Universidade Federal da Paraíba
Centro de Informática
Programa de Pós-Graduação em Informática

Graph Declawing Problem: Polyhedra and Exact Solutions

Felipe Crispim Fragoso

Master Thesis submitted for the Programm of MSc on Computer Science of the Universidade Federal da Paraíba as requirement for being acknowledged as a Master's Degree on Computer Science.

Research Field: Computer Science
Research Area: Optimization

Gilberto Farias de Sousa Filho (Supervisor)
Teobaldo Leite Bulhões Júnior (Co-supervisor)

João Pessoa, Paraíba, Brasil

Catalogação na publicação
Seção de Catalogação e Classificação

F811g Fragôso, Felipe Crispim.
Graph Declawing Problem: Polyhedra and Exact Solutions / Felipe Crispim Fragôso. - João Pessoa, 2020.
87 f. : il.

Orientação: Gilberto Farias de Sousa Filho.
Coorientação: Teobaldo Leite Bulhões Júnior.
Dissertação (Mestrado) - UFPB/CI.

1. Graph Declawing Problem. 2. Polyhedral Combinatorics. 3. Branch-and-Cut. 4. Branch-and-Price. I. Filho, Gilberto Farias de Sousa. II. Júnior, Teobaldo Leite Bulhões. III. Título.

UFPB/BC

UNIVERSIDADE FEDERAL DA PARAÍBA
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM INFORMÁTICA

PPGI
INFORMÁTICA • UFPB

Ata da Sessão Pública de Defesa de Dissertação de Mestrado de Felipe Crispim Fragôso, candidato ao título de Mestre em Informática na Área de Sistemas de Computação, realizada em 17 de fevereiro de 2020.

Aos dezessete dias do mês de fevereiro, do ano de dois mil e vinte, às quatorze horas, no Centro de Informática da Universidade Federal da Paraíba, em Mangabeira, reuniram-se os membros da Banca Examinadora constituída para julgar o Trabalho Final do Sr. Felipe Crispim Fragôso, vinculado a esta Universidade sob a matrícula nº 20181000608, candidato ao grau de Mestre em Informática, na área de "Sistemas de Computação", na linha de pesquisa "Computação Distribuída", do Programa de Pós-Graduação em Informática, da Universidade Federal da Paraíba. A comissão examinadora foi composta pelos professores: Gilberto Farias de Sousa Filho (PPGI-UFPB), Orientador e Presidente da Banca, Teobaldo Leite Bulhões Junior (UFPB), Examinador Externo ao Programa, Fabio Protti (UFF), Examinador Externo à Instituição, Thiago Gouveia da Silva (IFPB), Examinador Externo à Instituição. Dando início aos trabalhos, o Presidente da Banca cumprimentou os presentes, comunicou aos mesmos a finalidade da reunião e passou a palavra ao candidato para que o mesmo fizesse a exposição oral do trabalho de dissertação intitulado: "Problema Livre de Garras: Estudo Poliédrico e Soluções Exatas". Concluída a exposição, o candidato foi arguido pela Banca Examinadora que emitiu o seguinte parecer: "aprovado". Do ocorrido, eu, Ruy Alberto Pisani Altafim, Coordenador do Programa de Pós-Graduação em Informática, lavrei a presente ata que vai assinada por mim e pelos membros da banca examinadora. João Pessoa, 17 de fevereiro de 2020.

Prof. Dr. Ruy Alberto Pisani Altafim

Prof. Gilberto Farias de Sousa Filho
Orientador (PPGI-UFPB)

Prof. Teobaldo Leite Bulhões Junior
Examinador Externo ao Programa (UFPB)

Prof. Fabio Protti
Examinador Externo à Instituição (UFF)

Prof. Thiago Gouveia da Silva
Examinador Externo à Instituição (IFPB)

Resumo

O grafo bipartido completo $K_{1,3}$ é uma árvore denominada garra. Um grafo G é considerado Livre de Garra se não houver nenhum subgrafo induzido em G isomorfo ao grafo $K_{1,3}$. O Problema de Eliminação de Garras (PEG) é NP-Completo e consiste em encontrar o conjunto de cardinalidade mínima $S \subseteq V(G)$ tal que G - S é livre de garra. Este trabalho apresenta um estudo poliédrico do politopo associado ao PEG, explicitando sua dimensionalidade e revelando inequações definidoras de facetas. Além disso, instâncias aleatórias e intervalares foram construídas e utilizadas para execução de testes computacionais para quatro algoritmos *Branch-and-Cut* e dois algoritmos *Branch-and-Price* propostos.

Palavras-chave: Problema Livre de Garra, Combinatória Poliédrica, Branch-and-Cut, Branch-and-Price.

Abstract

The complete bipartite graph $K_{1,3}$ is a tree known as claw. A graph is considered to be claw-free if it does not contain any induced subgraph isomorphic to the complete bipartite graph $K_{1,3}$. Consider a graph G, the NP-Hard Graph Declawing Problem (GDP) consists of finding a minimum set $S \subseteq V(G)$ such that G - S is claw-free. This research is a polyhedral study of the GDP polytope, expliciting its full dimensionality, proposing instances and four Branch-and-Cut algorithms with facet inequalities. Alongside that, two Branch-and-Price algorithms are proposed. The results for each algorithm are studied.

Keywords: Graph Declawing Problem, Polyhedral Combinatorics, Branch-and-Cut, Branch-and-Price.

**** I don't know half of you half as well as I should like;*
*and I like less than half of you half as well as you deserve. ****
(Bilbo Baggins)

Acknowledgments

Agradeço ao meu orientador Gilberto Farias por me ajudar na demonstração das facetas, me apresentar a outros bons pesquisadores, ajudar na escrita/revisão dos artigos e pela paciência com meus sumiços ocasionais. Espero algum dia ter metade da capacidade intelectual e de produção desse cara.

Agradeço ao Prof. Anand Subramanian pelo excelente ensino dos conhecimentos básicos para implementar um algoritmo Branch-and-Price e Branch-and-Cut.

Também quero agradecer ao Prof. Teobaldo pela ajuda na implementação do Branch-and-Price no bapcod, ao Prof. Fábio Protti pela ótima revisão/melhoria das provas e Thiago Gouveia pelo pontapé inicial em um possível algoritmo Branch-and-Price para o problema.

Agradeço também a minha mãe Lucia e irmã Patrícia, que mesmo sem entender nada desse trabalho, sempre tiravam um tempo para me perturbar pra ir atrás dos professores 'Giba' (Gilberto) e 'Arnold' (Anand) para 'fazer o negócio direito'.

Agradeço a minha noiva Hozana por me aturar nos momentos que batia a melancolia total e por estar sempre arrumando tempo pra me apoiar mesmo com tantas dificuldades em sua própria jornada.

Contents

List of Figures

List of Tables

Chapter 1

The Graph Declawing Problem

Every graph property Π defines a graph class, namely the set of graphs that satisfy the property. Often in literature, characterizations of a graph with property Π are made by demonstration of its forbidden vertex-induced subgraphs. In such situations, graph modification problems arise.

Graph modification problems are usually in one of the following classes:

- **Vertex Deletion Problems for** Π : Given a Graph G, find the minimum subset $S \subset V(G)$, such that, $G - S$ has the property Π. For example, Boral et al. (2016) study a clustering problem by eliminating P_3 (paths on 3 vertices) subgraphs. Bevern et al. (2010) show 3 forbidden subgraphs for $2-plex$ vertex deletion problem, and Guo et al. (2010) go further and show a characterization of the forbidden subgraphs for the $s - plex$ problem. Lekkeikerker C. (1962) presents forbidden subgraphs for interval graphs.

- **Edge Deletion Problems for** Π : Given a Graph G, find the minimum subset $S \subseteq E(G)$, such that, $G-S$ has the property Π. For example, Guo (2007) presents edge deletion problems for several graph classes with its forbidden subgraphs.

- **Edge Completion Problems for** Π : Given a Graph G, find the minimum subset

$S \subset E(\overline{G})$, such that, $G + S$ has the property Π. For example, Peng e Chen (2006) show a completion problem for interval graphs.

- **Edge Edition Problems for** Π : Given a Graph G, find the minimum subsets $S_1 \subset E(\overline{G})$ and $S_2 \subset E(G)$, such that, $G' = G + S_1 - S_2$ has the property Π. For example, Filho et al. (2017) solve the Bicluster Editing Problem.

- **Edge Contraction Problems for** Π : Given a Graph G, find the minimum subset $S \subset E(G)$, such that, the contraction of S results in a new graph with property Π. For example, Krithika et al. (2016) study some edge contraction problems.

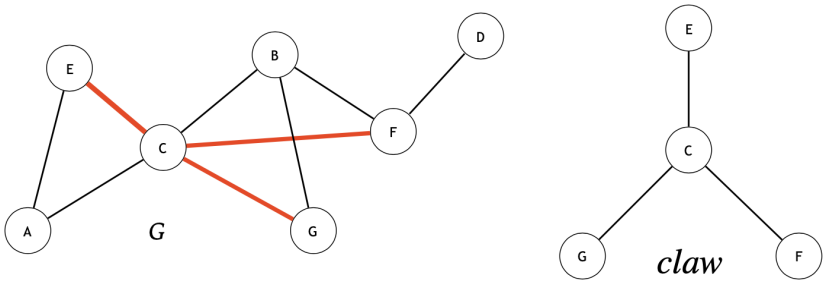
This research focuses on the study of a vertex deletion problem by eliminating claws of some input graph G in order to make G a *claw-free* graph. A Graph G is *claw-free* when there is no vertex-induced subgraph isomorphic to a complete bipartite graph $K_{1,3}$, a tree also called claw (See Figure 1.1 for an example of claw and a graph G with a claw as an induced subgraph). The Graph Declawing Problem (GDP, for short) aims at finding a minimum subset $S \subset V(G)$, such that, $G - S$ is claw-free.

Lewis e Yannakakis (1980) presented a generic proof about vertex deletion problems, showing that any vertex deletion problem on directed/undirected graphs for *non-trivial* and *hereditary* properties Π is NP-Hard. Π is considered to be nontrivial if it is true for infinitely many graphs and false for infinitely many graphs. Π is hereditary if, for any graph G satisfying Π, every vertex-induced subgraph of G also satisfies Π. Clearly, being claw-free is a hereditary and nontrivial property; therefore, the GDP is NP-Hard.

1.1 Motivation

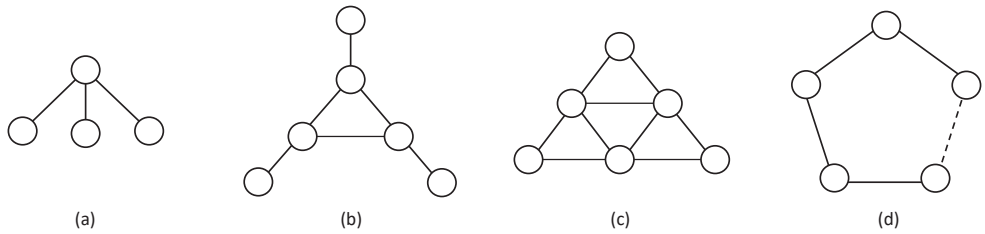
The GDP is one of the required steps for the resolution of the *Unit Interval Vertex Deletion Problem*, which consists of converting an input graph into a unit interval graph

Figure 1.1: Example of Claw in a Graph



by removing a minimum subset of vertices such that the resulting graph is free of the following forbidden induced subgraphs: claws, *nets*, *tents*, and induced cycles C_k ($k \geq 4$) (see Figure 1.2). This problem is studied by Bevern et al. (2010).

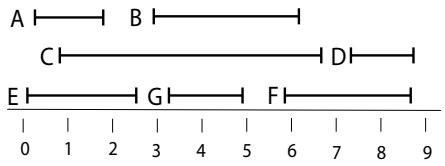
Figure 1.2: Forbidden subgraphs for unit interval graphs: (a) claw, (b) net, (c) tent, (d) $C_k, k \geq 4$.



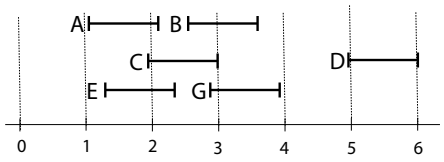
The GDP is also related with the problem of converting an *interval graph* into a *unit interval graph*. An interval graph is the intersection graph of a family of intervals. Formally speaking, G is an interval graph if its vertices can be associated with intervals on the real line such that $uv \in E(G)$ if and only if $I_u \cap I_v \neq \emptyset$, where I_u and I_v are the intervals associated with u and v. A unit interval graph is an interval graph where the intervals can be chosen so that all of them have the same length (or, equivalently, length one). Figure 1.3 shows a transformation from an interval graph to a unit interval graph.

Unit interval graphs coincide with *proper interval graphs* (interval graphs where the

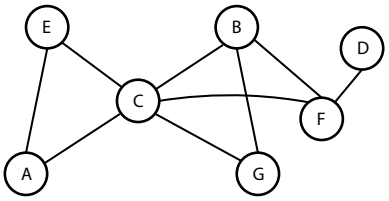
Figure 1.3: Converting an interval graph (c) associated with (a) into a unit interval graph (d) associated with (b) by removing F from (c).



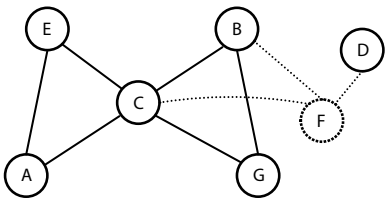
(a) Intervals on the real line.



(b) Unit intervals on the real line.



(c) Interval graph *G*.



(d) A unit interval graph, obtained by removing *F*.

intervals can be chosen so that no interval properly contains another) and *indifference graphs* (defined below). Read Bogart e West (1999) for a proof that G is a unit interval graph if and only if G is a proper interval graph, and Roberts (1979) for a proof that G is a unit interval graph if and only if G is an indifference graph.

Indifference graphs are defined in the context of Social Sciences, as discussed in the work of Roberts (1979). A graph G is an indifference graph if there is a function $f : V(G) \to \mathbb{R}$ such that

$$uv \in E(G) \iff |f(u) - f(v)| \leq \epsilon,$$

where ϵ is a positive number that measures *closeness*. Informally, this means that u and v are indistinguishable items if and only if, according to f, there is a difference at most ϵ between their values. But since item values are in general empirically obtained, there is the possibility that some presumably close items are distinguishable. Therefore, one

may be interested in removing the fewer items possible to generate a real indifference graph and thus solving the Unit Interval Vertex Deletion Problem. Fishburn (1985) poses the problem of converting an *interval order* into an *indifference order* by the removal of some elements; in terms of graphs, this problem is equivalent to the process exemplified in Fig. 1.3.

Properties of claw-free graphs have been deeply investigated by the research community (see the survey by Faudree, Flandrin e Ryjáček (1997)). For example, Minty (1980) presents a polynomial-time algorithm for the problem of finding an independent set with maximum weight in a claw-free graph. HSU e Nemhauser (1982) present a polynomial-time algorithm for the Minimum Weighted Clique Cover Problem in claw-free perfect graphs. To the author's knowledge, there is no work in the literature presenting a polyhedral study of the GDP or a method to solve it via integer programming. Therefore, we believe that the studies developed in this work are fully justified.

1.2 Aim and Objectives

1.2.1 Thesis Aim

This thesis aims at proposing optimal algorithms through mathematical formulations for the GDP.

1.2.2 Study Objectives

1. Polyhedral study for the GDP.
2. Propose mathematical formulations with facets.
3. Propose mathematical formulation for column generation.
4. Compare models using computational experiments.

1.3 Thesis Outline

There are 7 remaining chapters on this thesis. On Chapter 2 there is an introduction to some necessary concepts to understand this thesis. Chapter 3 presents a polytope for the GDP alongside with some of its facet-inducing inequalities. Chapter 4 has four Branch-and-Cut algorithms for solving the GDP. Chapter 5 presents a integer formulation suited for a Branch-and-Price approach. Chapter 6 presents the computational results. Chapter 7 presents the conclusions of this thesis. Chapter 8 shows generalizations for facet-inducing inequalities and Branch-and-Price formulations for vertex deletion problems to eliminate any forbidden subgraph Q, such generalizations were observed to be possible during the study of the mathematical formulations for the GDP.

Chapter 2

Theoretical Background

2.1 Linear and Affine Algebra

Take $v^1 \dots v^m$ as vectors (or points) in $\mathbb{R}^n$. A point v is a *linear combination* of the points $v^1 \dots v^m$, if and only if, there is a $\lambda_1 \dots \lambda_m \in \mathbb{R}$ such that:

$$v = \sum_{i=1}^{m} \lambda_i v^i.$$

If $\sum_{i=1}^{m} \lambda_i = 1$, then v is a *affine combination* of $v^1 \dots v^m$.

A set is *linear independent* if no point in the set can be described as a linear combination of the remaining points. Alternatively, $v^1 \dots v^m$ are linear independent if $\lambda_1 = \dots = \lambda_m = 0$ is the unique solution for $\sum_{i=1}^{m} \lambda_i v^i = 0$. The maximum number of linear independent points in $\mathbb{R}^n$ is n. Similarly, a set of point is *affinely independent* if no point in the set can be described as an affine combination of the remaining points or $\lambda_1 = \dots = \lambda_m = 0$ is the unique satisfying solution for both $\sum_{i=1}^{m} \lambda_i v^i = 0$ and $\sum_{i=1}^{m} \lambda_i = 0$. $n + 1$ is the maximum number of affine points in $\mathbb{R}^n$.

2.2 Polyhedral Theory

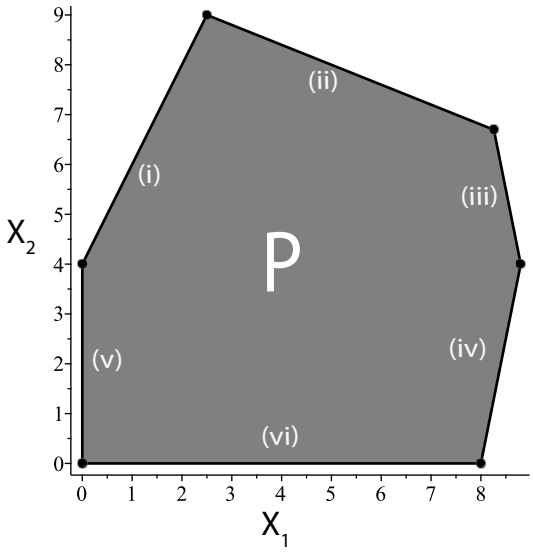
A polyhedra P can be defined by the solution set of a system of linear inequalities. The dimension $dim(P)$ of a polyhedra is k when it has exactly $k+1$ affinely independent points. P is *full dimensional* when $P \in \mathbb{R}^n$ and $dim(P) = n$. The full dimensionality of a polyhedra implies that its facets are uniquely defined by the product of some number α with some inequality.

P can also be described as $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$, where A is a matrix and b is a vector. $Ax \leq b$ is a system of linear inequalities. For example, the inequality system:

$$P = \begin{cases} -2x_1 + x_2 \leq 4, & \text{(i)} \\ 2x_1 + 5x_2 \leq 50, & \text{(ii)} \\ 5x_1 + x_2 \leq 48, & \text{(iii)} \\ 5x_1 - x_2 \leq 40, & \text{(iv)} \\ x_1 \geq 0, & \text{(v)} \\ x_2 \geq 0, & \text{(vi)} \end{cases}$$

Defines the polyhedra P of the Figure 2.1.

Figure 2.1: Polyhedra Example



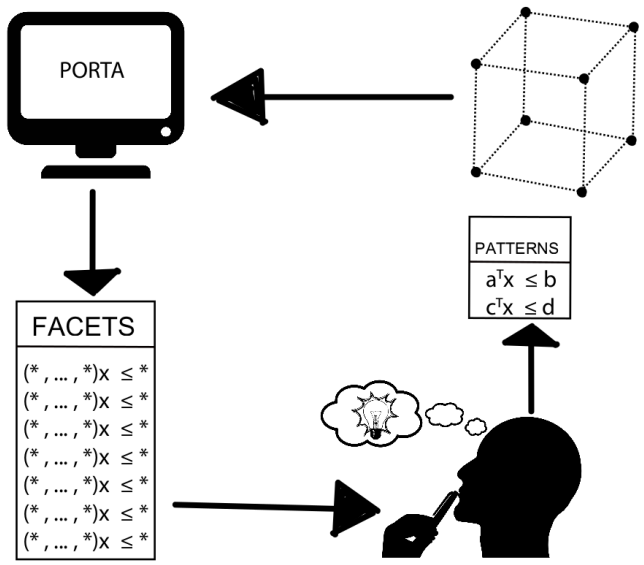
Any inequality in the form $ax \geq b$ can be changed to the form $-ax \leq -b$ multiplying it by -1. In addition, any equality of the form $ax = b$ can be replaced by two inequalities of the form $ax \leq b$ e $ax \geq b$. Therefore, any system of equalities and inequalities can be changed to describe P in the form $P = \{x \in \mathbb{R}^n \mid Ax \leq b\}$.

If $P = \emptyset$, the system of inequalities $\{x \in \mathbb{R}^n \mid Ax \leq b\}$ is *infeasible*. Otherwise, it is *feasible*. If there is some value ω bounding P such that $P \subset [-\omega, \omega]^n$, then P is a *polytope*. A *valid inequality* is an inequality that is satisfied by every point in P.

The set $F = \{x \in P \mid ax = b\}$ is a *face* of P. All faces of a polyhedra are also a polyhedra. Let F be a face of P, if $dim(F) = dim(P) - 1$, then F is *facet* of P. Facet defining inequalities are the inequalities that describe P with the minimum redundancy, thereby find them is always desirable.

Softwares like PORTA (Christof e Schenker) describes a polytope by the list of its internal points. Figure 2.2 resumes the process of identification of facets for a polytope using PORTA. At the beginning, PORTA receives all internal points of the polytope. Next, PORTA lists all facets that describes the polytope. Finally, the user tries to identify the pattern of construction for some of the listed facets.

Figure 2.2: Process of facet identification



2.3 Linear Programming

Linear Programming solves the maximization or minimization of some linear function over a polyhedra $P \in \mathbb{R}^n$. It is usually described as:

$$\begin{aligned} \text{Maximize or Minimize } z &= \sum_{i=1}^{n} c_i x_i \\ \text{s. t.} Ax &\leq b \\ x &\geq 0 \end{aligned}$$

z is the *objective function* and $c_i \in \mathbb{R}$ are the *coefficients* of the objective function. It is common to have integrality or non-negativity constraints in the variables x_i, such constraints are explicitly written on a linear program description. x^* will be used to denotate the optimal solution of a linear program and z^* for the value of the objective function on x^*. Efficient algorithms for linear programming emerged on last decades. State of art *solvers* as CPLEX or GUROBI use the *Simplex* algorithm to solve linear programming. These solvers are capable of solving linear programs with a huge number of variables or inequalities on a few seconds. Simplex algorithm is also used as subroutine for solving integer linear programs.

2.4 Integer Linear Programming

Some real world problems require that a solution for a linear program must be integral. A *Integer Linear Programming* problem arises when this requirement is made for all variables in a linear program. If a problem has all variables as binary integers, then it is a *Binary Integer Programming* problem. Finally, if the problem mixes real and integer variables, it is named *Mixed Integer Programming* problem.

This thesis will discuss the algorithms *Branch-and-Bound*, *Cutting Planes*, *Branch-and-Cut* and *Branch-and-Price* for solving integer programming problems.

2.4.1 *Branch-and-Bound*

Assuming a minimization problem over a linear function z applied on an integer set S, i.e., $z^* = minimize\{\sum_{i=1}^{n} c_i x_i \mid x \in S\}$ must be solved. It is possible to find a solution z^* by separating S on two sets S_1 and S_2 such that $S = S_1 \cup S_2$, and afterwards, solving the problems $z_1^* = minimize\{\sum_{i=1}^{n} c_i x_i \mid x \in S_1\}$ and $z_2^* = minimize\{\sum_{i=1}^{n} c_i x_i \mid x \in S_2\}$. z^* is given by $z^* = minimum\{z_1^*, z_2^*\}$. This divide and conquer idea plays a central key on *Branch-and-Bound*. It is the "*branching*".

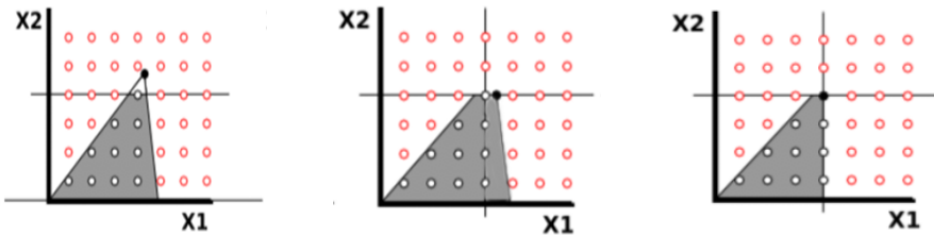
Assuming again the problem $z^* = minimize\{\sum_{i=1}^{n} c_i x_i \mid x \in S\}$. Using simplex it is possible to solve $\overline{z}^* = minimizar\{\sum_{i=1}^{n} c_i x_i \mid x \in P\}$, where P is the polyhedra that emerges from S when the integrality constraints are removed, observe that $S \subset P$. Though the initial problem of determine z^* was not solved, $\overline{z}^*$ is a lower bound for z^*, that means z^* can not be lesser than $\overline{z}^*$. It is possible to find an integral solution for z using the solution $\overline{x}^*$ of $\overline{z}^*$ by making it integral. Take $\overline{x}_i^* = f$ as a coordinate with fractional value, P can be splitted on two new regions P_1 and P_2 where $P_1 = P \cap \{x \mid x_i \leq \lfloor f \rfloor\}$ and $P_2 = P \cap \{x \mid x_i \geq \lceil f \rceil\}$, therefore the algorithm will find all integer points of P by dividing it on new regions and successively dividing them on another regions until integral solutions are found. When the algorithm finds an integral solution on a region P_i, it stores the solution as a candidate solution if it has the best objective function value and use it to “bound” the search, avoiding the exploration of unnecessary integer points of a region. This can be done by comparing the objective function on the best candidate solution z' with the value of the objective function $\overline{z}_j^*$ of the region P_j, if $\overline{z}_j^*$ is greater than z', there is no necessity to search integral points in P_j, because every integral point in P_j has objective value worse than $\overline{z}_j^*$, which has value worse than z'. Therefore, *Branch-and-Bound* searches, implicitly, all integer points on S and by the

end $z^* = z'$.

2.4.2 Cutting Planes

Cutting planes solve integer linear programs of the form $z^* = minimize\{\sum_{i=1}^{n} c_i x_i \mid x \in S\}$ by eliminating fractional solutions from the polyhedra made by the integrality constraints relaxation. Figure 2.3 shows an example of how it works. In (a), the fractional solution $\overline{z}^* = minimize\{\sum_{i=1}^{n} c_i x_i \mid x \in P\}$ was found by simplex. (b) shows the new polyhedra after insertion of a cutting plane that eliminated the previous optimal fractional solution. (c) shows a new cut that reveals the optimal integer solution for S.

Figure 2.3: Cutting Plane Algorithm



(a) Initial Polyhedra (b) Added Cut (c) Integer Solution

Reference: Cunha, N. S. Um Framework para a geração semiautomática de solos de guitarra. Dissertação (Mestrado em Informática) – Universidade Federal da Paraíba. 2016.

These cuts must be *valid*. They must not remove integral solutions on S.

2.4.3 *Branch-and-Cut*

The *Branch-and-Cut* algorithm applies cutting planes on each polyhedral P_i of a *Branch-and-Bound* to make it converge faster.

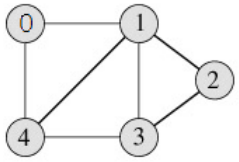
2.4.4 *Branch-and-Price*

Branch-and-Price is a *Branch-and-Bound* where variables are added on each problem P_i. On big linear programming problems, most variables are 0-valued on the optimal solution. Such variables are irrelevant for the simplex algorithm. Therefore, a simplified version of the problem, named *Master Problem*, take just a few variables on its formulation, but this leads to a new problem of detecting when a solution of the master problem is optimal. Then, a new algorithm named *Pricing* finds new variables not present on the master that improves its objective function value.

2.5 Graph Theory

A graph G is a data structure with $V(G)$ representing a set of vertices and $E(G)$ representing a set of edges connecting these vertices. Figure 2.4 shows a graph with $V(G) = \{0, 1, 2, 3, 4\}$ and $E(G) = \{01, 04, 12, 13, 14, 23, 34\}$.

Figure 2.4: Graph Example

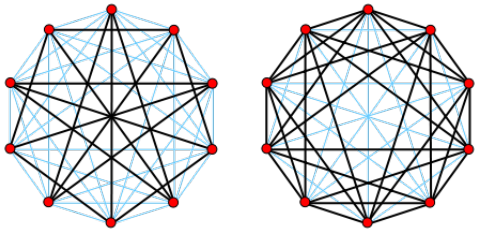


Reference: Cormen, Thomas H. et al. Algoritmos: Teoria e prática

2.5.1 Complementary Graphs

The *complementary graph* $\overline{G}$ of G, showed at Figure 2.5, is obtained by adding all edges that are not in $E(G)$ in the set $E(\overline{G})$.

Figure 2.5: Complementary Graphs



Reference: Retrieved from URL[1]

Black lines are the real edges. Blue lines are the complementary edges.

2.5.2 Subgraph

A *subgraph* G' of a graph G is a graph such that $V(G') \subseteq V(G)$ and $E(G') \subseteq E(G)$. A subgraph G'of G is an *induced by vertex subgraph*, for convenience notated as $G[V(G')]$, if all edges $uw \in E(G)$ with $u \in V(G')$ and $w \in V(G')$ are on $E(G')$. In addition, we may just write subgraphs formed by the elimination of vertices or edges in G as $G' = G - S$ where $S \subset V(G)$ or $S \subset E(G)$. The red portion of the Figure 2.6 shows a subgraph (a) and an induced subgraph (b), observe that (b) is induced by the inclusion of two missing edges on (a).

Figure 2.6: Subgraph Examples

(a) Subgraph (b) Vertex-induced Subgraph

Reference: Retrieved from URL[2]

1 - Available on: <pt.wikipedia.org/wiki/Grafo_complementar>
2 - Available on: <www.math.ucdenver.edu/~wcherowi/courses/m4408/gtaln2.html>

2.5.3 Supergraphs

G' is supergraph of G when G is a subgraph of G'. For convenience, we may just write supergraphs formed by the adition of edges in G as $G' = G + S$ where $S \subset E(\overline{G})$.

2.5.4 Edge Contraction

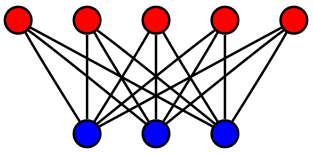
The edge contraction operation occurs relative to a particular edge e. The edge e is removed and its two incident vertices, u and v, are merged into a new vertex w, where each edge incident to w correspond to an edge incident to either u or v.

An edge contraction operation may result in a graph with parallel edges even if the original graph is a simple graph. Nevertheless, it is possible to disallow the creation of multiple edges, so that edge contractions performed on simple graphs always produce simple graphs.

2.5.5 Bipartite Graph

A graph G is *bipartite* when its possible to find X and Y such that for all edges $uw \in E(G)$, $u \in X$ and $w \in Y$. That means that every vertex on X is neighbor of a vertex on Y. Bipartite graphs can also be *complete* if all vertices on X are neighbors of all vertices on Y. Complete bipartite graphs are denoted as $K_{\|X\|,\|Y\|}$.

Figure 2.7: Bipartite Graph $K_{5,3}$



Reference: Retrieved from URL[3]

3 - Available on: <https://pt.wikipedia.org/wiki/Grafo_bipartido_completo>. Visited on 1th June 2019.

2.5.6 Clique and Independent Set

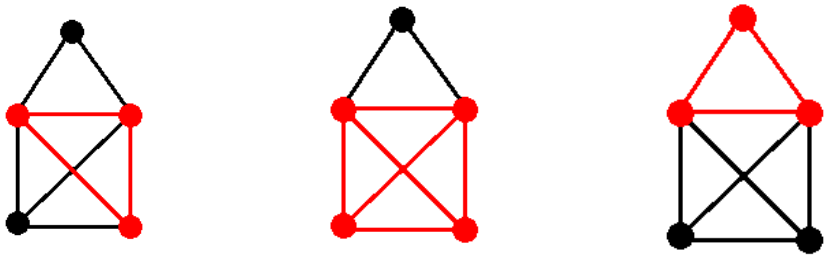
A *clique* C is a complete subgraph G' of G.

Maximal cliques are the set of cliques on G that can not be extended by a new vertex $u \in V(G)$.

Maximum cliques are the set of maximal cliques on G with the greatest size $\|V(G')\|$ possible.

Figure 2.8 shows, on red colors, a maximal and maximum clique. Subfigure (a) presents a non maximal clique, because a vertex can extend it to the maximum and maximal clique on (b). (c) presents a maximal clique that is not maximum.

Figure 2.8: Cliques



(a) Non-maximal Clique (b) Maximum Clique (c) Maximal Clique

Reference: Retrieved from URL [4]

The *independent set* I_k of G is a set of k vertices where each pair $uw \in I_k$ is not in $E(G)$. Like cliques, independent sets can be *maximal* or *maximum*. The maximum is the greatest I_k in G, similarly, the maximal independent set can not be extended by the inclusion of some vertex. Independent sets and cliques are related problems, since to find maximum/maximal cliques on a graph G can be reduced to find maximum/minimum independent sets on the complementary graph $\overline{G}$. Figure 2.9 shows an independent set example G (Subfigure (a)) and a clique constructed with same vertices on $\overline{G}$ (Subfigure (b)).

4 - Available on: <https://i.stack.imgur.com/MrlSG.png>. Visited on 1th June 2019.

Figure 2.9: Independent Set and Clique

(a) G (b) $\overline{G}$

Reference: Retrieved from URL[5]

5 - Available on: <https://en.wikipedia.org/wiki/Maximal_independent_set>. Visited on 1th June 2019.

Chapter 3

Facet-Inducing Inequalities for the Polytope of the GDP

3.1 Polytope for the GDP

From now on, G denotes a graph with $|V(G)| = \{1, \ldots, n\}$. Let P_G be the convex hull of incidence vectors χ^H that represent claw-free subgraphs H of a graph G, that is:

$$P_G = conv\{\chi^H \in \{0,1\}^n \mid H \text{ is a claw-free subgraph of } G\}.$$

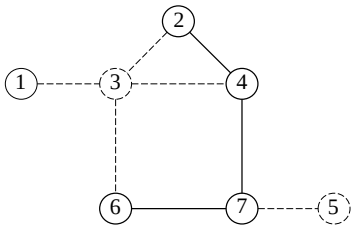
Say that P_G is the *GDP polytope* of G. Figure 3.1 shows a claw-free subgraph H induced by $\{1, 2, 4, 6, 7\}$ and its incidence vector.

The GDP can be described as an optimization problem over its polytope:

$$\min x^T w$$
$$s.t. \quad x \in P_G$$

In order to apply binary integer programming techniques for the GDP, mathematical

Figure 3.1: (a) A claw-free subgraph $H = G[\{1, 2, 4, 6, 7\}]$ and (b) its incidence vector.



(a) Claw-free subgraph H

$$\chi^H = \begin{vmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \\ 1 \end{vmatrix}$$

(b) Incidence Vector $\chi^H \in \{0,1\}^7$

formulations of P_G are required. Clearly, P_G is n-dimensional and contains the null and all the unitary vectors, summing up to $n + 1$ affinely independent vectors. This means that P_G is full dimensional.

3.2 Trivial Inequalities

Theorem 3.2.1. *Following items holds for P_G.*

1. *Binary Inequalities associated with a vertex $0 \leq x_v$ and $x_v \leq 1$ are valid;*

2. *$0 \leq x_v$ is a facet-inducing inequality;*

3. *$x_v \leq 1$ is a facet-inducing inequality;*

Proof.

1. Since all the coordinates of the incidence vectors of claw-free subgraphs are binary, it is clear that the trivial inequalities are valid.

2. $x_v = 0$ is satisfied by the null vector and all the unit vectors with $\chi^H_u = 1, u \neq v$. These n vectors belong to P_G and are affinely independent.

3. $x_v = 1$ is satisfied by the unit vector $\chi^H_v = 1$ and by the vectors $\chi^H_v = 1, \chi^H_u = 1, \chi^H_t = 0$ where $u \neq v$ and $t \notin \{u, v\}$. These n vectors belong to P_G and are affinely

independent.

□

3.3 Claw Inequalities

For simplicity, take $abcd$ as a claw subgraph with central vertex c (the vertex with degree three). In addition, for a subset S of vertices, we may just write χ^S to mean the incidence vector of the induced subgraph $G[S]$. Let

$$x_a + x_b + x_c + x_d \leq 3 \tag{3.1}$$

be the *claw inequality* associated with a claw subgraph of G.

Theorem 3.3.1. *Following items holds for P_G.*

1. *The claw inequality is valid.*
2. *The claw inequality is facet-inducing.*

Proof.

1. Immediate. A claw inequality forces at least one of x_a, x_b, x_c, x_d to be zero.

2. Let w be a vector such that $w_i = 1$ for $i \in \{a, b, c, d\}$ and $w_i = 0$ for $i \notin \{a, b, c, d\}$. Let $u^T x \leq u_0$ be a facet-inducing inequality for the GDP polytope such that $F_a = \{\chi^H \in P_G \mid w^T \chi^H = 3\} \subseteq F_b = \{\chi^H \in P_G \mid u^T \chi^H = u_0\}$. Clearly, $F_a \neq P_G$ and $F_a \neq \emptyset$. Thus, a proof that $u = \alpha w$ for some positive $\alpha \in \mathbb{R}$ shows that F_a is facet of P_G.

For a claw subgraph $abcd$, the incidence vectors $\chi^{\{a,b,c\}}$, $\chi^{\{a,b,d\}}$, $\chi^{\{b,c,d\}}$, and $\chi^{\{a,c,d\}}$ are in $F_a \subseteq F_b$. Then, the following equalities hold: $0 = u_0 - u_0 = u^T \chi^{\{a,b,c\}} -$

$u^T\chi^{\{a,b,d\}} = u_c - u_d$; $0 = u_0 - u_0 = u^T\chi^{\{a,b,c\}} - u^T\chi^{\{a,c,d\}} = u_b - u_d$; $0 = u_0 - u_0 = u^T\chi^{\{a,b,c\}} - u^T\chi^{\{b,c,d\}} = u_a - u_d$. This implies $u_a = u_b = u_c = u_d = \alpha$.

For a node $v \notin \{a, b, c, d\}$, if its neighborhood satisfies $|N(v) \cap \{a, b, d\}| \leq 2$ then $\chi^{\{a,b,d,v\}} \in F_a \subseteq F_b$. Then, the following holds: $0 = u_0 - u_0 = u^T\chi^{\{a,b,d,v\}} - u^T\chi^{\{a,b,d\}} = u_v$; That means $u_v = 0$.

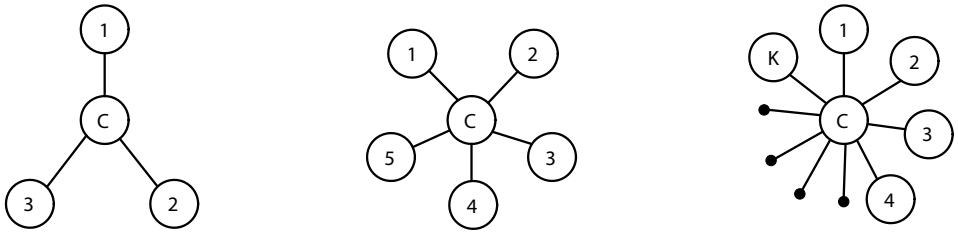
Finally, for a vertex $v \notin \{a, b, c, d\}$ such that $|N(v) \cap \{a, b, d\}| = 3$, $\chi^{\{a,b,c,v\}} \in F_a \subseteq F_b$. Then, the following holds: $0 = u_0 - u_0 = u^T\chi^{\{a,b,c,v\}} - u^T\chi^{\{a,b,c\}} = u_v$. □

3.4 Star Inequalities

Definition 3.4.1. *A* star graph S_k *is a complete bipartite graph* $K_{1,k}$*, for* $k \geq 3$*.* S_k *has a central vertex* c *whose neighborhood is an independent set* I_k *of size* k*.*

Possible topologies for S_k are illustrated in Figure 3.2.

Figure 3.2: Star graphs S_3, S_5, and S_k.



(a) S_3 (b) S_5 (c) S_k

A *star subgraph* is an induced subgraph of G isomorphic to a star graph.

Theorem 3.4.1. *Let* S_k *be a star subgraph, and consider the corresponding* star inequality*:*

$$\sum_{v \in I_k} x_v + (k-2)x_c \leq k, \tag{3.2}$$

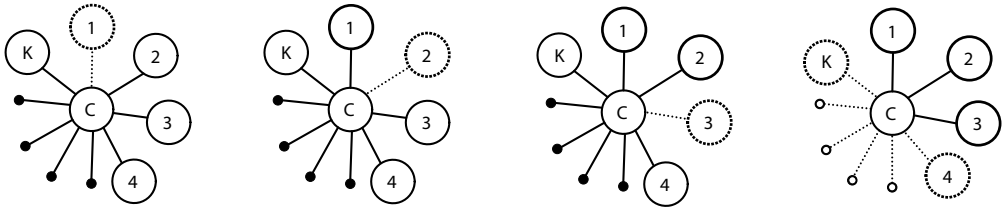
Then:

1. *The star inequality is valid for P_G.*

2. *The star inequality is facet-inducing for P_G.*

Proof. (1) The case $k = 3$ corresponds to the claw inequality. For $k \geq 4$, the proof is done by induction. Let $S_k^v = S_k - v$ and $S_3' = S_k[\{i, j, \ell, c\}]$, where i, j, ℓ are distinct vertices in I_k. Note that S_k^i, S_k^j, S_k^ℓ, and S_3' are subgraphs of S_k. See Figure 3.3, where $i = 1, j = 2, \ell = 3$.

Figure 3.3: Subgraphs S_k^i, S_k^j, S_k^ℓ, and S_3' of S_k.



(a) S_k^i (b) S_k^j (c) S_k^ℓ (d) S_3'

By induction, the inequalities associated with S_k^i, S_k^j, S_k^ℓ, and S_3' are valid for P_G, and summing them up leads to the valid inequality

$$\sum_{v \in I_k} 3x_v + (3k - 8)x_c \leq 3k.$$

Adding $2x_c \leq 2$ to the above inequality gives

$$\sum_{v \in I_k} 3x_v + (3k - 6)x_c \leq 3k + 2,$$

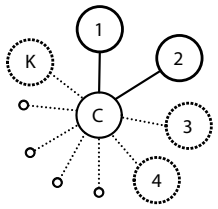
and, therefore,

$$\sum_{v \in I_k} x_v + (k-2)x_c \leq k + \left\lfloor \frac{2}{3} \right\rfloor = k$$

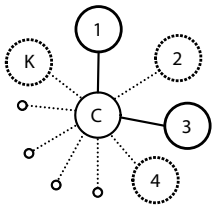
is valid for P_G.

(2) Let w be a vector such that $w_c = k-2$, $w_i = 1$ for $i \in I_k$, and $w_v = 0$ for $v \notin I_k \cup \{c\}$. Let $u^T x \leq u_0$ be a facet-inducing inequality for the GDP polytope P_G such that $F_a = \{\chi^H \in P_G \mid w^T \chi^H = k\} \subseteq F_b = \{\chi^H \in P_G \mid u^T \chi^H = u_0\}$. Clearly, $F_a \neq P_G$ and $F_a \neq \emptyset$. Thus, a proof that $u = \alpha w$ for some $\alpha \in \mathbb{R}$ shows that F_a is facet-inducing for P_G. Let $L_{i,j}$ be a claw-free subgraph of S_k such that $L_{i,j} = S_k[\{i, j, c\}]$, where $\{i, j\} \subset I_k$ and c is the center of S_k. Figure 3.4 shows two examples of such subgraphs.

Figure 3.4: Subgraphs $L_{1,2}$ and $L_{1,3}$ of S_k.



(a) $L_{1,2}$



(b) $L_{1,3}$

The incidence vectors χ^{I_k}, $\chi^{L_{i,j}}$, and $\chi^{L_{j,k}}$ are in $F_a \subseteq F_b$. Then, the following equalities hold: $0 = u_0 - u_0 = u^T \chi^{L_{i,j}} - u^T \chi^{L_{j,k}} = u_i - u_k$. Therefore, $u_i = u_k = \alpha$. By applying this process to all $v \in I_k$ we get $u_v = \alpha$. Additionaly, $0 = u_0 - u_0 = u^T \chi^{I_k} - u^T \chi^{L_{i,j}} = (\sum_{v \in I_k \setminus \{i,j\}} u_v) - u_c$, and this implies $u_c = \alpha(k-2)$.

For a node $v \notin I_k \cup \{c\}$, if its neighborhood satisfies $|N(v) \cap I_k| < 3$ then $\chi^{I_k \cup \{v\}} \in F_a \subseteq F_b$. Thus, the following equality holds: $0 = u_0 - u_0 = u^T \chi^{I_k \cup \{v\}} - u^T \chi^{I_k} = u_v$.

Finally, for a node $v \notin I_k \cup \{c\}$ such that $|N(v) \cap I_k| \geq 3$, the incidence vec-

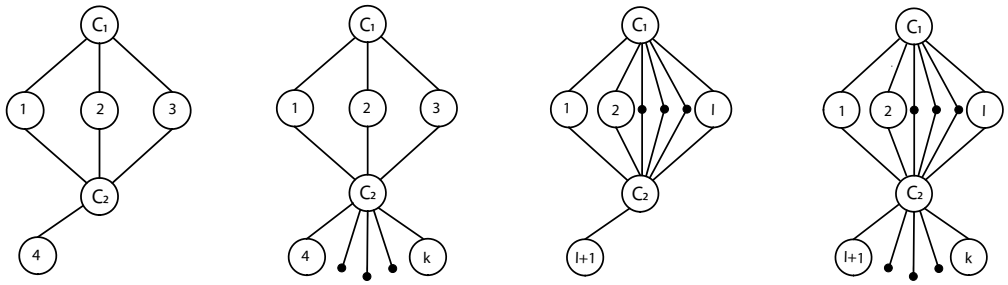
tor $\chi^{V(L_{i,j})\cup\{v\}}$, where $\{i,j\} \subset N(v) \cap I_k$, is in $F_a \subseteq F_b$. Then, $0 = u_0 - u_0 = u^T\chi^{V(L_{i,j})\cup\{v\}} - u^T\chi^{L_{i,j}} = u_v$.

□

3.5 Lantern inequalities

Definition 3.5.1. *Let S_k and S_ℓ, $k > \ell \geq 3$, be two star graphs such that S_ℓ has central node c_1 and independent set I_ℓ, S_k has central node c_2 and independent set I_k, and $I_\ell \subset I_k$. A* lantern graph *$L_{\ell k}$ is the union of S_ℓ and S_k. Figure 3.5 shows some topologies of lantern graphs.*

Figure 3.5: Lantern graphs L_{34}, L_{3k}, $L_{\ell(\ell+1)}$, and $L_{\ell k}$.



(a) L_{34} (b) L_{3k} (c) $L_{\ell(\ell+1)}$ (d) $L_{\ell k}$

A *lantern subgraph* is an induced subgraph isomorphic to a lantern graph.

Theorem 3.5.1. *Let $L_{\ell k}$ be a lantern subgraph, and consider the corresponding* lantern inequality*:*

$$\sum_{v\in I_k} x_v + (\ell-2)x_{c_1} + (k-\ell)x_{c_2} \leq k. \tag{3.3}$$

Then:

1. *The lantern inequality is valid for P_G.*

2. *The lantern inequality induces a facet of P_G, except when $k = \ell + 1$ and there is a vertex $v \notin V(L_{\ell k})$ such that $N(v) \cap I_k = I_k$ and $N(v) \cap \{c_1, c_2\} = \emptyset$.*

Proof.

1. The validity of the lantern inequality for $\ell = 3$ and $k = 4$ can be checked by adding the star inequalities associated with the subgraphs $S'_3 = L_{34}[\{1, 2, 3, c_1\}]$ and $S'_4 = L_{34}[\{1, 2, 3, 4, c_2\}]$ and the inequality $x_{c_1} + x_4 \leq 2$, obtaining the following valid inequality:

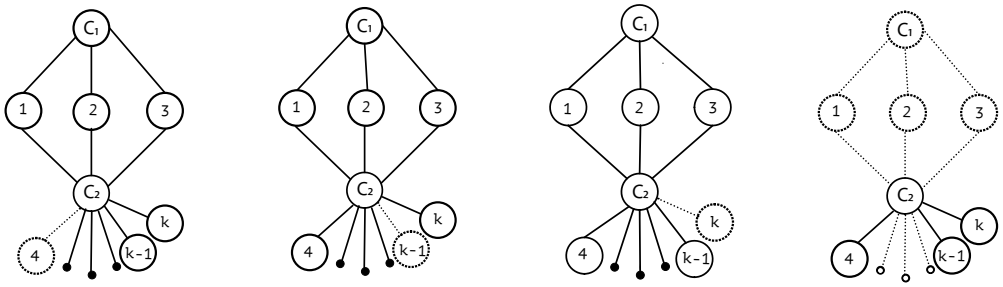
$$\sum_{v \in I_4} x_v + x_{c_1} + x_{c_2} \leq 4 + \left\lfloor \frac{1}{2} \right\rfloor = 4.$$

Now, consider $L^v_{\ell k} = L_{\ell k} - v$. The case $\ell = 3, k = 5$ can be checked by adding the inequalities associated with subgraphs L^4_{35}, L^5_{35} (where $\{4, 5\} \subset I_5$), $S'_3 = L_{35}[\{1, 4, 5, c_2\}]$, $S'_5 = L_{35}[\{1, 2, 3, 4, 5, c_2\}]$ and the valid inequality $x_{c_1} - x_1 \leq 1$, obtaining the following inequality:

$$\sum_{v \in I_5} x_v + x_{c_1} + 2x_{c_2} \leq 5 + \left\lfloor \frac{2}{3} \right\rfloor = 5.$$

For the case $\ell = 3$ and $k > 5$, the proof is done by induction. Note that L^i_{3k}, L^j_{3k}, L^h_{3k}, and $S'_3 = L_{3k}[\{i, j, h, c_2\}]$, for distinct $i, j, h \in I_k \setminus I_3$, are subgraphs of L_{3k}. Figure 3.6 shows subgraphs L^i_{3k}, L^j_{3k}, L^h_{3k}, and S'_3 for $i = 4, j = k - 1, h = k$.

Figure 3.6: Subgraphs of L_{3k}, $k > 5$.



(a) L^i_{3k} (b) L^j_{3k} (c) L^h_{3k} (d) S'_3

By induction, the inequalities associated with L^i_{3k}, L^j_{3k}, L^h_{3k}, and S'_3 are valid for P_G, and summing them up leads to the valid inequality

$$3\sum_{v\in I_k} x_v + 3(3-2)x_{c_1} + (3k-11)x_{c_2} \leq 3k.$$

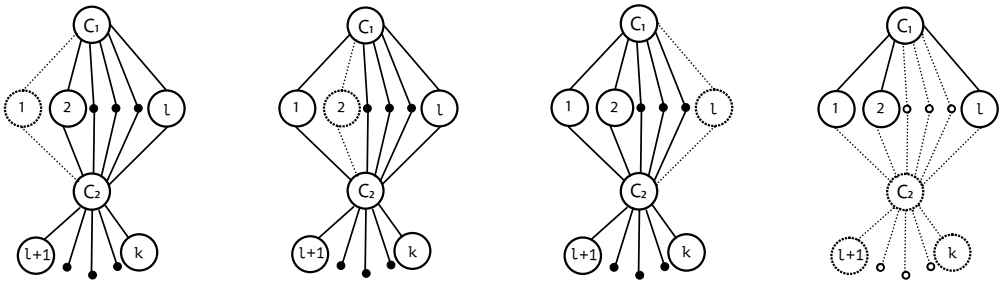
Adding $2x_{c_2} \leq 2$, we get that

$$\sum_{v\in I_k} x_v + (3-2)x_{c_1} + (k-3)x_{c_2} \leq k + \left\lfloor \frac{2}{3} \right\rfloor = k$$

is valid for P_G.

For the case $\ell > 3$ and $k \geq \ell + 1$, note that $L^i_{\ell k}$, $L^j_{\ell k}$, $L^h_{\ell k}$, and $S'_3 = L_{\ell k}[\{i, j, h, c_1\}]$, for distinct $i, j, h \in I_\ell$, are subgraphs of $L_{\ell k}$. Figure 3.7 shows subgraphs $L^i_{\ell k}$, $L^j_{\ell k}$, $L^h_{\ell k}$, and S'_3 for $i = 1, j = 2, h = \ell$.

Figure 3.7: Subgraphs of $L_{\ell k}$ for $l > 3$, $k \geq \ell + 1$.



(a) $L^i_{\ell k}$ (b) $L^j_{\ell k}$ (c) $L^h_{\ell k}$ (d) S'_3

By induction, the inequalities associated with $L^i_{\ell k}$, $L^j_{\ell k}$, $L^h_{\ell k}$, and S'_3 are valid for P_G, and summing them up leads to

$$3\sum_{v \in I_k} x_v + (3\ell - 8)x_{c_1} + 3(k - \ell)x_{c_2} \leq 3k.$$

Adding $2x_{c_1} \leq 2$ we get that

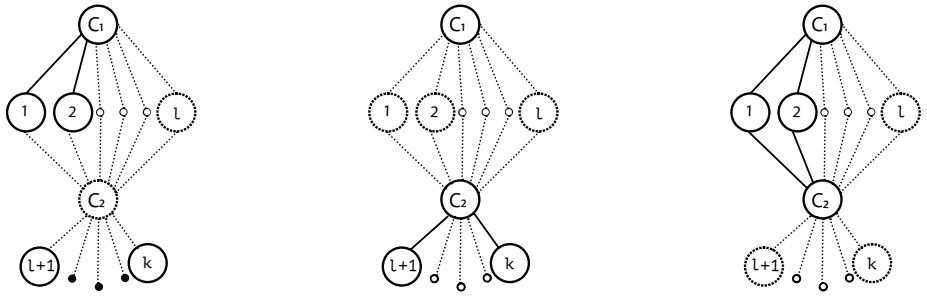
$$\sum_{v \in I_k} x_v + (\ell - 2)x_{c_1} + (k - \ell)x_{c_2} \leq k + \left\lfloor \frac{2}{3} \right\rfloor = k$$

is valid for P_G.

2. Let w be a vector such that $w_{c_1} = \ell - 2$, $w_{c_2} = k - \ell$, $w_i = 1$ for $i \in I_k$, and $w_v = 0$ for $v \notin V(L_{\ell k})$. Let $u^T x \leq u_0$ be a facet-inducing inequality for the GDP polytope P_G such that $F_a = \{\chi^H \in P_G \mid w^T \chi^H = k\} \subseteq F_b = \{\chi^H \in P_G \mid u^T \chi^H = u_0\}$. Clearly, $F_a \neq P_G$ and $F_a \neq \emptyset$. Thus, a proof that $u = \alpha w$ for some $\alpha \in \mathbb{R}$ shows that $w^T \chi^H = k$ defines a facet of P_G.

Let $N_{i,j}$ and $M_{i,j}$ be claw-free subgraphs of $L_{\ell k}$ such that $N_{i,j} = L_{\ell k}[(I_k \setminus I_\ell) \cup \{i, j, c_1\}]$ and $M_{i,j} = L_{\ell k}[\{i, j, c_1, c_2\}]$, where $i, j \in I_k$, $i \neq j$. Figure 3.8 shows examples of subgraphs $N_{i,j}$ and $M_{i,j}$.

Figure 3.8: Subgraphs of $L_{\ell k}$.



(a) $N_{1,2}$ (b) $M_{\ell+1,k}$ (c) $M_{1,2}$

Then:

1. The incidence vectors $\chi^{M_{i,j}}$ and $\chi^{M_{j,h}}$ with $\{i,j,h\} \subset I_k$ are in $F_a \subseteq F_b$. Thus, $0 = u_0 - u_0 = u^T\chi^{M_{i,j}} - u^T\chi^{M_{j,h}} = u_i - u_h$. This implies $u_i = u_h = \alpha$ for all choices of $\{i,j,h\} \subset I_k$.

2. The incidence vectors χ^{I_k} and $\chi^{N_{i,j}}$ with $\{i,j\} \subset I_\ell$ are in $F_a \subseteq F_b$. Thus, $0 = u_0 - u_0 = u^T\chi^{I_k} - u^T\chi^{N_{i,j}} = (\sum_{v \in I_\ell \setminus \{i,j\}} u_v) - u_{c_1}$. This implies $u_{c_1} = \alpha(\ell - 2)$.

3. The incidence vectors χ^{I_k} and $\chi^{M_{i,j}}$ with $\{i,j\} \subset I_k$ are in $F_a \subseteq F_b$. Thus $0 = u_0 - u_0 = u^T\chi^{I_k} - u^T\chi^{M_{i,j}} = (\sum_{v \in I_k \setminus \{i,j\}} u_v) - u_{c_1} - u_{c_2}$. This implies $u_{c_2} = \alpha(k-2) - \alpha(\ell - 2) = \alpha(k - \ell)$.

For a node $v \notin V(L_{\ell k})$ such that $|N(v) \cap I_k| < 3$, $\chi^{\{v\} \cup I_k}$ is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T\chi^{\{v\} \cup I_k} - u^T\chi^{I_k} = u_v$.

For a node $v \notin V(L_{\ell k})$ such that $|N(v) \cap I_k| > 3$, we have that:

1. If $k = \ell + 1$ and $k \notin N(v)$, $\chi^{\{v\} \cup V(N_{i,j})}$ with $i \in I_\ell \cap N(v)$ and $j \in I_\ell$ is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T\chi^{\{v\} \cup V(N_{i,j})} - u^T\chi^{N_{i,j}} = u_v$.

2. If $k = \ell + 1$, $k \in N(v)$, and $N(v) \cap I_k \neq I_k$, $\chi^{\{v\} \cup V(N_{i,j})}$ with $i \in I_\ell \cap N(v)$ and $j \notin I_\ell \cap N(v)$ is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T \chi^{\{v\} \cup V(N_{i,j})} - u^T \chi^{N_{i,j}} = u_v$.

3. If $k = \ell + 1$, $N(v) \cap I_k = I_k$, and $c_1 \in N(v)$ or $c_2 \in N(v)$, $\chi^{\{v\} \cup V(M_{i,j})}$ with $i \in I_\ell$ and $j = k$ is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T \chi^{\{v\} \cup V(M_{i,j})} - u^T \chi^{M_{i,j}} = u_v$.

4. If $k > \ell + 1$ and there is no vertex $i \in (I_k \setminus I_\ell) \cap N(v)$, $\chi^{\{v\} \cup V(N_{i,j})}$ with $i \in I_\ell \cap N(v)$ and $j \in I_\ell$ is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T \chi^{\{v\} \cup V(N_{i,j})} - u^T \chi^{N_{i,j}} = u_v$.

5. If $k > \ell + 1$, $c_1 \notin N(v)$, and there is a vertex $i \in (I_k \setminus I_\ell) \cap N(v)$, the incidence vector $\chi^{\{v\} \cup V(M_{i,j})}$ is in $F_a \subseteq F_b$, for $j \in I_k \setminus I_\ell$. Then, the following equality holds: $0 = u_0 - u_0 = u^T \chi^{\{v\} \cup V(M_{i,j})} - u^T \chi^{M_{i,j}} = u_v$.

6. If $k > \ell + 1$, $c_1 \in N(v)$, and there is a vertex $i \in (I_k \setminus I_\ell) \cap N(v)$, the incidence vector $\chi^{\{v\} \cup V(M_{i,j})}$ is in $F_a \subseteq F_b$, for $j \in I_\ell$. Then, the following equality holds: $0 = u_0 - u_0 = u^T \chi^{\{v\} \cup V(M_{i,j})} - u^T \chi^{M_{i,j}} = u_v$.

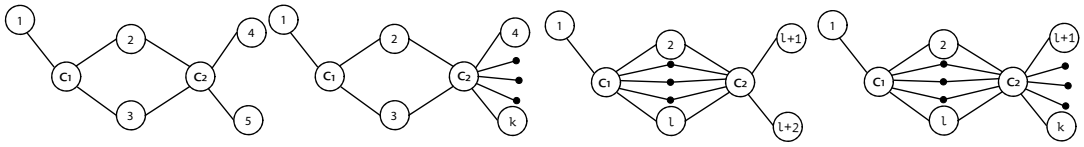
Note that the only case not covered by the above items (a)-(f) is $k = \ell + 1$, $N(v) \cap I_k = I_k$, and $N(v) \cap \{c_1, c_2\} = \emptyset$.

□

3.6 Binary star inequalities

Definition 3.6.1. *Let S_ℓ and S_k be two star graphs, $k > \ell \geq 3$, such that S_ℓ has central node c_1 with independent set I_ℓ, S_k has central node c_2 with independent set I_k, $|I_\ell \cap I_k| \geq 2$, and $|I_\ell \setminus I_k| = 1$. A* binary star graph *$B_{\ell k}$ is the union of S_ℓ and S_k. Figure 3.9 shows some topologies of binary star graphs.*

Figure 3.9: Examples of binary stars.



(a) B_{34} (b) $B_{3(k-1)}$ (c) $B_{\ell(\ell+1)}$ (d) $B_{\ell(k-1)}$

A *binary star subgraph* is an induced subgraph isomorphic to a binary star graph.

Theorem 3.6.1. *Let $B_{\ell k}$ be a binary star subgraph, and consider the corresponding* binary star inequality*:*

$$\sum_{v \in I_k \cup I_\ell} x_v + (\ell - 2)x_{c_1} + (k - \ell)x_{c_2} \leq k + 1. \tag{3.4}$$

Then:

1. *The binary star inequality is valid for P_G.*
2. *The binary star inequality induces a facet of P_G, except when there is a node v such that $N(v) \cap (I_k \cup I_\ell) = I_k \cup I_\ell$.*

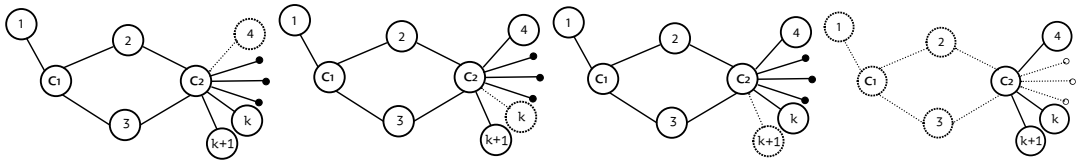
Proof.

1. For $\ell = 3$ and $k = 4$, the validity can be checked by summing up the claw inequality associated with $S'_3 = B_{34}[\{1, 2, 3, c_1\}]$, the star inequality associated with $S'_4 = B_{34}[\{2, 3, 4, 5, c_2\}]$, and the valid inequality $x_1 + x_4 + x_5 + x_{c_1} \leq 4$, yielding the valid inequality:

$$\sum_{v \in I_3 \cup I_4} x_v + x_{c_1} + x_{c_2} \leq 5 + \left\lfloor \frac{1}{2} \right\rfloor = 5.$$

For $\ell = 3$ and $k \geq 5$, the proof is done by induction. Let $B^v_{\ell k} = B_{\ell k} - v$. Note that B^i_{3k}, B^j_{3k}, B^h_{3k}, and $S'_3 = B_{3k}[\{i, j, h, c_2\}]$, for distinct $i, j, h \in I_k \setminus I_\ell$, are subgraphs of B_{3k}. Figure 3.10 shows subgraphs B^i_{3k}, B^j_{3k}, B^h_{3k}, and S'_3 for $i = 4$, $j = k$, and $h = k + 1$.

Figure 3.10: Subgraphs of B_{3k}.



(a) B^i_{3k} (b) B^j_{3k} (c) B^h_{3k} (d) S'_3

By induction, the inequalities associated with B^i_{lk}, B^j_{lk}, B^h_{lk}, and S'_3 are valid for P_G, and summing them up leads to

$$3\Big(\sum_{v \in I_k \cup I_\ell} x_v\Big) + 3(\ell - 2)x_{c_1} + (3k - 3\ell - 2)x_{c_2} \leq 3k + 3.$$

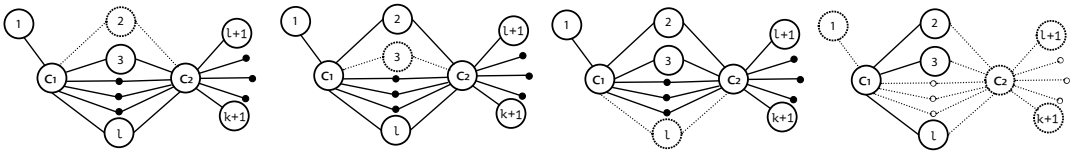
Adding $2x_{c_2} \leq 2$ we get that

$$\sum_{v \in I_k \cup I_\ell} x_v + (\ell - 2)x_{c_1} + (k - \ell)x_{c_2} \leq (k + 1) + \left\lfloor \frac{2}{3} \right\rfloor = k + 1$$

is valid for P_G.

For $\ell > 3$ and $k \geq \ell + 1$, the proof is similarly done by induction. Consider the following subgraphs of $B_{\ell k}$: $B^i_{\ell k}$, $B^j_{\ell k}$, $B^h_{\ell k}$, and $S'_3 = B_{\ell k}[\{i, j, h, c_1\}]$, for distinct $i, j, h \in I_\ell$. See Figure 3.11, where $i = 2$, $j = 3$, and $h = \ell$.

Figure 3.11: Subgraphs of $B_{\ell k}$.



(a) $B^i_{\ell k}$ (b) $B^j_{\ell k}$ (c) $B^h_{\ell k}$ (d) S'_3

By induction, the inequalities associated with $B^i_{\ell k}$, $B^j_{\ell k}$, $B^h_{\ell k}$, and S'_3 are valid for P_G, and summing them up leads to

$$3\Big(\sum_{v\in I_\ell\cup I_k} x_v\Big) + (3\ell - 6 + 1)x_{c_1} + 3(k-\ell)x_{c_2} \leq 3k + 3.$$

Adding $-x_{c_1} \leq 0$ we get that

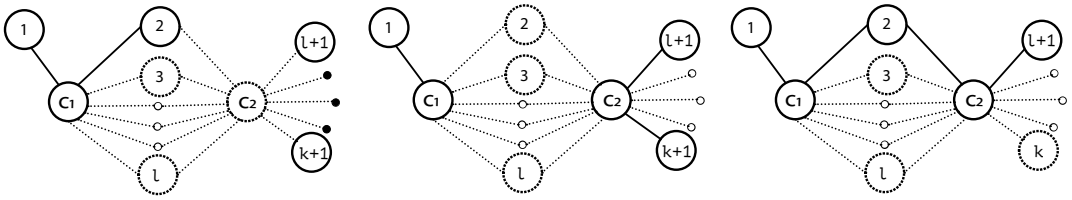
$$\sum_{v\in I_\ell\cup I_k} x_v + (\ell - 2)x_{c_1} + (k-\ell)x_{c_2} \leq k + 1$$

is valid for P_G.

2. Let w be a vector such that: $w_{c_1} = \ell - 2$, $w_{c_2} = k - \ell$, $w_i = 1$ for $i \in I_k \cup I_\ell$, and $w_v = 0$ for $v \notin V(B_{\ell k})$. Let $u^T x \leq u_0$ be a facet-inducing inequality for the GDP polytope P_G such that $F_a = \{\chi^H \in P_G \mid w^T\chi^H = k+1\} \subseteq F_b = \{\chi^H \in P_G \mid u^T\chi^H = u_0\}$. Clearly, $F_a \neq P_G$ and $F_a \neq \emptyset$. Thus, a proof that $u = \alpha w$ for some $\alpha \in \mathbb{R}$ shows that $w^T\chi^H = k + 1$ defines a facet of P_G.

Let $N_{i,j} = B_{\ell k}[(I_k \setminus I_\ell) \cup \{i, j, c_1\}]$ and $M_{i,j} = B_{\ell k}[(I_\ell \setminus I_k) \cup \{i, j, c_1, c_2\}]$ be subgraphs of $B_{\ell k}$ as exemplified in Figure 3.12. Observe that $M_{i,j}$ is claw-free when $\{i, j\} \subset I_k \setminus I_\ell$ or $i \in I_\ell \cap I_k$ and $j \in I_k \setminus I_\ell$. Figure 3.12 shows the graphs $N_{1,2}$, $M_{\ell+1,k}$, and $M_{2,\ell+1}$.

Figure 3.12: Subgraphs of $B_{\ell k}$.



(a) $N_{1,2}$ (b) $M_{\ell+1,k}$ (c) $M_{2,\ell+1}$

Then:

1. The incidence vectors $\chi^{N_{i,j}}$ and $\chi^{N_{j,h}}$ with $\{i,j,h\} \subset I_\ell$ are in $F_a \subseteq F_b$. Thus $0 = u_0 - u_0 = u^T\chi^{N_{i,j}} - u^T\chi^{N_{j,h}} = u_i - u_h$. This implies $u_i = u_h = \alpha$ for all choices of $\{i,j,h\} \subset I_\ell$.

2. The incidence vectors $\chi^{M_{i,j}}$ and $\chi^{M_{j,h}}$ with $\{i,j,h\} \subset I_k \setminus I_\ell$ are in $F_a \subseteq F_b$. Thus $0 = u_0 - u_0 = u^T\chi^{M_{i,j}} - u^T\chi^{M_{j,h}} = u_i - u_h$. This implies $u_i = u_h = \alpha$ for all choices of $\{i,j,h\} \subset I_k \setminus I_\ell$.

3. The incidence vectors $\chi^{I_k \cup I_\ell}$ and $\chi^{N_{i,j}}$ with $\{i,j\} \subset I_\ell$ are in $F_a \subseteq F_b$. Thus $0 = u_0 - u_0 = u^T\chi^{I_k \cup I_\ell} - u^T\chi^{N_{i,j}} = (\sum_{v \in I_\ell \setminus \{i,j\}} u_v) - u_{c_1}$. This implies $u_{c_1} = \alpha(\ell - 2)$.

4. The incidence vectors $\chi^{I_k \cup I_\ell}$ and $\chi^{M_{i,j}}$ with $\{i,j\} \subset I_k \setminus I_\ell$ are in $F_a \subseteq F_b$. Thus, $0 = u_0 - u_0 = u^T\chi^{I_k \cup I_\ell} - u^T\chi^{M_{i,j}} = (\sum_{v \in (I_k \cup I_\ell) \setminus \{i,j\}} u_v) - u_{c_1} - u_{c_2}$. This implies $u_{c_2} = \alpha(k-2) - \alpha(\ell - 2) = \alpha(k - \ell)$.

For a node $v \notin V(B_{\ell k})$ such that $|N(v) \cap (I_k \cup I_\ell)| < 3$, $\chi^{\{v\} \cup I_k \cup I_\ell}$ is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T\chi^{\{v\} \cup I_k \cup I_\ell} - u^T\chi^{I_k \cup I_\ell} = u_v$.

For a node $v \notin V(B_{\ell k})$ such that $|N(v) \cap (I_k \cup I_\ell)| > 3$, we have that:

1. If there is no vertex $i \in N(v) \cap (I_k \setminus I_\ell)$, the incidence vector $\chi^{\{v\}\cup V(N_{i,j})}$, with $i \in N(v) \cap I_\ell$ and $j \in I_\ell$, is in $F_a \subseteq F_b$. Then, the following equality holds: $0 = u_0 - u_0 = u^T\chi^{\{v\}\cup V(N_{i,j})} - u^T\chi^{N_{i,j}} = u_v$.

2. If there are vertices $i \in N(v) \cap (I_k \setminus I_\ell)$ and $j \in (I_\ell \cap I_k) \setminus N(v)$, the incidence vector $\chi^{\{v\}\cup V(M_{i,j})}$ is in $F_a \subseteq F_b$. Therefore, the following equality holds: $0 = u_0 - u_0 = u^T\chi^{\{v\}\cup V(M_{i,j})} - u^T\chi^{M_{i,j}} = u_v$.

3. If there is a vertex $i \in N(v) \cap (I_k \setminus I_\ell)$ and, in addition, $(I_\ell \cap I_k) \setminus N(v) = \emptyset$ and $N(v) \cap (I_k \cup I_\ell) \neq I_k \cup I_\ell$, the incidence vector $\chi^{\{v\}\cup V(M_{i,j})}$, with $i \in N(v) \cap (I_k \setminus I_\ell)$ and $j \in (I_k \setminus I_\ell) \setminus N(v)$, is in $F_a \subseteq F_b$. Thus, the following equality holds: $0 = u_0 - u_0 = u^T\chi^{\{v\}\cup V(M_{i,j})} - u^T\chi^{M_{i,j}} = u_v$.

□

Chapter 4

Branch-and-Cut Algorithms

4.1 Objective Function

The GDP can be formulated as the optimization problem over the incidence vectors of its polytope. The objective function searches for the incidence vector with minimum number of zeros (removed vertices):

$$\text{Minimize} \sum_{i=1}^{|V(G)|} (1 - x_i)$$

$$x \in P_G$$

4.2 Claw Model

P_G can be described by claw inequalities, generating the following model:

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{|V(G)|} (1 - x_i) \\ & x_a + x_b + x_c + x_d \leq 3, && \forall abcd \subset G \\ & x \in \{0, 1\} \end{aligned}$$

4.3 Star Model

P_G can be described by star inequalities, generating the following model:

$$\begin{aligned} \text{Minimize} \quad & \sum_{i=1}^{|V(G)|} (1 - x_i) \\ & \sum_{v \in I_k} x_v + (k-2)x_c \leq k, && \forall S_k \subset G \\ & x \in \{0, 1\} \end{aligned}$$

4.4 Lantern Model

P_G can be described by lantern and star inequalities, generating the following model::

$$\begin{aligned}
\text{Minimize } & \sum_{i=1}^{|V(G)|} (1 - x_i) \\
& \sum_{v \in I_k} x_v + (k-2)x_c \leq k, && \forall S_k \subset G \\
\sum_{v \in I_k} & x_v + (l-2)x_{cl} + (k-l)x_{ck} \leq k, && \forall L_{lk} \subset G \\
& x \in \{0,1\}
\end{aligned}$$

Observe that star inequalities are included because lanterns do not forbid infeasible integer points.

4.5 Binary Star Model

P_G can be described by binary star and star inequalities, generating the following model:

$$\begin{aligned}
\text{Minimize } & \sum_{i=1}^{|V(G)|} (1 - x_i) \\
& \sum_{v \in I_k} x_v + (k-2)x_c \leq k, && \forall S_k \subset G \\
\sum_{v \in \{I_k \cup I_l\}} & x_v + (l-2)x_{cl} + (k-l)x_{ck} \leq k+1, && \forall B_{lk} \subset G \\
& x \in \{0,1\}
\end{aligned}$$

Observe that star inequalities are included because binary stars do not forbid infeasible integer points.

4.6 *Branch-and-Cut* with Maximal Stars

Considering only maximal stars in G, its possible to reduce the number of inequalities of the star model. According to Wood (2011), the number of maximal independent sets $MIS(G)$ on G is :

$$|MIS(G)| \leq \begin{cases} 3^{n/3}, & \text{if } n \equiv 0(\text{mod } 3) \\ 4 \cdot 3^{(n-4)/3}, & \text{if } n \equiv 1(\text{mod } 3) \\ 2 \cdot 3^{(n-2)/3}, & \text{if } n \equiv 2(\text{mod } 3) \end{cases}$$

The set of maximal stars $STA(G)$ in G is found with all maximal independent sets for each induced by vertex subgraph of G, that is:

$$STA(G) = \cup_{v \in V(G)}(G[MIS(G[v]) \cup v])$$

Therefore, the number of maximal stars in G is bounded by an exponential number. So, a *Branch-and-Cut* approach is best suited for the star model. Since lanterns and binary stars are built by combining stars, they are also bounded by an exponential number.

4.7 Claw Extraction

There is a central vertex $v \in V(G)$ and an independent set of size three associated with each claw in G. Claws are extracted by an $O(n^4)$ algorithm that searches for all triples adjacent to a vertex v that are independent (not adjacent to each other). Each extracted claw is added to a support graph set $\mathcal{S}$.

4.8 Maximal Star Extraction

There is a central vertex $v \in V(G)$ and an independent set $I_k \subset G[N(v)]$ associated with each star $S_k \subset G$. Figure 4.1 shows two stars S_3 and S_4 centered at vertex v whose maximal independent sets on $G[N(v)]$ are used to build the stars.

Algorithm 1 resumes the star extraction for a graph G. Line 1 creates the support graph set $\mathcal{S}$ with an empty value. Line two iterates through each vertex $v \in V(G)$, fixing it as a possible star center. Line three finds the complementary graph $\overline{G[N(v)]}$ of the induced by vertex subgraph associated with v. Line four extracts all cliques on $\overline{G[N(v)]}$ with the algorithm $hybrid(.)$ presented by Eppstein e Strash (2011), this process has the same effect of extracting maximal independent sets in $G[N(v)]$. Lines five and six are selecting the maximal independent sets of size greater than two. Line seven generates the support subgraphs with center v combining it with its maximal independent sets. Line eight adds the support graph to the list of support graphs $\mathcal{S}$.

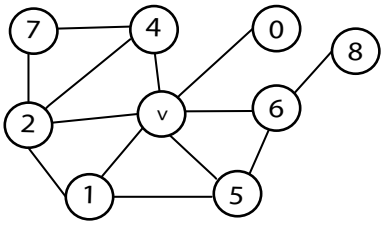
Algorithm 1: MAXIMAL STARS EXTRACTION ALGORITHM

```
1 S = ∅
2 for v ∈ V(G) do
3 |   G' = complement(G[N(v)])
4 |   S_temp = hybrid(G')
5 |   for I_k ∈ S_temp do
6 |   |   if k ≥ 3 then
7 |   |   |   S_k = G[I_k ∪ v]
8 |   |   |   S = S ∪ S_k
9 return S
```

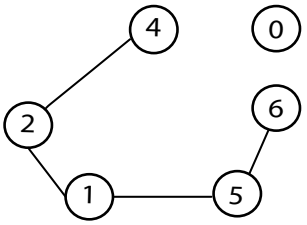
4.9 Separation for the claw and star models

With the current collection of support graphs at hand, the Branch-and-Cut algorithm solves the relaxed linear programming problem and then searches for all inequalities

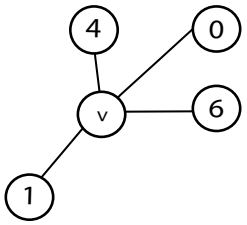
Figure 4.1: Extraction of maximal stars centered at v (c) and (d) from (a) by using the neighborhood of v (b).



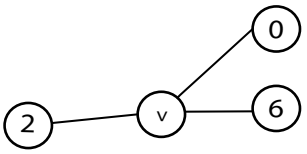
(a) Graph G



(b) $G[N(v)]$



(c) Star subgraph S_4



(d) Star subgraph S_3

violated by the current optimal solution x^*, measures their violation λ, and adds the one hundred most violated inequalities to the current region. The equation $\lambda = \sum_{v \in I_k} x_v^* + (k-2)x_c^* - k$ is used to measure the violation. If $\lambda > 0,0001$ the inequality is considered violated and is added to a pool of inequalities, in order to choose the one hundred best ones (those with the largest values of λ).

4.10 Separation for lantern and binary star models

It is not reasonable to extract all lantern and binary star subgraphs previously to the Branch-and-Cut routines. Thus, a separation heuristic is executed during the Branch-and-Cut algorithm. It begins by finding the one hundred most violated star inequalities using the same strategy described in section 4.9. Let $\mathcal{S}_{violated}$ be the collection of support star graphs associated with such inequalities. Then, for each distinct pair S_B, S_L of $\mathcal{S}_{violated}$, new lantern (binary star) subgraphs are constructed and added to the collection of support lantern (binary star) graphs, and the violation of the associated

new inequalities is measured. Finally, the one hundred most violated inequalities are added to the current region See Algorithms 2 and 3 for a brief description of the process used to obtain the new support graphs and associated inequalities. Figures 4.2 and 4.3 also illustrate this process. The violation λ for both models is measured as follows:

$$\lambda = \begin{cases} \sum_{v \in I_b} x^*_v + (l-2)x^*_{c_1} + (k-l)x^*_{c_2} - k, & \text{for Lanterns} \\ \sum_{v \in \{I_k \cup I_l\}} x^*_v + (l-2)x^*_{c_1} + (k-l)x^*_{c_2} - (k+1), & \text{for Binary Stars} \end{cases}$$

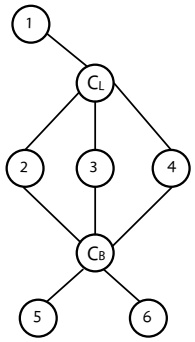
If $\lambda > 0,0001$ the inequality is considered to be violated.

Algorithm 2: LANTERN SEPARATION ALGORITHM

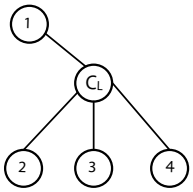
```
1  𝒮ℒ = 𝒮_violated
2  for distinct S_B, S_L ∈ 𝒮_violated do
3  |   Suppose w.l.o.g. |V(S_B)| ≥ |V(S_L)|
4  |   Let c_B, c_L be the centers of S_B, S_L, respectively
5  |   Let I_B, I_L be the independent sets of S_B, S_L, respectively
6  |   if |I_B ∩ I_L| ≥ 3 then
7  |   |   L_1 = G[V(S_B) ∪ {c_L}]
8  |   |   𝒮ℒ = 𝒮ℒ ∪ {L_1}
9  |   |   if |I_L| > 3 then
10 |   |   |   L_2 = G[V(S_L) ∪ {c_B}]
11 |   |   |   𝒮ℒ = 𝒮ℒ ∪ {L_2}
12 Measure λ for all L_ℓk ∈ 𝒮ℒ and add the one hundred best cuts to the current
   region
```

Algorithm 2 summarizes how to construct the collection $\mathscr{L}$ of support lantern graphs and evaluate the violation of the inequalities associated with such graphs. Line 1 initializes $\mathscr{L}$ with $\mathcal{S}_{violated}$. Lines 2 goes through each possible pair S_B, S_L of stars in $\mathcal{S}_{violated}$. Line 6 checks if the independent sets of S_B and S_L have at least three vertices in common, in order to construct valid lantern graphs. Line 7 creates the support lantern graph L_1 by taking induced subgraph $G[V(S_B) \cup \{c_L\}]$ formed by S_B and the center of S_L, and in line 8 the graph L_1 is stored in the collection $\mathscr{L}$ of support lantern graphs. Lines 9–11 create and store the support lantern graph $L_2 = G[V(S_L) \cup \{c_B\}]$, provided that I_L contains vertices outside $I_B \cap I_L$. Fig. 4.2 illustrates the construction of L_1 and L_2.

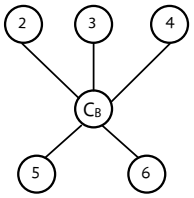
Figure 4.2: Figure (a) shows graph G, figures (b) and (c) show two maximal stars S_L and S_B in G, figures (d) and (e) show the two lantern subgraphs L_1 and L_2 constructed from S_L and S_B.



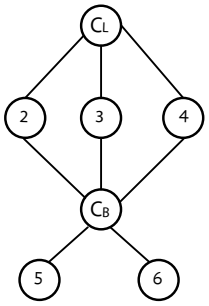
(a) Graph G



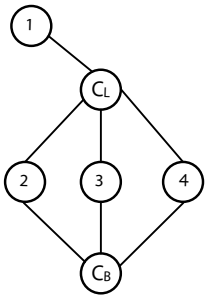
(b) S_L



(c) S_B



(d) L_1



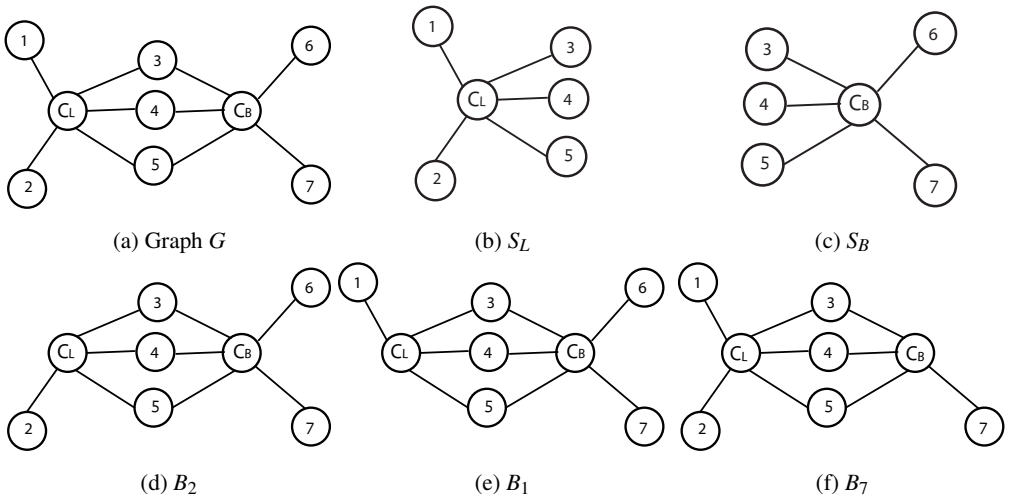
(e) L_2

Algorithm 3 is analogous to Algorithm 2. For each v outside $I_B \cap I_L$, it tries to construct the binary star subgraph $B_v = G[V(S_B) \cup \{v\} \cup \{c_L\}]$ (if $v \in I_L \setminus I_B$) or $B_v = G[V(S_L) \cup \{v\} \cup \{c_B\}]$ (if $v \in I_B \setminus I_L$). The collection $\mathscr{B}$ stores the support binary star graphs.

Algorithm 3: BINARY STAR SEPARATION ALGORITHM

$\mathscr{B} = \mathcal{S}_{violated}$
for *distinct* $S_B, S_L \in \mathcal{S}_{violated}$ **do**
    Let c_B, c_L be the centers of S_B, S_L, respectively
    Let I_B, I_L be the independent sets of S_B, S_L, respectively
    if $|I_B \cap I_L| \geq 2$ **then**
        for $v \in V(G) \setminus (I_B \cap I_L)$ **do**
            if $v \in I_L$ *and* $|I_B| > |I_B \cap I_L| + 1$ **then**
                $B_v = G[V(S_B) \cup \{v\} \cup \{c_L\}]$
                $\mathscr{B} = \mathscr{B} \cup \{B_v\}$
            if $v \in I_B$ *and* $|I_L| > |I_B \cap I_L| + 1$ **then**
                $B_v = G[V(S_L) \cup \{v\} \cup \{c_B\}]$
                $\mathscr{B} = \mathscr{B} \cup \{B_v\}$
Measure λ for all $B_{\ell k} \in \mathscr{B}$ and add the one hundred best cuts to the current region

Figure 4.3: Figure (a) shows graph G, figures (b) and (c) show two maximal stars S_L and S_B in G, figures (d), (e), and (f) show three binary star subgraphs constructed from S_L and S_B.



(a) Graph G (b) S_L (c) S_B

(d) B_2 (e) B_1 (f) B_7

Chapter 5

Column Generation for the GDP

5.1 Formulation for the Master Problem

The GDP can be formulated as:

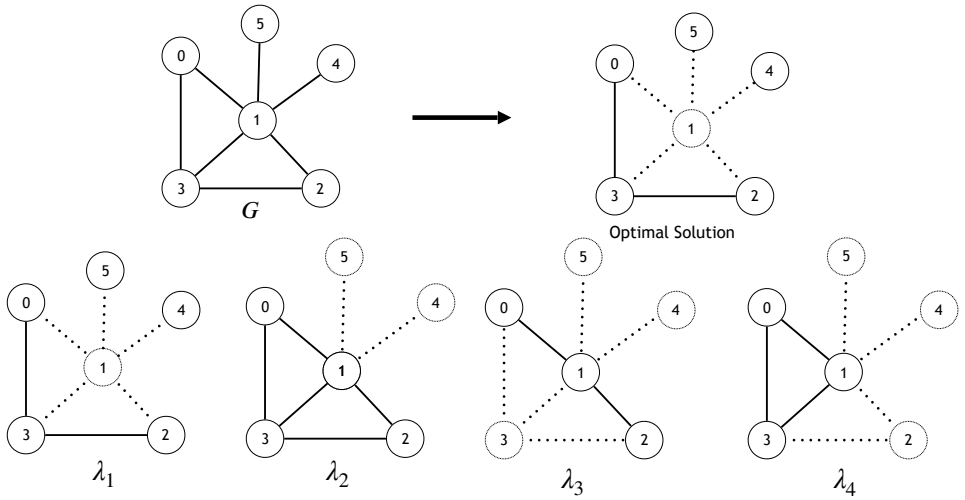
$$
\begin{aligned}
\text{Minimize } & \sum_{i=1}^{|V(G)|} (1 - x_i) \\
x_u = & \sum_{k \in C_v} a_u^{kv} \lambda_{kv}, && \forall v \in \mathscr{V}, \forall u \in \mathbb{S}_v \\
& \sum_{k \in C_v} \lambda_{kv} = 1, && \forall v \in \mathscr{V} \\
& x, \lambda \in \{0, 1\}
\end{aligned}
\tag{5.1}
$$

- G is an input graph.
- x_v is 1 if the vertex v is on solution.
- $\mathbb{S}_v$ is the set of all vertices of all stars S_k that contains vertex v as a center vertex, i.e. $\mathbb{S}_v = \cup_{\forall S_k | v=c} V(S_k)$.

- $\mathscr{V}$ is the set made by all centers of stars in G.
- C_v are the columns that forbids all S_k such that $v \in V(S_k)$.
- a_u^{kv} is 1 if vertex u is included on the k-esimal solution for the center v.
- λ_{kv} is used to know which solution is used for each vertex.

Figure 5.1 shows an example of how the mathematical formulation is build from a subset of possible columns associated with the vertex 1. On the figure, there is an input graph, its optimal solution, some claw-free columns associated with the center 1 and an equality forcing one claw-free column to be chosen. If there were more centers of stars, it could be possible that some vertex would be associated with more than one equality, but the nature of the simplex algorithm naturally avoid inconsistent values for the variable λ.

Figure 5.1: Example of the formulation for a subset of columns of a graph.



5.2 Formulation for the *Pricing*

$$\begin{aligned} &\text{Minimize } 0 - \pi_{vu} y_u \\ &(k-2)y_c + \sum_{u \in I_k} y_u \leq k, && \forall S_k \mid v = c \\ &y \in \{0,1\} \end{aligned}$$

- π_{vu} are the duals associated with constraints 5.1 of the master formulation, with v being the 'claw central vertex and u is a neighbor of v associated with the constraint.
- y_u is a vector that represents a claw-free solution centered at v.

5.3 Branching

The branchings were made on variable x.

5.4 Pricing Heuristics

Two pricing heuristics were made for speed up the resolution of the pricing. Algorithm 4 shows a *pricing* heuristic for generating columns with the central vertex removed. The algorithm takes the central vertex v as input and creates an empty column (line 2). The remotion of v grants a claw-free solution, so lines 3-5 add nodes on $S_v - v$ with positive dual, generating a column with better reduced cost.

Algorithm 5 shows a heuristic *pricing* for generating columns with the central vertex included. It searches for u, w in $\mathscr{V}$ with greatest dual values (line 2) and creates the columns with center and those two vertices (line 3).

Algorithm 4: COLUMNS WITHOUT CENTER PRICING

Subproblema *(v)*:
    $Col = \emptyset$
    for $u \in \mathbb{S}_v - v$ **do**
        if $(\pi_{vu} > 0)$ **then**
            $Col = Col \cup u$
    return Col

Algorithm 5: COLUMNS WITH CENTER PRICING

Subproblema *(v)*:
    Let u, w be two distinct vertices with greatest dual value in $\mathbb{S}_v - v$
    $Col = Col \cup \{v, u, w\}$
    return Col

Chapter 6

Computational Results

6.1 Random graph instances

Forty random graph instances generated by Bastos et al. (2016) are used to test the four proposed models, with $n \in \{50, 100\}$. All of the benchmark instances are available at

`http://sites.google.com/site/biclustereditingproblem/documents`.

6.2 Interval graph instances

A random generator is used to create interval graph instances. It receives the number n of intervals, their maximum length, and a time ruler as input parameters. See Algorithm 4 below.

The algorithm maintains a list of intervals $\mathscr{I}$ during its execution. It creates the initial and end times for each interval i and checks if i intersects any previously created interval j; if an intersection happens, edge ij is added to $E(G)$.

Twenty seven groups of interval graph instances were created by combining the parameters $n \in \{100, 200, 300\}$, $time\_ruler = 100$, and $interval\_max\_length \in$

Algorithm 6: INTERVAL GRAPH GENERATION ALGORITHM

```
1  Interval_graph_generator (n, interval_max_length, time_ruler):
2      I = ∅; G = ∅; i = 1
3      for i = 1, ..., n do
4          initial_time = rand() mod (time_ruler − interval_max_length)
5          end_time = initial_time + (rand() mod interval_max_length)
6          if initial_time = end_time then
7              end_time = end_time + 1
8          add interval i = (initial_time, end_time) to I
9          add vertex i to V(G)
10         for j ∈ I, j ≠ i do
11             if (j.initial_time ≤ i.end_time) and (i.initial_time ≤
                j.end_time) then
12                 add edge (j, i) to E(G)
13     return G
```

$\{10, 20, 30, 40, 50, 60, 70, 80, 90\}$. For each combination (group), nine instances were created. Interval graph instances are named following the pattern x_y_z-d where $x = n$, $y = time\_ruler$, $z = interval\_max\_length$, and d is the identification of the test instance.]

6.3 Experimental setup

All *Branch-and-Cut* algorithms proposed in this research have been developed in C++ with the aid of the mathematical solver CPLEX 12. The computational experiments have been performed on an Intel i7 processor running at 3.4 GHz with 16 GB of RAM, and executing the operating system Linux Ubuntu 14.04.

CPLEX has two internal callbacks to cope with Branch-and-Cut. The first one is the *lazy callback*, which verifies if any integer solution found by the Branch-and-Cut algorithm is feasible. The second one is the *cut callback*, which adds valid cuts to the current region being solved at any time during the execution of the Branch-and-Cut algorithm. For the claw and star models, the lazy and cut callbacks were implemented

with the same separation routines described in chapter five. For the lantern and binary star models, the lazy callback uses the separation routine of the star model, and the cut callback uses proper separation routines described on chapter five.

Two *Branch-and-Price* formulations, one with subproblem solved by integer programming and another with integer programming and heuristics were developed on C++ with aid of the *solvers* BapCod and CPLEX 12. These computational experiments were performed on a Linux Ubuntu 16.04 with processor i7 - 3.4 GHz and 32gb of RAM. The algorithms begin with artificial columns of cost 1000. Trivial *Upper Bounds* were used for the *Branch-and-Price* with heuristic *pricing*, trivial upper bounds are given by the solution with all vertices removed. The heuristic for pricing tries to find columns with negative reduced cost using algorithm 4, if no negative cost column is found, algorithm 5 is called. If, again, no negative cost column is found, the *pricing* is solved as an integer program by CPLEX.

The stop criteria used for the Branch-and-Cut tree are the use of a total memory of 10 Gb and a total time of 2 hours to solve the instances, using only one thread for the processing.

6.4 *Branch-and-Cut* Results

6.4.1 Results on Random Graphs

Tables 6.1 and 6.2 present results for random graph instances with $n = 50$. All models have been able to solve them optimally. Both tables show the running time, the number of nodes solved, and the number of cuts added by the Branch-and-Cut algorithm for each model. For this set of instances, the star model, when compared with the other models, achieves better running times for all random graph instances.

Comparing only the star and claw models, we observe that the star model has a fewer number of cuts and expanded nodes for all random graph instances. This is an evidence

Table 6.1: Computational results for random graph instances with $n = 50$.

			Claw Model			**Star Model**		
Instance	Density	OPT	**T (s)**	**#Nodes**	**#Cuts**	**T (s)**	**#Nodes**	**#Cuts**
105_1_50	0,05	16,00	0,09(0,01)	110	1085	0,02(0,00)	2	337
105_2_50	0,10	19,00	0,18(0,01)	315	747	0,03(0,00)	127	115
105_3_50	0,15	23,00	2,25(0,07)	3415	1650	0,16(0,01)	776	208
105_4_50	0,22	28,00	32,92(0,87)	22593	4351	1,58(0,08)	4073	612
105_5_50	0,24	29,00	116,06(3,40)	61758	5579	3,40(0,17)	6177	913
105_6_50	0,32	32,00	363,81(9,47)	111681	8675	15,19(0,57)	13713	2016
105_7_50	0,32	32,00	278,16(7,77)	86531	9002	21,69(0,93)	20387	2057
105_8_50	0,42	35,00	643,82(17,15)	134288	13036	47,43(1,24)	22035	3694
105_9_50	0,43	35,00	727,60(20,67)	146154	13769	68,22(1,90)	28127	3990
105_10_50	0,47	35,00	666,90(22,94)	136030	13218	57,96(1,55)	21185	4588
106_1_50	0,11	12,00	0,02(0,00)	3	217	0,01(0,00)	4	103
106_2_50	0,16	21,00	0,70(0,02)	972	1246	0,09(0,00)	236	273
106_3_50	0,20	25,00	7,21(0,22)	7254	2810	0,40(0,02)	897	576
106_4_50	0,25	28,00	46,02(1,49)	29146	4913	2,75(0,11)	3853	1142
106_5_50	0,30	31,00	161,31(4,56)	59852	7689	11,08(0,49)	11397	1818
106_6_50	0,32	32,00	209,15(5,31)	65449	8636	9,82(0,35)	8575	2022
106_7_50	0,40	34,00	536,74(15,30)	124642	12034	18,79(0,54)	9289	3382
106_8_50	0,42	35,00	870,03(26,62)	191671	12187	123,19(2,99)	53993	3852
106_9_50	0,44	35,00	712,37(20,09)	147306	13218	67,79(1,83)	28457	4164
106_10_50	0,50	34,00	371,87(13,53)	79052	13039	25,60(0,84)	9194	4797

Table 6.2: Computational results for random graph instances with $n = 50$.

			Lantern Model			**Binary Star Model**		
Instance	Density	OPT	**T (s)**	**#Nodes**	**#Cuts**	**T (s)**	**#Nodes**	**#Cuts**
105_1_50	0,05	16,00	0,02(0,00)	2	500	0,02(0,00)	3	345
105_2_50	0,10	19,00	0,03(0,00)	127	115	0,03(0,00)	127	115
105_3_50	0,15	23,00	0,14(0,01)	771	211	0,21(0,01)	917	351
105_4_50	0,22	28,00	4,24(0,21)	9214	924	4,88(0,19)	7676	1386
105_5_50	0,24	29,00	5,27(0,23)	7442	1339	7,50(0,25)	7395	2085
105_6_50	0,32	32,00	29,53(0,92)	19662	3186	33,62(0,83)	16529	4205
105_7_50	0,32	32,00	37,33(1,21)	24895	3340	30,25(0,76)	14680	4303
105_8_50	0,42	35,00	80,08(1,41)	21447	6142	81,83(1,32)	17773	7478
105_9_50	0,43	35,00	108,13(2,11)	27877	6580	89,81(1,58)	19757	7262
105_10_50	0,47	35,00	95,15(1,89)	22746	6991	133,74(0,02)	28272	8071
106_1_50	0,11	12,00	0,01(0,00)	4	103	0,01(0,00)	4	103
106_2_50	0,16	21,00	0,08(0,00)	212	307	0,12(0,01)	245	393
106_3_50	0,20	25,00	0,64(0,03)	1304	743	0,77(0,03)	925	1181
106_4_50	0,25	28,00	2,61(0,10)	2734	1495	3,53(0,11)	2365	2336
106_5_50	0,30	31,00	10,90(0,31)	7127	3022	19,09(0,55)	11013	3742
106_6_50	0,32	32,00	15,56(0,46)	10030	2882	17,90(0,38)	7569	4529
106_7_50	0,40	34,00	27,95(0,01)	8615	5731	33,90(0,69)	8351	6715
106_8_50	0,42	35,00	147,42(2,50)	41542	6221	172,75(2,95)	43498	6554
106_9_50	0,44	35,00	86,12(1,61)	23229	6619	101,66(1,80)	23936	7004
106_10_50	0,50	34,00	50,32(0,02)	11254	8268	54,98(1,36)	11063	8271

that star inequalities dominate claw inequalities and thus reach optimal solutions faster. For example, for instance 105_6_50, the star model took nearly eight times fewer nodes and four times fewer cuts. This explains the gain in time to find the optimal solution.

The comparison between the star model and the lantern/binary star models is more involved, because for each model there are instances for which it performs well, in terms of number of cuts or number of expanded nodes. But, in general, the star model is still the one with best results. It seems that star inequalities are simpler to manage, and provide a very strong relaxation. For example, instance 106_8_50 generates fewer nodes for the lantern and binary star models, but a worse execution time than the star model. Still for instance 106_8_50, the star model adds fewer cuts – possibly lantern and binary stars inequalities are making the feasible region more difficult to dealt with, forcing the Branch-and-Cut algorithm to take more time to find an optimal solution.

Tables 6.3 and 6.4 present results for random graph instances with $n = 100$. Both tables show the lower bound, upper bound, running time, number of nodes, and number of cuts for each model. In general, the star model is again the one with best results, although the lantern and binary star models have produced a better lower bound for some instances. Probably, this is due to the time increase these inequalities impose on the Branch-and-Cut tree by reducing the number of nodes.

6.4.2 Interval Graphs Results

Only the claw and star models were used to solve interval graph instances, since chordless cycles with four vertices are forbidden structures for interval graphs, and both lantern and binary star subgraphs contains such cycles.

For instances optimally solved by both models, see Table A.1. Instances with density higher than 60% were easier to solve (most of them were solved at the root node).

Table 6.5 shows the instances not solved optimally by at least one of the models. The star model was able to solve optimally all the instances with $n = 200$ and some with

Table 6.3: Computational results for random graph instances with $n = 100$.

		Claw Model					Star Model				
Instance	Density	**LB**	**UB**	**T (s)**	**#Nodes**	**#Cuts**	**LB**	**UB**	**T (s)**	**#Nodes**	**#Cuts**
109_1_100	0,05	45,00	63,00	680,63(211,42)	27062	54678	60,00	60,00	4,07(0.48)	256	6885
109_2_100	0,11	39,51	59,00	596,37(21,99)	100594	11858	55,00	55,00	3296,91(99.59)	3387399	895
109_3_100	0,15	40,75	67,00	554,64(37,07)	46700	25418	56,19	64,00	971.75(49.50)	359124	3470
109_4_100	0,20	40,60	76,00	518,45(51,97)	28998	42083	58,57	72,00	1057,89(55.47)	86873	12149
109_5_100	0,25	41,25	79,00	532,63(70,92)	23391	55790	59,79	75,00	1325,18(56.88)	43298	25334
109_6_100	0,29	41,70	81,00	631,58(87,40)	20251	65533	61,11	78,00	1531,49(59.12)	30837	37710
109_7_100	0,36	42,67	84,00	658,79(122,85)	18571	76663	60,19	82,00	1606,44(69.09)	22971	52368
109_8_100	0,39	43,04	85,00	697,71(131,90)	17215	79505	61,02	83,00	1644,75(70.48)	21432	56833
109_9_100	0,45	44,12	86,00	678,57(159,58)	17296	85094	60,28	85,00	1596,33(71.80)	19387	65034
109_10_100	0,50	44,07	85,00	705,25(158,11)	16198	88850	61,22	85,00	1614,32(70.32)	19117	66042
110_1_100	0,13	43,10	49,00	1647,33(34,45)	253453	8783	48,00	48,00	668,77(15.82)	352227	1742
110_2_100	0,17	41,28	63,00	595,50(40,33)	53454	22135	54,42	59,00	1893,47(62.09)	313549	5195
110_3_100	0,21	41,04	72,00	663,39(49,37)	30194	40564	58,08	69,00	1281,57(57.43)	76838	14558
110_4_100	0,26	41,75	79,00	553,02(66,43)	23599	54720	57,63	77,00	1288,43(56.99)	43005	25475
110_5_100	0,30	42,00	81,00	655,51(85,08)	19422	69145	60,64	79,00	1413,11(55.85)	29670	39072
110_6_100	0,34	42,26	83,00	635,52(111,21)	18460	75576	60,10	82,00	1551,39(65.61)	25240	47224
110_7_100	0,38	43,17	86,00	692,43(124,12)	17623	78625	60,03	83,00	1608,75(69.97)	21903	55566
110_8_100	0,43	43,67	85,00	672,47(142,52)	16945	83828	61,10	83,00	1637,98(72.85)	20265	61687
110_9_100	0,46	43,69	85,00	688,50(160,54)	17271	85066	60,15	85,00	1610,98(72.27)	19061	65935
110_10_100	0,50	44,13	85,00	704,46(155,52)	16141	91062	63,01	84,00	1770,65(70.14)	18597	68311

Table 6.4: Computational results for random graph instances with $n = 100$.

		Lantern Model					**Binary Star Model**				
Instance	Density	**LB**	**UB**	**T (s)**	**#Nodes**	**#Cuts**	**LB**	**UB**	**T (s)**	**#Nodes**	**#Cuts**
109_1_100	0,05	60,00	60,00	9,13(0.66)	206	18805	60,00	60,00	16,05(1.34)	349	18415
109_2_100	0,11	55,00	55,00	2343,30(64.85)	2200393	1100	55,00	55,00	8456,15(119.88)	3632702	2474
109_3_100	0,15	55,43	65,00	900,24(40.13)	271320	4726	55,13	65,00	897,43(31.14)	190931	6640
109_4_100	0,20	58,29	71,00	1179,08(50.73)	72063	17052	57,05	72,00	1088,06(44.26)	58367	20170
109_5_100	0,25	57,10	76,00	1376,73(46.74)	34687	37667	57,70	75,00	1391,88(44.55)	31053	40466
109_6_100	0,29	58,64	78,00	1543,56(52.09)	25496	51650	59,51	77,00	1671,17(49.14)	21923	59654
109_7_100	0,36	60,07	82,00	1762,05(58.26)	18460	75405	60,45	82,00	1801,79(54.12)	16527	83097
109_8_100	0,39	60,97	83,00	1833,58(58.03)	16701	84905	59,98	84,00	1821,07(56.45)	15275	91959
109_9_100	0,45	61,14	85,00	1853,25(60.31)	14924	96929	61,31	84,00	2065,95(59.15)	13577	105852
109_10_100	0,50	61,67	84,00	1940,53(58.82)	14044	106672	59,73	85,00	1794,74(56.96)	14185	100095
110_1_100	0,13	48,00	48,00	717,94(16.36)	369592	1829	48,00	48,00	743,66(11.39)	252114	2716
110_2_100	0,17	54,69	59,00	2377,35(64.99)	327720	6285	53,44	61,00	1221,98(38.00)	173841	7671
110_3_100	0,21	57,91	69,00	1338,68(51.09)	67397	18348	57,37	69,00	1243,53(45.09)	56163	21658
110_4_100	0,26	57,95	77,00	1371,90(47.30)	35000	34661	58,57	77,00	1344,79(45.05)	32580	36849
110_5_100	0,30	59,24	78,00	1635,61(51.79)	24497	54099	59,17	79,00	1622,86(48.13)	21566	59856
110_6_100	0,34	59,17	82,00	1666,62(55.66)	20077	65862	59,47	81,00	1622,35(52.66)	18285	73634
110_7_100	0,38	61,10	83,00	1799,67(57.76)	17132	81057	58,30	83,00	1694,40(57.24)	15973	85594
110_8_100	0,43	59,97	84,00	1800,39(63.49)	15497	94666	59,68	85,00	1712,03(58.01)	15071	94293
110_9_100	0,46	60,96	84,00	1839,95(57.54)	14692	100936	60,93	85,00	1840,75(56.44)	13871	103703
110_10_100	0,50	61,07	84,00	1912,82(57.53)	14094	106656	61,63	84,00	1880,32(54.19)	13070	107612

$n = 300$. But it could not solve instances 300_100_40-9, 300_100_50-1, 300_100_50-3, 300_100_50-7, and 300_100_50-9, because of the large amount of maximal independent sets. In such cases, the claw model achieves better results.

6.5 *Branch-and-Price* Results

Column generation was applied only on random graph instances, because they were harder to solve by *Branch-and-Cut*. Table 6.6 presents results for the proposed *Branch-and-Prices*. The table has the *Lower Bounds* and running time for root/tree.

Table 6.6 shows that the relaxed value found on root by the *Branch-and-Cut* approach is close to the root value for the *Branch-and-Price* and in some cases is even better (See instances 110_4_100 or 109_3_100). The *Branch-and-Price* algorithm needs greater running time to finish root processing. The *Branch-and-Price* algorithm without heuristic was not capable of solve the instances 109_1_100, 109_10_100 and 110_10_100 on root.

Table 6.6 shows that the *Branch-and-Price* with heuristics was capable of solving all roots for all instances, but star model keeps with better time and upper bounds.

Table 6.5: Computational results for interval graph instances not solved to optimality.

		Claw Model					Star Model				
Instance	Density	**LB**	**UB**	**T (s)**	**#Nodes**	**#Cuts**	**LB**	**UB**	**T (s)**	**#Nodes**	**#Cuts**
200_100_20-1	0.24	66.00	74.00	2034.05(170.92)	36027	40512	71.00	71.00	111.14(10.36)	2620	22471
200_100_20-2	0.23	65.25	73.00	1834.78(155.62)	36531	42987	72.00	72.00	749.30(38.80)	11796	30244
200_100_20-3	0.24	63.02	76.00	1278.98(154.39)	25503	56054	73.00	73.00	880.04(41.02)	10717	35238
200_100_20-4	0.25	64.51	76.00	1686.41(178.19)	26857	53606	72.00	72.00	251.89(22.30)	3778	35330
200_100_20-5	0.22	64.50	82.00	1386.28(149.89)	23664	53248	77.00	77.00	964.83(39.80)	8447	43613
200_100_20-6	0.23	64.51	74.00	1606.42(152.13)	38307	41065	72.00	72.00	2063.31(97.87)	27343	35161
200_100_20-9	0.24	64.39	81.00	1673.53(158.20)	24106	52477	74.00	74.00	1144.33(50.63)	10311	40877
200_100_30-4	0.36	65.00	73.00	3169.96(404.59)	21809	76124	72.00	72.00	526.27(32.61)	3379	51897
300_100_10-1	0.11	87.03	104.00	1238.27(84.14)	38769	28306	100.00	100.00	1634.40(110.16)	32404	20011
300_100_10-2	0.11	81.86	115.00	1338.51(80.68)	33456	33231	94.81	107.00	1743.09(214.21)	38686	30236
300_100_10-3	0.11	87.38	117.00	1322.46(79.72)	31354	33589	99.33	113.00	1849.01(238.41)	40496	28361
300_100_10-4	0.10	86.26	105.00	1393.94(85.41)	38921	28885	101.00	101.00	6699.80(407.60)	136886	22858
300_100_10-5	0.11	85.36	107.00	1092.95(94.91)	36853	29062	97.35	102.00	3603.79(385.95)	72159	24523
300_100_10-6	0.11	84.25	110.00	1115.63(101.55)	35754	30432	93.73	108.00	1685.14(219.17)	36830	31026
300_100_10-7	0.11	85.50	113.00	1458.77(81.19)	31898	33532	100.82	106.00	3240.54(329.69)	61706	25918
300_100_10-8	0.11	82.50	107.00	1307.44(73.40)	38752	28258	94.23	99.00	3198.63(288.53)	60986	24908
300_100_10-9	0.10	85.60	114.00	1350.88(80.06)	34688	30690	97.82	107.00	2176.65(213.80)	43763	28088
300_100_20-1	0.23	87.25	138.00	3322.03(406.15)	10236	117088	104.44	137.00	4543.62(254.13)	8939	134686
300_100_20-2	0.24	89.91	130.00	3573.12(387.06)	10926	111728	106.95	126.00	5276.96(260.27)	11086	113284
300_100_20-3	0.24	88.51	131.00	3397.13(410.17)	10675	115273	106.78	127.00	4769.55(282.55)	10287	122609
300_100_20-4	0.24	88.17	136.00	3075.90(326.58)	9854	122795	109.28	124.00	5871.60(334.99)	11221	116440
300_100_20-5	0.24	88.77	142.00	3538.09(485.36)	9405	127700	108.06	136.00	6142.51(321.06)	7862	156232
300_100_20-6	0.22	84.88	127.00	2892.44(328.80)	12542	97161	103.23	124.00	4687.96(199.84)	10451	120772
300_100_20-7	0.24	87.50	128.00	3300.29(412.23)	10854	107685	115.00	115.00	10634.78(339.91)	15259	103696
300_100_20-8	0.24	89.63	132.00	3467.63(408.72)	9788	118011	111.96	124.00	6524.05(378.29)	11529	114525
300_100_20-9	0.23	86.25	140.00	2954.84(329.00)	10136	117450	108.13	127.00	6145.07(295.07)	9689	129303
300_100_30-1	0.37	90.01	110.00	5212.03(1231.42)	8628	153587	108.00	108.00	2153.47(448.59)	1800	136634
300_100_30-2	0.38	87.67	120.00	5908.49(1162.00)	8076	149072	115.14	117.00	21608.28(1755.68)	12600	201256
300_100_30-3	0.36	93.13	122.00	7066.39(1100.16)	7599	161233	117.00	117.00	1057.29(175.00)	923	101502
300_100_30-4	0.36	85.00	118.00	4820.33(1077.95)	10050	131147	99.69	115.00	6548.15(944.36)	9981	151365
300_100_30-5	0.39	93.00	127.00	7478.74(1626.75)	7603	168243	118.00	118.00	19150.25(3800.82)	9786	210441
300_100_30-6	0.39	97.76	112.00	8207.66(1994.37)	12057	112884	111.00	111.00	2036.75(336.03)	2054	111079
300_100_30-7	0.36	96.01	114.00	6462.65(1580.45)	9978	128686	114.00	114.00	2571.46(473.80)	1933	124882
300_100_30-9	0.36	91.00	110.00	5842.56(981.55)	10054	138267	106.00	106.00	980.75(189.24)	1300	92365
300_100_40-6	0.50	104.51	107.00	21605.69(4181.36)	16700	129677	107.00	107.00	1088.44(654.58)	442	83640
300_100_40-9	0.53	99.50	112.00	13800.68(4484.70)	12462	126926	-	-	-	-	-
300_100_50-1	0.66	70.00	70.00	44.76(38.08)	2	7800	-	-	-	-	-
300_100_50-3	0.66	66.00	66.00	45.63(38.79)	2	7800	-	-	-	-	-
300_100_50-7	0.62	80.00	80.00	233.69(186.36)	167	20156	-	-	-	-	-
300_100_50-9	0.66	70.00	70.00	61.50(52.43)	3	9400	-	-	-	-	-

Table 6.6: Results for Column Generation in Random Instances.

	Star Model				Branch-and-Price (MIP)				Branch-and-Price (HEUR)			
	Root		Tree		Root		Tree		Root		Tree	
Instance	LB	T(s)	UB	T (s)	LB	T(s)	UB	T (s)	LB	T(s)	UB	T (s)
105_1_50	16.00	0.02	16	0.02	16.00	133.03	16	3073.19	16.00	26.70	16	2766.17
105_2_50	15.35	0.01	19	0.03	16.40	5.18	19	156.23	16.40	2.51	19	39.44
105_3_50	17.66	0.02	23	0.14	18.33	8.47	23	958.72	18.33	4.65	23	120.56
105_4_50	19.94	0.04	28	1.58	19.96	15.55	-	7200.00	19.96	7.00	28	1438.10
105_5_50	20.39	0.05	29	3.40	20.24	19.55	-	7200.00	20.24	7.45	29	2194.52
105_6_50	20.47	0.05	32	15.19	20.43	27.89	-	7200.00	20.43	11.19	32	5991.76
105_7_50	20.61	0.04	32	21.69	20.60	24.78	-	7200.00	20.60	11.75	32	5084.51
105_8_50	20.52	0.04	35	47.43	20.51	55.73	-	7200.00	20.51	23.66	38	7200.00
105_9_50	20.19	0.04	35	68.22	20.56	55.69	-	7200.00	20.56	25.27	38	7200.00
105_10_50	20.19	0.04	35	57.96	20.15	100.97	-	7200.00	20.15	33.31	49	7200.00
106_1_50	11.48	0.00	12	0.01	11.80	2.06	12	6.35	11.25	1.01	12	1.01
106_2_50	17.05	0.03	21	0.09	17.04	5.04	21	575.62	17.04	3.33	21	95.10
106_3_50	18.80	0.04	25	0.40	18.85	8.69	25	2357.67	18.85	5.21	25	342.93
106_4_50	19.90	0.04	28	2.75	19.89	12.35	-	7200.00	19.89	7.43	28	1361.44
106_5_50	20.39	0.05	31	11.08	20.38	17.19	-	7200.00	20.38	9.45	31	3096.97
106_6_50	20.55	0.05	32	9.82	20.55	32.31	-	7200.00	20.55	11.09	32	5487.38
106_7_50	20.56	0.04	34	18.79	20.57	41.95	-	7200.00	20.57	16.31	36	7200.00
106_8_50	20.17	0.05	35	123.19	20.14	55.20	-	7200.00	20.14	21.52	38	7200.00
106_9_50	20.48	0.04	35	67.79	20.30	65.18	-	7200.00	20.30	31.29	36	7200.00
106_10_50	19.97	0.04	34	25.60	20.02	95.51	-	7200.00	20.02	39.70	48	7200.00
109_1_100	42.43	0.19	60	4.07	36.95	7200.00	-	-	42.38	4610.93	96	7200.00
109_2_100	40.56	0.19	55	3296.90	40.97	33.28	-	7200.00	40.97	16.15	75	7200.00
109_3_100	43.43	0.22	64	971.75	43.39	62.78	-	7200.00	43.39	28.92	94	7200.00
109_4_100	43.95	0.25	72	1057.89	43.95	152.78	-	7200.00	43.95	55.81	95	7200.00
109_5_100	44.02	0.33	75	1325.18	44.02	348.62	-	7200.00	44.02	105.12	99	7200.00
109_6_100	44.06	0.32	78	1531.49	44.04	834.67	-	7200.00	44.04	371.68	99	7200.00
109_7_100	43.66	0.27	82	1606.44	43.79	1897.49	-	7200.00	43.79	917.93	97	7200.00
109_8_100	43.56	0.28	83	1644.75	43.71	3081.38	-	7200.00	43.71	1292.23	99	7200.00
109_9_100	43.06	0.25	85	1596.33	43.35	5610.75	-	7200.00	43.35	2822.15	99	7200.00
109_10_100	42.54	0.28	85	1614.32	42.88	7200.00	-	-	42.99	4240.89	99	7200.00
110_1_100	36.92	0.13	48	668.77	37.30	32.91	-	7200.00	37.30	11.04	95	7200.00
110_2_100	41.39	0.23	59	1893.47	41.37	69.34	-	7200.00	41.37	26.56	97	7200.00
110_3_100	43.26	0.16	69	1281.57	43.26	174.58	-	7200.00	43.26	55.73	99	7200.00
110_4_100	43.69	0.28	77	1288.43	43.56	352.90	-	7200.00	43.56	102.00	98	7200.00
110_5_100	43.70	0.31	79	1413.11	43.74	848.48	-	7200.00	43.74	306.43	99	7200.00
110_6_100	43.59	0.31	82	1551.39	43.71	1839.16	-	7200.00	43.71	842.12	96	7200.00
110_7_100	43.52	0.28	83	1608.75	43.66	3143.45	-	7200.00	43.66	1402.11	98	7200.00
110_8_100	43.15	0.28	83	1637.98	43.40	4777.19	-	7200.00	43.40	2152.26	98	7200.00
110_9_100	42.96	0.25	85	1610.56	43.30	5871.48	-	7200.00	43.30	3038.39	99	7200.00
110_10_100	42.70	0.26	84	1770.65	42.90	7200.00	-	-	43.07	3830.92	99	7200.00

Chapter 7

Conclusions

In this thesis we studied the Graph Declawing Problem (GDP). Claw free graph properties have been studied by almost thirty years and the optimization version for it was never studied using a polyhedral approach. Some facet-defining inequalities of the GDP polytope were presented. Separation algorithms were proposed and used into a Branch-and-Cut procedure for solving the GDP.

Interval graph instances were created. Computational experiments were carried out and the results show that the algorithms are able to solve many interval graph instances with $n \in \{100, 200, 300\}$ using fewer than two hours of processing. The computational results also show that random graph instances were harder to solve. Random graph instances with $n = 50$ were optimally solved but most random graph instances with $n = 100$ were not solved to optimality.

A Branch-and-Price algorithm was also proposed. The Branch-and-Price and Branch-and-Cut presented an equal relaxation on root node, but the Branch-and-Price algorithm had a worse resolution time. In general, the Branch-and-Cut with the star model presented best results.

The next chapter proposes possibilities that will be explored on future works.

Chapter 8

Future Work

8.1 Families of Facets For Vertex Deletion Problems

Let Q be a undirected graph. The vertex deletion problem for property Π associated with Q (VDP-Q) searches for the minimum subset of vertices $S \subseteq V(G)$ such that the new graph $G' = G - S$ has no vertex-induced subgraph isomorphic to Q. Let $\mathscr{P}_{G,Q}$ be the convex hull of the incidence vectors H that represents subgraphs of G not containg Q as a vertex-induced subgraph, i.e. $\mathscr{P}_{G,Q} = conv.hull\{\chi^H \in \{0,1\}^n \mid H \text{ is a Q-free subgraph of } G\}$, $\mathscr{P}_{G,Q}$ is the polytope associated with any VDP-Q. $\mathscr{P}_{G,Q}$ has the following properties:

Theorem 8.1.1.

1. *Full dimensionality.*
2. *Binary inequalities (i.e.* $0 \leq x_v$ *and* $x_v \leq 1$ *for all* $v \in V(G)$*) are facet-inducing.*
3. *Inequalities of the form:* $\sum_{v \in V(Q)} x_v \leq |V(Q)| - 1$*, are valid.*
4. *Inequalities of the form:* $\sum_{v \in V(Q)} x_v \leq |V(Q)| - 1$*, are facet-inducing. Excepting on cases of clique or independent set graphs.*

Proof.

1. The null and unitary vectors are on $\mathscr{P}_{G,Q}$ and they sum up to $n+1$ affinelly independent points. Since $\mathscr{P}_{G,Q} \in \mathbb{R}^n$, $\mathscr{P}_{G,Q}$ is a full dimensional polytope.

2. $x_v = 0$ is attended at equality by the null vector and all unitary vectors where $x_v \neq 1$ summing up to n affinely independent points; $x_v = 1$ is attended at equality by the unitary vector $x_v = 1$ and all binary vectors where $x_v = 1$ and $x_k = 1$ with $k \neq v$ summing up to n affinely independent points.

3. These inequalities are forcing the elimination of one vertex $v \in V(Q)$. Therefore, they are forbidding only graphs of the form Q and allowing any subgraph of Q to be part of $\mathscr{P}_{G,Q}$.

4. Let a be a vector with $a_i = 1$ for $i \in V(Q)$ and $a_i = 0$ otherwise. Let $b^T x \leq b_0$ be a facet-inducing inequality for $\mathscr{P}_{G,Q}$ such that $F_a = \{\chi^H \in \mathscr{P}_{G,Q} \mid a^T\chi^H = |V(Q)| - 1\} \subseteq F_b = \{\chi^H \in \mathscr{P}_{G,Q} \mid b^T\chi^H = b_0\}$. Clearly, $F_a \neq \emptyset$ and $F_a \neq P$; since $\mathscr{P}_{G,Q}$ is full dimensional, finding the existence of some positive $\alpha \in \mathbb{R}$ such that $b = \alpha a$ is enough to state $\sum_{v \in V(Q)} x_v = |V(Q)| - 1$ as a facet-inducing inequality.

We have that $\chi^{V(Q)\setminus x}$ and $\chi^{V(Q)\setminus y}$ are in $F_a \subseteq F_b$ for any $x, y \in V(Q)$ with $x \neq y$; therefore, $0 = b_0 - b_0 = b^T\chi^{V(Q)\setminus x} - b^T\chi^{V(Q)\setminus y} = b_y - b_x$; and so, $b_x = b_y = \alpha$ for any $x, y \in V(Q)$.

For any $w \notin V(Q)$, let Q' be the supergraph of Q that contains w, i.e. $Q' = Q + w$, now let T_w be the subset of "twin vertices" of w in Q' such that T_w is composed by all vertices $v \in V(Q)$ where $Q'[w \cup N(w)]$ is isomorphic to $Q'[v \cup N(v)]$. For $x \in V(Q) \setminus T_w$ we have that $\chi^{(V(Q)\setminus x)\cup w} \in F_a \subseteq F_b$; therefore, $0 = \chi^{(V(Q)\setminus x)\cup w} - \chi^{V(Q)\setminus x} = b_w$; and so, $b_w = 0$ for any $w \notin V(Q)$.

□

Note that the above proof do not cover cases where all vertices on Q' are twin vertices (For example, this happens for families of cliques, holes and independent sets).

8.2 Branch-and-Price For Vertex Deletion Problems

8.2.1 Master Problem

Any VDP-Q can be formulated as a column generation problem using the following formulation:

$$
\begin{aligned}
\text{Minimize } & \sum_{i=1}^{|V(G)|} (1 - x_i) \\
x_u &= \sum_{k \in C_v} a_u^{kv} \lambda_{kv}, && \forall v \in \mathscr{V}, \forall u \in \mathbb{S}_v \\
& \sum_{k \in C_v} \lambda_{kv} = 1, && \forall v \in \mathscr{V} \\
& x, \lambda \in \{0, 1\}
\end{aligned} \tag{8.1}
$$

- G is an input graph.
- x_v is 1 if the vertex v is on solution.
- $\mathbb{S}_v$ is the set of all vertices of all subgraphs Q that contains vertex v, i.e. $\mathbb{S}_v = \cup_{\forall Q | v \in V(Q)} V(Q)$.
- $\mathscr{V}$ is a subset of $V(G)$ such that any $Q \subset G$ has some vertex $v \in \mathscr{V}$ in $V(Q)$.
- C_v are the columns that forbids all Q such that $v \in V(Q)$.
- a_u^{kv} is 1 if vertex u is included on the k-esimal solution for vertex $v \in \mathscr{V}$.
- λ_{kv} is used to know which solution is used for each vertex.

8.2.2 Pricing

The master formulation has an exponential number of variables λ. Its possible to associate a subproblem for each vertex on $\mathscr{V}$, the following formulation can be used to generate λ for the master formulation:

$$\text{Minimize } 0 - \sum_{u \in \mathbb{S}_v} \pi_{vu} y_u$$

$$\sum_{i \in V(Q)} y_i \leq |V(Q)| - 1, \qquad \forall Q \mid v \in V(Q)$$

- π_{vu} are the duals associated with constraints 7.1 of the master formulation, with v being the vertex associated with the subproblem and u is a vexter of $\mathbb{S}_v$.
- y_u is a vector that represents a Q-free solution for all Q that contains v.

8.2.3 Branching

As *branching* rule, it is possible to use *Branching* on *x*.

8.3 Complexity on Interval Graph Instances

The computational results obtained for the interval graph instances foster the possibility that the problem is solvable on polynomial time because of the great advance on the resolution of bigger size instances. Therefore a future work will address the complexity class for the GDP on interval graph instances.

Bibliography

[Bastos et al. 2016]BASTOS, L. et al. Efficient algorithms for cluster editing. *Journal of Combinatorial Optimization*, v. 31, n. 1, p. 347–371, Jan 2016. ISSN 1573-2886.

[Bevern et al. 2010]BEVERN, R. van et al. Measuring indifference: Unit interval vertex deletion. In: THILIKOS, D. M. (Ed.). *Graph Theoretic Concepts in Computer Science*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 232–243. ISBN 978-3-642-16926-7.

[Bogart e West 1999]BOGART, K. P.; WEST, D. B. A short proof that proper = unit. *Discrete Mathematics*, v. 201, n. 1, p. 21 – 23, 1999. ISSN 0012-365X.

[Boral et al. 2016]BORAL, A. et al. A fast branching algorithm for cluster vertex deletion. *Theory of Computing Systems*, v. 58, n. 2, p. 357–376, Feb 2016.

[Christof e Schenker]CHRISTOF, T.; SCHENKER, S. *PORTA - POlyhedron Representation Transformation Algorithm*.

[Eppstein e Strash 2011]EPPSTEIN, D.; STRASH, D. Listing all maximal cliques in large sparse real-world graphs. In: ____. *Experimental Algorithms: 10th International Symposium, SEA 2011, Kolimpari, Chania, Crete, Greece, May 5-7, 2011. Proceedings*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2011. p. 364–375. ISBN 978-3-642-20662-7.

[Faudree, Flandrin e Ryjáček 1997]FAUDREE, R.; FLANDRIN, E.; RYJáčEK, Z. Claw-free graphs - a survey. *Discrete Mathematics*, v. 164, n. 1, p. 87 – 147, 1997. ISSN 0012-365X. The second Krakow conference of graph theory.

[Filho et al. 2017]FILHO, G. et al. New heuristics for the bicluster editing problem. *Annals of Operations Research*, v. 258, 11 2017.

[Fishburn 1985]FISHBURN, P. C. Interval graphs and interval orders. *Discrete Applied Mathematics*, v. 55, n. 2, p. 135–149, 1985.

[Guo 2007]GUO, J. Problem kernels for np-complete edge deletion problems: Split and related graphs. In: *Proceedings of the 18th International Conference on Algorithms and Computation*. Berlin, Heidelberg: Springer-Verlag, 2007. (ISAAC'07), p. 915–926. ISBN 3-540-77118-2, 978-3-540-77118-0.

[Guo et al. 2010]GUO, J. et al. A more relaxed model for graph-based data clustering: s-plex cluster editing. *SIAM Journal on Discrete Mathematics*, v. 24, n. 4, p. 1662–1683, 2010.

[HSU e Nemhauser 1982]HSU, W.-L.; NEMHAUSER, G. L. A polynomial algorithm for the minimum weighted clique cover problem on claw-free perfect graphs. *Discrete Mathematics*, v. 38, n. 1, p. 65 – 71, 1982. ISSN 0012-365X.

[Krithika et al. 2016]KRITHIKA, R. et al. Lossy Kernels for Graph Contraction Problems. In: LAL, A. et al. (Ed.). *36th IARCS Annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS 2016)*. Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2016. (Leibniz International Proceedings in Informatics (LIPIcs), v. 65), p. 23:1–23:14. ISBN 978-3-95977-027-9. ISSN 1868-8969. Disponível em: <http://drops.dagstuhl.de/opus/volltexte/2016/6858>.

[Lekkeikerker C. 1962]LEKKEIKERKER C., B. J. Representation of a finite graph by a set of intervals on the real line. *Fundamenta Mathematicae*, v. 51, n. 1, p. 45–64, 1962.

[Lewis e Yannakakis 1980]LEWIS, J. M.; YANNAKAKIS, M. The node-deletion problem for hereditary properties is np-complete. *Journal of Computer and System Sciences*, v. 20, n. 2, p. 219 – 230, 1980. ISSN 0022-0000.

[Minty 1980]MINTY, G. J. On maximal independent sets of vertices in claw-free graphs. *Journal of Combinatorial Theory, Series B*, v. 28, n. 3, p. 284 – 304, 1980. ISSN 0095-8956.

[Peng e Chen 2006]PENG, S.-L.; CHEN, C.-K. On the interval completion of chordal graphs. *Discrete Applied Mathematics*, v. 154, n. 6, p. 1003 – 1010, 2006. ISSN 0166-218X.

[Roberts 1979]ROBERTS, F. S. Indifference and seriation. *Annals of the New York Academy of Sciences*, Blackwell Publishing Ltd, v. 328, n. 1, p. 173–182, 1979. ISSN 1749-6632.

[Wood 2011]Wood, D. R. On the number of maximal independent sets in a graph. *ArXiv e-prints*, abr. 2011.

Appendix A

Computational Results for Branch-and-Cut on Interval Instances

Table A.1: Computational results for interval graph instances optimally solved.

			Claw Model			**Star Model**		
Instance	Density	OPT	**T (s)**	**#Nodes**	**#Cuts**	**T (s)**	**#Nodes**	**#Cuts**
100_100_10-1	0.12	22.00	0.06(0.00)	36	628	0.04(0.00)	42	595
100_100_10-2	0.10	21.00	0.04(0.00)	7	465	0.02(0.00)	15	255
100_100_10-3	0.11	22.00	0.04(0.00)	12	445	0.02(0.00)	8	372
100_100_10-4	0.11	17.00	0.06(0.00)	4	620	0.03(0.00)	3	533
100_100_10-5	0.10	19.00	0.05(0.00)	2	466	0.03(0.00)	3	383
100_100_10-6	0.11	23.00	0.07(0.00)	52	538	0.04(0.00)	21	450
100_100_10-7	0.10	21.00	0.05(0.00)	13	511	0.03(0.00)	5	328
100_100_10-8	0.11	21.00	0.05(0.00)	32	567	0.03(0.00)	19	392
100_100_10-9	0.11	22.00	0.05(0.00)	14	537	0.02(0.00)	2	345
100_100_20-1	0.23	30.00	0.47(0.03)	146	3096	0.12(0.00)	4	1438
100_100_20-2	0.24	30.00	0.98(0.04)	430	3201	0.17(0.01)	39	1975
100_100_20-3	0.23	30.00	0.70(0.03)	362	2861	0.31(0.02)	105	2354
100_100_20-4	0.25	32.00	1.70(0.09)	538	4583	0.77(0.05)	311	2825

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	T (s)	#Nodes	#Cuts	T (s)	#Nodes	#Cuts
100_100_20-5	0.21	35.00	11.33(0.52)	4321	6095	1.91(0.12)	876	3871
100_100_20-6	0.23	29.00	0.42(0.02)	161	2644	0.19(0.01)	51	2280
100_100_20-7	0.23	34.00	2.65(0.13)	898	5083	0.45(0.03)	141	2867
100_100_20-8	0.23	27.00	0.16(0.01)	22	1783	0.15(0.01)	18	1799
100_100_20-9	0.22	32.00	1.40(0.05)	484	3654	0.26(0.01)	66	2294
100_100_30-1	0.43	37.00	20.45(2.16)	1997	13645	2.94(0.28)	249	7672
100_100_30-2	0.38	29.00	0.47(0.03)	44	3344	0.18(0.01)	2	1606
100_100_30-3	0.36	30.00	0.88(0.08)	161	4261	0.30(0.02)	11	2602
100_100_30-4	0.33	27.00	0.51(0.04)	90	3929	0.19(0.01)	16	1982
100_100_30-5	0.39	34.00	2.92(0.40)	287	8772	0.95(0.11)	49	5797
100_100_30-6	0.41	28.00	0.56(0.07)	57	4729	0.18(0.02)	13	2078
100_100_30-7	0.33	31.00	2.51(0.24)	630	6330	0.40(0.03)	50	3329
100_100_30-8	0.38	33.00	1.75(0.19)	177	6473	0.56(0.05)	17	3347
100_100_30-9	0.39	31.00	3.73(0.36)	772	6762	0.75(0.07)	107	4019
100_100_40-1	0.56	27.00	0.56(0.16)	19	4600	0.30(0.06)	4	2200
100_100_40-2	0.51	30.00	1.13(0.30)	96	4572	0.47(0.07)	2	2753
100_100_40-3	0.50	31.00	1.63(0.32)	172	6856	1.08(0.18)	47	6535
100_100_40-4	0.52	33.00	2.99(0.76)	255	7549	1.02(0.19)	39	5259
100_100_40-5	0.54	28.00	0.71(0.16)	28	4364	0.46(0.06)	5	2773
100_100_40-6	0.47	34.00	5.16(1.02)	473	9017	1.10(0.17)	38	5931
100_100_40-7	0.51	35.00	7.06(1.62)	730	8534	0.66(0.12)	23	3731
100_100_40-8	0.51	32.00	2.98(0.62)	275	7664	1.14(0.13)	27	5180
100_100_40-9	0.50	35.00	6.71(1.54)	513	12294	1.27(0.25)	53	5569
100_100_50-1	0.73	14.00	0.08(0.02)	0	1100	0.09(0.02)	0	900
100_100_50-2	0.60	25.00	0.58(0.18)	37	4356	0.37(0.06)	2	2660
100_100_50-3	0.69	16.00	0.12(0.05)	0	1500	0.10(0.02)	0	800
100_100_50-4	0.57	22.00	0.25(0.07)	2	2100	0.20(0.07)	0	1700
100_100_50-5	0.73	19.00	0.18(0.09)	0	1600	0.14(0.04)	0	1300
100_100_50-6	0.64	25.00	0.34(0.12)	0	2400	0.30(0.12)	0	1800
100_100_50-7	0.58	27.00	0.59(0.16)	5	3227	0.23(0.08)	0	1800

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	T (s)	#Nodes	#Cuts	T (s)	#Nodes	#Cuts
100_100_50-8	0.68	20.00	0.14(0.06)	0	1900	0.14(0.04)	0	1769
100_100_50-9	0.66	26.00	1.05(0.43)	64	5321	0.55(0.14)	2	3000
100_100_60-1	0.85	10.00	0.06(0.01)	0	800	0.05(0.00)	0	500
100_100_60-2	0.82	12.00	0.13(0.05)	1	1400	0.09(0.02)	0	1000
100_100_60-3	0.83	14.00	0.12(0.03)	4	1200	0.07(0.01)	0	800
100_100_60-4	0.78	13.00	0.11(0.04)	0	1600	0.09(0.02)	0	1000
100_100_60-5	0.86	9.00	0.08(0.02)	2	900	0.05(0.01)	0	700
100_100_60-6	0.83	9.00	0.05(0.01)	0	600	0.05(0.01)	0	600
100_100_60-7	0.79	13.00	0.09(0.03)	0	1200	0.07(0.02)	0	900
100_100_60-8	0.78	11.00	0.07(0.02)	0	1100	0.05(0.01)	0	600
100_100_60-9	0.78	10.00	0.08(0.02)	0	800	0.07(0.01)	0	700
100_100_70-1	0.89	5.00	0.05(0.00)	0	500	0.04(0.00)	0	300
100_100_70-2	0.90	7.00	0.05(0.01)	0	600	0.04(0.00)	0	400
100_100_70-3	0.87	10.00	0.08(0.01)	4	700	0.04(0.00)	0	500
100_100_70-4	0.82	8.00	0.06(0.01)	0	800	0.05(0.01)	0	800
100_100_70-5	0.85	7.00	0.06(0.01)	0	600	0.04(0.01)	0	518
100_100_70-6	0.88	5.00	0.04(0.00)	0	300	0.04(0.00)	0	300
100_100_70-7	0.90	6.00	0.05(0.00)	0	600	0.03(0.00)	0	300
100_100_70-8	0.86	7.00	0.05(0.01)	0	800	0.04(0.00)	0	400
100_100_70-9	0.83	6.00	0.06(0.01)	0	500	0.04(0.00)	0	300
100_100_80-1	0.95	1.00	0.04(0.00)	0	100	0.02(0.00)	0	100
100_100_80-2	0.93	3.00	0.04(0.00)	0	200	0.02(0.00)	0	200
100_100_80-3	0.93	4.00	0.04(0.00)	0	200	0.02(0.00)	0	200
100_100_80-4	0.90	4.00	0.04(0.00)	0	400	0.03(0.00)	0	200
100_100_80-5	0.93	4.00	0.04(0.00)	0	300	0.02(0.00)	0	100
100_100_80-6	0.92	5.00	0.05(0.00)	0	400	0.03(0.00)	0	400
100_100_80-7	0.91	6.00	0.05(0.00)	3	400	0.03(0.00)	0	400
100_100_80-8	0.94	3.00	0.04(0.00)	0	200	0.02(0.00)	0	200
100_100_80-9	0.91	5.00	0.04(0.00)	0	300	0.02(0.00)	0	200
100_100_90-1	0.98	0.00	0.04(0.00)	0	0	0.01(0.00)	0	0

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	T (s)	#Nodes	#Cuts	T (s)	#Nodes	#Cuts
100_100_90-2	0.97	1.00	0.04(0.00)	0	100	0.02(0.00)	0	100
100_100_90-3	0.95	2.00	0.04(0.00)	0	200	0.02(0.00)	0	200
100_100_90-4	0.98	1.00	0.04(0.00)	0	100	0.01(0.00)	0	100
100_100_90-5	0.98	0.00	0.04(0.00)	0	0	0.02(0.00)	0	0
100_100_90-6	0.98	0.00	0.04(0.00)	0	0	0.02(0.00)	0	0
100_100_90-7	0.96	0.00	0.04(0.00)	0	0	0.02(0.00)	0	0
100_100_90-8	0.99	0.00	0.04(0.00)	0	0	0.02(0.00)	0	0
100_100_90-9	0.95	3.00	0.04(0.00)	0	200	0.02(0.00)	0	200
200_100_10-1	0.10	64.00	1739.87(36.54)	222770	7980	50.27(3.95)	11273	5515
200_100_10-2	0.11	59.00	264.09(5.98)	42434	6584	7.05(0.83)	2566	4468
200_100_10-3	0.11	58.00	14.09(0.57)	4204	5745	0.98(0.11)	170	4155
200_100_10-4	0.10	54.00	11.31(0.53)	4391	5477	2.37(0.20)	754	3717
200_100_10-5	0.11	53.00	3.66(0.24)	1419	4855	0.43(0.05)	94	2762
200_100_10-6	0.11	57.00	110.91(3.61)	31488	6020	6.84(0.70)	2379	4887
200_100_10-7	0.11	66.00	11108.05(198.95)	1123275	9034	277.40(20.01)	56740	5799
200_100_10-8	0.11	55.00	22.71(0.94)	7160	5655	2.10(0.17)	510	4089
200_100_10-9	0.11	61.00	149.70(3.45)	22209	7201	23.05(2.12)	6574	5143
200_100_20-7	0.24	73.00	12124.31(355.81)	68427	54986	111.77(9.87)	1598	25471
200_100_20-8	0.24	72.00	6868.43(249.22)	57879	41997	127.89(10.88)	2077	27212
200_100_30-1	0.36	73.00	3637.97(345.51)	18592	53877	52.03(8.60)	332	28733
200_100_30-2	0.38	69.00	1524.23(140.71)	8295	54410	55.79(9.36)	406	29103
200_100_30-3	0.38	71.00	1033.69(132.92)	7326	41407	42.87(8.88)	420	21181
200_100_30-5	0.38	75.00	10623.76(746.76)	33921	82695	177.12(32.72)	1033	44690
200_100_30-6	0.40	69.00	695.09(93.03)	4574	38305	24.48(5.21)	113	20301
200_100_30-7	0.35	72.00	1743.49(224.45)	9509	55409	89.24(19.48)	536	35222
200_100_30-8	0.39	68.00	316.24(65.14)	2906	34547	8.74(1.96)	24	9449
200_100_30-9	0.37	67.00	992.39(82.99)	8235	43334	72.60(11.97)	643	33810
200_100_40-1	0.47	76.00	5448.62(690.29)	13836	79218	208.30(87.08)	859	40355
200_100_40-2	0.51	63.00	145.49(67.80)	1345	21445	28.21(12.26)	10	12911
200_100_40-3	0.53	59.00	25.56(12.75)	206	14870	9.56(3.69)	15	10209

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	T (s)	#Nodes	#Cuts	T (s)	#Nodes	#Cuts
200_100_40-4	0.56	69.00	379.56(148.43)	2334	32692	31.78(11.54)	19	13802
200_100_40-5	0.55	59.00	28.44(16.79)	200	12800	17.05(8.00)	6	10448
200_100_40-6	0.50	72.00	2311.61(363.05)	9032	57260	190.61(57.30)	856	52817
200_100_40-7	0.51	68.00	594.76(169.30)	3396	38716	27.87(11.44)	71	16872
200_100_40-8	0.52	67.00	469.21(164.48)	2731	38712	25.69(11.05)	29	17154
200_100_40-9	0.53	75.00	5305.61(796.45)	16385	71752	222.80(68.72)	784	55488
200_100_50-1	0.68	45.00	10.64(8.13)	7	6500	24.90(14.38)	0	4000
200_100_50-2	0.66	43.00	6.88(4.97)	4	5700	4.84(2.71)	0	3700
200_100_50-3	0.69	39.00	4.80(3.67)	7	4300	8.50(4.91)	0	3800
200_100_50-4	0.61	46.00	10.91(8.85)	2	5800	17.55(11.04)	0	5600
200_100_50-5	0.71	39.00	4.42(3.40)	0	4000	6.76(3.91)	0	3800
200_100_50-6	0.63	51.00	7.96(5.55)	3	5700	10.61(6.50)	0	5000
200_100_50-7	0.61	53.00	22.02(15.04)	40	10636	23.97(15.90)	0	6300
200_100_50-8	0.67	42.00	4.84(3.69)	0	4800	3.84(2.17)	0	4100
200_100_50-9	0.68	46.00	10.38(7.76)	4	6400	13.72(8.58)	0	4500
200_100_60-1	0.79	26.00	1.76(0.99)	0	2200	1.49(0.65)	0	2000
200_100_60-2	0.82	28.00	4.32(3.22)	2	3800	5.45(2.92)	0	2800
200_100_60-3	0.77	26.00	2.89(2.04)	0	2900	3.63(1.90)	0	2800
200_100_60-4	0.79	24.00	2.27(1.47)	0	2900	1.26(0.49)	0	1600
200_100_60-5	0.82	23.00	2.26(1.40)	2	2800	1.73(0.75)	0	1900
200_100_60-6	0.84	19.00	1.21(0.51)	0	1600	0.80(0.26)	0	1300
200_100_60-7	0.80	32.00	4.17(2.95)	5	4700	3.93(2.14)	0	3000
200_100_60-8	0.77	20.00	1.48(0.76)	0	2100	1.05(0.40)	0	1500
200_100_60-9	0.79	29.00	2.64(1.79)	0	3100	2.00(0.97)	0	2300
200_100_70-1	0.85	17.00	1.63(0.89)	0	1900	1.16(0.45)	0	1500
200_100_70-2	0.89	12.00	0.89(0.24)	0	1500	0.42(0.10)	0	800
200_100_70-3	0.84	23.00	2.05(1.20)	8	2200	1.76(0.79)	0	1700
200_100_70-4	0.84	15.00	1.36(0.66)	0	2200	0.82(0.28)	0	1100
200_100_70-5	0.85	16.00	1.45(0.72)	0	1800	1.12(0.41)	0	1500
200_100_70-6	0.87	13.00	0.88(0.22)	0	1400	0.50(0.14)	0	1100

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	T (s)	#Nodes	#Cuts	T (s)	#Nodes	#Cuts
200_100_70-7	0.92	11.00	0.82(0.18)	0	1400	0.34(0.08)	0	700
200_100_70-8	0.86	16.00	1.21(0.51)	0	1900	0.76(0.25)	0	1300
200_100_70-9	0.84	14.00	1.17(0.47)	0	1600	0.82(0.30)	0	1500
200_100_80-1	0.90	14.00	0.97(0.28)	4	1600	0.46(0.14)	0	1000
200_100_80-2	0.92	8.00	0.75(0.12)	0	800	0.31(0.06)	0	500
200_100_80-3	0.93	7.00	0.64(0.03)	0	400	0.23(0.04)	0	600
200_100_80-4	0.90	8.00	0.69(0.06)	0	700	0.26(0.05)	0	500
200_100_80-5	0.92	10.00	0.76(0.12)	3	800	0.27(0.06)	0	800
200_100_80-6	0.93	8.00	0.71(0.09)	0	700	0.25(0.04)	0	500
200_100_80-7	0.93	8.00	0.67(0.06)	0	600	0.25(0.04)	0	500
200_100_80-8	0.92	11.00	0.78(0.13)	0	1000	0.36(0.09)	0	900
200_100_80-9	0.92	9.00	0.75(0.11)	2	1100	0.26(0.05)	0	500
200_100_90-1	0.96	2.00	0.59(0.00)	0	200	0.16(0.01)	0	200
200_100_90-2	0.96	3.00	0.59(0.00)	0	200	0.15(0.01)	0	200
200_100_90-3	0.96	3.00	0.60(0.00)	0	200	0.15(0.01)	0	200
200_100_90-4	0.97	4.00	0.61(0.00)	0	300	0.14(0.01)	0	300
200_100_90-5	0.99	1.00	0.59(0.00)	0	100	0.12(0.00)	0	100
200_100_90-6	0.98	0.00	0.59(0.00)	0	0	0.12(0.00)	0	0
200_100_90-7	0.97	4.00	0.61(0.01)	0	300	0.16(0.02)	0	300
200_100_90-8	0.98	1.00	0.59(0.00)	0	100	0.13(0.00)	0	100
200_100_90-9	0.97	3.00	0.59(0.00)	0	200	0.14(0.01)	0	200
300_100_30-8	0.37	99.00	2603.03(485.71)	4502	73133	151.19(47.30)	101	41264
300_100_40-1	0.54	92.00	707.34(408.74)	1035	47910	176.34(109.72)	8	18742
300_100_40-2	0.52	89.00	559.04(395.71)	1192	32296	193.20(133.66)	2	16900
300_100_40-3	0.52	92.00	1545.94(368.24)	1321	104765	161.33(102.55)	33	23800
300_100_40-4	0.57	98.00	2934.19(1539.56)	5102	49422	154.01(91.30)	11	19300
300_100_40-5	0.55	89.00	322.71(234.56)	464	30100	380.51(256.80)	2	24600
300_100_40-7	0.53	102.00	11961.18(3536.73)	11038	98403	291.06(177.96)	61	31011
300_100_40-8	0.54	104.00	6258.65(2470.50)	7155	67871	1286.73(689.74)	255	106531
300_100_50-2	0.70	63.00	49.56(43.54)	9	7900	86.22(57.25)	0	6500

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	T (s)	#Nodes	#Cuts	T (s)	#Nodes	#Cuts
300_100_50-4	0.61	69.00	66.23(57.11)	6	10200	180.22(124.37)	0	8300
300_100_50-5	0.69	65.00	35.13(30.15)	0	6900	98.94(66.27)	0	7200
300_100_50-6	0.62	81.00	112.54(89.12)	38	16800	274.04(204.17)	0	11900
300_100_50-8	0.66	65.00	37.81(32.84)	0	7200	43.65(27.78)	0	6200
300_100_60-1	0.82	35.00	15.45(11.58)	0	4400	14.16(8.17)	0	3500
300_100_60-2	0.82	43.00	23.77(19.42)	4	4500	44.37(27.80)	0	4700
300_100_60-3	0.79	42.00	31.73(27.18)	2	5200	57.32(36.06)	0	4400
300_100_60-4	0.81	34.00	15.98(12.19)	0	3900	13.69(7.85)	0	3400
300_100_60-5	0.84	28.00	11.00(7.34)	0	3000	7.79(3.54)	0	2400
300_100_60-6	0.82	29.00	9.12(5.57)	0	3300	6.86(3.08)	0	2600
300_100_60-7	0.79	47.00	29.33(24.46)	2	5900	43.03(25.07)	0	3400
300_100_60-8	0.77	37.00	15.27(11.53)	0	3800	10.29(5.27)	0	2800
300_100_60-9	0.78	44.00	22.39(18.27)	0	4800	28.06(18.20)	0	4600
300_100_70-1	0.87	32.00	15.68(11.58)	6	4600	19.43(11.74)	0	3700
300_100_70-2	0.88	20.00	7.25(3.73)	4	3200	3.53(1.44)	0	1700
300_100_70-3	0.86	29.00	13.34(9.54)	2	3500	13.41(7.66)	0	3000
300_100_70-4	0.85	26.00	11.89(8.19)	0	4700	5.22(2.40)	0	2100
300_100_70-5	0.87	27.00	8.86(5.29)	8	2900	5.69(2.84)	0	2700
300_100_70-6	0.88	20.00	5.85(2.44)	2	2300	3.23(1.23)	0	1800
300_100_70-7	0.91	14.00	4.44(1.20)	0	2000	1.76(0.47)	0	1000
300_100_70-8	0.86	27.00	13.15(9.43)	0	4200	12.31(6.16)	0	2400
300_100_70-9	0.84	25.00	12.81(9.15)	0	3800	9.29(4.55)	0	2300
300_100_80-1	0.92	18.00	5.32(1.98)	5	2600	2.07(0.70)	0	1300
300_100_80-2	0.93	12.00	4.28(1.05)	2	1500	1.40(0.32)	0	700
300_100_80-3	0.94	10.00	3.31(0.24)	0	800	0.91(0.20)	0	900
300_100_80-4	0.92	12.00	3.64(0.54)	0	1600	0.95(0.23)	0	800
300_100_80-5	0.92	13.00	3.77(0.58)	0	1000	1.49(0.51)	0	1300
300_100_80-6	0.91	14.00	5.17(1.83)	0	2400	2.57(0.68)	0	1000
300_100_80-7	0.93	13.00	3.86(0.65)	0	1400	1.48(0.41)	0	1200
300_100_80-8	0.92	18.00	5.12(1.77)	2	2300	2.78(1.13)	0	1800

Continues on next page ...

			Claw Model			Star Model		
Instance	Density	OPT	**T (s)**	**#Nodes**	**#Cuts**	**T (s)**	**#Nodes**	**#Cuts**
300_100_80-9	0.91	20.00	5.54(2.18)	11	2700	2.40(0.99)	0	1800
300_100_90-1	0.97	7.00	3.08(0.06)	2	600	0.50(0.06)	0	600
300_100_90-2	0.96	6.00	3.21(0.12)	0	800	0.59(0.07)	0	400
300_100_90-3	0.96	3.00	3.07(0.02)	0	300	0.52(0.04)	0	300
300_100_90-4	0.97	8.00	3.22(0.12)	3	700	0.61(0.10)	0	600
300_100_90-5	0.99	1.00	3.00(0.00)	0	100	0.42(0.01)	0	100
300_100_90-6	0.98	1.00	3.02(0.00)	0	100	0.40(0.00)	0	100
300_100_90-7	0.96	6.00	3.18(0.11)	0	600	0.61(0.09)	0	500
300_100_90-8	0.98	3.00	3.05(0.01)	0	300	0.42(0.02)	0	200
300_100_90-9	0.97	6.00	3.15(0.09)	0	700	0.55(0.07)	0	400