\!]$ was not an integer, then the convex combination would be non-trivial, which would contradict the fact that (4.3) is facet-defining. Concluding our induction, we obtain that for all $(i, l) \in [\![1, n]\!] \times [\![1, k]\!]$, β_l^i is an integer. And thus that $\beta \in \Lambda$, i.e. (4.3) belongs to the set of inequalities defining $\tilde{P}$. □

Lemma 64. *Let*

$$\sum_{l=1}^{k} \sum_{i=1}^{n} \beta_l^i y_l^i + \gamma \leq h, \tag{4.4}$$

be a facet-defining inequality of P, with $\beta_l^i \in \mathbb{R}$, $\forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!]$, $\gamma \in \mathbb{R}$. Then it is equivalent to $h_{\max} - \sum_{l=1}^{h_{\max}-1}(h_{\max}-l)\sum_{i=1}^n y_l^i \leq h$ for some $h_{\max} \in [\![1,k]\!]$.

Proof. Since for each $i \in [\![1,n]\!]$, we have $\sum_{l=1}^k y_l^i = 1$, we can replace β_l^i by $\tilde{\beta}_l^i = \beta_l^i - v_i$, for some $v_i \geq 0$ such that $\tilde{\beta}_l^i \leq 0$, $\forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!]$, and γ by $\tilde{\gamma} = \gamma + \sum_{i=1}^n v_i$ and thus get new coefficients $\tilde{\beta}$ which are non-positive without changing (4.4). So without loss of generality, we can assume that β is non-positive and furthermore that γ is minimal for a non-positive β.

For $(\tilde{i},\tilde{l}) \in [\![1,n]\!] \times [\![2,k]\!]$, we take an extreme point (y,h) of P saturating (4.4) such that $y_{\tilde{l}}^{\tilde{i}} = 1$ and $(y',h') = (y_{\tilde{l}\xrightarrow{\tilde{i}}\tilde{l}-1}, h_{\tilde{l}\xrightarrow{\tilde{i}}\tilde{l}-1})$. We have $\sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l^i + \gamma = h$ and $\sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l'^i + \gamma \leq h' \leq h$, which yields $\beta_{\tilde{l}-1}^{\tilde{i}} \leq \beta_{\tilde{l}}^{\tilde{i}}$. In other words, β_l^i is increasing with l, for all $i \in [\![1,n]\!]$. This implies that for all $i \in [\![1,n]\!]$, there exists $l_i \in [\![1,k]\!]$ such that $\beta_l^i = 0$, $\forall l \in [\![l_i,k]\!]$ and $\beta_l^i < 0$ for $l < l_i$ because suppose there exists $i \in [\![1,n]\!]$ for which $\beta_k^i < 0$, then we can replace β_l^i by $\beta_l^i - \beta_k^i$ for all $l \in [\![1,k]\!]$ and add β_k^i to γ and thus get new non-positive coefficients $\tilde{\beta}$ with a $\tilde{\gamma} < \gamma$, which contradicts the minimality of γ.

Let $(\tilde{i},\tilde{l}) \in [\![1,n]\!] \times [\![1,k]\!]$ such that $\beta_{\tilde{l}}^{\tilde{i}} < 0$ (i.e., $\tilde{l} < l_{\tilde{i}}$), we know there exists an extreme point (y,h) of P saturating (4.4) such that $y_{\tilde{l}}^{\tilde{i}} = 1$. Suppose that $h < \tilde{l}$, we take (y',h') an extreme point of P such that $y_l'^i = y_l^i$, $\forall (i,l) \in [\![1,n]\!] \backslash \{\tilde{i}\} \times [\![1,k]\!]$, $y_{l_{\tilde{i}}}'^{\tilde{i}} = 1$ and obtain $h = \sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l^i + \gamma \leq \sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l'^i + \gamma \leq h' = h$, that yields $\beta_{\tilde{l}}^{\tilde{i}} = 0$, which is a contradiction. Hence $h = \tilde{l}$, so if we take the extreme point (y',h') of P such that $y_{\tilde{l}}'^{\tilde{i}} = 1$, $y_k'^i = 1$, $\forall i \in [\![1,n]\!] \backslash \{\tilde{i}\}$ and $y_l^i = 0$, $\forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!] \backslash (\{(\tilde{i},\tilde{l})\} \cup \{(i,k),\ i \in [\![1,n]\!] \backslash \{\tilde{i}\}\})$, we have

$$\tilde{l} = \sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l^i + \gamma \leq \sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l'^i + \gamma = \beta_{\tilde{l}}^{\tilde{i}} + \gamma \leq h' = \tilde{l}.$$

This gives us that for all $i \in [\![1,n]\!]$, $\beta_l^i = l - \gamma$, $\forall l \in [\![1, l_i - 1]\!]$. Consider the extreme point (y,h) of P such that for all $i \in [\![1,n]\!]$, $y_{l_i}^i = 1$, we have $\gamma \leq \min_{i \in [\![1,n]\!]} l_i$. We call $h_{\max}$ the maximum value of h among the extreme points (y,h) of P saturating (4.4).

If we take an extreme point (y,h) of P realizing $h_{\max}$, i.e. saturating (4.4) and such that $h = h_{\max}$, we have $\sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l^i + \gamma = h_{\max}$ which, since $\sum_{l=1}^k \sum_{i=1}^n \beta_l^i y_l^i$ is non-positive, yields $h_{\max} \leq \gamma$. Now for $\tilde{i} \in [\![1,n]\!]$, there exists an extreme point (y,h) of P saturating (4.4) such that $y_{l_{\tilde{i}}-1}^{\tilde{i}} = 1$. Suppose that $h < l_{\tilde{i}} - 1$, we take (y',h') an extreme point of P such that $y_l'^i = y_l^i$, $\forall (i,l) \in [\![1,n]\!] \backslash \{\tilde{i}\} \times [\![1,k]\!]$, $y_{l_{\tilde{i}}}^{\tilde{i}} = 1$

and obtain

$$h = \sum_{l=1}^{k}\sum_{i=1}^{n} \beta_l^i y_l^i + \gamma \leq \sum_{l=1}^{k}\sum_{i=1}^{n} \beta_l^i y_l'^i + \gamma \leq h' = h,$$

that yields $\beta_{l_{\tilde{i}}-1}^{\tilde{i}} = 0$, which is a contradiction. So $h = l_{\tilde{i}} - 1$, hence $h_{\max} \geq l_{\tilde{i}} - 1$. We obtain $\max_{i\in[\![1,n]\!]} l_i - 1 \leq h_{\max} \leq \gamma \leq \min_{i\in[\![1,n]\!]} l_i \leq \max_{i\in[\![1,n]\!]} l_i$. There are two possibilities, either $\min_{i\in[\![1,n]\!]} l_i = \max_{i\in[\![1,n]\!]} l_i$, or $\max_{i\in[\![1,n]\!]} l_i - 1 = \min_{i\in[\![1,n]\!]} l_i$. Suppose $\max_{i\in[\![1,n]\!]} l_i - 1 = h_{\max} = \gamma = \min_{i\in[\![1,n]\!]} l_i$, then there exists $\tilde{i} \in [\![1,n]\!]$ such that $l_{\tilde{i}} = 1 + h_{\max}$, which implies that $\beta_{h_{\max}}^{\tilde{i}} \neq 0$ and $\beta_{h_{\max}}^{\tilde{i}} = h_{\max} - \gamma = 0$ which is a contradiction. So $\min_{i\in[\![1,n]\!]} l_i = \max_{i\in[\![1,n]\!]} l_i =: L$ and

$$L - 1 \leq h_{\max} \leq \gamma \leq L.$$

Let us assume that $h_{\max} = L-1$. Then we know that $\gamma < L$, otherwise if $\gamma = L$, the extreme point of $P\,(y_{\to L}, L)$ would saturate (4.4) and thus contradict the maximality of $h_{\max}$. We consider the following inequality

$$\sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-L)y_l^i + L \leq h. \tag{4.5}$$

Let us show that it is a valid inequality for P, that is to say, that every extreme point (y,h) of P verifies it. If $h \geq L$, then $\sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-L)y_l^i = 0$ and we are done. If $h \leq L-1$, then there exists $\tilde{i} \in [\![1,n]\!]$ such that $y_h^{\tilde{i}} = 1$. Combining this with the validity of (4.4) implies

$$\sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-L)y_l^i+L = \sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-\gamma)y_l^i+\gamma+\sum_{l=1}^{L-1}\sum_{i=1}^{n}(\gamma-L)y_l^i+L-\gamma \leq h+\gamma-L+L-\gamma = h.$$

Moreover, if (y,h) is an extreme point of P saturating (4.4), then there exists $\tilde{i} \in [\![1,n]\!]$ such that $y_h^{\tilde{i}} = 1$ and $h \leq h_{\max} = L-1$ which yields $\sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-\gamma)y_l^i+\gamma = h = \sum_{l=1}^{L-1}\sum_{\substack{i=1\\ i\neq\tilde{i}}}^{n}(l-\gamma)y_l^i + h - \gamma + \gamma$. So we have $\sum_{l=1}^{L-1}\sum_{\substack{i=1\\ i\neq\tilde{i}}}^{n}(l-\gamma)y_l^i = 0$ which implies that $y_l^i = 0,\ \forall (i,l) \in ([\![1,n]\!]\backslash\{\tilde{i}\}) \times [\![1, L-1]\!]$. Thus

$$\sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-L)y_l^i+L = \sum_{l=1}^{L-1}\sum_{i=1}^{n}(l-\gamma)y_l^i+\gamma+\sum_{l=1}^{L-1}\sum_{i=1}^{n}(\gamma-L)y_l^i+L-\gamma = h+\gamma-L+L-\gamma = h.$$

Consequently, all points saturating (4.4) saturate (4.5), furthermore, $(y_{\to L}, L)$ saturates (4.5) and not (4.4). This means that the face of the polyhedron defined by (4.4) is strictly contained in the face defined by (4.5) contradicting the fact that (4.4) is facet-defining, hence $h_{\max} = \gamma = L$ and (4.4) becomes $\sum_{l=1}^{h_{\max}-1}\sum_{i=1}^{n}(l -$

$h_{\max})y_l^i + h_{\max} \leq h$ with $h_{\max} \in [\![1, n]\!]$. □

The folllowing then naturally flows.

Theorem 65.

$$P = \tilde{P}$$

Proof. With Lemma 61, we know that $P \subseteq \tilde{P}$. Take any facet-defining inequality of P $\sum_{l=1}^{k}\sum_{i=1}^{n}\beta_l^i y_l^i + \gamma \geq \alpha h$, where $\beta_l^i \in \mathbb{R}$, $\forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!]$, $(\gamma, \alpha) \in \mathbb{R} \times \{-1, 0, 1\}$. If $\alpha = 0$ (resp. $\alpha = 1$, $\alpha = -1$), then Lemma 62 (resp. Lemma 63, Lemma 64) gives us that this inequality is equivalent to one of those defining $\tilde{P}$. □

4.3 Separation problem

In Section 1.3, we explained the importance for a polytope to have an inherent separation problem solvable in polynomial time in the scope of optimizing any (linear) objective function over said polytope. The performance of any cutting plane method, let alone Branch & Cut algorithm depend on the ability to solve the separation problem inherent to the solution polytope.

P is defined by n equalities, kn non-negativity constraints, k constraints of type $\sum_{l=1}^{h_{\max}-1}\sum_{i=1}^{n}(l-h_{\max})y_l^i+h_{\max} \leq h$ and n^{k-1} of inequalities of type $\sum_{l=2}^{k}\sum_{i=1}^{n}\lambda_l^i y_l^i \geq h-1$. The total number of inequalities is then exponential. However, the following holds.

Theorem 66. *The separation problem which consists in deciding if a vector $(y,h) \in \mathbb{R}^{nk+1}$ is in P, and if not in returning a constraint of P violated by (y,h) can be solved in polynomial time.*

Proof. Let $(y,h) \in \mathbb{R}^{nk+1}$, first, one can verify in linear time if $(y,h) \in [0,1]^{nk} \times [1,k]$ is such that $\sum_{l=1}^{k} y_l^i = 1$, $\forall i \in [\![1,n]\!]$ and verifies the k inequalities of type $\sum_{l=1}^{h_{\max}-1}\sum_{i=1}^{n}(l-h_{\max})y_l^i + h_{\max} \leq h$. If not, we return a violated constraint. Otherwise, we build $\tilde{\lambda} \in \Lambda$ as follows: $\tilde{\lambda}_1^i = 0$, $\forall i \in [\![1,n]\!]$, and for $l = 2, \cdots, k$, let $\tilde{i}_l = \arg\min_{i \in [\![1,n]\!]} y_l^i + y_{l+1}^i + \cdots + y_k^i$ and set $\tilde{\lambda}_l^{\tilde{i}_l} = \tilde{\lambda}_{l-1}^{\tilde{i}_l} + 1$ and $\tilde{\lambda}_l^i = \tilde{\lambda}_{l-1}^i$, $\forall i \in [\![1,n]\!] \backslash \{\tilde{i}_l\}$. We will show that if (y,h) satisfies the inequality of P corresponding to $\tilde{\lambda}$, then it satisfies all the inequalities corresponding to an element of Λ. Suppose $\sum_{l=2}^{k}\sum_{i=1}^{n}\tilde{\lambda}_l^i y_l^i \geq h-1$ and let $\lambda \in \Lambda$ and $(i_2, \cdots, i_k) \in [\![1,n]\!]^{k-1}$ the indices corresponding to the building of λ, i.e. for $l = 2, \cdots, k$, $\lambda_l^{i_l} = \lambda_{l-1}^{i_l} + 1$ and $\lambda_l^i = \lambda_{l-1}^i$, $\forall i \in [\![1,n]\!] \backslash \{i_l\}$. By construction of $i_2, \cdots, i_k$ and $\tilde{i}_2, \cdots, \tilde{i}_k$, we have

$$\sum_{l=2}^{k}\sum_{i=1}^{n}\lambda_l^i y_l^i = \sum_{l=2}^{k}(y_l^{i_l} + y_{l+1}^{i_l} + \cdots + y_k^{i_l}) \geq \sum_{l=2}^{k}(y_l^{\tilde{i}_l} + y_{l+1}^{\tilde{i}_l} + \cdots + y_k^{\tilde{i}_l}) = \sum_{l=2}^{k}\sum_{i=1}^{n}\tilde{\lambda}_l^i y_l^i \geq h-1,$$

hence the inequality of P corresponding to λ is satisfied. So we can conclude that if (y, h) satisfies the inequality of P corresponding to $\tilde{\lambda}$, then it satisfies all the inequalities of P corresponding to an element of Λ. And since the construction of $\tilde{\lambda}$ is done in polynomial time, the separation problem is indeed polynomial. □

4.4 Variants

4.4.1 Modified Polytopes

Linear programming formulations aiming to maximize (resp. minimize) the index of the lowest (resp. highest) nonzero row of an assignment matrix are related to polytope Q (resp. Q') described below. Observe that h (resp. g) is only required to be less (resp. more) than or equal to $\min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i$ (resp. $\max_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l x_l^i$).

$$Q = Conv\left\{ \left(\begin{bmatrix} y_k^1 & y_k^2 & \cdots & y_k^n \\ \vdots & \vdots & \ddots & \vdots \\ y_2^1 & y_2^2 & \cdots & y_2^n \\ y_1^1 & y_1^2 & \cdots & y_1^n \end{bmatrix}, h \right) \in M_{k\times n}(\{0,1\}) \times \mathbb{N}\backslash\{0\} \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1,\ \forall i \in [\![1,n]\!], \\ h \leq \min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i, \end{array} \right\}$$

$$Q' = Conv\left\{ \left(\begin{bmatrix} x_k^1 & x_k^2 & \cdots & x_k^n \\ \vdots & \vdots & \ddots & \vdots \\ x_2^1 & x_2^2 & \cdots & x_2^n \\ x_1^1 & x_1^2 & \cdots & x_1^n \end{bmatrix}, g \right) \in M_{k\times n}(\{0,1\}) \times \mathbb{N}\backslash\{0\} \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} x_l^i = 1,\ \forall i \in [\![1,n]\!], \\ g \geq \max_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l x_l^i, \end{array} \right\}$$

A full description of Q is given below.

Theorem 67.

$$Q = \left\{ \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1,\ \forall i \in [\![1,n]\!], \\ \sum_{l=2}^{k} \sum_{i=1}^{n} \lambda_l^i y_l^i \geq h - 1,\ \forall \lambda \in \Lambda, \\ y_l^i \geq 0,\ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!],\ h \geq 1. \end{array} \right.$$

Proof. It is a simple fact that $h \geq 1$ is the only possible facet of type (4.4) while the positivity constraints are the only possible facets of type (4.2). Let us consider an inequality of type (4.3) defining a facet of Q. Any extreme point of Q saturating such a facet necessarily satisfies $h = \min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i$ implying that it is also a point of P. Using this observation and the fact that Q and P have the same dimension, we deduce that any facet of Q of type (4.3) is also a facet of P. Using the description of P, we get the result. □

Similarly to Theorem 66 we can deduce that the separation problem with respect to Q is solvable in polynomial time as well.

4.4.2 Opposite polytopes

We can also derive a full description of P' and Q' from the previous results.

Theorem 68.

$$P' = \begin{cases} \sum_{l=1}^{k} x_l^i = 1, \ \forall i \in [\![1,n]\!], \\ \sum_{l=1}^{k-1} \sum_{i=1}^{n} \lambda_l^i x_l^i \leq g-k, \ \forall \lambda \in \tilde{\Lambda}, \\ \sum_{l=g_{\min}+1}^{k} \sum_{i=1}^{n} (l-g_{\min}) x_l^i + g_{\min} \geq g, \ \forall g_{\min} \in [\![1,k]\!], \\ x_l^i \geq 0, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!], \ g \in \mathbb{R}, \end{cases}$$

$$Q' = \begin{cases} \sum_{l=1}^{k} x_l^i = 1, \ \forall i \in [\![1,n]\!], \\ \sum_{l=1}^{k-1} \sum_{i=1}^{n} \lambda_l^i x_l^i \leq g-k, \ \forall \lambda \in \tilde{\Lambda}, \\ x_l^i \geq 0, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!], \ g \leq k \end{cases}$$

where

$$\tilde{\Lambda} = \left\{ \lambda = (\lambda_l^i)_{(i,l) \in [\![1,n]\!] \times [\![1,k]\!]} \in \mathbb{N}^{nk} \;\middle|\; \begin{array}{l} \lambda_{l+1}^i \leq \lambda_l^i, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k-1]\!] \\ \sum_{i=1}^{n} \lambda_l^i = k-l, \ \forall l \in [\![1,k]\!] \end{array} \right\}.$$

Proof. Take an extreme point (y,h) of P, and let $(x,g) \in M_{k \times n}(\{0,1\}) \times \mathbb{N} \backslash \{0\}$ such that $x_l^i = y_{k-l+1}^i, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!]$ and $g = k-h+1$, then $g = \max_{i \in [\![1,n]\!]} \sum_{l=1}^{k} l x_l^i$, hence $(x,g) \in P'$. Conversely, any extreme point (x,g) of P' can be obtained from an extreme point of P in this manner. So P' is obtained from P doing the change of variables $x_l^i = y_{k-l+1}^i, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!]$ and $g = k-h+1$. Therefore its hyperplane representation is obtained from that of P in Theorem 65. Similarly, Q' is obtained from Q doing the same change of variables and its hyperplane representation is thus obtained from Theorem 67. □

The previous results imply that the separation problems related to P' and Q' can be solved in polynomial time.

4.5 An alternative description by Balas's lift-and-project technique

In 1998, Egon Balas [9] presented a way to derive the description of the convex hull of the union of polyhedra from their description. This technique can be used to derive a description of P described as such a convex hull.

4.5.1 P as the convex hull of the union of easily describable polyhedra

Take for any $h' \in [\![1,k]\!]$ the set X'_h consisting in the $\{0,1\}$ assignment matrices which lowest row not identically equal to the zero row is the one corresponding to h', appended with h' in order to fit the elements of P :

$$X_{h'} = \left\{ \left(\begin{bmatrix} y_k^1 & y_k^2 & \cdots & y_k^n \\ \vdots & \vdots & \ddots & \vdots \\ y_2^1 & y_2^2 & \cdots & y_2^n \\ y_1^1 & y_1^1 & \cdots & y_1^n \end{bmatrix}, h' \right) \in M_{k\times n}(\{0,1\}) \times \mathbb{N}\backslash\{0\} \;\middle|\; \begin{array}{l} \sum_{l=1}^{k} y_l^i = 1, \ \forall i \in [\![1,n]\!], \\ \min_{i\in[\![1,n]\!]} \sum_{l=1}^{k} l y_l^i = h', \end{array} \right\}.$$

We consider the polytope $P_{h'} = Conv(X_{h'})$, it is then clear that

$$P = Conv\left(\bigcup_{h'=1}^{k} P_{h'} \right).$$

In order to make use of Balas's result, we need an hyperplane description of P'_h. One can be found quite easily by construction of $P_{h'}$.

Proposition 69. *Given $h' \in [\![1,k]\!]$,*

$$P_{h'} = \begin{cases} \sum_{l=1}^{k} y_l^i = 1, \ \forall i \in [\![1,n]\!], \\ y_l^i = 0, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,h'-1]\!], \\ \sum_{i=1}^{n} y_{h'}^i \geq 1, \\ y_l^i \geq 0, \ \forall (i,l) \in [\![1,n]\!] \times [\![1,k]\!] \end{cases} \tag{4.6}$$

Proof. Let us call $\tilde{P}_{h'}$ the bounded set of points of $\mathbb{R}^{kn+1}$ satisfying all the equalities and inequalities of the right-hand side of 4.6. It is clear that any element of $P_{h'}$ verifies all the afore-mentioned constraints, yielding that $P_{h'} \subseteq \tilde{P}_{h'}$. It remains to show that these constraints suffice to define $P_{h'}$, in other words, that $\tilde{P}_{h'} \subseteq P_{h'}$. One way to do so is to show that any extreme point of $\tilde{P}_{h'}$ belongs to $X_{h'}$. Let $y \in M_{k\times n}(\mathbb{R})$ such that (y,h') is an extreme point of $\tilde{P}_{h'}$. All the components of y are then positive, and since $\sum_{l=1}^{k} y_l^i = 1$ for all $i \in [\![1,n]\!]$, then we have $y \in M_{k\times n}([0,1])$. If we have that $y \in M_{k\times n}(\{0,1\})$, then we can directly conclude that $(y,h') \in X'_h$. Let us show that y cannot be fractional while (y,h') os an extreme point of $\tilde{P}_{h''}$. Suppose that it is, that is to say that there exists $(i,l) \in [\![1,n]\!] \times [\![1,k]\!]$ such that $0 < y_l^i < 1$. Let us build y' and y'' two distinct points of $\tilde{P}_{h'}\backslash\{y\}$ such that (y,h') is a non-trivial convex combination of (y',h') and (y'',h'). Since $\sum_{l=1}^{k} y_l^i = 1$, there exists $l' \in [\![1,k]\!]\backslash\{l\}$ such that $0 < y_{l'}^i < 1$. We know that both l and l' are larger

than or equal to h', we distinguish two cases on the coincidence of h' with l or l'.

$\underline{l \neq h' \neq l'}$**:** we take $\varepsilon > 0$ small enough such that $\{y^i_l - \varepsilon, y^i_l + \varepsilon, y^i_{l'} - \varepsilon, y^i_{l'} + \varepsilon\} \subset [0,1]$ and y' (resp. y'') coinciding with y on all components but y^i_l and $y^i_{l'}$ for which we set $y'^i_l = y^i_l - \varepsilon$ and $y'^i_{l'} = y^i_{l'} + \varepsilon$ (resp. $y''^i_l = y^i_l + \varepsilon$ and $y''^i_{l'} = y^i_{l'} - \varepsilon$).

$\underline{h' \in \{l, l'\}}$**:** without loss of generality, we assume that $l \neq h' = l'$. Since $\sum_{i=1}^n y^i_{h'} \geq 1$, there exists $i' \in [\![1,n]\!] \backslash \{i\}$ such that $y^{i'}_{h'} > 0$. If $y^{i'}_{h'} = 1$, then we take $\varepsilon > 0$, y' and y'' exactly as defined in the case above. On the other hand, if $0 < y^{i'}_{h'} < 1$, then there exists $l'' \in [\![h'+1, k]\!]$ such that $y^{i'}_{l''} > 0$. Then we take $\varepsilon > 0$ small enough such that $\{y^i_l - \varepsilon, y^i_l + \varepsilon, y^i_{h'} - \varepsilon, y^i_{h'} + \varepsilon, y^{i'}_{h'} - \varepsilon, y^{i'}_{h'} + \varepsilon, y^{i'}_{l''} - \varepsilon, y^{i'}_{l''} + \varepsilon\} \subset [0,1]$ and y' (resp. y'') coinciding with y on all components but y^i_l, $y^i_{h'}$, $y^{i'}_{h'}$ and $y^{i'}_{l''}$ for which we set $y'^i_l = y^i_l - \varepsilon$, $y'^i_{h'} = y^i_{h'} + \varepsilon$, $y'^{i'}_{h'} = y^{i'}_{h'} - \varepsilon$ and $y'^{i'}_{l''} = y^{i'}_{l''} + \varepsilon$ (resp. $y''^i_l = y^i_l + \varepsilon$, $y''^i_{h'} = y^i_{h'} - \varepsilon$, $y''^{i'}_{h'} = y^{i'}_{h'} + \varepsilon$ and $y''^{i'}_{l''} = y^{i'}_{l''} - \varepsilon$).

In both cases, $(y, h') = \frac{1}{2}(y', h') + \frac{1}{2}(y'', h')$ with (y', h') and (y'', h') in $\tilde{P}_{h'}$, thus contradicting the fact that (y, h) is an extreme point of $\tilde{P}_{h'}$. Consequently, (y, h) belongs necessarily to $M_{k\times n}(\{0,1\})$ and hence to $X_{h'}$. □

4.5.2 Deriving an alternative description for P and its variants

Using Balas's technique on P as the convex hull of the union of the $P_{h'}$, we obtain the following description for P :

Theorem 70.

$$P = \begin{cases} (y,h) = \sum_{h'=1}^k (y^{(h')}, \lambda^{(h')} h'), \\ \sum_{l=1}^k y_l^{(h'),i} = \lambda^{(h')}, \ \forall (i,h') \in [\![1,n]\!] \times [\![1,k]\!], \\ y_l^{(h'),i} = 0, \ \forall (i,l,h') \in [\![1,n]\!] \times [\![1,k]\!]^2, \ s.t. \ l < h', \\ \sum_{i=1}^n y_{h'}^{(h'),i} \geq \lambda^{(h')}, \ \forall h' \in [\![1,k]\!], \\ \sum_{h'=1}^k \lambda^{(h')} = 1, \\ y_l^{(h'),i} \geq 0, \ \forall (i,l,h') \in [\![1,n]\!] \times [\![1,k]\!]^2, \\ \lambda \in \{0,1\}^k \end{cases}$$

The main difference between our first description of P and this one is the fact it involves a polynomial number of constraints, making its inherent separation problem naturally tractable. On the other hand, it involves significantly more variables. Using the same method, we can derive alternative descriptions for the variants of P.

Theorem 71.

$$Q = \begin{cases} (y,h) = \sum_{h'=1}^{k} (y^{(h')}, \lambda^{(h')} h'), \\ \sum_{l=1}^{k} y_l^{(h'),i} = \lambda^{(h')}, \ \forall (i,h') \in [\![1,n]\!] \times [\![1,k]\!], \\ y_l^{(h'),i} = 0, \ \forall (i,l,h') \in [\![1,n]\!] \times [\![1,k]\!]^2, \ s.t. \ l < h', \\ \sum_{h'=1}^{k} \lambda^{(h')} = 1, \\ y_l^{(h'),i} \geq 0, \ \forall (i,l,h') \in [\![1,n]\!] \times [\![1,k]\!]^2, \\ \lambda \in \{0,1\}^k \end{cases}$$

$$P' = \begin{cases} (x,g) = \sum_{g'=1}^{k} (x^{(g')}, \lambda^{(g')} g'), \\ \sum_{l=1}^{k} x_l^{(g'),i} = \lambda^{(g')}, \ \forall (i,g') \in [\![1,n]\!] \times [\![1,k]\!], \\ x_l^{(g'),i} = 0, \ \forall (i,l,g') \in [\![1,n]\!] \times [\![1,k]\!]^2, \ s.t. \ l > g', \\ \sum_{i=1}^{n} x_{g'}^{(g'),i} \geq \lambda^{(g')}, \ \forall g' \in [\![1,k]\!], \\ \sum_{g'=1}^{k} \lambda^{(g')} = 1, \\ x_l^{(g'),i} \geq 0, \ \forall (i,l,g') \in [\![1,n]\!] \times [\![1,k]\!]^2, \\ \lambda \in \{0,1\}^k \end{cases}$$

$$Q' = \begin{cases} (x,g) = \sum_{g'=1}^{k} (x^{(g')}, \lambda^{(g')} g'), \\ \sum_{l=1}^{k} x_l^{(g'),i} = \lambda^{(g')}, \ \forall (i,g') \in [\![1,n]\!] \times [\![1,k]\!], \\ x_l^{(g'),i} = 0, \ \forall (i,l,g') \in [\![1,n]\!] \times [\![1,k]\!]^2, \ s.t. \ l > g', \\ \sum_{g'=1}^{k} \lambda^{(g')} = 1, \\ x_l^{(g'),i} \geq 0, \ \forall (i,l,g') \in [\![1,n]\!] \times [\![1,k]\!]^2, \\ \lambda \in \{0,1\}^k \end{cases}$$

Chapter 5

Conclusion and future research

The choice of our generic problem $\max_{\Lambda \in \vec{\mathcal{O}}(G)} f(d^+_\lambda(v_1) - d^-_\lambda(v_1), \cdots, d^+_\lambda(v_n) - d^-_\lambda(v_n))$ for some objective function f turns out to be quite relevant since when we study various possible objective functions f, we reach a fairly wide range of graph optimization problems such as the maximum cut problem or the isoperimetric number. We managed to obtain good-performance formulations and bounds from the orientation-based formulations of these problems and described a family of polyhedra emerging in very diverse combinatorial optimization problems including the most imbalanced orientation of a graph, the minimum span frequency assignment and some scheduling problems.

Concerning the problem (MaxIm) (Chapter 2), while computing the most imbalanced orientation of a graph is generally difficult, the problem turns out to be easy for cacti. It may be the same for other graph classes. Characterizing such graph classes would be interesting. We can think of families defined by a structural condition and give a polynomial time algorithm for these graphs precisely as we have done for cacti in Section 2.3. And we can also think of families defined by the value of MaxIm(G) as we characterized the graphs for which MaxIm$(G) = 0$ in Section 2.2. For example, it would be interesting to characterize the graphs for which MaxIm$(G) = \delta_G$. It is clear that the bipartite graphs belong to this family and that if a regular graph (i.e. a graph where all the vertices have equal degrees) belongs to this family, then it is bipartite. Moreover, with the characterization given for the graphs such that MaxIm$(G) = 0$, we know that for any graph G, if $\delta(G) = 1$, then any orientation of G has an imbalance equal to 1 (implying MaxIm$(G) = 1$). Two mixed integer linear programming formulations of MaxIm have been presented. Several families of valid inequalities have been presented to strengthen one of the two formulations. Exhibition of other families of valid inequalities might be helpful to solve larger size problems. It would also be interesting to derive similar results of all the edge-weighted version of (MaxIm).

Starting from our generic graph orientation problem, we have seen in Chapter 3 that the maximum cut problem can be modeled in several new ways. By lifting, one can get some bounds that are stronger than the standard semidefinite bound of [47]. The bounds are even exact for some graph classes. Several new mixed integer programming formulations have been obtained using discretization and aggregation techniques. The performance of these formulations compares to and is often better than the performance of the BiqCrunch solver on many graph families, it can even be improved if we strengthen the formulations either by adding valid inequalities or using other lifting techniques. Also the new formulations we introduced here for the unweighted maximum cut problem may lead to similar formulations for the weighted case.

We exhibited in Chapter 4 a family of polyhedra emerging in very diverse combinatorial optimization problems including the most imbalanced orientation of a graph, the minimum span frequency assignment and some scheduling problems. Then a full description of these polyhedra has been derived. We also proved that the separation problems related to these polyhedra can be solved in polynomial time and thus optimization over them can be done in polynomial time. We think that, like MaxIm, many combinatorial optimization problems where discretization techniques are used can benefit from the description of the polyhedra we introduced. Future work may be directed towards investigations on extensions of the polyhedra we considered in order to get better approximations while still keeping the feature of computational tractability. One can, for example, study $\{0,1\}$ assignment matrices appended with both h and g (the index of the lowest (resp. highest) nonzero row of the matrix). The related polytope is included in the intersection of P or Q. Some preliminary investigations show that more inequalities are necessary to describe the polytope.

Bibliography

[1] Karen Aardal, Stan Van Hoesel, Arie Koster, Carlo Mannino & Antonio Sassano: 'Models and solution techniques for frequency assignment problems', *Annals of Operations Research, Volume 153-1*, pp. 79–129 (2007)

[2] Manindra Agrawal, Neeraj Kayal & Nitin Saxena: 'PRIMES is in P', *Annals of Mathematics, Volume 160-2*, pp. 781–793 (2002)

[3] Miguel Anjos & Henry Wolkowicz: 'Strengthened semidefinite relaxations via a second lifting for the max-cut problem', *Discrete Applied Mathematics, Volume 119*, pp. 79–106 (2002)

[4] Yuichi Asahiro, Eiji Miyano, Hirotaka Ono & Kouhei Zenmyo: 'Graph Orientation Algorithms to Minimize the Maximum Outdegree', *Proceedings of Computin: the Australian Theory Symposium (CATS2006), Hobart Australia*,(2006)

[5] Yuichi Asahiro, Eiji Miyano, Hirotaka Ono & Kouhei Zenmyo: 'Approximation algorithms for the graph orientation minimizing the maximum weighted outdegree', *Proceedings of the 3rd Int. Conf. on Algorithmic Aspects in Information and Management (AAIM2007), LNCS 4508*, pp. 167–177 (2007)

[6] Yuichi Asahiro, Eiji Miyano & Hirotaka Ono: 'Graph classes and the complexity of the graph orientation minimizing the maximum weighted outdegree', *Proceedings of the fourteenth Computing: the Australasian Theory Symposium(CATS2008), Wollongong, NSW, Australia*, (2008)

[7] Yuichi Asahiro, Jesper Jansson, Eiji Miyano & Hirotaka Ono: 'Deegree constrained graph orientation: maximum satisfaction and minimum violation', *WAOA 2013, LNCS 8447*, pp. 24–36 (2014)

[8] Cleve Ashcraft & Joseph Liu: 'Using domain decomposition to find graph bisectors', *Technical report CS-95-08, York University, North York, Canada*, (1995)

[9] Egon Balas: 'Disjunctive Programming: Properties of the Convex Hull of Feasible Points', Invited paper, with a foreword by G. Cornuéjols and W. Pulleyblank, *Discrete Applied Mathematics, Volume 89*, pp. 4-44 (1998)

[10] Jøseph Bang-Jensen & Gregory Gutin: Orientations of graphs and digraphs in 'Digraphs: Theory, Algorithms and applications', *Springer, 2nd edition*, pp. 417–472 (2009)

[11] Francisco Barahona, Martin Grötschel, Michael Jünger & Gerhard Reinelt: 'An application of combinatorial optimization to statistical physics and circuit layout design', *Operations Research, Volume 36*, pp. 493–513 (1998)

[12] Francisco Barahona: 'The max-cut problem on graphs not contractible to K_5', *Operations Resarch Letters, Volume 2*, pp.107–111 (1983)

[13] Francisco Barahona & Ali Ridha Mahjoub: 'On the cut polytope', *Mathematical Programming, Volume 36*, pp.157–173 (1986)

[14] Walid Ben-Ameur, Antoine Glorieux & José Neto: 'On the most imbalanced orientation of a graph', *Proceedings of COCOON2015, LNCS*, pp. 16–29 (2015)

[15] Walid Ben-Ameur, Antoine Glorieux & José Neto: 'A full description of polytopes related to the index of the lowest nonzero row of an assignment matrix', *Proceedings of ISCO2016, LNCS*, (2016)

[16] Walid Ben-Ameur, Antoine Glorieux & José Neto: 'From graph orientation to the unweighted maximum cut problem', *Proceedings of COCOON2016, LNCS*, pp. 370–384 (2016)

[17] Walid Ben-Ameur & Makhlouf Hadji: 'Designing Steiner Networks with Unicyclic Connected Components: An Easy Problem', *SIAM Journal on Discrete Mathematics, Society for Industrial and Applied Mathematics, Volume 24-4*, pp. 1541–1557 (2010)

[18] Walid Ben-Ameur, Ali Ridha Mahjoub & José Neto: 'The Maximum Cut Problem, in Paradigms of Combinatorial Optimization', *V. Paschos (Ed), Wiley-ISTE*, pp. 131–172 (2010)

[19] Walid Ben-Ameur & José Neto: 'Spectral bounds for the maximum cut problem', *Networks , Volume 52-1*, pp.8–13 (2008)

[20] Walid Ben-Ameur & José Neto: 'Spectral bounds for unconstrained (-1, 1)-quadratic optimization problems', *European Journal of Operational Research, Volume 207-1*, pp.15–24 (2010)

[21] Walid Ben-Ameur & José Neto: 'On a gradient-based randomized heuristic for the maximum cut problem', *International journal of Mathematics in Operational Research, Volume 4-3*, pp. 276–293 (2012)

[22] Therese Biedl , Timothy Chan, Yashar Ganjali, Mohammad Hajiaghayi & David Wood: 'Balanced vertex-orderings of graphs', *Discrete Applied Mathematics, Volume 48-1*, pp. 27–48 (2005)

[23] Leonard Brooks: 'On Colouring the nodes of a network', *Mathematical Proceedings Cambridge Philosophical Society, Volume 37*, pp. 194–197 (1941)

[24] Maria Calzarossa & Giuseppe Serazzi: 'Workload characterization: a survey', *Proceedings of the IEEE, Volume 81*, pp.1136–1150 (1993)

[25] Jeff Cheeger: 'A lower bound for the smallest eigenvalue of the Laplacian', *Proceedings of the Princeton conference in honor of Professor S. Bochner*, pp. 195–199 (1969)

[26] Marek Chrobak & David Eppstein: 'Planar orientations with low out-degree and compaction of adjacency matrices', *Theoretical Computer Sciences, Volume 86*, pp. 243–266 (1991)

[27] Fan Chung: 'Spectral Graph Theory', *volume 92 of CBMS Regional Conference Series in Mathematics. Published for the Conference Board of the Mathematical Sciences, Washington, DC*, (1997)

[28] Václav Chvátal & Carsten Thomassen: 'Distances in orientation of graphs', *Journal of Combinatorial Theory, series B, Volume 24*, pp. 61–75 (1978)

[29] William Cook, William Cunningham, William Pulleyblank & Alexander Schrijver: 'Combinatorial Optimization', *ISBN: 978-0-471-55894-1*, (1997)

[30] George Dantzig: 'Maximization of a linear function of variables subject to linear inequalities', *Activity analysis of Production and Allocation*, pp. 339-347 (1951)

[31] Maurits Degraaf & Alexander Schrijver: 'Grid Minors of Graphs on the Torus', *Journal of Combinatorial Theory, series B, Volume 61*, pp. 57–62 (1994)

[32] Charles Delorme & Svatopluk Poljak: 'Combinatorial Properties and the Complexity of a Max-cut Approximation', *European Journal of Combinatorics, Volume 14, Issue 4*, pp. 313–333 (1993)

[33] Michel Deza & Monique Laurent: 'Geometry of cuts and metrics', *Springer, Berlin*, (1997)

[34] Reinhard Diestel: 'Graph Theory, 4th edition', *Springer* (2010)

[35] Oliver Dolezal, Thomas Hofmeister & Hanno Lefmann: 'A comparison of approximation algorithms for the MaxCut problem', *Report CI-/99, Universität Dortmund*, (1999)

[36] Jack Edmonds & Ellis Johnson: 'Matching, Euler tours and the Chinese postman problem', *Mathematical Programming, Volume 5*, pp. 88–24 (1973)

[37] Paul Erdős & Tibor Gallai: 'Gráfok előírt fokszámṕontokkal', *Matematikai Lapok, Volume 5*, pp.88–124 (1960)

[38] Leonhard Euler: 'Solutio problematis ad geometriam situs pertinentis', *Commentarii academiae scientiarum Petropolitanae, Volume 8*, pp. 128–140 (1741)

[39] Paola Festa, Panos Pardalos, Mauricio Resende & Celso Ribeiro: 'Randomized heuristics for the MAX-CUT problem', *Optimization Methods and Software, Volume 7*, pp.1033–1058 (2002)

[40] Ilse Fischer, Gerald Gruber, Franz Rendl & Renata Sotirov: 'Computational experience with a bundle approach for semidefinite cutting plane relaxations of Max-Cut and Equipartition', *Mathematical Programming, Volume 105*, pp.451–469 (2006)

[41] Fedor Fomin, Martín Matamala & Ivan Rapaport: 'Complexity of approximating the oriented diameter of chordal graphs', *Journal of Graph Theory, Volume 45-4*, pp. 255–269 (2004)

[42] Lester Ford & Delbert Fulkerson: 'Flows in networks', *Princeton University Press, Princeton, NJ*, (1962)

[43] András Frank & András Gyárfás: 'How to orient the edges of a graph?', *Colloquia Mathematica Societatis János Bolyai, Volume 18*, pp. 353–364 (1976)

[44] Robert Freund: 'Introduction to Semidefinite Programming (SDP)', *Manuscript, available from http://ocw.mit.edu* (2004)

[45] Michael Garey & David Johnson: 'Computers and Intractability; A Guide to the Theory of NP-Completeness', *W.H.Freeman & Co.*, (1990)

[46] Fred Glover: 'Improved linear integer programming formulations of nonlinear integer problems', *Management Science, Volume 22*, pp. 455–460 (1975)

[47] Michel Goemans & David Williamson: 'Improved approximation algorithms for maximum cut and satisfiability problems using semidefinite programming', *Journal of the ACM, Volume 42*, pp. 1115–1145 (1995)

[48] Michel Goemans: 'Semidefinite programming in combinatorial optimization', *Mathematical Programming, Volume 79, Issue 1*, pp.143 –161 (1997)

[49] Petr Golovach: 'Computing the isoperimetric number of a graph', *Cybernetics and Systems Analysis, Volume 30*, (1994)

[50] Martin Grötschel & William Pulleyblank: 'Weakly bipartite graphs and the max-cut problem', *Operations Research Letters, Volume 1*, pp.23–27 (1981)

[51] Martin Grötschel, László Lovász & Alexander Schrijver: 'The ellipsoid method and its consequences in combinatorial optimization', *Combinatorica, Volume 1*, pp.70–89 (1981)

[52] Bertrand Guenin: 'A Characterization of Weakly Bipartite Graphs', *Journal of Combinatorial Theory, Series B, Volume 83*, pp.112-168 (2001)

[53] Frank Hadlock: 'Finding a maximum cut of a planar graph in polynomial time', *SIAM Journal on Computing, Volume 4*, pp. 221–225 (1975)

[54] Seifollah Hakimi: 'On realizability of a set of integers as degrees of the vertices of a linear graph. I', *Journal of the Society for Industrial and Applied Mathematics, Volume 10*, pp. 496–506 (1962)

[55] Peter Hammer: 'Some network flow problems solved with pseudo-boolean programming', *Operations Research, Volume 32*, pp. 388–399 (1965)

[56] Frank Harary & Leo Moser: 'The theory of round robin tournaments', *American Mathematical Monthly, Volume 73*, pp. 231–246 (1971)

[57] Frank Harary, Jakob Krarup, & Allen Schwenk: 'Graphs suppressible to an edge', *Canadian Mathematical Bulletin, Volume 15*, pp. 201–204 (1971)

[58] Frank Harary: 'Graph Theory', *Addison-Wesley*, (1969)

[59] Johan Håstad: 'Some optimal inapproximability results', *Journal of the ACM, Volume 48*, pp. 798–859 (2001)

[60] Václav Havel: 'A remark on the existence of finite graphs', *Časopis pro pěstování matematiky, Volume 80*, pp. 477–480 (1955)

[61] Christoph Helmberg: 'A cutting plane algorithm for large scale semidefinite relaxations, Technical report ZR-01-26', *Konrad-Zuse-Zentrum Berlin*, (2001)

[62] Carl Hierholzer: 'Ueber die Mglichkeit, einen Linienzug ohne Wiederholung und ohne Unterbrechung zu umfahren', *Mathematische Annalen, Volume 6*, pp. 30–32. (1873)

[63] Dorit Hochbaum & David Shmoys: 'A polynomial approximation scheme for scheduling on uniform processors: Using the dual approximation approach', *SIAM Journal on Computing, Volume 17-3*, pp. 539–551 (1988)

[64] Ellis Horowitz & Sartaj Sahni: 'Exact and approximate algorithms for scheduling nonidentical processors', *Journal of ACM, Volume 23-2*, pp. 317–327(1976)

[65] Jan Kára, Jan Kratochvíl & David Wood: 'On the complexity of the balanced vertex ordering problem', *Proceedings of COCOON2005, LNCS 3595*, pp. 849–858 (2005)

[66] Narendra Karmarkar: 'A new polynomial time algorithm for linear programming', *Combinatorica, Volume4*, pp. 373–395 (1984)

[67] Richard Karp: 'Reducibility among combinatorial problems', *Complexity of computer computation, R.E. Miller and J.W. Thatcher (eds.), Plenum Press, New York*, pp. 85–103 (1972)

[68] Leonid Khachiyan: 'A Polynomial Algorithm in Linear Programming', *Zhurnal Vychisditel'noi Matematiki i Matematicheskoi Fiziki, Volume 20*, pp. 51–68 (1979)

[69] Arie Koster: 'Frequency Assignment - Models and Algorihtms', *PhD thesis, Univ. Maastricht, the Netherlands*, (1999)

[70] Nathan Krislock, Jérôme Malick & Frédéric Roupin: 'Improved semidefinite bounding procedure for solving Max-Cut problems to optimality', *Mathematical Programming, Volume143-1,2*, pp.61–86 (2014)

[71] Alisa Land & Alison Doig: 'An automatic method of solving discrete programming problems', *Econometrica, Volume 28*, pp. 497–520 (1960)

[72] Dominic Lanphier & Jason Rosenhouse: 'Lower Bounds on the Cheeger Constants of Highly Connected Regular Graphs', *Congressus Numerantium 173*, pp. 65–74 (2005)

[73] Hyman Landau: 'On dominance relations and the structure of animal societies III. The condition for a score structure', *The Bulletin of Mathematical Biophysics, Volume 15*, pp. 143–148 (1953)

[74] Monique Laurent & Franz Rendl: 'Semidefinite Programming and Integer Programming', *Handbooks in Operations Research and Management Science, Volume 12*, pp. 393–514 (2005)

[75] Monique Laurent: 'Tighter linear and semidefinite relaxations for max-cut based on the Lovász-Schrijver lift-and-project procedure', *SIAM Journal on Optimization, Volume 2*, pp. 345–375 (2001)

[76] Adam Letchford & Andrea Lodi: 'Strengthening Chvátal-Gomory cuts and Gomory fractional cuts', *Operations Research Letters, Volume 30-2*, pp. 74-82 (2002)

[77] László Lovász: 'Three short proofs in graph theory', *Journal of Combinatorial Theory, Series B, Volume 19*, pp. 269–271 (1975)

[78] Bin Luo, Richard Wilson & Edwin Hancock: 'Spectral Clustering of Graphs', *Graph Based Representations in Pattern Recognition*, pp.190–201 (2003)

[79] Zoltan Mann & Aniko Szajkó: 'Complexity of different ILP models of the frequency assignment problem', *Linear Programming - New frontiers in Theory and Applications*, pp. 305–326 (2012)

[80] Pedro Martins: 'Extended and discretized formulations for the maximum clique problem', *Computers & Operations Research, Volume 37-7*, pp. 1348–1358 (2011)

[81] Bojan Mohar: 'Isoperimetric Numbers of Graphs', *Journal of Combinatorial Theory, Series B, Volume 47*, pp. 274–291 (1989)

[82] Dhruv Mubayi, Todd Will & Douglas West: 'Realizing Degree Imbalances in Directed Graphs', *Discrete Mathematics, Volume 239-173*, pp. 147–153 (2001)

[83] Crispin Nash-Williams: 'On orientations, connectivity and odd vertex pairings in finite graphs', *Canadian Journal of Mathematics, Volume 12*, pp. 555-567 (1960)

[84] G.I. Orlova & Y.G. Dorfman: 'Finding the maximal cut in a graph', *Engineering Cybernetics*, pp. 502–506 (1972)

[85] James Park & Cynthia Phillips: 'Finding minimum-quotient cuts in planar graphs', *Proceedings of the twenty-fifth annual ACM symposium on Theory of computing (STOC '93)*, pp. 766–775 (1993)

[86] Ronlad Read & Robin Wilson: 'An atlas of graphs', *Clarendon press*, (1998)

[87] Franz Rendl, Giovanni Rinaldi & Angelika Wiegele: 'Solving Max-Cut to optimality by intersecting semidefinite and polyhedral relaxations', *IASI research report 08-11*, (2008)

[88] Herbert Robbins: 'A theorem on graphs with an application to a problem of traffic control', *American Mathematical Monthly, Volume 46*, pp. 281–283 (1939)

[89] Thomas Rothvoß: 'The Lasserre hierarchy in Approximation algorithms', *Lecture notes for the MAPSP Tutorial*, preliminary version (2013)

[90] Sartaj Sahni & Teofilo Gonzalez: 'P-complete approximation algorithms', *Journal of the Association for Computing Machinery, Volume 23-3*, pp.555–565 (1976)

[91] Thomas Schaefer: 'The complexity of satisfiability problems', *Proceedings of the 10th Annual ACM Symposium on Theory of Computing*, pp. 216–226 (1978)

[92] Alexander Schrijver: 'Combinatorial optimization: polyhedra and efficiency', *Springer*, (2003)

[93] Lieven Vandenberghe & Stephen Boyd: 'Semidefinite Programming' *SIAM*, pp. 49–95 (1996)

[94] Vijay Vazirani: 'Minimum Makespan Scheduling', *Approximations Algorithms, Springer, Chapter 10*, pp. 79–83 (2003)

[95] Venkat Venkateswaran: 'Minimizing maximum indegree', *Discrete Applied Mathematics, Volume 143*, pp. 374–378 (2004)

[96] Angelika Wiegele: 'Nonlinear Optimization Techniques Applied to Combinatorial Optimization Problems', *PhD thesis, Alpen-Adria-Universität Klagenfurt*, (2006)

[97] Laurence Wolsey: 'Integer Programming', *ISBN: 978-0-471-28366-9*, (1998)

[98] Mihalis Yannakakis: 'Node-and-edge deletion NP-complete problems', *Proceedings of the 10th annual ACM Symposium on the Theory of Computing*, pp. 253–264 (1978)

Appendix A

Proof of Lemma 21

Not-all-equal at most 3-SAT is a NP-complete satisfiability problem [91] for which the formulas must respect the following constraints:

▷ Each clause contains at most three literals.

▷ In each clause, not all the literals can be true. In other words, for the formula to be satisfiable, there must exist an assignment of the variables such that in each clause, at least one of the literal is true and at least one of the literal is false.

Let φ be a not-all-equal at most 3-SAT formula with n variables $\{x_1, \cdots, x_n\}$ and m clauses $\{c_1, \cdots, c_m\}$. For all $i \in [\![1, n]\!]$, let $k_i^+ \in \mathbb{N}$ (resp. $k_i^- \in \mathbb{N}$) be the number of occurrences of x_i as a positive (resp. negative) literal in φ and $k_i = k_i^+ + k_i^-$. We assume that φ is not an instance of not-all-equal at most 3-SAT(3V) which means that there is at least one variable x_i for which $k_i \geq 4$, $k_i^+ = 0$ or $k_i^- = 0$ and we will build from φ a not-all-equal at most 3-SAT(3V) φ' such that φ and φ' are equisatisfiable as follows.

- For all $i \in [\![1, n]\!]$, if $k_i \geq 4$, $k_i^+ = 0$ or $k_i^- = 0$ then we introduce k_i new variables $\{x_i^1, \cdots, x_i^{k_i}\}$ and for $l \in [\![1, k_i]\!]$ we replace the l-th occurence of x_i in φ with x_i^l.

- For all $i \in [\![1, n]\!]$, if $k_i \geq 4$, $k_i^+ = 0$ or $k_i^- = 0$ then we add k_i new clauses $\{c_{x_i}^1, \cdots, c_{x_i}^{k_i}\}$ where for $l \in [\![1, k_i - 1]\!]$, $c_{x_i}^l = (x_i^l \vee \neg x_i^{l+1})$ and $c_{x_i}^l = (x_i^l \vee \neg x_i^1)$.

Suppose there exists an assignment $v : \{x_1, \cdots, x_n\} \to \{\text{TRUE}, \text{FALSE}\}$ of $x_1, \cdots, x_n$ satisfying φ. Then

$$v' : \begin{array}{ll} x_i \mapsto v(x_i) & \forall i \in [\![1, n]\!] \text{ s.t. } k_i \leq 3,\ k_i^+ > 0 \text{ and } k_i^- > 0; \\ x_i^l \mapsto v(x_i) & \forall i \in [\![1, n]\!] \text{ s.t. } k_i \geq 4,\ k_i^+ = 0 \text{ or } k_i^- = 0 \text{ and } \forall l \in [\![1, k_i]\!]; \end{array}$$

is an assignment of the variables x_i and x_i^l satisfying φ' for

- $\forall j \in [\![1, m]\!]$, the values of the literals of c_j w.r.t. v and v' are piecewise equal so $v'(c_j) = v(c_j) = \text{TRUE}$ and v' is not-all-equal for c_j as well as v is;

- $\forall i \in [\![1, n]\!]$ s.t. $k_i \geq 4$, $k_i^+ = 0$ or $k_i^- = 0$, $\forall l \in [\![1, k_i - 1]\!]$, $v'(x_i^l) = v'(x_i^{l+1}) = v(x_i)$ and $v'(x_i^{k_i}) = v'(x_i^1) = v(x_i)$ so we directly have $\forall l \in [\![1, k_i - 1]\!]$, $v'(c_{x_i}^l) = \text{TRUE}$ and $v'(c_{x_i}^{k_i}) = \text{TRUE}$ and v' is not-all-equal for each of these clauses since they all consist of two literals having opposite values w.r.t. v'.

As an example, for a formula

$$\varphi = (x_1 \vee \neg x_2 \vee x_3) \wedge (\neg x_1 \vee \neg x_3 \vee x_4) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_3 \vee \neg x_4) \wedge (x_1 \vee x_3),$$

where x_1 occurs five times, x_2 occurs only as negative literal and x_3 occurs four times so we add eleven new variables x_1^1, x_1^2, x_1^3, x_1^4, x_1^5, x_2^1, x_2^2, x_3^1, x_3^2, x_3^3 and x_3^4 and eleven new clauses:

$$\begin{aligned}\varphi' =& (x_1^1 \vee \neg x_2^1 \vee x_3^1) \wedge (\neg x_1^2 \vee \neg x_3^2 \vee x_4) \wedge (x_1^3 \vee \neg x_2^2) \wedge (\neg x_1^4 \vee \neg x_3^3 \vee \neg x_4) \wedge (x_1^5 \vee x_3^4) \\ & \wedge (x_1^1 \vee \neg x_1^2) \wedge (x_1^2 \vee \neg x_1^3) \wedge (x_1^3 \vee \neg x_1^4) \wedge (x_1^4 \vee \neg x_1^5) \wedge (x_1^5 \vee \neg x_1^1) \\ & \wedge (x_2^1 \vee \neg x_2^2) \wedge (x_2^2 \vee \neg x_2^1) \\ & \wedge (x_3^1 \vee \neg x_3^2) \wedge (x_3^2 \vee \neg x_3^3) \wedge (x_3^3 \vee \neg x_3^4) \wedge (x_3^4 \vee \neg x_3^1).\end{aligned}$$

Now suppose there exists an assignment v' of the x_i and x_i^l satisfying φ' and let $i \in [\![1, n]\!]$ such that $k_i \geq 4$, $k_i^+ = 0$ or $k_i^- = 0$. If we take a look at the clauses $c_{x_i}^1, \cdots, c_{x_i}^{k_i}$, we notice that if $v'(x_i^1) = \text{FALSE}$ then for $c_{x_i}^1$ to be satisfied, $v'(\neg x_i^2) = \text{TRUE}$, i.e. $v'(x_i^2) = \text{FALSE}$, then for $c_{x_i}^2$ to be satisfied, $v'(\neg x_i^3) = \text{TRUE}$, etc. Repeating this argument, we obtain that if $v'(x_i^1) = \text{FALSE}$ then $v'(x_i^1) = v'(x_i^2) = \cdots = v'(x_i^{k_i}) = \text{FALSE}$. Similarly, if $v'(x_i^{k_i}) = \text{TRUE}$ then for $c_{x_i}^{k_i}$ to be satisfied, $v'(\neg x_i^{k_i - 1}) = \text{FALSE}$, i.e. $v'(x_i^{k_i - 1}) = \text{TRUE}$, then for $c_{x_i}^{k_i - 1}$ to be satisfied, $v'(\neg x_i^{k_i - 2}) = \text{FALSE}$, etc. Hence if $v'(x_i^{k_i}) = \text{TRUE}$ then $v'(x_i^{k_i}) = v'(x_i^{k_i - 1}) = \cdots = v'(x_i^1) = \text{TRUE}$. This yields that

$$\forall i \in [\![1, n]\!] \text{ s.t. } k_i \geq 4,\ k_i^+ = 0 \text{ or } k_i^- = 0,\ v'(x_i^1) = v'(x_i^2) = \cdots = v'(x_i^{k_i}).$$

Hence for all $i \in [\![1, n]\!]$ such that $k_i \geq 4$, $k_i^+ = 0$ or $k_i^- = 0$, we can replace $x_i^1, \cdots, x_i^{k_i}$ by a unique variable x_i and doing so the clauses $c_{x_i}^1, \cdots, c_{x_i}^{k_i}$ become trivial and can

be removed and only φ remains. So the following assignment of $x_1, \cdots, x_n$:

$$v : \begin{array}{ll} x_i \mapsto v'(x_i) & \forall i \in [\![1, n]\!] \text{ s.t. } k_i \leq 3,\ k_i^+ > 0 \text{ and } k_i^- > 0; \\ x_i \mapsto v'(x_i^1) & \forall i \in [\![1, n]\!] \text{ s.t. } k_i \geq 4,\ k_i^+ = 0 \text{ or } k_i^- = 0; \end{array}$$

satisfies φ. We have just shown that φ and φ' are equisatisfiable.

Appendix B

Publications, Conferences and Award

February 2015 **"Nouvelles formulations et bornes pour le problème de la coupe maximum" (extended abstract)**
W. Ben-Ameur, A. Glorieux & J. Neto
16ème Congrès de la Société Française de Recherche Opérationnelle et Aide à la Décision (ROADEF 2015), Marseille (France)

August 2015 **"On the most imbalanced orientation of a graph"**
W. Ben-Ameur, A. Glorieux & J. Neto
Computings & Combinatorics, Proceedings of the 21st International Computing and Combinatorics Conference (COCOON 2015), Beijing (China), LNCS 9198, pp. 16–29

February 2016 **"Graphes et équilibres d'orientations" (extended abstract)**
W. Ben-Ameur, A. Glorieux & J. Neto
17ème Congrès de la Société Française de Recherche Opérationnelle et Aide à la Décision (ROADEF 2016), Compiègne (France)

February 2016 **"Une description complète de polytopes liés à lindice minimum dune ligne non identiquement nulle dune matrice daffectation" (extended abstract)**
W. Ben-Ameur, A. Glorieux & J. Neto
17ème Congrès de la Société Française de Recherche Opérationnelle et Aide à la Décision (ROADEF 2016), Compiègne (France)

May 2016 **SAMOVAR Laboratory (CNRS) PhD student award 2nd prize**
Journée Jeunes Chercheurs SAMOVAR, Évry (France)

May 2016 **"A full description of polytopes related to the index of the lowest nonzero row of an assignment matrix"**
W. Ben-Ameur, A. Glorieux & J. Neto
Combinatorial Optimization, Proceedings of the 4th International Symposium on Combinatorial Optimization (ISCO 2016), Vietri sul Mare (Italy), LNCS 9849, pp. 13-25

August 2016 **"From graph orientation to the unweighted maximum cut problem"**,
W. Ben-Ameur, A. Glorieux & J. Neto,
Computings & Combinatorics, Proceedings of the 22nd International Computing and Combinatorics Conference (COCOON 2016), ho Chi Minh City (Vietnam), LNCS 9797, pp. 370–384

February 2017 **"Graph orientation and some related problems" (extended abstract)**,
W. Ben-Ameur, A. Glorieux & J. Neto
8th International Network Optimization Conference (INOC 2017), Lisbon (Portugal)

February 2017 **"On the most imbalanced orientation of a graph"**
W. Ben-Ameur, A. Glorieux & J. Neto
Journal of Combinatorial Optimization (JOCO), Special Issue: Selected Papers from COCOON 2015, Springer, pp. 1–33


Calina
nec
Adsitus