UNIVERSIDAD DE BUENOS AIRES

Facultad de Ciencias Exactas y Naturales

Departamento de Computación

Estudio poliedral y algoritmo branch-and-cut para el problema de coloreo equitativo en grafos

Tesis presentada para optar al título de Doctor de la Universidad de Buenos Aires en el área Ciencias de la Computación

Daniel E. Severin

Directoras de Tesis:	Isabel Méndez-Díaz
	Graciela L. Nasini
Consejera de Estudios:	Isabel Méndez-Díaz

Lugar de trabajo:	Departamento de Matemática
	Facultad de Ciencias Exactas, Ingeniería y Agrimensura
	Universidad Nacional de Rosario

Buenos Aires, 2012

Estudio poliedral y algoritmo Branch-and-Cut para el Problema de Coloreo Equitativo en Grafos

Resumen

Los problemas de coloreo de grafos constituyen una familia de problemas de una gran relevancia tanto teórica como práctica. Todos ellos son variaciones del problema del coloreo clásico, cuyo estudio se inició en el Siglo XIX. El origen de estas variaciones radica en las restricciones adicionales que imponen las aplicaciones a problemas de la vida real. En esta tesis abordamos en particular el *Problema de Coloreo Equitativo en Grafos*.

Como muchos problemas de Optimización Combinatoria, el Problema de Coloreo Equitativo es un problema NP-difícil. Los algoritmos Branch-and-Cut basados en el estudio poliedral de una formulación del problema como programa lineal entero, son la herramienta más efectiva que se conoce para la resolución exacta de problemas NP-difíciles.

En esta tesis se propone un modelo de programación lineal entera para el Problema del Coloreo Equitativo y se estudia el poliedro asociado. Se derivan varias familias de desigualdades validas y se prueba que siempre definen caras de alta dimensión, lo cual es un buen indicador para la utilización de las mismas como planos de corte. Finalmente, se desarrolla e implementa un algoritmo Branch-and-Cut para el Problema de Coloreo Equitativo que resulta altamente competitivo con los algoritmos exactos conocidos.

Palabras claves: Branch-and-Cut
Coloreo Equitativo de Grafos
Programación Lineal Entera
Poliedros
Planos de Corte

A polyhedral study and a Branch-and-Cut Algorithm for the Equitable Graph Coloring Problem

Abstract

The graph coloring problems constitute a family of problems of great theoretical and practical relevance. All of them are variations of the classical coloring problem, whose study began in the nineteenth century. The origin of these variations arises from the additional restrictions imposed by applications to real life problems. In particular, this thesis deals with the *Equitable Graph Coloring Problem*.

Like many combinatorial optimization problems, the Equitable Coloring Problem is NP-hard. It is known that Branch-and-Cut algorithms based on the polyhedral study of a formulation of the problem as an integer linear program, are the most effective tool for solving NP-hard problems.

This thesis proposes a linear integer programming model for the Equitable Coloring Problem and studies the polytope associated to that model. Several families of valid inequalities that define faces of high dimension are derived in order to use them later as cutting planes. Finally, a Branch-and-Cut algorithm for the Equitable Coloring Problem that is highly competitive against known exact algorithms is developed.

Keywords: Branch-and-Cut
Equitable Graph Coloring
Integer Linear Programming
Polyhedra
Cutting Plane

Agradecimientos

Agradezco principalmente a mis directoras por el amplio apoyo que me dieron desde que inicié mi doctorado.

Agradezco también a los integrantes del Grupo de Optimización Combinatoria de la Universidad Nacional de Rosario y del Departamento de Computación de la Universidad de Buenos Aires, por haberme acompañado a lo largo de este largo proceso generando un agradable entorno de trabajo y haciéndome más sencillo el día a día.

A mi papá Aurelio
A mi hermana María Julia
A mi novia Diana
A la memoria de mi mamá Adriana

Índice general

Lista de Algoritmos

Índice de cuadros

Índice de figuras

Capítulo 1

Introducción

1.1. Problemas de coloreo en grafos

Varios autores coinciden en indicar como el primer trabajo de Teoría de Grafos el documento sobre los 7 puentes de Königsberg, publicado en 1736 por Leonhard Euler [34]. Sin embargo, no fue hasta 1878 que James Joseph Sylvester, en un artículo publicado en la revista *Nature* [73], acuña el término *grafo* para denomimar a estas estructuras definidas por un número finito de *vértices* y una familia de pares de vértices, denominados *aristas*. Hoy en día, los grafos permiten modelar muchos y muy diversos problemas de interés práctico, incluyendo la dinámica de procesos en sistemas físicos, biológicos y sociales.

Uno de los problemas más famosos y conocidos de la Teoría de Grafos es el Problema de los Cuatro Colores: ¿*Es cierto que 4 colores son suficientes para pintar las regiones de cualquier mapa dibujado sobre un plano y de manera que dos regiones con una frontera común tengan diferentes colores*? Esta pregunta fue formulada por Francis Guthrie en 1852 cuando intentaba colorear el mapa de los condados de Inglaterra.

Si se construye un grafo con un vértice por cada región a colorear y una arista entre cada par de vértices correspondientes a regiones linderas, el problema se reduce a preguntarse si es posible colorear con cuatro colores los vértices de este grafo, con la restricción de que dos vértices unidos por una arista no compartan el mismo color. Los grafos así obtenidos resultan ser los *grafos planares* y el Problema de los Cuatro Colores plantea determinar si todo grafo planar puede ser coloreado con 4 colores.

El problema permaneció sin resolverse hasta 1976 cuando Kenneth Appel y Wolfgang Haken hacen uso de las nociones desarrolladas por Heesch en [43] y reducen el problema a chequear ciertas propiedades sobre 1.936 configuraciones. Esta tarea se realiza con la asistencia de una computadora y permite contestar afirmativamente a la pregunta original [7]. La prueba no fue plenamente aceptada por la comunidad científica debido a la complejidad de verificar manualmente los resultados de la computadora sobre cada una de las 1.936 configuraciones diferentes. Posteriormente, en 1997, Neil Robertson, Daniel Sanders, Paul Seymour y Robin Thomas simplifican la prueba a 633 configuraciones [70].

La extensión natural del Problema de los Cuatro Colores es el problema conocido como *Coloreo de Grafos* (PCG): *¿cuál es la cantidad mínima de colores necesaria para colorear los vértices de un grafo de tal manera que vértices conectados por una arista no compartan el mismo color?*

El PCG es uno de los problemas de Teoría de Grafos de mayor impacto debido al gran número de aplicaciones que pueden ser modeladas como instancias del mismo. Sólo a modo de ejemplo mencionamos las siguientes: Programación de Tareas [32], Asignación de Frecuencias [53] y Asignación de Registros [23]. Una lista más completa de aplicaciones se encuentra en [67].

El impacto de la resolución de problemas de la vida real a través de su modelización como instancias del PCG llevó naturalmente a intentar resolver otros problemas cuyas soluciones podían también modelarse como coloreos de un grafo aunque con ciertas restricciones adicionales. Esto ha dado lugar a numerosas

variantes del PCG como por ejemplo *List Coloring*, en donde cada vértices tiene especificada una lista de colores admisibles, o *T-Coloring* donde, si los colores se representan con números naturales, la diferencia entre colores asignados a vértices adyacentes no debe pertenecer a una lista preestablecida [33]. En particular, ciertas aplicaciones del PCG requieren *equilibrar* la cantidad de vértices pintados con cada color, lo cual da lugar al concepto de *coloreos equitativos*.

1.2. El Problema de Coloreo Equitativo de Grafos

En los problemas de Programación de Tareas, cada vértice del grafo es una tarea a realizar, las aristas indican ciertas incompatibilidades que impidan que ambas puedan ser realizadas por el mismo trabajador (por ejemplo, en el caso en que ambas tareas deban llevarse a cabo en el mismo horario) y cada color es un trabajador al cual se le asigna un conjunto de vértices o tareas. El objetivo es contratar la menor cantidad de trabajadores posibles.

En este problema es natural requerir que la distribución de la carga laboral sea uniforme. Esta restricción adicional de *equitatividad* da lugar a una variante del PCG llamada *Problema de Coloreo Equitativo en Grafos* (PCEG), introducida por Meyer en 1973 [65]. Más precisamente, en el PCEG se requiere que, dados dos colores cualesquiera, la cantidad de vértices pintados con cada uno de estos colores difiera a lo sumo en una unidad.

Entre otras aplicaciones del PCEG podemos mencionar:

- *Asignación de aulas a asignaturas* [33]. Consideremos una universidad que necesita determinar en qué aulas se van a impartir ciertas asignaturas. La universidad puede tener como política adicional que el uso de las aulas sea homogéneo, evitando que algunas de ellas se deterioren más que otras con el paso del tiempo. Construyamos un grafo en donde cada vértice representa una asignatura y dos vértices son adyacentes cuando las asignaturas que representan no se pueden impartir simultáneamente en la misma aula debido a una incompatibilidad horaria. Si representamos a las aulas con colores entonces un coloreo equitativo del grafo indica cómo debe efectuarse tal asignación.

- *Recolección de residuos* [74, 65]. Supongamos que las rutas de recolección de residuos de una ciudad están previamente establecidas y todas deben ser cubiertas en una semana (de lunes a sábado). Ademas, si dos rutas cubren parcialmente un mismo tramo, no pueden ser recorridas el mismo día. El objetivo es que el número de rutas que se recorran cada día sea balanceado. Entonces construyamos un grafo en donde cada vértice represente una ruta y dos vértices son adyacentes cuando las rutas que representan no se pueden recorrer el mismo día. Teniendo en cuenta que cada día puede ser representado con un color, un coloreo equitativo de 6 colores indica en cúales días se debe recorrer cada ruta.

- *Esquemas de distribución paralela de datos en memoria* [31]. En arquitecturas multiprocesadores (es decir, sistemas que deben correr sobre varios procesadores que acceden paralelamente a varios módulos de memoria), un módulo de memoria puede alojar distintos bloques de datos en distintos momentos de la ejecución de un programa. Se espera utilizar la menor cantidad de módulos posibles siempre que, en todo momento, los procesadores que corran el programa tengan acceso a los bloques de datos necesarios. Este concepto es conocido como *acceso de datos libre de conflictos* y modelado como un grafo donde los vértices representan bloques de datos y dos vértices son adyacentes si dos bloques de datos no pueden almacenarse en el mismo módulo de memoria. Si representamos con un color a cada módulo de memoria, un coloreo equitativo del grafo señala cómo deben asignarse los módulos de memoria a los bloques de datos de forma tal que no se produzcan conflictos y que la memoria se utilice uniformemente, a fín de minimizar las esperas en los accesos entre procesadores y memoria. Esto último es conocido en la jerga como *balanceo de carga*.

Tambien en [68, 47] pueden encontrarse aplicaciones teóricas del PCEG en Teoría de Probabilidades.

1.3. Complejidad de la resolución del PCEG

El PCEG es un problema NP-difícil [36] y, por lo tanto, no se conoce un algoritmo que, para cualquier instancia, encuentre la solución óptima en tiempo polinomial. Como es habitual en esta clase de problemas, las aplicaciones concretas imponen la necesidad de contar con herramientas que nos den alguna respuesta en tiempos viables. Una forma de abordar esta problemática es proponer algoritmos *heurísticos* que encuentren soluciones relativamente buenas en un tiempo razonable pero sin garantizar la optimalidad de la solución hallada.

La otra posibilidad es proponer algoritmos *exactos* que encuentren una solución óptima del problema a sabiendas de que el tiempo requerido para alcanzarla podría ser muy grande o incluso prohibitivo en algunos casos. En esta línea, los algoritmos exactos que han demostrado mejor performance son los basados en una formulación del problema original como un problema de *Programación Lineal Entera*. En los últimos años, las técnicas de *Branch-and-Cut* que explotan la estructura poliedral del modelo de Programación Lineal Entera, han permitido resolver de forma exacta instancias de problemas NP-difíciles que sería imposible de lograr con otros medios. El ejemplo más relevante de la potencialidad de estas herramientas es la resolución de instancias del problema del viajante de comercio [8]. Por ejemplo, el programa *Concorde* desarrollado por David Applegate, Robert E. Bixby, Vašek Chvátal y William J. Cook resolvió en 2006 una instancia con 85900 ciudades por optimalidad [9]. También son importantes de mencionar las experiencias para la resolución del problema del máximo conjunto estable [69] y del PCG [64].

En lo que se refiere al PCEG, hasta donde llega nuestro conocimiento, sólo se ha propuesto un algoritmo Branch-and-Cut en [11, 12]. Sin embargo, sólo utiliza resultados del estudio poliedral realizado en [22] para el PCG.

1.4. Objetivos de la tesis

El objetivo general de esta tesis es:

implementar un algoritmo competitivo de Ramificación y Corte (Branch-and-Cut) que resuelva el Problema de Coloreo Equitativo de Grafos.

Los objetivos específicos son:

- Proponer diferentes modelos de Programación Lineal Entera para el PCEG y evaluar empíricamente la performance de los mismos.
- Estudiar el poliedro asociado al modelo de Programación Lineal Entera elegido, con el fin de determinar familias de desigualdades válidas que puedan ser empleadas como cortes en el marco de un algoritmo Branch-and-Cut.
- Diseñar heurísticas para el PCEG que permitan mejorar las cotas involucradas en las diferentes etapas de un algoritmo Branch-and-Cut.
- Implementar un algoritmo Branch-and-Cut para el PCEG que incorpore los resultados derivados de los objetivos anteriores y comprobar su competitividad frente a otros algoritmos ya existentes.

La tesis está organizada de la siguiente manera:

En el Capítulo 2 relevamos algunos resultados teóricos del PCEG. En el Capítulo 3 presentamos una heurística para el PCEG y evaluamos la calidad de las soluciones obtenidas por el mismo. En el Capítulo 4 proponemos y analizamos distintos modelos de Programación Lineal Entera para el PCEG. Presentamos también la estructura general de un algoritmo Branch-and-Cut. En el Capítulo 5 desarrollamos el estudio poliedral de uno de los modelos presentados en el Capítulo 4. En el Capítulo 6 describimos en profundidad el algoritmo Branch-and-Cut propuesto para resolver el PCEG. En el Capítulo 7 ajustamos nuestro algoritmo mediante la experimentación computacional y comparamos el mismo con otros algoritmos Branch-and-Cut existentes. El Capítulo 8 está destinado a conclusiones y líneas de investigación futuras relacionadas con este trabajo. El Capítulo 9 es un compendio de demostraciones de teoremas presentados en el Capítulo 5.

1.5. Definiciones y notaciones

En esta sección presentamos las definiciones y notaciones utilizadas a lo largo de esta tesis.

En este trabajo todos los grafos son simples y los notamos $G = (V,E)$, donde $V = \{1,\dots,n\}$ es el conjunto de vértices y E el conjunto de aristas. Si $(u,v) \in E$, llamamos *extremos de la arista* a los vértices u y v, y decimos que u y v son adyacentes. Si $E = \varnothing$, decimos que G es un *grafo aislado*. Si $E = \{(u,v) : u,v \in V,\ u \neq v\}$ decimos que G es un *grafo completo*. Denotamos con K_n al grafo completo de n vértices.

El *complemento de G* es el grafo que resulta de invertir la relación de adyacencia de los vértices de G, y lo denotamos con $\overline{G}$. Es decir, $\overline{G} = (V,\overline{E})$ donde $\overline{E} = \{(u,v) \notin E : u,v \in V,\ u \neq v\}$. Claramente, si G es un grafo aislado entonces $\overline{G}$ es un grafo completo.

Dado $V' \subset V$, denotamos por $G[V']$ al grafo cuyo conjunto de vértices es V' y el conjunto de aristas es el subconjunto de E cuyos extremos son vértices de V', i.e. $G[V'] = (V',E')$ donde $E' = \{(u,v) : u,v \in V',\ u \neq v\} \cap E$. A $G[V']$ lo denominamos *subgrafo de G inducido por V'*. Si G' es isomorfo a algún subgrafo de G inducido por un subconjunto de vértices, notamos $G' \subseteq G$.

Dado $S \subset V$, denotamos con $G - S$ al grafo que resulta de borrar los vértices S y las aristas cuyos extremos están en S, i.e. $G - S = G[V \setminus S]$. En el caso en que S sea un único vértice, $S = \{u\}$, escribimos simplemente $G - u$ en vez de $G - \{u\}$.

Decimos que $S \subset V$ es un *conjunto estable de G* si $G[S]$ es un grafo aislado. Un conjunto estable S es maximal si para todo vértice $v \in V \setminus S$, $S \cup \{v\}$ no es un conjunto estable. La máxima cardinalidad de un estable en G se denota con $\alpha(G)$, también llamado *número de estabilidad de G*. Si S es un conjunto estable de G tal que su cardinal es $\alpha(G)$ decimos que S es un *estable máximo*. Por simplicidad en la notación, dado $V' \subset V$ acordamos notar con $\alpha(V')$ a $\alpha(G[V'])$. Decimos que V' es *α-maximal* si para todo vértice $v \in V \setminus V'$, $\alpha(V' \cup \{v\}) > \alpha(V')$.

Decimos que $Q \subset V$ es una *clique de G* si $G[Q]$ es un grafo completo o, equivalentemente, si $\alpha(Q) = 1$. La máxima cardinalidad de una clique en G se denota con $\omega(G)$. Si Q es una clique cuyo cardinal es $\omega(G)$, diremos que Q es una *clique máxima*. Una *clique maximal* es una clique α-maximal.

La *vecindad* de un vértice v es el conjunto de todos los vértices adyacentes a v y se denota con $N(v)$. La *vecindad cerrada* de v es la unión entre v y $N(v)$ y se denota con $N[v]$. El *grado* de un vértice v es el cardinal de $N(v)$ y se denota con $\delta(v)$. Si $\delta(v) = 0$ decimos que v es *aislado en G* y si $\delta(v) = n-1$ decimos que v es *universal en G*. Denotamos con $\Delta(G)$ al grado del vértice con mayor grado en G. Si, en particular, todos los vértices de G tienen el mismo grado r entonces decimos que G es r-regular.

Un *matching* de G es un subconjunto de aristas sin extremos en común. Con $\nu(G)$ denotamos el máximo número de aristas de un matching de G.

Sea $k \geq 2$. Un grafo P_k es un *camino de longitud k* si tiene $k+1$ vértices que pueden ser indexados como $\{v_0, v_1, v_2, \dots, v_k\}$ de manera que $\{(v_0,v_1)$, $\{(v_1,v_2)$, (v_2,v_3), …, $(v_{k-1},v_k)\}$ es su conjunto de aristas. A los

vértices v_0 y v_k se los llama *extremos* de P_k. Dados $u, v \in V$, un *camino de G entre u y v* es un conjunto $P \subset V$ tal que $G[P]$ es un grafo camino cuyos extremos son u y v. La *distancia* entre los vértices u y v es la longitud mínima entre todos los caminos de G entre u y v.

Sea $k \geq 3$. Un grafo C_k es un *ciclo de longitud k* si tiene k vértices que pueden ser indexados como $\{v_1, v_2, \ldots, v_k\}$ de manera que $\{(v_1,v_2), (v_2,v_3), \ldots, (v_{k-1},v_k), (v_1,v_k)\}$ es su conjunto de aristas. Un *agujero de G* (de longitud k) es un conjunto $C \subset V$ tal que $G[C]$ es un grafo ciclo de longitud k.

Un grafo G es *conexo* si, para dos vértices cualesquiera $u, v \in V$, existe un camino entre u y v en G. Dado $G' = (V', E') \subseteq G$, G' es una *componente conexa de G* si G' es conexo y, para todo $v \in V \backslash V'$, $G[V' \cup \{v\}]$ no es conexo.

Un grafo G es un *grafo árbol* si es conexo y tiene exactamente $n-1$ aristas.

Un grafo G es *bipartito* si existe una partición de su conjunto de vértices en dos conjuntos estables de G.

Denotamos con $K_{p_1,p_2,\ldots,p_r}$ al *grafo r-partito completo,* definido como el grafo cuyo complemento está conformado por r componentes completas $K_{p_1}, K_{p_2}, \ldots, K_{p_r}$. Para el caso $r = 2$, éste se llama *grafo bipartito completo*.

Un *coloreo de G* es una asignación c entre vértices de V y números de $\{1, \ldots, n\}$, tal que todo par de vértices adyacentes reciben distintos números, i.e. $\forall\ (u,v) \in E : c(u) \neq c(v)$. Al número $c(v)$ lo llamamos *color de v*. Si la cardinalidad de la imagen de c es k, entonces decimos que c es un *k-coloreo de G* y también que *G admite un k-coloreo*. El *número cromático de G* se define como el mínimo k para el cual G admite un k-coloreo, y se escribe $\chi(G)$. Dado un coloreo c y un color j, llamamos *clase de color j* al conjunto de vértices que reciben el color j y lo denotamos con C_j, i.e. $C_j = \{v \in V : c(v) = j\}$.

Un coloreo c de G es un *coloreo equitativo* si la diferencia entre los tamaños de dos clases de color no vacías es a lo sumo de una unidad, i.e. $\forall\ i, j$ tales que $C_i \neq \varnothing$, $C_j \neq \varnothing$, se verifica $||C_i| - |C_j|| \leq 1$. Esta imposición se llama *restricción de equidad*. Si c es un k-coloreo que es equitativo entonces decimos que c es un *k-coloreo equitativo,* o para abreviar, un *k-eqcol*. El *número cromático equitativo de G* se define como el mínimo k para el cual G admite un k-eqcol, y se escribe $\chi_{eq}(G)$.

Dado x un número real, $\lfloor x \rfloor$ es el mayor número entero menor o igual que x, y $\lceil x \rceil$ es el menor número entero mayor o igual que x. Sea y un número real distinto de cero, entonces definimos $x \bmod y = x - y\lfloor x/y \rfloor$.

De la definición de coloreo equitativo se deducen fácilmente las siguientes propiedades:

Observación 1.5.1. *Sea G un grafo de n vértices, c un k-coloreo de G para algún* $1 \leq k \leq n$ *y* $r = n \bmod k$. *Entonces, las siguiente afirmaciones son equivalentes:*

- *c es un k-eqcol,*
- *toda clase de color no vacía en c tiene tamaño* $\lfloor n/k \rfloor$ *o* $\lceil n/k \rceil$,
- *en c hay r clases de tamaño* $\lfloor \frac{n}{k} \rfloor + 1$ *y* $k - r$ *clases de tamaño* $\lfloor \frac{n}{k} \rfloor$,
- *si los tamaños de las clases decrecen respecto a los colores, esto es,* $|C_j| \geq |C_{j+1}|$ *para todo* $1 \leq j \leq n-1$, *entonces* $|C_j| = \lceil \frac{n-j+1}{k} \rceil$ *para todo* $1 \leq j \leq k$.

Un conjunto de puntos $x^1, x^2, \ldots, x^k \in \mathbb{R}^n$ es *afínmente independiente* si el sistema

$$\begin{cases} \sum_{i=1}^{k} \alpha_i x^i = 0, \\ \sum_{i=1}^{k} \alpha_i = 0, \end{cases}$$

tiene solución única $\alpha_i = 0$ para todo $1 \leq i \leq k$.

Sea $P \subset \mathbb{R}^n$. Decimos que P es un *poliedro* si es el conjunto solución de un sistema finito de desigualdades lineales; esto es, existen $A \in \mathbb{R}^{m \times n}$ y $b \in \mathbb{R}^m$ tales que $P = \{x \in \mathbb{R}^n : Ax \leq b\}$.

Un poliedro P tiene *dimensión k* si el número máximo de puntos afínmente independientes que se pueden encontrar en P es $k+1$. La dimensión de P se denota con $dim(P)$.

Sea P un poliedro y $A'x = b'$ un sistema de ecuaciones que se verifica para todo $x \in P$. Si el rango de la matríz A' es $n - dim(P)$ entonces $A'x = b'$ es un *sistema minimal de ecuaciones* de P. Toda igualdad lineal satisfecha por todo $x \in P$ puede obtenerse como combinación lineal de las ecuaciones de un sistema minimal de P.

Sea S un conjunto finito de puntos $x^1, x^2, \ldots, x^k \in \mathbb{R}^n$. La *cápsula convexa* de S se define como:

$$conv(S) = \left\{ \sum_{i=1}^{k} \alpha_i x^i : \sum_{i=1}^{k} \alpha_i = 1,\ \alpha_i \geq 0 \ \text{ para todo } 1 \leq i \leq k \right\}.$$

Es sabido que $conv(S)$ es un poliedro para todo S finito.

Decimos que $\pi x \leq \pi_0$ es una *desigualdad válida* para P si se satisface para todo $x \in P$. En tal caso, $F = \{x \in P : \pi x = \pi_0\}$ es la *cara definida* por $\pi x \leq \pi_0$.

Una desigualdad válida $\pi x \leq \pi_0$ define una *cara propia* F de P si $F \neq \varnothing$ y $F \neq P$. En tal caso, existen puntos $x_1, x_2 \in P$ tales que $\pi x_1 = \pi_0$ y $\pi x_2 < \pi_0$. De ahora en más, cuando nos referimos a una cara de P nos estamos refiriendo a una cara propia de P. Decimos que una desigualdad válida define una *faceta* de P si define una cara F tal que $dim(F) = dim(P) - 1$.

Capítulo 2

Problema de Coloreo Equitativo: algunos resultados previos

En este capítulo presentamos algunos resultados generales de la literatura referentes al PCEG. Se presentan similitudes y diferencias entre coloreo clásico y coloreo equitativo, algunas familias de grafos en dónde el PCEG se puede resolver polinomialmente y finalmente, resultados relativos a cotas del número cromático equitativo.

2.1. Coloreo clásico vs. coloreo equitativo

Es sabido que, cualquiera sea el grafo G, si G admite un k-coloreo entonces también admite un $(k+1)$-coloreo. Esto no es válido para los coloreos equitativos [33]. El ejemplo más pequeño que viola la propiedad anterior es el grafo bipartito completo $K_{3,3}$ que puede colorearse equitativamente con 2 colores pero no con 3. En la Figura 2.1 se muestran coloreos equitativos admisibles para $K_{3,3}$.

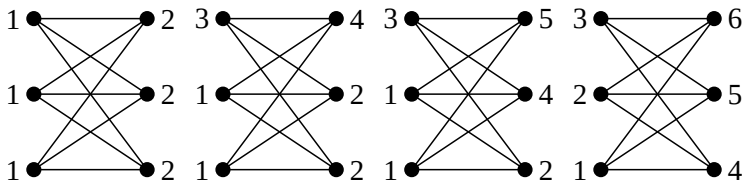


Figura 2.1: Coloreos equitativos de $K_{3,3}$

Esto justifica la definición de *umbral cromático de G* como el mínimo k para el cual G admite un l-eqcol para todo $l \geq k$, y se escribe $\chi^*_{eq}(G)$.

Otra definición de la que haremos un uso intensivo en los capítulos siguientes es el conjunto de aquellos k para los cuales el grafo no admite un k-eqcol:

Definición 2.1.1. *Dado un grafo G,* $\mathscr{S}(G) = \{k : \chi_{eq}(G) \leq k \leq n, G \text{ no admite un } k\text{-eqcol}\}$.

En el caso de coloreo clásico, el umbral cromático coincide siempre con el número cromático. Sin embargo, $\chi_{eq}(K_{3,3}) = 2$ mientras que $\chi^*_{eq}(K_{3,3}) = 4$. Además, $\mathscr{S}(K_{3,3}) = \{3\}$.

Los resultados presentados en [54], trabajando con grafos aleatorios distribuidos uniformemente, indican que los parámetros χ_{eq} y χ^*_{eq} *no se alejan mucho* entre sí. El comportamiento asintótico de los mismos se

expresa a través de la siguiente relación:

$$\chi(G) \leq \chi_{eq}(G) \leq \chi^*_{eq}(G) = (1+o(1))\chi(G).$$

Esta quiere decir que, a medida que el tamaño del grafo aumenta, tanto χ_{eq} como χ^*_{eq} tienden a separarse de χ en una distancia constante.

Los grafos para los cuales el número cromático equitativo coincide con el umbral cromático equitativo (o equivalentemente, los grafos G tales que $\mathscr{S}(G) = \varnothing$) se denominan *monótonos*.

Determinar si un grafo G es monótono o explicitar $\mathscr{S}(G)$ son problemas tan difíciles como el propio PCEG. Sin embargo, existen familias de grafos donde estas propiedades son conocidas. Por ejemplo, es sabido que los grafos árboles [57] y los grafos de intervalos [26] son monótonos.

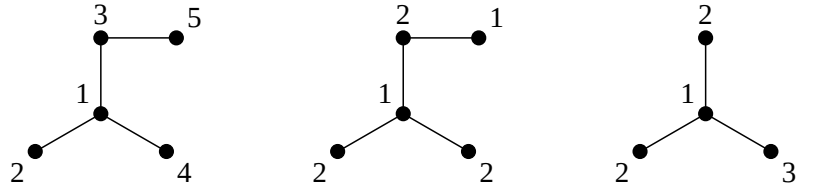
En los grafos bipartitos completos es trivial ver que, para todo $m \geq 3$, $\chi_{eq}(K_{m,m}) = 2$, mientras que en [58] se prueba que $K_{m,m}$ admite un k-eqcol si y sólo si

$$\left\lceil \frac{m}{\lfloor k/2 \rfloor} \right\rceil - \left\lfloor \frac{m}{\lceil k/2 \rceil} \right\rfloor \leq 1.$$

Luego, $\chi^*_{eq}(K_{m,m}) = \min\{r : \lceil \frac{m}{\lfloor k/2 \rfloor} \rceil - \lfloor \frac{m}{\lceil k/2 \rceil} \rfloor \leq 1 \quad \forall\, k \geq r\}$.

Otra diferencia entre coloreos clásicos y equitativos es la referida a la relación entre los números cromáticos de un grafo y sus subgrafos.

Sabemos que si G' es un subgrafo de G entonces $\chi(G') \leq \chi(G)$. De esta manera, el número crómatico de cualquier subgrafo de G nos aporta una cota inferior de $\chi(G)$. Esta propiedad no es válida en los coloreos equitativos. Por ejemplo, si G es el grafo de la figura 2.2(a), $\chi_{eq}(G) = 2$ y sin embargo, $\chi_{eq}(G-5) = 3$.



(a) vértices de G (b) 2-eqcol de G (c) 3-eqcol de $G-5$

Figura 2.2: Coloreos equitativos de un grafo y un subgrafo

Otra implicancia negativa de esta propiedad es el hecho de que no nos permite restringirnos a trabajar con grafos conexos. En efecto, si G es un grafo no conexo con componentes G_i donde $i \in \{1,\ldots,m\}$, es sabido que:

$$\chi(G) = \max\{\chi(G_1), \chi(G_2), \ldots, \chi(G_m)\}.$$

Lamentablemente la propiedad anterior no se verifica para $\chi_{eq}(G)$ como lo muestra el siguiente ejemplo. Sea G un grafo compuesto de 3 componentes conexas isomorfas a $K_{1,5}$. Entonces $\chi_{eq}(K_{1,5}) = 4$ pero $\chi_{eq}(G) = 3$.

Sin embargo, los coloreos equitativos de las componentes conexas permiten construir coloreos equitativos del grafo en el sentido expresado en el siguiente resultado:

Teorema 2.1.2. *[79] Sean G_1, G_2, …, G_m las componentes conexas de G. Si G_i admite un k-eqcol para todo $i \in \{1,\ldots,m\}$, entonces G también admite un k-eqcol.*

Del resultado anterior se desprende también el siguiente corolario.

Corolario 2.1.3. $\chi_{eq}(G) \leq \chi^*_{eq}(G) \leq \max\{\chi^*_{eq}(G_1), \chi^*_{eq}(G_2), \ldots, \chi^*_{eq}(G_m)\}$.

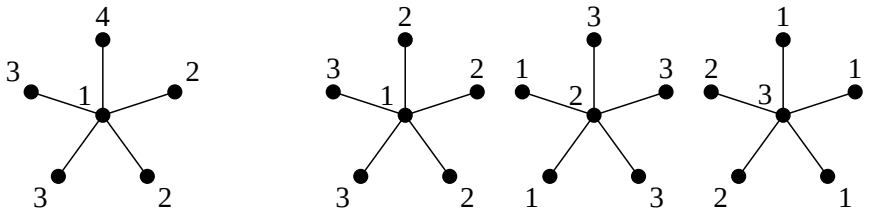


Figura 2.3: Coloreos equitativos de $K_{1,5}$ y G

2.2. Familias de grafos con números cromáticos equitativos conocidos

Si bien resolver el PCEG en grafos arbitrarios es NP-difícil, existen familias de grafos en donde resulta sencillo calcular su número cromático equitativo. Una lista de estas familias puede ser consultada en el libro *Graph Colorings* de Kubale [33]. A continuación enumeramos algunas de ellas.

- **Grafos completos, casi completos y aislados**. Tenemos que $\chi_{eq}(G) = n$ si y sólo si $G = K_n$. También, $\chi_{eq}(G) = n-1$ si y sólo si $\omega(G) = n-1$. Además, $\chi_{eq}(G) = 1$ si y sólo si $\omega(G) = 1$.

- **Ciclos, Complementos de Ciclos y Ruedas**. [33] Es sencillo verificar que para ciclos de longitud par se satisface $\chi_{eq}(C_{2k}) = 2$ mientras que para los de longitud impar se satisface $\chi_{eq}(C_{2k+1}) = 3$. Para los complementos de ciclos se verifica $\chi_{eq}(\overline{C_n}) = \lceil n/2 \rceil$. Sea W_n un grafo de n vértices que tiene un agujero de longitud $n-1$ y el vértice que no pertenece al agujero es universal en W_n. A este grafo se le llama *rueda* y verifica $\chi_{eq}(W_n) = \lceil \frac{n-1}{2} \rceil + 1$.

- **Hipercubos**. [33] Un *hipercubo* Q_d es un grafo definido recursivamente de la siguiente forma: Q_2 es un grafo completo de 2 vértices y Q_d se forma a partir unir 2 copias de Q_{d-1}. Si $Q^1_{d-1} = (V^1, E^1)$ y $Q^2_{d-1} = (V^2, E^2)$ representan cada copia de Q_{d-1} entonces $Q_d = (V, E)$ donde $V = V^1 \cup V^2$ y $E = E^1 \cup E^2 \cup \{(v^1, v^2) : v^1 \in V^1, v^2 \in V^2\}$. Es simple corroborar que $\chi_{eq}(Q_d) = 2$.

- **Grafos árboles**. [57] Sea T un árbol y X e Y una partición en conjuntos estables de sus vértices. Entonces

$$\chi_{eq}(T) = \begin{cases} 2, & \text{si } ||X| - |Y|| \leq 1, \\ \text{máx}\left\{3, \text{máx}\left\{\left\lceil \dfrac{n+1}{\alpha(V \backslash N[v]) + 2} \right\rceil : v \in V\right\}\right\}, & \text{si no.} \end{cases}$$

  El cálculo de $\chi_{eq}(T)$ involucra conocer el número de estabilidad de $V \backslash N[v]$ que se puede obtener polinomialmente, pues las componentes conexas de $V \backslash N[v]$ son árboles.

- **Grafos r-partitos completos**. [76] Sea $K_{p_1,p_2,\ldots,p_r}$ un grafo r-partito completo con $r > 1$. Si M es el mayor número entero tal que $p_i \; mod \; M < \lceil p_i/M \rceil$ para todo $1 \leq i \leq r$ entonces

$$\chi_{eq}(K_{p_1,p_2,\ldots,p_r}) = \sum_{i=1}^{r} \left\lceil \frac{p_i}{M+1} \right\rceil.$$

- **Productos cartesianos de caminos de longitud par**. [36] Sean p, q números impares y sea $G_{pq} = (V, E)$ el grafo tal que $V = \{v_{ij} : i \in \{1, \ldots, p\} \wedge j \in \{1, \ldots, q\}\}$ y $E = \{(v_{ij}, v_{i+1j}) : i \in \{1, \ldots, p-1\} \wedge j \in \{1, \ldots, q\}\} \cup \{(v_{ij}, v_{ij+1}) : i \in \{1, \ldots, p\} \wedge j \in \{1, \ldots, q-1\}\}$. Entonces $\chi_{eq}(G_{pq}) = 2$.

- **Grafos con vértices universales**. Si G presenta un vértice universal u entonces, en un coloreo equitativo, dicho vértice forma una clase de color. Entonces las clases de color no pueden superar el tamaño 2 y, por lo tanto, el número cromático equitativo de G puede calcularse a partir de conocer un máximo matching en el complemento de G. Más precisamente, $\chi_{eq}(G) = n - \nu(\overline{G - u})$.

- **Grafos con número de estabilidad 2**. Para aquellos grafos G tales que $\alpha(G) = 2$, las clases de color sólo tienen tamaño 1 y 2. Luego, el número cromático puede calcularse conociendo un máximo matching en el complemento de G: $\chi_{eq}(G) = n - \nu(\overline{G})$.

Teniendo en cuenta las propiedades mencionadas, en este tesis abordamos el PCEG sobre grafos G sin vértices universales y con $2 \leq \omega(G) \leq n-2$. Por lo tanto, también verifican $2 \leq \chi_{eq}(G) \leq n-2$.

2.3. Cotas para el número cromático equitativo

Claramente, como todo coloreo equitativo de G es un coloreo de G en el sentido tradicional, sabemos que $\chi_{eq}(G) \geq \chi(G)$. De esta manera todas las cotas inferiores conocidas para $\chi(G)$ son a la vez cotas inferiores de $\chi_{eq}(G)$. Por ejemplo, $\omega(G)$ es una cota inferior de $\chi(G)$ y $\chi_{eq}(G)$. Sin embargo, calcular $\omega(G)$ es tan difícil como calcular $\chi_{eq}(G)$ [48]. Una cota inferior más débil que $\omega(G)$ y que puede obtenerse en tiempo polinomial es el cardinal de cualquier clique maximal de G.

Al final del capítulo presentamos el algoritmo FINDCLIQUE que permite obtener una clique maximal de un grafo G. Para poder obtener una cota más ajustada, podemos generar varias cliques maximales de G y luego quedarnos con la de mayor cardinal. Este es el propósito del procedimiento BESTCLIQUE que también presentamos al final del capítulo.

Existen casos en donde $\chi(G)$ (y por lo tanto cualquier cota inferior de éste) puede llegar a ser muy pobre para acotar a $\chi_{eq}(G)$ como ocurre con los grafos estrella $K_{1,m}$ donde se verifica [65]:

$$\chi_{eq}(K_{1,m}) - \chi(K_{1,m}) = (\lceil m/2 \rceil + 1) - 2 = \lceil m/2 \rceil - 1.$$

Esto justifica la búsqueda de cotas inferiores para $\chi_{eq}(G)$ que no sean válidas para $\chi(G)$.

Es claro que, en todo coloreo, el tamaño de la clase de color a la que pertenece un vertice v está acotado superiormente por $\alpha(G - N[v]) + 1$. Además, en un $\chi_{eq}(G)$-eqcol, ese tamaño también está acotado inferiormente por $\lfloor n/\chi_{eq}(G) \rfloor$, por lo cual tenemos que

$$\left\lfloor \frac{n}{\chi_{eq}(G)} \right\rfloor \leq \alpha(V \backslash N[v]) + 1,$$

pudiéndose concluir:

Teorema 2.3.1. *[57] Sea G un grafo de n vértices, entonces* $\chi_{eq}(G) \geq \text{máx}\left\{ \left\lceil \frac{n+1}{\alpha(V \backslash N[v]) + 2} \right\rceil : v \in V \right\}$.

Esta cota es ajustada en el caso de los grafos $K_{1,m}$ ya que coincide con el número cromático equitativo de ellos. Además, es evidente que no es cota inferior del número cromático clásico para estos mismos grafos.

La cota dada por el teorema anterior requiere de conocer el número de estabilidad de $G[V \backslash N[v]]$ para todo $v \in V$. Similarmente a lo que sucede con la cota provista por $\omega(G)$, computar estos parámetros puede ser tan difícil como computar χ_{eq} mismo [48].

Es sabido que, para cualquier grafo G, si $P = \{Q_1, Q_2, \ldots, Q_r\}$ es una partición en cliques de G de su conjunto de vértices, entonces $\alpha(G) \leq r$ [77]. Por lo tanto, una cota inferior de χ_{eq} más débil que la provista por el Teorema 2.3.1 se obtiene reemplazando, para cada $v \in V$, $\alpha(V \backslash N[v])$ por el cardinal de una partición en cliques del conjunto de vértices de $G[V \backslash N[v]]$. Particiones del conjunto de vértices de un grafo en cliques pueden ser halladas en tiempo polinomial con el procedimiento CLIQUEPART descripto al final del capítulo.

En la Sección 3.3 analizamos empíricamente la bondad de las cotas inferiores generadas por los algoritmos propuestos.

En lo que se refiere a cotas superiores es bien sabido que, en todo grafo G, $\chi(G) \leq \Delta(G)+1$. Para el caso de coloreos equitativos, Hajnal y Szemerédi [41] probaron que $\chi^*_{eq}(G) \leq \Delta(G)+1$ y, por lo tanto, $\Delta(G)+1$ es también una cota superior de $\chi_{eq}(G)$. Más aún, en 2008 se prueba que, dado $r \geq \Delta(G)$, existe un algoritmo de complejidad $O(n^5)$ que construye un $(r+1)$-eqcol de G [49]. Este resultado fue mejorado en 2010, pues se propone un algoritmo de complejidad $O(rn^2)$ para realizar la misma tarea [51].

El Teorema de Brooks [19] afirma que, para todo grafo G conexo que no es completo ni ciclo de longitud impar, $\chi(G) \leq \Delta(G)$. Este hecho motivó a Meyer [65] a conjeturar que el mismo resultado es válido para coloreo equitativo. Esto es decir que para todo grafo G conexo que no es completo ni ciclo de longitud impar, $\chi_{eq}(G) \leq \Delta(G)$. La conjetura ha sido probada sólo en determinadas familias de grafos como los grafos bipartitos conexos [58], grafos de intervalos [26], grafos árboles [57] y grafos *split* [25], e intentar demostrarla para el caso general es el tópico más activo en lo que a coloreo equitativo se refiere. Para mayores detalles, se puede consultar en [56].

Recientemente en [50] se prueba el siguiente resultado que generaliza el teorema de Hajnal y Szemerédi:

Teorema 2.3.2. *[50] Sea G un grafo y r un número. Si para toda arista $(u,v) \in E$, $\delta(u)+\delta(v) \leq 2r+1$, entonces G admite un $(r+1)$-eqcol.*

A diferencia de lo que sucede con los $(\Delta(G)+1)$-eqcols, no se conoce un algoritmo polinomial que construya el coloreo equitativo cuya existencia se garantiza en el Teorema 2.3.2 [51]. Sin embargo, de este teorema, es inmediato obtener la siguiente cota superior para χ_{eq}:

$$\chi_{eq}(G) \leq \left\lceil \frac{\max\{\delta(u)+\delta(v) : (u,v) \in E\}-1}{2} \right\rceil + 1. \tag{2.1}$$

Es fácil comprobar que la cota dada en (2.1) siempre es al menos tan ajustada como $\Delta(G)+1$.

Estas cotas superiores tienen la ventaja de que pueden ser calculadas en tiempo lineal. No obstante, en general suelen ser muy débiles.

Claramente, la cantidad de colores usados por cualquier coloreo equitativo de G es naturalmente una cota superior de $\chi_{eq}(G)$. Por esta razón, es conveniente contar con algoritmos que generen *buenos* coloreos equitativos en un tiempo razonable. Estos se denominan *algoritmos heurísticos* para el PCEG y el próximo capítulo está dedicado a los mismos.

A continuación presentamos un pseudocódigo de los algoritmos FINDCLIQUE, BESTCLIQUE y CLIQUE-PART mencionados anteriormente.

Algoritmo 1: FINDCLIQUE

```
Data: grafo G
Result: Q = clique maximal de G
1  begin
2      v ← vértice de mayor grado en G
3      Q ← {v}
4      candidatos ← N(v)
5      while candidatos ≠ ∅ do
6          v ← vértice de mayor grado en candidatos
7          Q ← Q ∪ {v}
8          candidatos ← candidatos ∩ N(v)
9      end
10 end
```

Algoritmo 2: BESTCLIQUE

```
Data: grafo G
Result: Q = clique maximal de G
1  begin
2      V ← vértices de G
3      Q ← ∅
4      while |V| ≥ |Q| do
5          Q' ← clique obtenida con FINDCLIQUE sobre el grafo G[V]
6          v ← vértice de mayor grado en Q'
7          V ← V\{v}
8          si Q' no es maximal en G, agregar vértices a Q' hasta que sea maximal
9          si |Q'| > |Q|, actualizar Q ← Q'
10     end
11 end
```

Algoritmo 3: CLIQUEPART

```
Data: grafo G
Result: part = cardinal de una partición en cliques de G
1  begin
2      V ← vértices de G
3      part ← 0
4      while V ≠ ∅ do
5          Q ← clique obtenida con FINDCLIQUE sobre el grafo G[V]
6          V ← V\Q
7          part ← part + 1
8      end
9  end
```

Capítulo 3

Algoritmos heurísticos para el Problema de Coloreo Equitativo

En este capítulo desarrollamos una búsqueda Tabú para obtener coloreos equitativos de un grafo y analizamos la bondad de las soluciones suministradas por la búsqueda Tabú en un conjunto de instancias de prueba.

3.1. Heurísticas y Metaheurísticas

Cuando lidiamos con instancias medianas y grandes de un problema NP-difícil, sabemos lo complejo (y muchas veces inviable) que resulta obtener una solución óptima del mismo en un tiempo razonable. En muchas aplicaciones es necesario contar rápidamente con una solución y la condición de optimalidad pierde relevancia frente al tiempo de cómputo. En este sentido, contar con herramientas que nos proporcionen soluciones factibles de buena calidad en tiempos breves tiene un gran valor práctico y, en estos casos, el desafío es facilitar la mejor solución posible en el tiempo preestablecido.

Más aún, en los casos donde la optimalidad es requerida, las *buenas* soluciones que seamos capaces de encontrar se utilizan como cotas del valor óptimo buscado que, generalmente, resultan efectivas para mejorar el rendimiento del algoritmo exacto que estemos utilizando. En particular, en los algoritmos Branch-and-Cut como el que desarrollaremos en esta tesis, cotas superiores e inferiores de calidad implican generalmente una importante disminución del número de nodos a explorar y, por ende, en el tiempo total de cómputo.

Los algoritmos que generan soluciones factibles no necesariamente óptimas se denominan *algoritmos heurísticos*, o simplemente *heurísticas*. En ciertos casos, es posible establecer teóricamente la calidad de la solución que genera un algoritmo heurístico y en esos casos hablamos de *algoritmos aproximados*. Ejemplos de algoritmos aproximados son conocidos para el problema de la cobertura de vértices (en inglés, *cardinality vertex cover*), el problema del árbol Steiner métrico (en inglés, *metric Steiner tree*) y el problema del viajante de comercio métrico (en inglés, *metric TSP*) y pueden encontrarse en [75]. En lo que respecta a coloreo equitativo, no encontramos algoritmos aproximados en la literatura.

Una técnica habitual es la de combinar un algoritmo heurístico de construcción de soluciones con un *algoritmo de mejora*. Estas combinaciones suelen denominarse en algunos casos *metaheurísticas*. Los algoritmos de mejora parten de una solución factible y exploran la posibilidad de obtener una mejor. Entre las técnicas de mejora más habituales están las llamadas *búsquedas locales*. Estas últimas responden al siguiente esquema.

Sea S el conjunto de soluciones factibles del problema, también conocido como *espacio de soluciones*. Asumiendo que se trate de un problema de minimización, sea $f : S \to \mathbb{R}$ la función objetivo que se desea minimizar. Entonces, para cada solución $s \in S$ se define el *vecindario* de s, $N(s)$, como un conjunto de

soluciones que resultan de aplicar alguna operación a s. Se pretende también que $N(s)$ satisfaga las siguientes propiedades [16]:

- No debe ser costoso computacionalmente calcular cada solución de $N(s)$.
- Dada dos soluciones $s, s' \in S$, debe existir un *camino* de soluciones $s_1 = s, s_2, \ldots, s_m = s'$ tal que $s_{i+1} \in N(s_i)$ para todo $i = 1, \ldots, m-1$.

Partiendo de una solución s_1, una *iteración* de la búsqueda local consiste en calcular una nueva solución $s_2 \in N(s_1)$ que verifique $f(s_2) < f(s_1)$ y $f(s_2) \leq f(s')$ para todo $s' \in N(s_1)$. Este procedimiento se repite hasta que no se pueda encontrar una solución mejor. En ese caso, el algoritmo habrá encontrado un *mínimo local* (respecto a las vecindades definidas).

En este tipo de algoritmos cabe la posibilidad de que el mínimo local hallado por la búsqueda local pueda encontrarse lejos del mínimo global. Sin embargo, existen búsquedas locales más sofisticadas que permiten lidiar con este inconveniente. Entre ellas se encuentran las búsquedas Tabú, los algoritmos de recocido simulado (*Simulated Annealing*), los algoritmos genéticos, los algoritmos basados en colonias de hormigas y los GRASP (*Greedy Randomized Adaptive Search Procedure*) [16]. Todos estos enfoques han sido utilizados para resolver el PCG [60], pero no se conocen trabajos de la literatura que ataquen al PCEG con estas técnicas.

La búsqueda Tabú fue introducida por Glover [38] y consiste en una búsqueda local con ciertas reglas adicionales que evitan que la búsqueda *cicle* o quede *atrapada* en un óptimo local. Las soluciones consideradas del vecindario de la solución actual son determinadas a través del uso de una memoria. Esta memoria es conocida como *lista tabú* y generalmente es un conjunto de reglas utilizadas para filtar entre soluciones que deben ser consideradas (en el vecindario de la solución actual) de otras que se deben prohibir y que reciben el nombre de *soluciones tabú*. En su forma más simple, una lista tabú es un conjunto de soluciones que fueron visitadas en el pasado reciente. Cada solución es acompañada de un valor que determina por cuántas iteraciones debe permanecer tabú. Una vez transcurrida esa cantidad de iteraciones, la solución se elimina de la lista tabú. Coloquialmente se le llama *tiempo de vida* de la solución a la cantidad de iteraciones que faltan para que la solución deje de ser tabú.

La cantidad de iteraciones que una solución permanece tabú es denominada *tabu tenure*. Este es un parámetro que mide el tiempo de vida de un elemento de la lista tabú y su elección impacta en la *diversificación* de algoritmo. Una búsqueda Tabú con tabu tenure muy bajo se comportará como una búsqueda local tradicional, en donde el proceso se estancará fácilmente en óptimos locales. Por el contrario, usar un tabu tenure muy elevado hará divagar al proceso de búsqueda por el espacio S sin converger a la solución óptima.

A continuación presentamos un esquema rudimentario de una búsqueda Tabú. Los pasos 2 a 5 conforman la iteración de la búsqueda:

1. *Inicialización.* Se parte de una solución inicial $s_1 \in S$ y una lista tabú L vacía.
   La mejor solución s^* conocida hasta el momento es s_1 por lo que se efectúa $s^* \leftarrow s_1$.
2. *Búsqueda.* Se calcula la solución $s_2 \in N(s_1) \backslash L$ que satisfaga $f(s_2) < f(s_1)$ y $f(s_2) \leq f(s')$ para todo $s' \in N(s_1) \backslash L$.
3. *Criterio de aspiración.* Si, durante el cómputo de $N(s_1)$, se encuentra una solución mejor que s^* entonces, sin importar si es o nó tabú, se asigna esta solución a s_2 y se actualiza s^*.
4. *Actualización de la lista tabú.* Con algún criterio determinado se generan un conjunto de soluciones tabú, de entre las cuales la solución s_1 tiene que estar incluída. Estas soluciones se incorporan a la lista L con un tiempo de vida equivalente al tabu tenure. Además se eliminan aquellas soluciones de L que tengan tiempo de vida igual a cero. Al tiempo de vida de las soluciones restantes de L se lo reduce en una unidad.

5. *Transición*. Se realiza la asignación $s_1 \leftarrow s_2$ y se vuelve al paso 2.

Esta técnica fue aplicada por Hertz y de Werra [44] para el PCG. Su implementación es conocida como TABUCOL y consiste en hallar un k-coloreo de un grafo a partir de un coloreo *infactible* de k colores, en el sentido de que pueden haber vértices adyacentes que compartan el mismo color. En realidad, TABUCOL ataca un problema de optimización diferente que tiene por conjunto de soluciones a particiones $(V_1, V_2, \ldots, V_k)$ del conjunto de vértices, y por función objetivo a la cantidad de aristas tales que sus extremos pertenecen al mismo conjunto V_i, llamadas *aristas conflictivas*. Claramente, si no hay aristas conflictivas entonces todos los conjuntos V_1, V_2, ..., V_k son estables y, por lo tanto, conforman un k-coloreo. La búsqueda Tabú finaliza cuando se alcanza un k-coloreo o cuando se verifica algún criterio de parada que, por lo general, es un límite a la cantidad de iteraciones o tiempo de CPU utlizado.

Una forma de usar TABUCOL para el PCG es obteniendo una cota superior k del número cromático y luego aplicar la búsqueda Tabú a una partición arbitraria $(V_1, V_2, \ldots, V_{k-1})$ del conjunto de vértices. En caso de que la búsqueda Tabú arroje un $(k-1)$-coloreo, se puede volver a repetir el procedimiento reduciendo k en una unidad. Cuando la búsqueda tabú finaliza sin poder obtener un coloreo factible, el procedimiento acaba.

Posteriormente, Galinier y Hao [37] mejoran este método considerando una componente dinámica en el tabu tenure haciendo la búsqueda más *adaptativa* al tamaño de los grafos. Se han propuesto otras modificaciones de TABUCOL (como, por ejemplo, la búsqueda *reactiva* [15]) pero la de Galinier y Hao ha sido la más competitiva.

Para el coloreo equitativo, sólo conocemos una mención de una búsqueda Tabú en [11] que no se profundiza.

En las próximas secciones presentamos un algoritmo para el PCEG que denominamos TABUEQCOL. Este algoritmo es una adaptación de la búsqueda Tabú dinámica propuesta en [37].

3.2. Descripción de TABUEQCOL

El algoritmo que presentaremos a continuación tiene como objetivo generar el mejor coloreo equitativo posible en un tiempo determinado aplicando una búsqueda Tabú reiteradamente.

Puesto que nos interesa generar coloreos que sean equitativos, el espacio de soluciones S de la búsqueda Tabú es, para un k dado, el conjunto de particiones $(V_1, V_2, \ldots, V_k)$ tales que $\lfloor n/k \rfloor \leq |V_i| \leq \lceil n/k \rceil$ para todo $1 \leq i \leq k$. Es decir que los conjuntos V_i satisfacen la restricción de equidad. Por eso diremos que la partición es *equitativa*. La función objetivo $f(s)$ es la misma que para TABUCOL, la cantidad de aristas conflictivas de la partición s.

TABUEQCOL comienza hallando un coloreo equitativo inicial de G mediante una heurística propuesta por Furmańczyk [36] que es llamada NAIVE y detallada en la Sección 3.2.1. Una vez conocido este coloreo equitativo inicial, el siguiente paso es intentar obtener un coloreo equitativo que utilice un color menos. Precísamente, si k es la cantidad de colores del coloreo equitativo inicial, el procedimiento GENINITSOL de la Sección 3.2.2 construye una partición equitativa $(V_1, V_2, \ldots, V_{k-1})$ del conjunto V y la búsqueda Tabú *mejora* aquella partición hasta obtener un $(k-1)$-eqcol. A su vez, este $(k-1)$-eqcol permite generar una nueva partición de $k-2$ conjuntos que luego se mejora hasta alcanzar un $(k-2)$-eqcol, y este proceso se vuelve a repetir hasta que se verifique uno de los 3 criterios de parada siguientes:

- Que la suma de las iteraciones ejecutadas en todas las búsqueda Tabú supere el parámetro $MAXITER$.
- Que el tiempo transcurrido total supere el parámetro $MAXTIME$.
- Que la cantidad de colores del coloreo equitativo actual sea igual a una cota inferior de $\chi_{eq}(G)$ calculada previamente.

En la práctica resultó adecuado fijar $MAXTIME$ en 30 segundos y $MAXITER$ en 50000 iteraciones.

Podría pensarse que, disponiendo de suficiente tiempo, TABUEQCOL sería siempre capaz de dar un coloreo óptimo de G. Pero esto no es así pues, aún considerando $MAXTIME = MAXITER = \infty$, se puede llegar al caso en que la búsqueda Tabú entre en un bucle infinito mientras intenta hallar un k-eqcol de G con k no admisible. Claramente, lo que ocurre en este caso es que toda partición s satisface $f(s) \geq 1$.

En las subsecciones siguientes detallamos los procedimientos involucrados en TABUEQCOL. Al final de la sección decribimos TABUEQCOL con pseudocódigo.

3.2.1. Heurística NAIVE

La heurística NAIVE consiste básicamente en producir un coloreo clásico del grafo con el algoritmo goloso SL-COLOR de Matula, Marble e Isaacson [46], para luego someterlo a un procedimiento de balanceo entre las clases de colores, hasta que se verifique la condición de equidad. La complejidad temporal de esta heurística es de $O(n^4)$ en el peor caso.

Al final de la sección se describen con más detalle los algoritmos SL-COLOR y NAIVE.

3.2.2. Generación de la partición equitativa inicial

En [15] se sugiere usar una heurística constructiva sencilla para producir la partición inicial con la que TABUCOL arranca. Siguiendo esta sugerencia, implementamos una heurística constructiva que llamamos GENINITSOL para generar una partición equitativa $(V_1, V_2, \ldots, V_{k-1})$ a partir de un k-eqcol, descripta a continuación.

Sea $(C_1, C_2, \ldots, C_k)$ un k-eqcol. Comenzamos copiando los vértices de las clases C_i con $1 \leq i \leq k-1$ a los conjuntos V_i y asignamos una etiqueta "$-$" o "$+$" a cada conjunto V_i según si el tamaño de la clase C_i es $\lfloor n/k \rfloor$ o $\lceil n/k \rceil$ respectivamente. El próximo paso es distribuir los vértices de la clase C_k según el siguiente criterio. Se elige un vértice v que pertenezca a la clique maximal inicial computada según lo visto en el Capítulo 2. En caso de que ningún vértice de C_k contenga vértices de la clique maximal, se elige el de mayor grado presente en C_k. Entonces se quita v de C_k y se agrega a un conjunto V_i etiquetado con "$-$". Luego se modifica la etiqueta de V_i a a "$+$". Si todos los conjuntos $V_1, \ldots, V_{k-1}$ tienen etiqueta "$+$" se los re-etiqueta a "$-$". Este proceso se repite hasta que no queden vértices en C_k por distribuir.

No es difícil de ver que la partición generada por GENINITSOL resulta equitativa.

3.2.3. Búsqueda Tabú

En las secciones siguientes explicamos los detalles de la búsqueda Tabú utilizada en TABUEQCOL.

Vecindario de una solución

Sean $s_1 = (V_1, V_2, \ldots, V_k)$ y $s_2 = (V'_1, V'_2, \ldots, V'_k)$ particiones equitativas. Consideramos dos esquemas para establecer si s_1 y s_2 son vecinas:

- *Intercambio de vértices de distintas clases*: Dados i, j tales que $1 \leq i \leq k$, $1 \leq j \leq k$ y $i \neq j$, y vértices $u \in V_i$, $v \in V_j$, se debe satisfacer que $V'_i = (V_i \backslash \{u\}) \cup \{v\}$, $V'_j = (V_j \backslash \{v\}) \cup \{u\}$, $V'_l = V_l$ para todo $l \in \{1, \ldots, k\} \backslash \{i, j\}$.

- *Movimiento de un vértice de una clase mayor a otra menor:* Dados i, j tales que $1 \leq i \leq k$, $1 \leq j \leq k$, $i \neq j$ y $|V_i| = |V_j| + 1$, y un vértice $v \in V_i$, se debe satisfacer que $V'_i = V_i \backslash \{v\}$, $V'_j = V_j \cup \{v\}$, $V'_l = V_l$ para todo $l \in \{1, \ldots, k\} \backslash \{i, j\}$.

El primer esquema se puede ver como una operación $I_{u,v} : S \to S$ y el segundo como una operación $M_{v,j} : S \to S$. En la primera se deben probar pares de vértices u, v donde al menos uno de ellos sea conflictivo, y en la segunda se deben probar vértices v que pertenezcan a un conjunto de tamaño $\lfloor n/k \rfloor + 1$, y conjuntos V_j de tamaño $\lfloor n/k \rfloor$. Observemos que la segunda operación sólo puede ser aplicada cuando k no divide a n.

Ambas operaciones producen particiones que satisfacen la restricción de equidad y tienen la característica de poder ser evaluadas ágilmente desde el punto de vista de complejidad temporal. Además el valor de $f(s_2)$ se calcula en tiempo lineal ya que, para la primer operación es

$$f(s_2) = f(s_1) + |\{uw \in E : w \in V_j\}| - |\{uw \in E : w \in V_i\}| + |\{vw \in E : w \in V_i\}| - |\{vw \in E : w \in V_j\}|,$$

mientras que para la segunda es

$$f(s_2) = f(s_1) + |\{vw \in E : w \in V_j\}| - |\{vw \in E : w \in V_i\}|.$$

Tabu tenure

En la implementación de TABUCOL se fija el tabu tenure en 7 (es decir que un elemento que ingresa a la lista L permanece allí por 7 iteraciones). Sin embargo, usar un tabu tenure estático no permite que el algoritmo se adapte bien a instancias de distintos tamaños. En [37] se propuso modificar el tabu tenure de TABUCOL de forma que dependa de alguna característica de la solución actual. Nosotros vamos a aplicar este esquema, pero con la salvedad de que emplearemos una fórmula *determinística* para calcular el tabu tenure, en vez de una fórmula basada en la elección de un número aleatorio distribuido uniformemente como es planteado en [37]. La razón de esta decisión es que nuestro algoritmo debe exhibir siempre el mismo comportamiento ante una misma entrada.

Dada la solución $s = (V_1, V_2, \ldots, V_k)$, para obtener el tabu tenure $\mathfrak{t}(s)$ asociado a ella, evaluamos

$$\mathfrak{t}(s) = \lfloor n(s) TENURE_\alpha + TENURE_\beta \rfloor$$

donde $TENURE_\alpha$ y $TENURE_\beta$ son parámetros que deben ser ajustados, y $n(s)$ es la cantidad de vértices conflictivos de s, i.e. extremos de aristas conflictivas.

En resumen, simplemente usamos como tabu tenure una función lineal de la cantidad de vértices conflictivos. Así, cuando trabajamos sobre un grafo de mayor tamaño, sus soluciones suelen estar lejos de ser un coloreo equitativo y hace que el tabu tenure se eleve. Esto genera una mayor diversificación del algoritmo. Nuestra experiencia nos sugiere fijar $TENURE_\alpha$ = 0.9 y $TENURE_\beta$ = 9.0, lo que resulta adecuado a nuestras necesidades.

Lista tabú

Una lista tabú no necesariamente tiene que contener a las soluciones prohibidan si no que, simplemente, puede contener características para considerar prohibida a una solución. Este es el caso de TABUCOL y TABUEQCOL.

La lista tabú consiste en ternas de la forma (v, j, t) que indican que un vértice v no puede pintarse con un color j durante las próximas t iteraciones.

Sean $s_1 = (V_1, V_2, \ldots, V_k)$ y $s_2 = (V'_1, V'_2, \ldots, V'_k)$ particiones equitativas tales que s_2 pertenece al vecindario de s_1. Si existe un vértice $v \in V'_j$ para el cual hay una entrada en la lista tabú (v, j, t), entonces la solución s_2 debe ser considerada tabú y descartada.

Si, en cambio, la solución s_2 no es tabú entonces se agrega una nueva terna a la lista tabú en función de qué operación generó la nueva solución. En el caso en que $s_2 = I_{u,v}(s_1)$ se debe incorporar a la lista el elemento $(u, i, \mathfrak{t}(s_1))$ donde i es el color de u en s_1, i.e. $u \in V_i$. En el caso en que $s_2 = M_{v,j}(s_1)$ se debe incorporar a la lista el elemento $(v, i, \mathfrak{t}(s_1))$ donde i es el color de v en s_1, i.e. $v \in V_i$.

A continuación presentamos los pseudocódigos correspondientes a los algoritmos SL-COLOR, NAIVE y TABUEQCOL mencionados en esta sección.

Algoritmo 4: SL-COLOR

Data: grafo G
Result: c = coloreo de G

begin
  inicializamos K como una lista vacía
  while $V \backslash K \neq \varnothing$ **do**
    $v \leftarrow$ vértice de grado mínimo en el subgrafo $G[V \backslash K]$
    agregamos v al final de K
  end
  sea $c : V \to \{1, \ldots, n\} \cup \{\mathsf{undefined}\}$
  inicializamos $c(v) = \mathsf{undefined}$ para todo $v \in V$
  while $K \neq \varnothing$ **do**
    $v \leftarrow$ último vértice de K
    quitar v de la lista K
    $j \leftarrow$ color mínimo que puede recibir v sin que exista un vecino con el mismo color en c
    $c(v) \leftarrow j$
  end
end

Algoritmo 5: NAIVE

Data: grafo G
Result: c = coloreo equitativo de G

begin
  $c \leftarrow$ coloreo de G dado por el algoritmo SL-COLOR
  while *c no es equitativo* **do**
    $\mathsf{colmin} \leftarrow$ color menos utilizado
    $\mathsf{colmax} \leftarrow$ color más utilizado
    buscar vértice $v_1 \in C_{\mathsf{colmin}}$ que pueda ser coloreado con colmax
    buscar vértice $v_2 \in C_{\mathsf{colmax}}$ que pueda ser coloreado con colmin
    if *es posible intercambiar los colores de v_1 y v_2* **then**
      $C_{\mathsf{colmin}} \leftarrow C_{\mathsf{colmin}} \cup \{v_2\} \backslash \{v_1\}$
      $C_{\mathsf{colmax}} \leftarrow C_{\mathsf{colmax}} \cup \{v_1\} \backslash \{v_2\}$
    else
      asignar un nuevo color a algún vértice de C_{colmax}
    end
  end
end

Algoritmo 6: TABUEQCOL

```
Data: grafo G
Result: coloreo = mejor coloreo equitativo encontrado
1  begin
2      cotainf ← cota inferior calculada con los procedimientos dados en Sección 2.3
3      coloreo ← coloreo equitativo generado por NAIVE
4      cotasup ← cantidad de colores de coloreo
5      if cotasup = cotainf then
6          el coloreo generado es óptimo, stop
7      end
8      iter ← 0
9      while tiempo transcurrido ≤ MAXTIME ∧ iter ≤ MAXITER do
10         a partir de coloreo obtener solución de cotasup − 1 colores con GENINITSOL
11         coloreo ← (cotasup − 1)-eqcol generado por la búsqueda Tabú
12         iter ← iter + iteraciones realizadas por la búsqueda Tabú
13         cotasup ← cotasup − 1
14         if cotasup = cotainf then
15             el coloreo generado es óptimo, stop
16         end
17     end
18 end
```

3.3. Resultados computacionales

Esta sección tiene como objetivo principal analizar la calidad de los coloreos obtenidos por TABUEQCOL. También examinaremos las cotas inferiores dadas por los procedimientes propuestos en la Sección 2.3.

Para efectuar las mediciones consideraremos dos conjuntos de instancias. El primero de ellos está compuesto de 25 grafos de 100 vértices generados aleatoriamente y el segundo contiene grafos Kneser [24] y parte de la librería COLORLIB de DIMACS [5]. Esta librería es una colección de instancias de diferentes fuentes que los investigadores suelen usar para realizar benchmarks de problemas de optimización que involucran grafos. En particular, estos grafos y los grafos Kneser fueron utilizados para evaluar la performance de un algoritmo Branch-and-Cut para el PCEG [11].

En los casos en donde evaluamos grafos generados al azar, los agrupamos según su *grado de densidad*, i.e. porcentaje de aristas de un grafo de n vértices respecto de la cantidad de aristas de un grafo completo K_n: $\frac{100|E|}{n(n-1)/2}$. En general las densidades de los grafos se fijan en 10%, 30%, 50%, 70% y 90%.

En el Cuadro 3.1 mostramos los resultados obtenidos por las heurísticas sobre las instancias generadas al azar. Las primeras dos columnas dan el número de la instancia y el grado de densidad. Las columnas 3 y 4 dan cotas inferiores de χ_{eq} proporcionadas por el procedimiento BESTCLIQUE y el que calcula una cota basada en el Teorema 2.3.1 (ambos descriptos en Sección 2.3). Las columnas 5, 6 y 7 dan cotas superiores proporcionadas por $\Delta(G)+1$, heurística NAIVE y TABUEQCOL en un lapso de no más de 30 segundos.

En lo que respecta a las cotas inferiores, BESTCLIQUE da mejores resultados en instancias de hasta 50% de densidad mientras que en las de mayor densidad la situación se revierte. La generación de ambas cotas se producen en menos de un segundo de tiempo, por lo que que es preferible ejecutar ambos procedimientos y retener la mejor cota en vez de elegir uno por sobre el otro.

		Cota inferior		Cota superior		
Instancia	% Densidad	BESTCLIQUE	Teorema 2.3.1	$\Delta(G)+1$	NAIVE	TABUEQCOL
1	10	4	3	18	7	5
2	10	4	3	18	6	5
3	10	3	3	19	5	5
4	10	4	3	17	7	5
5	10	3	3	20	6	5
6	30	6	5	38	13	10
7	30	6	5	47	14	10
8	30	5	5	43	15	10
9	30	6	5	41	20	10
10	30	6	6	42	16	10
11	50	9	8	63	32	15
12	50	9	9	65	29	16
13	50	8	8	61	21	15
14	50	9	10	65	29	16
15	50	10	8	60	22	16
16	70	14	17	80	36	23
17	70	13	15	77	35	22
18	70	13	15	80	37	22
19	70	14	17	86	36	23
20	70	12	15	79	36	22
21	90	27	34	96	53	41
22	90	28	34	96	55	39
23	90	28	34	98	52	42
24	90	30	34	95	55	39
25	90	27	34	97	55	40

Cuadro 3.1: Instancias aleatorias de 100 vértices

En los resultados referidos a las cotas superiores, es evidente que NAIVE siempre obtiene un coloreo equitativo inicial bastante mejor que si se usase alguno de los algoritmos para obtener $(\Delta(G)+1)$-eqcols (comentados en la Sección 2.3). La cantidad de colores usados en los coloreos generados por NAIVE son, a su vez, reducidos por TABUEQCOL hasta un 53% (instancia 11). Esta reducción es más significativa en los grafos de densidad media, lo que justifica aún más usar TABUEQCOL pues es sabido que en problemas de coloreo las instancias aleatorias de densidad media son las más difíciles de resolver y es donde se requieren cotas iniciales de mayor calidad.

En los Cuadros 3.2 y 3.3 mostramos las características de las instancias DIMACS y grafos Kneser, junto con los resultados obtenidos por las heurísticas. El formato de los cuadros son similares al del Cuadro 3.1 con la diferencia de que, en las primeras tres columnas, se reporta el nombre de la instancia, la cantidad de vértices y la cantidad de aristas. En los casos donde la mejor cota inferior y superior coinciden, ambas cotas se muestran en **negrita** dándonos a entender que alcanzamos la optimalidad.

Como puede observarse, de un total de 62 instancias evaluadas se alcanza la optimalidad en 11 de ellas. El óptimo es dado por NAIVE en 6 instancias mientras que en las otras 5 la optimalidad sólo es alcanzada

por la búsqueda Tabú. Por otro lado, la instancia `david` se resuelve por optimalidad gracias a la cota inferior dada por Teorema 2.3.1.

Existen también muchas instancias que, si bien no son resueltas, las mejores cotas están muy próximas entre sí. Las siguientes 21 instancias tienen una diferencia de una unidad entre su mejor cota inferior y su mejor cota superior: `miles750`, `queen6_6`, `queen8_8`, `1-FullIns_3`, `2-FullIns_3`, `3-FullIns_3`, `4-FullIns_3`, `5-FullIns_3`, `mug88_1`, `mug88_25`, `mug100_1`, `mug100_25`, `school1`, `2-Insertions_3`, `3-Insertions_3`, `DSJC125.1`, `le450_15a`, `le450_15b`, `will199GPIA`, `kneser7_3` y `kneser9_4`.

En resumen, del total de instancias evaluadas se resuelven por optimalidad un 18% de ellas y se logra una diferencia de una unidad entre las cotas en un 34% de ellas. Podemos concluir entonces que vale la pena invertir 30 segundos en ejecutar TABUEQCOL pues, junto con las heurísticas para calcular cotas inferiores de χ_{eq}, generan cotas de buena calidad y, eventualmente, resuelven la instancia sin necesidad de recurrir a un algoritmo exacto.

			Cota inferior		Cota superior		
Instancia	$\|V\|$	$\|E\|$	BESTCLIQUE	Teorema 2.3.1	$\Delta(G)+1$	NAIVE	TABUEQCOL
miles750	128	2113	30	11	65	33	31
miles1000	128	3216	40	17	87	47	43
miles1500	128	5198	69	43	107	74	73
zeroin.i.1	211	4100	49	3	112	51	51
zeroin.i.2	211	3541	30	4	141	51	50
zeroin.i.3	206	3540	30	4	141	49	48
queen6_6	36	290	6	5	20	10	7
queen7_7	49	476	**7**	6	25	12	**7**
queen8_8	64	728	8	8	28	18	9
queen8_12	96	1368	**12**	11	32	20	**12**
queen9_9	81	1056	9	8	33	15	11
queen10_10	100	1470	10	10	36	18	12
jean	80	254	**10**	3	37	**10**	10
anna	138	493	**11**	3	72	**11**	11
david	87	406	11	**30**	83	40	**30**
games120	120	638	**9**	5	14	**9**	9
homer	561	1628	**13**	2	100	**13**	13
huck	74	301	**11**	6	54	**11**	11
1-FullIns_3	30	100	3	3	12	7	4
2-FullIns_3	52	201	4	3	16	9	5
3-FullIns_3	80	346	5	3	20	7	6
4-FullIns_3	114	541	6	3	24	12	7
5-FullIns_3	154	792	7	3	28	9	8
1-FullIns_4	93	593	3	3	33	7	5
mug88_1	88	146	3	3	5	4	4
mug88_25	88	146	3	3	5	4	4
mug100_1	100	166	3	3	5	4	4
mug100_25	100	166	3	3	5	4	4
mulsol.i.1	197	3925	49	4	122	63	53
mulsol.i.2	188	3885	31	11	157	58	51
school1	385	19095	14	9	283	49	15
school1_nsh	352	14612	**14**	8	233	40	**14**

Cuadro 3.2: Instancias DIMACS (parte 1)

			Cota inferior		Cota superior		
Instancia	$\|V\|$	$\|E\|$	BESTCLIQUE	Teorema 2.3.1	$\Delta(G)+1$	NAIVE	TABUEQCOL
fpsol2.i.1	496	11654	65	3	253	85	74
fpsol2.i.2	451	8691	30	5	347	62	60
fpsol2.i.3	425	8688	30	7	347	80	73
1-Insertions_4	67	232	2	3	23	5	5
2-Insertions_3	37	72	2	3	10	4	4
3-Insertions_3	56	110	2	3	12	4	4
4-Insertions_3	79	156	2	2	14	4	4
DSJC125.1	125	736	4	3	24	8	5
DSJC125.5	125	3891	9	9	76	27	19
DSJC125.9	125	6961	30	42	121	66	47
DSJC250.1	250	3218	4	3	39	13	9
DSJC250.5	250	15668	10	11	148	65	34
DSJC250.9	250	27897	37	63	235	136	93
le450_25a	450	8260	**25**	5	129	26	**25**
le450_25b	450	8263	**25**	6	112	**25**	25
le450_15a	450	8168	15	5	100	18	16
le450_15b	450	8169	15	5	95	17	16
le450_5a	450	5714	5	3	43	12	7
le450_5b	450	5734	5	4	43	12	7
inithx.i.1	864	18707	54	3	503	70	69
inithx.i.2	645	13979	30	8	542	158	122
myciel4	23	71	2	3	12	5	5
myciel5	47	236	2	3	24	9	6
myciel6	95	755	2	3	48	11	7
flat300_20_0	300	21375	10	11	161	81	38
will199GPIA	701	6772	6	4	39	9	7
kneser7_2	21	105	3	3	11	8	6
kneser7_3	35	70	2	2	5	3	3
kneser9_4	126	315	2	2	6	4	3
kneser11_5	462	1386	2	2	7	4	4

Cuadro 3.3: Instancias DIMACS (parte 2) y grafos Kneser

Capítulo 4

Modelos de Programación Lineal Entera para el Problema de Coloreo Equitativo en Grafos

En este capítulo analizamos distintos modelos de Programación Lineal Entera para el PCEG con el objetivo de elegir uno de ellos como formulación inicial en nuestro algoritmo Branch-and-Cut. En las primeras secciones damos una breve introducción a la Programación Lineal Entera y presentamos la estructura general de un algoritmo Branch-and-Cut.

4.1. Programación Lineal Entera

Un *programa lineal* consiste en determinar una solución que maximiza o minimiza el valor de una función objetivo lineal en un dominio definido por un sistema finito de desigualdades lineales.

El problema de resolver un sistema de desigualdades lineales se remonta al siglo XIX con la aparición del método de eliminación de Fourier-Motzkin, pero la programación lineal fue mayormente desarrollada durante la Segunda Guerra Mundial para planificar los gastos y reducir costos, lo que la hizo una actividad secreta hasta 1947 [27]. Los fundadores de este campo fueron Leonid Kantorovich, un matemático ruso que desarrolló problemas de programación lineal en 1939, George B. Dantzig, que publicó el método Simplex en 1947, y John von Neumann, que propuso la teoría de dualidad en el mismo año. Sin embargo, no fue hasta 1979 que Leonid Khachiyan demostró que el problema de resolver programas lineales puede ser llevado a cabo en tiempo polinomial [40].

En muchas aplicaciones se requiere que la solucion óptima sea entera. El agregado de las restricciones de integralidad de las variables a un programa lineal, da lugar a un *programa lineal entero*. A diferencia de los primeros, no se conocen algoritmos que resuelvan estos últimos en tiempo polinomial. Más aún, se sabe que la Programación Lineal Entera es un problema NP-difícil.

En nuestro caso, trabajaremos con problemas de programación entera de minimización. O sea, un *programa entero* de la forma

$$\textit{Calcular} \quad z_{PE} = \min\{cx : Ax \leq b, x \in \mathbb{Z}^n\},$$

donde A es una matríz $m \times n$ y $b \in \mathbb{R}^n$. Si eliminamos las restricciones de integralidad, obtenemos un programa lineal de la forma

$$\textit{Calcular} \quad z_{PL} = \min\{cx : Ax \leq b, x \in \mathbb{R}^n\},$$

al cual denomimamos *relajación lineal* de nuestro programa entero.

Claramente, $z_{LP} \leq z_{PE}$ con lo cual la resolución de la relajación lineal de nuestro programa entero nos brinda una cota inferior del mismo, que puede ser obtenida en tiempo polinomial.

A pesar de que la Programación Lineal Entera sea NP-difícil, el importantísimo avance en el desarrollo de técnicas para su resolución y, en particular, los algoritmos Branch-and-Cut que explotan la estructura poliedral del modelo elegido, han consolidado a esta área como el mejor abordaje de una gran cantidad de problemas de Optimización Combinatoria y de Teoría de Grafos [66, 78].

En la próxima sección presentamos la estructura general de un algoritmo Branch-and-Cut.

4.2. Algoritmos Branch-and-Cut

Los algoritmos exactos para resolver problemas de Programación Lineal Entera son, en esencia, *algoritmos enumerativos*. Es decir que exploran las soluciones factibles del problema en base a un árbol de posibilidades, pero guiándose por reglas que permitan determinar zonas del árbol en donde no es necesario explorar, ya sea por que el problema presenta simetrías que pueden ser evitadas, o simplemente por que se tiene la certeza de que las soluciones óptimas no están en dicha zona.

Un algoritmo Branch-and-Cut es una clase particular de algoritmo enumerativo que conjuga principalmente dos métodos: *Branch-and-Bound* y *Planos de Corte*.

4.2.1. Branch-and-Bound

Este método consiste en la enumeración sistemática de todas las soluciones factibles del programa entero, aprovechando el hecho de que un número muy grande de soluciones infructuosas pueden ser descartadas mediante la estimación de cotas inferiores y superiores del valor que desea encontrarse. El esquema básico es el siguiente:

1. *Inicialización*. Se crea una lista L de problemas que, en un principio, sólo contiene la relajación lineal del programa entero y se incializa la cota superior global de z_{PE}: $\overline{z} \leftarrow \infty$ (o cualquier otra cota que se tenga disponible).

2. *Elección de un subproblema*. Si L está vacía o las cotas coinciden, el algoritmo termina. Si no, se elige un problema P de L y se lo elimina de la lista. Luego se resuelve ese problema lineal.

3. *Poda por infactibilidad.* Si el problema P resulta infactible, se vuelve al paso 2.

4. *Poda por cota*. Si se obtiene una solución factible x^* de P que satisface $\overline{z} \leq cx^*$, se vuelve al paso 2 (las soluciones de ese problema o las de los que desciendan de éste no son prometedoras).

5. *Poda por optimalidad*. Si la solución factible x^* que se obtiene de P es entera, x^* es también una solución del programa entero. Por lo tanto se actualiza la cota superior $\overline{z} \leftarrow cx^*$, y se vuelve al paso 2.

6. *Bifurcación*. En caso de que una de las componentes de la solución factible x^* no sea entera, se generan nuevos problemas lineales $P_1, P_2, \ldots, P_m$ con la siguiente propiedad: las soluciones enteras de cada problema lineal P_i conforman una partición de las soluciones enteras del problema P. Luego, los nuevos problemas se agregan al final de la lista L y se vuelve al paso 2.

La propiedad de que las soluciones enteras de cada P_i conforman una partición de las soluciones enteras de P asegura que no quede fuera del proceso ninguna solución entera, pero que tampoco se visite una solución entera más de una vez.

El esquema dado anteriormente resulta algo rudimentario, dado que no especificamos cómo elegir un subproblema particular de L en el paso 2, ni como generar subproblemas a partir de un problema dado en el paso 6. Tanto para la elección del subproblema como para la generación de nuevos subproblemas, podemos elegir entre diseñar criterios específicos para el problema de optimización en cuestión o utilizar criterios conocidos de propósito general.

Respecto a la elección de un subproblema, que se conoce como *Estrategia de Selección de Nodos*, se puede optar por las siguiente estrategias de uso general:

- **DFS** (*primero profundo*): Se elige el último problema de la lista.
- **BFS** (*primero a lo ancho*): Se elige el primer problema de la lista.
- **BestF** (*mejor cota*): Se elige el problema cuyo valor óptimo $c.x^*$ es mínimo entre los problemas de la lista.

El criterio usado para generar subproblemas a partir de un problema lineal P es conocido como *Estrategia de generación de nodos* o *Estrategia de Branching*. Una muy común es aplicar la dicotomía clásica. Es decir que, dada la solución óptima x^* del problema P, se toma una componente fraccionaria $x_i^* \notin \mathbb{Z}$ y se crean los subproblemas $\min\{cx : x \in P, x_i \leq \lfloor x_i^* \rfloor\}$ y $\min\{cx : x \in P, x_i \geq \lceil x_i^* \rceil\}$. Como puede verse, ambos subproblemas resultan lineales, sus soluciones enteras son disjuntas y la union de ellas conforman todas las soluciones enteras de P. En caso de que haya más de una componente fraccionaria, se debe decidir cuál elegir. Aquí también existen varios criterios al respecto. A modo de ejemplo mencionamos el más sencillo, que consiste en la elección de la *variable más fraccionaria*: aquella componente cuya parte fraccionaria esté más cercana a $\frac{1}{2}$.

En la terminología referida a algoritmos Branch-and-Bound es usual llamar *nodo* a cada subproblema generado, *nodo raíz* a la relajación lineal inicial del programa entero y *árbol de búsqueda* al árbol de posibilidades que explora el algoritmo.

4.2.2. Planos de corte

Un algoritmo de planos de corte es un algoritmo iterativo que tiene el propósito de ajustar la relajación lineal de un programa entero mediante la incorporación de desigualdades que resulten válidas para todas las soluciones enteras de aquel programa entero. En una iteración del algoritmo de planos de corte, se computa una solución óptima x^1 de un problema lineal $z^1 = \min\{cx : Ax \leq b, x \in \mathbb{R}\}$ y un conjunto de desigualdades $A'x \leq b'$ que suelen llamarse *cortes* por que cada desigualdad no es satisfecha por x^1. El resultado es otra relajación lineal de la forma $z^2 = \min\{cx : Ax \leq b, A'x \leq b', x \in \mathbb{R}\}$ de lo cual, obviamente, $z^2 \geq z^1$.

En la actualidad existen algoritmos de planos de corte que fueron diseñados para ser aplicados sobre cualquier programa entero. El primero surgió en la década de los 60 y es conocido como Algoritmo Fraccionario de Gomory [39]. A partir de aquí se propusieron otros, pero hay que destacar que todos ellos tienen una eficiencia limitada por el hecho de que no aprovechan la estructura inherente del problema en cuestión. Este motivo dió lugar a plantear algoritmos de planos de corte para problemas particulares, en donde se utilizan familias de desigualdades válidas que son determinadas a través de un *estudio poliedral* del problema. Ese estudio involucra caracterizar, al menos parcialmente, el poliedro que resulta de calcular la cápsula convexa de las soluciones enteras del modelo. En este contexto, la efectividad de una desigualdad válida guarda cierta relación con la dimensión de la cara que ella defina en el poliedro. De ahí surge la importancia de caracterizar desigualdades que definan caras con dimensión alta y, particularmente, facetas del poliedro (aquellas cuya dimensión es máxima).

4.2.3. Cut-and-Branch y Branch-and-Cut

A comienzo de los 80, se propuso un algoritmo para resolver programas enteros *grandes* (en donde sus variables eran binarias) [30] y tuvo bastante éxito. Este constaba de aplicar un algoritmo de planos de corte a la relajación lineal del problema y luego resolver la relajación más ajustada con un algoritmo Branch-and-Bound. Esta técnica es ahora conocida como *Cut-and-Branch*.

Ahora bien, si en cada nodo del árbol ejecutamos un algoritmo de planos de corte sobre la relajación lineal de dicho nodo, se pueden obtener cotas inferiores más ajustadas, lo que puede ocasionar que se incrementen los nodos que son podados por cota en el paso 4 del algoritmo Branch-and-Bound. Esta idea da origen a los algoritmos Branch-and-Cut. El esquema de un algoritmo Branch-and-Cut es similar al de Branch-and-Bound, pero el paso 6 debe ser reemplazado por los siguientes:

6. *Separar o Bifurcar*. En caso de que una de las componentes de la solución factible x^* no sea entera, debe decidirse si se buscarán planos de corte o se procederá a la bifurcación. En este último caso, se va al paso 8.

7. *Separación*. Identificar desigualdades válidas violadas por x^*. En caso de encontrarse, se agregan a P. Luego se resuelve P y se regresa al paso 3. Si no se encuentran desigualdades válidas, se continúa con el siguiente paso.

8. *Bifurcación*. Se generan nuevos problemas lineales $P_1, P_2, \ldots, P_m$ con la siguiente propiedad: las soluciones enteras de cada problema lineal P_i conforman una partición de las soluciones enteras del problema P. Luego, los nuevos problemas se agregan al final de la lista L y se vuelve al paso 2.

En esta descripción quedan nuevamente puntos sin especificar. En los pasos 6 y 7 se deben preestablecer criterios para decidir cuándo se buscan planos de cortes y en qué cantidad. Estos son factores a determinar que deben considerarse junto con los ya mencionados en la Sección 4.2.1. En la práctica, lograr un equilibrio entre todos ellos no resulta sencillo y, por lo general, depende del problema particular que se quiera resolver.

En general el proceso de separación mencionado en el paso 7 consiste de varios algoritmos de separación, cada uno de ellos asociado a una determinada familia de desigualdades válidas. Estas familias suelen surgir de un estudio poliedral del modelo de programación entera.

No hay un regla estricta que nos permita decidir cuáles son los mejores cortes de entre un conjunto de ellos, pero existen algunos criterios que suelen funcionar bien en la práctica:

- *Dimensión de la cara*. A mayor dimensión se espera un mejor rendimiento general en el algoritmo. De ahí que uno ponga énfasis en estudiar cuándo las desiguadades válidas definen facetas.

- *Violación de un corte*. Es la diferencia entre el lado izquierdo de la desigualdad evaluada en la solución fraccionaria y el lado derecho de la misma. A mayor diferencia se espera mayor eliminación de soluciones fraccionarias, y por ende, mayor crecimiento en la cota inferior cuando el corte es añadido. En general, las rutinas de separación intentan maximizar esta diferencia.

- *Ortogonalidad entre cortes*. La ortogonalidad se refiere al ángulo que forman dos hiperplanos entre sí y está relacionado directamente con el soporte de las desigualdades involucradas. Desigualdades con soporte similar suelen producir un comportamiento similar al ser agregadas como cortes y el rendimiento general decae. Además, en ciertos casos, dos desigualdades poco ortogonales pueden generar inestabilidad numérica. Las rutinas de separación que generen muchos cortes deben introducir un mecanismo adicional que evite agregar cortes que sean poco ortogonales entre ellos.

4.3. Modelos de Programación Lineal Entera para el PCEG

Dado que toda solución factible del PCEG es una solución factible de PCG que satisfaga las restricciones de equidad, es natural plantearse modelos como programas lineales enteros para el PCEG obtenidos a partir de un programa entero para el PCG con el agregado de las restricciones de equidad, modeladas como desigualdades lineales en las variables del programa original.

Existen muchos modelos para el PCG. En particular, nos interesa mencionar los propuestos en [62, 64], basados en el modelo de *asignación de colores a vértices* [6, 29] y el modelo de *representantes asimétricos* [21, 22] sobre los cuales existen antecedentes de su adaptación al PCEG. Otros modelos para el PCG que, según nuestro conocimiento, no han sido adapatados al PCEG pueden encontrarse en [20, 35, 42, 55, 61].

4.3.1. El modelo de representantes asimétricos y el PCEG

En [22] se propuso el *modelo de representantes* para resolver el PCG y consiste en elegir un vértice por cada color que utilice un coloreo. Ese vértice debe pertenecer a la clase de color que "representa". En este sentido, un vértice v que se coloree con un color j puede admitir dos estados únicamente: o bien v representa a j, o bien existe otro vértice que representa a j, de manera que no es importante aquí el conocimiento del valor de j sino del vértice que representa el color de v. Esta relación puede ser modelada como un problema de asignación.

Dado un grafo $G = (V,E)$ conexo, consideremos las siguientes variables binarias x_{uv} para todo $u,v \in V$:

$$x_{uv} = \begin{cases} 1 & \text{si el vértice } u \text{ representa el color del vértice } v, \\ 0 & \text{si no,} \end{cases}$$

Tenemos las siguientes restricciones asociadas a estas variables:

$$\sum_{u \in V}^{n} x_{uv} = 1, \qquad \forall\, v \in V \tag{4.1}$$

$$x_{uv} + x_{uw} \leq x_{uu}, \qquad \forall\, u \in V,\ (v,w) \in E \tag{4.2}$$

$$x_{uv} = 0, \qquad \forall\, u \in V,\ v \in N(u) \tag{4.3}$$

$$x_{uv} \in \{0,1\}, \qquad \forall\, u,v \in V$$

Las ecuaciones (4.1) aseguran que cada vértice admita un único representante, las inecuaciones (4.2) establecen que dos vértices adyacentes no pueden tener el mismo representante en común y las inecuaciones (4.3) afirman que un vértice no puede representar a ninguno de sus vecinos.

A partir de una solución factible de esta formulación se puede armar un coloreo cuya cantidad de colores es $\sum_{v \in V} x_{vv}$, simplemente asignando las distintas clases de colores a los vértices representados por un mismo vértice. Para resolver el PCG, minimizamos la función objetivo $\sum_{v \in V} x_{vv}$.

Posteriormente se presentó una modificación de este modelo con el nombre de *representantes asimétricos* [21]. Básicamente, lo que se propone es introducir las siguientes restricciones:

$$x_{uv} = 0, \qquad \forall\, u,v \in V,\ u > v \tag{4.4}$$

Estas restricciones fuerzan a que un vértice sólo pueda ser representado por un vértice cuyo número (o etiqueta) sea inferior. Así, el vértice que representa un color es justamente el vértice cuyo número es mínimo en dicha clase de color. Esto produce una reducción de la cantidad de soluciones simétricas. En la siguiente sección hablaremos sobre el concepto de simetría con mayor profundidad.

La formulación de representantes asimétricos fue modificada para que los coloreos intervinientes sean equitativos [11]. Consideremos el vector entero (x,w) que satisface las restricciones (4.1)-(4.4). Los autores introducen nuevas variables binarias y_i y z_{ui} donde $u \in V$ y $1 \leq i \leq n$ definidas como sigue:

$$y_i = \begin{cases} 1 & \text{si el cardinal de la clase de color de máximo cardinal es } i, \\ 0 & \text{si no,} \end{cases}$$

$$z_{ui} = x_{uu}\, y_i.$$

Con estas variables, las restricciones de equidad son modeladas a través de las siguientes restricciones lineales:

$$x_{uv} \leq x_{uu}, \qquad \forall\, u \in V,\ v \in I,\ u \neq v \tag{4.5}$$

$$x_{uu} + \sum_{v \in V} x_{uv} \leq \sum_{i=1}^{n} i\, z_{ui}, \qquad \forall\, u \in V \tag{4.6}$$

$$2x_{uu} + \sum_{v \in V} x_{uv} \geq \sum_{i=1}^{n} i\, z_{ui}, \qquad \forall\, u \in V \tag{4.7}$$

$$\sum_{i=1}^{n} y_i = 1, \tag{4.8}$$

$$z_{ui} \leq y_i, \qquad \forall\, u \in V,\ 1 \leq i \leq n \tag{4.9}$$

$$z_{ui} \leq x_{uu}, \qquad \forall\, u \in V,\ 1 \leq i \leq n \tag{4.10}$$

$$z_{ui} \geq y_i + x_{uu} - 1, \qquad \forall\, u \in V,\ 1 \leq i \leq n \tag{4.11}$$

$$y_i, z_{ui} \in \{0,1\}, \qquad \forall\, u \in V,\ 1 \leq i \leq n$$

Las restricciones (4.5), junto con las (4.2), indican que un vértice sólo puede ser representado por otro vértice si este último se representa a sí mismo. En las restricciones (4.6) y (4.7), la clase de color representada por u tiene cardinal entre i e $i-1$ sólo si $z_{ui} = 1$. Las restricciones (4.8)-(4.11) aseguran que y_i esté bien definida y que $z_{ui} = x_{uu}\, y_i$.

En [11, 12] se presenta un algoritmo Branch-and-Cut basado en este modelo. Si bien este algoritmo incorpora componentes que aprovechan características propias del PCEG (como una búsqueda Tabú y una estrategia de generación de nodos específica), en lo que respecta a la estructura poliedral del modelo sólo utiliza resultados del estudio realizado en [21] para el PCG.

Para nuestro algoritmo Branch-and-Cut utilizaremos un modelo de programación lineal entera para el PCEG, basado en los modelos de asignación de colores a vértices para el PCG propuestos en [62, 64].

4.3.2. Modelo de asignación de colores a vértices

Dado un grafo $G = (V,E)$ conexo de n vértices, los coloreos de G pueden modelarse a través de variables binarias x_{vj} y w_j para todo $v \in V$ y $1 \leq j \leq n$, definidas de la siguiente manera:

$$x_{vj} = \begin{cases} 1 & \text{si el vértice } v \text{ es coloreado con el color } j, \\ 0 & \text{si no,} \end{cases}$$

$$w_j = \begin{cases} 1 & \text{si el color } j \text{ es usado por algún vértice,} \\ 0 & \text{si no.} \end{cases}$$

Entonces toda solución del siguiente sistema representa un coloreo de G:

$$\sum_{j=1}^{n} x_{vj} = 1, \qquad \forall\, v \in V \tag{4.12}$$

$$x_{uj} + x_{vj} \leq w_j, \qquad \forall\, uv \in E,\ 1 \leq j \leq n \tag{4.13}$$

$$x_{vj}, w_j \in \{0,1\}, \qquad \forall\, v \in V,\ 1 \leq j \leq n$$

Las ecuaciones (4.12) reciben el nombre de *restricciones de asignación* y aseguran que cada vértice sea coloreado exactamente con un color. Las inecuaciones (4.13) reciben el nombre de *restricciones de adyacencia* y evitan que dos vértices adyacentes compartan el mismo color. Dado que el grafo G es conexo, las inecuaciones (4.13) además impiden que un vértice utilice el color j si $w_j = 0$ (pues cada vértice v es adyacente a algún otro y, por lo tanto, cada variable x_{vj} figura al menos una vez en estas restricciones).

El PCG puede resolverse, por lo tanto, minimizando la función objetivo $\sum_{j=1}^{n} w_j$ sobre el conjunto de soluciones enteras del sistema. Lo llamamos *modelo de asignación de colores a vértices*.

Ejemplo 4.3.1. *Consideremos un grafo G de 3 vértices y conjunto de aristas $E = \{(1,2),(2,3)\}$. En el modelo de asignación de colores a vértices, la cantidad de soluciones enteras que representan coloreos asciende a 18, de las cuales las que representan 2-coloreos son 12 (las variables que no son mencionadas tienen el valor cero):*

$$\begin{aligned}
&x_{11} = x_{22} = x_{31} = w_1 = w_2 = 1, && x_{11} = x_{23} = x_{31} = w_1 = w_3 = 1,\\
&x_{12} = x_{21} = x_{32} = w_1 = w_2 = 1, && x_{12} = x_{23} = x_{32} = w_2 = w_3 = 1,\\
&x_{13} = x_{21} = x_{33} = w_1 = w_3 = 1, && x_{13} = x_{22} = x_{33} = w_2 = w_3 = 1,\\
&x_{11} = x_{22} = x_{31} = w_1 = w_2 = w_3 = 1, && x_{11} = x_{23} = x_{31} = w_1 = w_2 = w_3 = 1,\\
&x_{12} = x_{21} = x_{32} = w_1 = w_2 = w_3 = 1, && x_{12} = x_{23} = x_{32} = w_1 = w_2 = w_3 = 1,\\
&x_{13} = x_{21} = x_{33} = w_1 = w_2 = w_3 = 1, && x_{13} = x_{22} = x_{33} = w_1 = w_2 = w_3 = 1,
\end{aligned}$$

y para el caso de los 3-coloreos, la cantidad de soluciones enteras que los representan es seis:

$$\begin{aligned}
&x_{11} = x_{22} = x_{33} = w_1 = w_2 = w_3 = 1, && x_{12} = x_{23} = x_{31} = w_1 = w_2 = w_3 = 1,\\
&x_{11} = x_{23} = x_{32} = w_1 = w_2 = w_3 = 1, && x_{13} = x_{21} = x_{32} = w_1 = w_2 = w_3 = 1,\\
&x_{12} = x_{21} = x_{33} = w_1 = w_2 = w_3 = 1, && x_{13} = x_{22} = x_{31} = w_1 = w_2 = w_3 = 1.
\end{aligned}$$

Cuando modelamos una aplicación real del PCEG, varias soluciones del modelo son traducidas al problema real como una misma solución debido a que los colores se consideran indistinguibles entre sí. En el ejemplo anterior se puede ver que los 3-coloreos guardan la misma esencia: sin importar qué color se asigne a cada vértice, se verifica que todos los colores asignados son distintos entre sí.

Entonces sería conveniente que pudiésemos eliminar, aunque sea, parte de estas soluciones para acelerar el proceso de búsqueda de la solución óptima.

Las soluciones que tienen el mismo valor de la función objetivo se llaman *soluciones simétricas*. En muchos casos estas soluciones pueden ser generadas a partir de otras soluciones mediante la permutación de sus componentes. Por lo general, la existencia de soluciones simétricas suele producir un deterioro en la performance de los algoritmos de resolución.

Un concepto que se empezó a usar hace unos años atrás es el de *romper simetrías* [28] y consiste justamente en construir modelos que eliminen la mayor cantidad de soluciones asegurándose de que, para cada valor posible de la función objetivo, exista al menos una solución factible del mismo.

Este concepto ya ha sido aplicado al PCG con éxito [64]. Aquí, los autores plantean varios enfoques para eliminar soluciones simétricas en el modelo de asignación de colores a vértices, que detallamos a continuación:

1. En el modelo de asignación de colores a vértices existen soluciones enteras en donde, para un color j cualquiera, $w_j = 1$ aunque el color j no sea utilizado por ningún vértice. Estas soluciones pueden eliminarse incorporando las restricciones:

$$\sum_{v \in V} x_{vj} \geq w_j, \qquad \forall\, 1 \leq j \leq n \tag{4.14}$$

   En el Ejemplo 4.3.1, el número total de soluciones enteras disminuye a 12, puesto que los 2-coloreos se representan sólo con:
   $x_{11} = x_{22} = x_{31} = w_1 = w_2 = 1$, $x_{11} = x_{23} = x_{31} = w_1 = w_3 = 1$,
   $x_{12} = x_{21} = x_{32} = w_1 = w_2 = 1$, $x_{12} = x_{23} = x_{32} = w_2 = w_3 = 1$,
   $x_{13} = x_{21} = x_{33} = w_1 = w_3 = 1$, $x_{13} = x_{22} = x_{33} = w_2 = w_3 = 1$.

2. También vemos que hay soluciones factibles que representan un coloreo en donde un color $j+1$ puede ser usado aún cuando el color j no lo es. Para evitar estas soluciones adicionales, podemos exigir que el color $j+1$ no pueda ser usado si los colores $1, \ldots, j$ no fueron usados en algún vértice. Para lograr esto, es suficiente con agregar las restricciones:

$$w_{j+1} \leq w_j, \qquad \forall\, 1 \leq j \leq n-1 \tag{4.15}$$

   Considerando las restricciones (4.14) y (4.15), la cantidad total de soluciones enteras del Ejemplo 4.3.1 se reduce a 8 debido a que los 2-coloreos se representan sólo con:
   $x_{11} = x_{22} = x_{31} = w_1 = w_2 = 1$, $x_{12} = x_{21} = x_{32} = w_1 = w_2 = 1$.

3. En la formulación anterior aún permanecen determinadas soluciones simétricas que pueden ser eliminadas si exigimos que el número de vértices coloreados con un color j sea mayor o igual al número de vértices coloreados con $j+1$. Entonces basta con agregar las siguientes restricciones:

$$\sum_{v \in V} x_{vj} \geq \sum_{v \in V} x_{vj+1}, \qquad \forall\, 1 \leq j \leq n-1 \tag{4.16}$$

   Esta vez, sólo uno de los 2-coloreos del Ejemplo 4.3.1 puede ser representado: $x_{11} = x_{22} = x_{31} = w_1 = w_2 = 1$, y por lo tanto el número total de soluciones enteras es de 7.

4. El modelo de asignación de colores a vértices no distingue los colores que le pueden ser asignados a conjuntos estables con igual cardinal. Para evitar estas soluciones, vamos a considerar los vértices de una clase de color y elegir aquel cuyo número (o etiqueta) sea el menor dentro de dicha clase. Las clases de color se pueden ordenar de acuerdo al número de estos vértices y, como el ordenamiento es único, permite eliminar en forma total las permutaciones de los colores: a la clase de color ubicada en la j-ésima posición (según este ordenamiento) se le asigna el color j. Para alcanzar este objetivo, agregamos las siguientes restricciones a la formulación:

$$x_{vj} = 0, \qquad \forall\, 1 \leq v < j \leq n \tag{4.17}$$

$$x_{vj} \leq \sum_{u=j-1}^{v-1} x_{uj-1}, \qquad \forall\, 2 \leq j < v \leq n \tag{4.18}$$

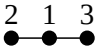
   La restricción (4.17) indica que a un vértice cuya etiqueta es v nunca se le puede asignar un color superior a v y la restricción (4.18) indica que a un vértice cuya etiqueta es v se le puede asignar el color j únicamente si algún vertice con etiqueta menor usa el color $j-1$. Un vértice que no cumpla esta condición está obligado a colorearse con $1, \ldots, j-1$.

Si consideramos las restricciones (4.14), (4.15), (4.17) y (4.18), sólo uno de los 2-coloreos del Ejemplo 4.3.1 puede ser representado con una solución: $x_{11} = x_{22} = x_{31} = w_1 = w_2 = 1$. Análogamente, sólo uno de los 3-coloreos del ejemplo puede ser representado: $x_{11} = x_{22} = x_{33} = w_1 = w_2 = w_3 = 1$. Por lo tanto, la cantidad total de soluciones enteras se reduce a 2.

La siguiente tabla da un resumen de las distintas formulaciones para el problema de coloreo clásico propuestas en [64], según las restricciones que la conforman:

Formulación	(4.12)	(4.13)	(4.14)	(4.15)	(4.16)	(4.17)	(4.18)
Estándar	•	•					
1	•	•	•				
2	•	•	•	•			
3	•	•	•		•		
4	•	•	•	•		•	•

Los criterios para romper simetrías involucrados en las formulaciones 3 y 4 no son compatibles entre sí, por lo que no es posible emplearlos de manera simultánea. Por ejemplo, consideremos el grafo de la Figura (i) que se diferencia del grafo del Ejemplo 4.3.1 en el etiquetado de sus vértices.

(i) 

(ii) 

(iii) 

En (ii) y (iii) se muestran los 2-coloreos posibles, teniendo en cuenta las restricciones (4.14) y (4.15). El coloreo dado en (iii) no tiene asociada una solución entera factible en la formulación 3 mientras que ocurre lo mismo con el coloreo de (ii) y la formulación 4. Si mezclásemos ambos criterios, no habría ninguna solución factible con 2 colores en la formulación resultante.

En [64] se analizaron las formulaciones 2, 3 y 4 antes mencionadas a los efectos de la implementacion de un Branch-and-Cut para el PCG. Nosotros analizaremos en forma similar las adaptaciones de estos modelos al PCEG.

4.3.3. Modelos para el Problema de Coloreo Equitativo

En esta sección utilizamos el modelo de asignación de colores a vértices que identifica cada coloreo con un vector entero (x, w) que satisface las restricciones (4.12) y (4.13).

Observemos que si G es un grafo conexo, la restricciones (4.13) implican que si un color j no es usado ($w_j = 0$) ningún vértice puede ser coloreado con ese color ($x_{vj} = 0$ para todo $v \in V$). Sin embargo, en caso de que G tenga un vértice aislado u, las restricciones (4.13) no evitan una solución con $w_j = 0$ y $x_{uj} = 1$. Esto no es tenido en cuenta en los modelos del PCG ya que para resolver este problema basta contemplar cada una de las componentes conexas de G. Sin embargo, como se ha visto en la Sección 2.1, esto no sucede cuando trabajamos con el PCEG. Por esta razón, para cualquier modelo del PCEG, debemos tener en cuenta restricciones adicionales que eviten este inconveniente, como las siguientes:

$$x_{vj} \leq w_j, \qquad \forall\, v \in I,\ 1 \leq j \leq n \tag{4.19}$$

donde $I = \{v \in V : \delta(v) = 0\}$ (I es el conjunto de vértices aislados de G).

Por otra parte recordemos que, para que un coloreo sea equitativo, debe satisfacer la restricción de equidad. Esto es, la diferencia entre los tamaños de dos clases de color no puede superar la unidad. Si somos capaces de escribir esta condición con desigualdades lineales en términos de las variables del modelo para el PCG, entonces podemos definir un modelo para el PCEG.

Observemos que si (x, w) satisface las restricciones (4.12), (4.13) y (4.19), y j es un color que se usa en el coloreo que (x, w) representa, entonces $w_j = 1$ y el tamaño de la clase j puede ser calculado con $\sum_{v \in V} x_{vj}$.

Entonces, la restricción de equidad puede ser reformulada en términos de las variables de (x,w):

$$w_i = w_j = 1 \quad \Rightarrow \quad -1 \leq \sum_{v \in V} x_{vi} - \sum_{v \in V} x_{vj} \leq 1 \;\; \forall\, 1 \leq i < j \leq n.$$

Para modelar esta restricción como una inecuación lineal, los autores de [13] introducen una constante M de valor muy grande (*big M*, coloquialmente) y proponen las siguientes restricciones:

$$\sum_{v \in V} x_{vi} - \sum_{v \in V} x_{vj} \leq (2M+1) - Mw_i - Mw_j, \qquad \forall\, 1 \leq i < j \leq n \tag{4.20}$$

$$\sum_{v \in V} x_{vi} - \sum_{v \in V} x_{vj} \geq -(2M+1) + Mw_i + Mw_j, \qquad \forall\, 1 \leq i < j \leq n \tag{4.21}$$

Asumiendo que se usen al menos dos colores para pintar a G, el tamaño de una clase de color no puede superar a $\lceil n/2 \rceil$. Entonces un valor apropiado para M puede ser $\lceil n/2 \rceil - 2$, dado que el resultado de evaluar $(2M+1) - Mw_i - Mw_j$ es $2\lceil n/2 \rceil - 1$, $\lceil n/2 \rceil$ o 1 dependiendo de si cero, uno o dos de los colores i y j son empleados. En los primeros casos, ambas restricciones son redundantes mientras que en el último imponen que $||C_i| - |C_j|| \leq 1$.

La formulación compuesta por las restricciones (4.12), (4.13), (4.19), (4.20) y (4.21) nos permite resolver el PCEG. A esta formulación la llamaremos *SECF* (*Symmetric Equitable Coloring Formulation*).

Sin embargo, es sabido que las restricciones que utilizan una *big M* generalmente producen relajaciones lineales muy pobres. Además de esto, la formulación *SECF* no evita soluciones simétricas con el agregado de restricciones como las vistas en la sección anterior, las cuales han funcionado muy bien en el marco de un algoritmo Branch-and-Cut para el PCG [64].

De hecho, esta formulación no se ha utilizado para resolver el PCEG más que en instancias muy pequeñas [13] (no más de 35 vértices, con excepción de algunas instancias mayores de muy baja densidad como el grafo `kneser9_4`).

Si, junto a las restricciones (4.12) y (4.13), consideramos también las restricciones (4.15) (aquellas que impedían que se use el color $j+1$ si no se usa el color j), entonces una solución entera (x,w) representa un r-eqcol si y sólo si los colores $1,\ldots,r$ son los únicos que se utilizan, es decir que $w_1 = w_2 = \ldots = w_r = 1$ y $w_{r+1} = w_{r+2} = \ldots = w_n = 0$. Equivalentemente, (x,w) representa un r-eqcol si y sólo si

$$w_k - w_{k+1} = \begin{cases} 1, & \text{si } k = r, \\ 0, & \text{si } k \neq r. \end{cases}$$

Según la Observación 1.5.1, las restricciones de equidad para un r-eqcol también pueden ser formuladas como $\lfloor n/r \rfloor \leq |C_j| \leq \lceil n/r \rceil$ para todo $1 \leq j \leq r$. De esta manera, las restricciones de equidad pueden modelarse con las siguientes desigualdades lineales:

$$\sum_{v \in V} x_{vj} \geq \sum_{k=j}^{n} \left\lfloor \frac{n}{k} \right\rfloor \left(w_k - w_{k+1}\right), \qquad \forall\, 1 \leq j \leq n \tag{4.22}$$

$$\sum_{v \in V} x_{vj} \leq \sum_{k=j}^{n} \left\lceil \frac{n}{k} \right\rceil \left(w_k - w_{k+1}\right), \qquad \forall\, 1 \leq j \leq n \tag{4.23}$$

En efecto, para cualquier solución que represente un r-eqcol, el lado derecho de las restricciones (4.22) y (4.23) será $\lfloor n/r \rfloor$ y $\lceil n/r \rceil$ respectivamente para todo $j \leq r$, mientras que para $j > r$ el lado derecho de ambas será cero. Para facilitar la escritura de estas restricciones se utiliza una variable w_{n+1} no definida, que se considera siempre con valor cero.

A la formulación compuesta por las restricciones (4.12), (4.13), (4.15), (4.19), (4.22) y (4.23) la llamaremos *ECF* (*Equitable Coloring Formulation*). Esta formulación tiene asociada el siguiente poliedro:

Definición 4.3.2. *El poliedro $\mathcal{ECP}$ es la cápsula convexa de las soluciones enteras de la formulación ECF:*

$$\mathcal{ECP} = conv\{(x,w) \in \{0,1\}^{n^2+n} : (x,w) \text{ satisface } (4.12),(4.13),(4.15),(4.19),(4.22),(4.23)\}$$

Puede observarse que no agregamos las restricciones (4.14) en ECF. En realidad, estas últimas son dominadas por las restricciones (4.22).

Tal como fue planteado anteriormente, en la formulación ECF aún permanecen determinadas soluciones simétricas que pueden ser eliminadas con la incorporación de las restricciones (4.16). De esta manera, nos aseguramos que los tamaños de las clases decrezcan en relación a los colores y, de acuerdo a lo observado en 1.5.1, un k-coloreo será equitativo si $|C_j| = \lceil \frac{n-j+1}{k} \rceil$ para todo $1 \leq j \leq k$. Es decir que conocemos exactamente el tamaño de cada clase en un k-eqcol, y las restricciones de equidad pueden modelarse con ecuaciones lineales de la forma:

$$\sum_{v \in V} x_{vj} = \sum_{k=j}^{n} \left\lceil \frac{n-j+1}{k} \right\rceil (w_k - w_{k+1}), \qquad \forall\ 1 \leq j \leq n \tag{4.24}$$

La igualdad anterior para el caso $j = 1$ es combinación lineal de las demás y, por lo tanto, puede omitirse.

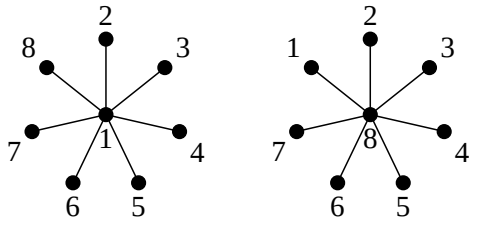
Llamaremos ECF_1 a la formulación que comprende (4.12), (4.13), (4.15), (4.19), y (4.24). Esta formulación tiene asociada el siguiente poliedro:

Definición 4.3.3. *El poliedro $\mathcal{ECP}_1$ es la cápsula convexa de las soluciones enteras de la formulación ECF_1:*

$$\mathcal{ECP}_1 = conv\{(x,w) \in \{0,1\}^{n^2+n} : (x,w) \text{ satisface } (4.12),(4.13),(4.15),(4.19),(4.24)\}$$

En el modelo ECF_1 aún subsisten coloreos simétricos debido a que las clases de color con el mismo cardinal son intercambiables entre sí, en el siguiente sentido. Para un k que divide a n, todas las clases de color de un k-eqcol tienen el mismo cardinal y ECF_1 no elimina más soluciones simétricas que ECF. En el caso en que k no divida a n, existen clases de color de tamaños $\lfloor n/k \rfloor$ y $\lfloor n/k \rfloor + 1$. Si bien ECF_1 elimina las soluciones simétricas que se desprenden de intercambiar los vértices entre una clase de tamaño $\lfloor n/k \rfloor$ con una de tamaño $\lfloor n/k \rfloor + 1$, aún mantiene las soluciones que surgen de intercambiar los vértices de dos clases de color del mismo tamaño.

Ya vimos que para evitar estas soluciones, agregamos las restricciones (4.17) y (4.18).



(a) 432 soluciones (b) 653 soluciones

Figura 4.1: Dos etiquetados para $K_{1,7}$

Es importante aclarar que el número de soluciones enteras de ECF_2 depende fuertemente de la forma en que etiquetamos los vértices. En la Figura 4.1 establecemos dos etiquetados diferentes del grafo $K_{1,7}$. El etiquetado de la izquierda da lugar a 432 soluciones diferentes mientras que el de la derecha produce 653.

Esto va a ser un factor a tener en cuenta cuando implementemos el modelo, puesto que distintos criterios de etiquetado pueden conducir a distintos comportamientos del algoritmo que resuelve el modelo.

Llamaremos ECF_2 a la formulación que consta de (4.12), (4.13), (4.15), (4.17), (4.18), (4.19), (4.22) y (4.23). Esta tiene asociada el siguiente poliedro.

Definición 4.3.4. *El poliedro $\mathcal{ECP}_2$ es la cápsula convexa de las soluciones enteras de la formulación ECF_2:*

$$\mathcal{ECP}_2 = conv\{(x,w) \in \{0,1\}^{n^2+n} : (x,w) \text{ satisface } (4.12),(4.13),\\ (4.15),(4.17),(4.18),(4.19),(4.22),(4.23)\}$$

4.4. Comparación de los modelos propuestos

A continuación analizamos los modelos de la Sección 4.3.3 en base a los siguientes parámetros: *cantidad de variables*, *cantidad de restricciones* y *cantidad de soluciones enteras*.

En primer lugar, podemos asumir que las variables x_{vj} con $v < j$ pueden ser eliminadas de la formulación ECF_2 debido a las restricciones (4.17), y estas últimas no se deben tener cuenta al contabilizar la cantidad de restricciones de ECF_2. También asumiremos que el grafo G en cuestión es conexo, pues de no serlo, simplemente debemos agregar las restricciones (4.19) a cualquiera de los modelos, que son $n|I|$ donde I es el conjunto de vértices aislados de G. Bajo estas consideraciones armamos el Cuadro 4.1.

	$SECF$	ECF	ECF_1	ECF_2
Variables	n^2+n	n^2+n	n^2+n	$\frac{1}{2}n^2+\frac{3}{2}n$
Restricciones	$n^2+\lvert E\rvert n$	$(\lvert E\rvert+4)n-1$	$(\lvert E\rvert+3)n-1$	$\frac{1}{2}n^2+(\lvert E\rvert+\frac{7}{2})n-1$

Cuadro 4.1: Cantidad de variables y restricciones

En lo que respecta a cantidad de soluciones enteras, no podemos dar un valor único para cada formulación pues éste depende de la estructura del grafo. Por ejemplo, para un grafo estrella $K_{1,m}$, la cantidad de soluciones enteras asociadas a la formulación ECF es

$$\sum_{k=\lceil\frac{m+2}{2}\rceil}^{m+1}\left(\prod_{r=m-k+2}^{k} r \times \prod_{r=1}^{m-k+1}\binom{m-2r+2}{2}\right),$$

mientras que para un grafo completo K_{m+1}, esta cantidad es $(m+1)!$.

En el cuadro 4.2 se pueden consultar los números de soluciones para cada modelo en grafos estrella $K_{1,m}$ con $3 \leq m \leq 7$. Para ECF_2 se muestran dos columnas. La columna (a) corresponde al caso en que el primer vértice es el centro de la estrella y la (b) corresponde al resto de los casos (véase la Figura 4.1 para el caso $m = 7$).

De los cuadros 4.1 y 4.2 podemos ver que la formulación $SECF$ es la que más restricciones tiene (de las cuales hay $n^2 - n$ restricciones que dependen de un valor *big M*) y el número de soluciones enteras crece más rápido que en los otros casos. Pruebas preliminares sobre instancias muy pequeñas ya descartan a $SECF$ como opción.

Los modelos restantes, sin embargo, deben ser analizados con mayor profundidad. Si bien ECF_2 tiene menos variables que el resto de los modelos y además muestra un número bajo de soluciones enteras, tiene más restricciones que ECF y ECF_1, lo que podría ocasionar que el cómputo de las relajaciones lineales consuma mayor tiempo. Por otro lado, ECF_1 podría presentar mejor performance que ECF porque involucra menos restricciones y rompe más simetrías. En el Capítulo 7 probaremos empíricamente los modelos ECF,

Grafo	$\chi_{eq}(G)$	*SECF*	*ECF*	ECF_1	ECF_2 (a)	ECF_2 (b)
$K_{1,3}$	3	96	42	30	4	5
$K_{1,4}$	3	1020	282	162	11	15
$K_{1,5}$	4	13320	2280	1020	33	48
$K_{1,6}$	4	206640	21600	7470	115	170
$K_{1,7}$	5	3709440	234360	61740	432	653

Cuadro 4.2: Cantidad de soluciones enteras para grafos estrella

ECF_1 y ECF_2 a fín de obtener aquel que pueda brindar el mayor beneficio a nuestro algoritmo Branch-and-Cut.

Respecto a los poliedros asociados a estos modelos, $\mathcal{ECP}$ es el que resulta ser más independiente de las particularidades del grafo, lo que lo vuelve más sencillo de analizar poliedralmente comparado con $\mathcal{ECP}_1$ y $\mathcal{ECP}_2$. En particular, la dimensión de $\mathcal{ECP}_2$ depende también del etiquetado de los vértices que hace aún más difícil su caracterización. Profundizaremos más sobre esta problemática en la Sección 5.5.

Por otro lado, cualquier desigualdad válida para $\mathcal{ECP}$ es también válida para los otros poliedros mencionados, ya que tanto $\mathcal{ECP}_1$ como $\mathcal{ECP}_2$ están contenidos en $\mathcal{ECP}$. En otras palabras, podemos estudiar el poliedro $\mathcal{ECP}$ independientemente del modelo que elijamos luego. Este va a ser el objetivo del Capítulo 5. Los resultados que obtengamos allí serán utilizados posteriormente en el Capítulo 6 para desarrollar las rutinas de separación vinculadas a cada familia que, junto a otros elementos, constituirán un algoritmo Branch-and-Cut que resuelva el PCEG.

Capítulo 5

Estudio del Poliedro de Coloreo Equitativo

En este capítulo presentaremos un detallado estudio poliedral de $\mathcal{ECP}$, el poliedro asociado a la formulación *ECF* vista en la Sección 4.3.3. Sobre el final haremos algunas observaciones acerca de los poliedros $\mathcal{ECP}_1$ y $\mathcal{ECP}_2$.

5.1. Dimensión del poliedro $\mathcal{ECP}$

De acuerdo a lo observado en la Sección 2.2, a partir de este momento trabajamos con grafos G sin vértices universales y tales que $2 \leq \chi_{eq}(G) \leq n-2$. En los casos restantes el problema PCEG se puede resolver en tiempo polinomial. Asumiremos además que G tiene al menos 5 vértices.

Para simplificar la notación vamos a escribir χ_{eq} en vez de $\chi_{eq}(G)$ y Δ en vez de $\Delta(G)$ siempre que G sea el grafo al que se hace referencia en el contexto.

Recordemos que I es el conjunto de vértices aislados del grafo G y $\mathcal{S}$ es el conjunto de enteros $k \geq \chi_{eq}(G)$ tales que G no admite un k-eqcol, según la Definición 2.1.1. Recordemos también que $\mathcal{ECP}$ es la cápsula convexa de las soluciones $(x,w) \in \{0,1\}^{n^2+n}$ que satisfacen:

$$\sum_{j=1}^{n} x_{vj} = 1, \qquad \forall\, v \in V \qquad (4.12)$$

$$x_{uj} + x_{vj} \leq w_j, \qquad \forall\, uv \in E,\ 1 \leq j \leq n \qquad (4.13)$$

$$x_{vj} \leq w_j, \qquad \forall\, v \in I,\ 1 \leq j \leq n \qquad (4.19)$$

$$w_{j+1} \leq w_j, \qquad \forall\, 1 \leq j \leq n-1 \qquad (4.15)$$

$$\sum_{v \in V} x_{vj} \geq \sum_{k=j}^{n} \left\lfloor \frac{n}{k} \right\rfloor \left(w_k - w_{k+1}\right), \qquad \forall\, 1 \leq j \leq n \qquad (4.22)$$

$$\sum_{v \in V} x_{vj} \leq \sum_{k=j}^{n} \left\lceil \frac{n}{k} \right\rceil \left(w_k - w_{k+1}\right), \qquad \forall\, 1 \leq j \leq n \qquad (4.23)$$

A partir de ahora, cada vez que nos refiramos a un coloreo equitativo, estaremos aludiendo a un coloreo asociado a un punto extremo del poliedro $\mathcal{ECP}$. Además, usaremos indistintamente las representaciones de los coloreos como vectores binarios, mapeos o clases de color. Si, por ejemplo, c es un k-eqcol tal que $c(1) = c(2) = 1$ entonces $\{1,2\} \subset C_1$ y $x_{11} = x_{21} = 1$, $x_{1j} = x_{2j} = 0$ para todo $2 \leq j \leq n$, $w_j = 1$ para todo $1 \leq j \leq k$ y $w_j = 0$ para todo $k+1 \leq j \leq n$.

Dos operadores entre coloreos serán de gran utilidad en este capítulo. El primero se basa en el hecho de que el intercambio de colores en un k-eqcol produce otro k-eqcol.

Definición 5.1.1. *Sea c un k-eqcol de G con clases de colores $C_1, \ldots, C_k$ y sea $L = (j_1, j_2, \ldots, j_r)$ una lista ordenada de distintos colores entre 1 y k. Se define como $swap_L(c)$ al k-eqcol cuyas clases de colores son $C'_1, \ldots, C'_k$ y satisfacen $C'_{j_t} = C_{j_{t+1}} \ \forall\, 1 \leq t \leq r-1, C'_{j_r} = C_{j_1}$ y $C'_i = C_i \ \forall\, i \notin L$.*

El otro operador nos permite construir un $(k+1)$-eqcol a partir de un k-eqcol con $\lceil n/2 \rceil \leq k \leq n-1$. Observemos que en todos los k-eqcols con $k \geq \lceil n/2 \rceil$, las clases de colores tienen uno o dos elementos, y asignándole el color $k+1$ a un vértice perteneciente a una clase de tamaño dos, obtenemos un $(k+1)$-eqcol. Formalmente:

Definición 5.1.2. *Sea c un k-eqcol de G con $\lceil n/2 \rceil \leq k \leq n-1$ y $v \neq v'$ tal que $c(v) = c(v')$. Se define $intro(c,v)$ como el $(k+1)$-eqcol c' que satisface $c'(v) = k+1$ y $c'(i) = c(i) \ \forall\, i \in V \backslash \{v\}$.*

En la Proposición 1 de [64] se presenta un procedimiento para generar coloreos afínmente independientes a partir de un n-coloreo. Al incorporar en éste la condición de equidad en los coloreos, obtenemos un nuevo procedimiento que nos permitirá generar coloreos equitativos afínmente independientes a partir de un n-eqcol.

Definición 5.1.3. *Sean u y u' vértices no adyacentes en G y sea c^1 un n-eqcol dado, en donde $c^1(u) = n$ y $c^1(u') = n-1$. Se define $PROC(c^1)$ como el conjunto de coloreos que incluye a c^1 y a los siguientes:*

- $c^2_{j,j'} = swap_{n,j,j'}(c^1)$ *para cada* $j, j' \in \{1, \ldots, n-1\}$ *tal que* $j \neq j'$. *De esta manera, construimos* $(n-1)(n-2)$ *coloreos.*
- $c^3_j = swap_{n,j}(c^1)$ *para cada* $j \in \{1, \ldots, n-1\}$. *Así construimos* $n-1$ *coloreos más.*
- *El* $(n-1)$*-eqcol* c^4 *tal que* $c^4(u) = n-1$ *y* $c^4(i) = c^1(i) \ \forall\, i \in V \backslash \{u\}$.
- $c^5_j = swap_{n-1,j}(c^4)$ *para cada* $j \in \{1, \ldots, n-2\}$. *Aquí tenemos* $n-2$ *coloreos más.*
- *Un k-eqcol arbitrario de G c^6_k para cada $k \in \{\chi_{eq}, \ldots, n-2\} \backslash \mathscr{S}$. Obtenemos $n - \chi_{eq} - |\mathscr{S}| - 1$ coloreos más.*

Lema 5.1.4. *Sea c^1 un n-eqcol de G tal que los vértices de colores n y $n-1$ no son adyacentes. Entonces $PROC(c^1)$ es un conjunto de $n^2 - \chi_{eq} - |\mathscr{S}|$ coloreos equitativos de G afínmente independientes.*

La prueba de este lema es casi idéntica a la de la Proposición 1 de [64], y puede encontrarse en el Capítulo 9.

Para el estudio de la estructura facial de $\mathcal{ECP}$ debemos comenzar identificando la dimensión de $\mathcal{ECP}$.

Teorema 5.1.5. *La dimensión de $\mathcal{ECP}$ es $n^2 - (\chi_{eq} + |\mathscr{S}| + 1)$ y un sistema minimal de ecuaciones para $\mathcal{ECP}$ es:*

$$\sum_{j=1}^{n} x_{vj} = 1, \qquad \forall\, v \in V \tag{5.1}$$

$$w_j = 1, \qquad \forall\, 1 \leq j \leq \chi_{eq} \tag{5.2}$$

$$w_j = w_{j+1}, \qquad \forall\, j \in \mathscr{S} \tag{5.3}$$

$$\sum_{v \in V} x_{vn} = w_n. \tag{5.4}$$

Demostración. Debemos probar que:

(i) Todo coloreo equitativo satisface las igualdades del sistema minimal.

(ii) Las igualdades son mutuamente independientes.

(iii) Existen $n^2 - \chi_{eq} - |\mathscr{S}|$ coloreos equitativos afínmente independientes.

La primera condición se cumple pues las igualdades (5.1) están explícitamente declaradas en *ECF* como restricciones de asignación, las igualdades (5.2) y (5.3) surgen de la definición de χ_{eq} y $\mathscr{S}$ respectivamente y la igualdad (5.4) afirma que la última clase de color tiene exactamente un elemento si el último color es usado. La segunda condición también se cumple dado que cada igualdad involucra una variable que no está presente en las demás igualdades. Finalmente, debido a que G no es un grafo completo, existen vértices u y u' no adyacentes. Sea c^1 un n-eqcol tal que $c^1(u) = n$ y $c^1(u') = n-1$. Con este coloreo, el Lema 5.1.4 prueba la condición (iii). □

En las próximas secciones analizaremos la dimensión de las caras de $\mathcal{ECP}$ definidas por diversas desigualdades válidas. Para ello, haremos uso reiterado de la siguiente observación.

Observación 5.1.6. *Dado un poliedro P y una cara $F = \{x \in P : \pi x = \pi_0\}$ definida por una desigualdad $\pi x \leq \pi_0$ válida en P, para probar que F es faceta de P, es suficiente con exhibir $dim(P)$ puntos afínmente independientes de F. Por el contrario, si deseamos probar que F no es faceta de P basta con exhibir una igualdad $\mu x = \mu_0$ que no pueda obtenerse mediante una combinación lineal de las ecuaciones del sistema minimal de P y la igualdad $\pi x = \pi_0$.*

5.2. Caras definidas por las desigualdades de la relajación lineal del modelo

Comenzamos por las restricciones de no negatividad de las variables.

Es fácil ver que las restricciones de no negatividad de las variables w_j nunca definen facetas. En efecto, $w_j \geq 0$ está dominada por la restricción (4.15) para $\chi_{eq} < j < n$, y nunca se satisface por igualdad para $j \leq \chi_{eq}$. Para $j = n$, tenemos $w_n = \sum_{v \in V} x_{vn} \geq 0$ por (5.4) y la no negatividad de x_{vn} para todo $v \in V$.

No sucede lo mismo con las variables x para las cuales podemos probar:

Teorema 5.2.1. *Sea $v \in V$ y $1 \leq j \leq n$. La restricción $x_{vj} \geq 0$ define una faceta de $\mathcal{ECP}$.*

Demostración. Para probar que $x_{vj} \geq 0$ define una faceta de $\mathcal{ECP}$, vamos a exhibir $n^2 - \chi_{eq} - |\mathscr{S}| - 1$ coloreos afínmente independientes que satisfacen $x_{vj} = 0$, acorde a la Observación 5.1.6.

Caso $j \leq n-2$. Sean $u, u' \in V \setminus \{v\}$ vértices no adyacentes y sea c^1 un n-eqcol tal que $c^1(u) = n$, $c^1(u') = n-1$ y $c^1(v) \neq j$. Consideramos los coloreos de $PROC(c^1)$, pero elegiendo los coloreos c^6_k de manera que no pinten a v de color j lo cual siempre es posible. Cada elemento de este conjunto de coloreos, excepto $c^2_{j,c^1(v)}$, satisface $x_{vj} = 0$.

Caso $j = n-1$. Sea S el conjunto de los n-eqcols y $(n-1)$-eqcols presentados en el caso anterior para $j = n-2$. Es claro que todo coloreo de S satisface $x_{vn-2} = 0$. Consideramos los coloreos $swap_{n-1,n-2}(\tilde{c})$ para cada $\tilde{c} \in S$ y un k-eqcol arbitrario de G para cada $k \in \{\chi_{eq}, \ldots, n-2\} \setminus \mathscr{S}$. Todos los coloreos satisfacen $x_{vn-1} = 0$.

Caso $j = n$. Sea u' un vértice no adyacente a v y sea c^1 un n-eqcol tal que $c^1(v) = n$ y $c^1(u') = n-1$. Consideremos los coloreos de $PROC(c^1)$ pero excluyendo a c^1. Los coloreos restantes satisfacen $x_{vn} = 0$. □

Analizamos ahora las desigualdades de cota superior de las variables. Las desigualdades $x_{vj} \leq 1$ nunca definen facetas porque son implicadas por la restricción de asignación (4.12) y las desigualdades $x_{vj} \geq 0$.

Respecto a las desigualdades $w_j \leq 1$, su facetitud será un corolario del resultado asociado a las desigualdades del orden establecido entre los colores.

Teorema 5.2.2. *Sea $1 \leq j \leq n-1$. La restricción (4.15), $w_{j+1} \leq w_j$, define faceta de $\mathcal{ECP}$ si y sólo si G admite un j-eqcol y $j \neq n-1$.*

Demostración. Sea F_j la cara de $\mathcal{ECP}$ definida por la desigualdad $w_{j+1} \leq w_j$.

Si G admite un j-eqcol y $j \neq n-1$, consideramos las soluciones de $PROC(c^1)$ para cualquier n-eqcol c^1 y excluimos el j-eqcol c^6_j. Es fácil ver que estos son $n^2 - \chi_{eq} - |\mathscr{S}| - 1$ coloreos afínmente independientes que satisfacen $w_{j+1} = w_j$.

Ahora demostramos que si G no admite un j-eqcol o $j = n-1$ entonces F_j no es una faceta de $\mathcal{ECP}$.

Si G no admite un j-eqcol, i.e. $j \in \{1, \ldots, \chi_{eq} - 1\} \cup \mathscr{S}$, entonces (4.15) es una combinación lineal de ecuaciones del sistema minimal y, por lo tanto, $F_j = \mathcal{ECP}$.

Por otro lado, la clase de color $n-1$ de todo coloreo (x, w) que satisface $w_n = w_{n-1}$ tiene a lo sumo un vértice. En este caso, (x, w) verifica $\sum_{v \in V} x_{vn-1} = w_{n-1}$ y, según la Observación 5.1.6, F_{n-1} no es una faceta de $\mathcal{ECP}$. □

Corolario 5.2.3. *La desigualdad $w_j \leq 1$ define faceta de $\mathcal{ECP}$ si y sólo si $j = \chi_{eq} + 1$.*

Demostración. Sea F_j la cara de $\mathcal{ECP}$ definida por la desigualdad $w_j \leq 1$.

Si $j \leq \chi_{eq}$, $F_j = \mathcal{ECP}$ por las ecuaciones (5.2). Si $j \geq \chi_{eq} + 2$, la desigualdad $w_j \leq 1$ es dominada por (4.15). Para el caso $j = \chi_{eq} + 1$, la cara F_j es la misma que la cara que define $w_j \leq w_{j-1}$, la cual es faceta por el teorema anterior. □

Los resultados anteriores (Teoremas 5.2.1 y 5.2.2) son, en realidad, adaptaciones de resultados similares propuestos en las Proposiciones 2 y 3 de [64] para el poliedro de coloreo clásico.

Los siguientes teoremas examinan la facetitud de las restricciones de equidad.

Teorema 5.2.4. *Sea $1 \leq j \leq n$. La restricción*

$$\sum_{v \in V} x_{vj} \geq \sum_{k=j}^{n} \left\lfloor \frac{n}{k} \right\rfloor (w_k - w_{k+1})$$

define una faceta de $\mathcal{ECP}$ si y sólo si $j \neq n$.

Demostración. Sea F_j la cara de $\mathcal{ECP}$ definida por (4.22). Vamos a demostrar que, para $j \leq n-1$, F_j es una faceta de $\mathcal{ECP}$ exponiendo $n^2 - \chi_{eq} - |\mathscr{S}| - 1$ coloreos afínmente independientes que caigan en F_j. Sean u, u' vértices no adyacentes y c^1 un n-eqcol tal que $c^1(u) = n$ y $c^1(u') = n-1$. Consideremos las soluciones de $PROC(c^1)$ y elijamos aquellos coloreos c^6_k de forma que satisfagan $|C_j| = \lfloor n/k \rfloor$ cuando $k \geq j$. Los coloreos propuestos, excepto aquel $(n-1)$-eqcol que asigna el color j a u y u' simultáneamente (si $j = n-1$ éste es c^4 y, si no, es c^5_j), pertenecen a F_j. Por lo tanto, F_j es una faceta de $\mathcal{ECP}$.

Si $j = n$, la desigualdad (4.22) dada por igualdad es equivalente a la ecuación (5.4) del sistema minimal y, consecuentemente, F_n no es faceta de $\mathcal{ECP}$. □

Teorema 5.2.5. *Sea $1 \leq j \leq n$. La restricción*

$$\sum_{v \in V} x_{vj} \leq \sum_{k=j}^{n} \left\lceil \frac{n}{k} \right\rceil (w_k - w_{k+1})$$

define una faceta de $\mathcal{ECP}$ si y sólo si $j = n-1$.

Demostración. Sea F_j la cara de $\mathcal{ECP}$ definida por (4.23). Vamos a exponer $n^2 - \chi_{eq} - |\mathscr{S}| - 1$ coloreos afínmente independientes que caigan en F_{n-1}, es decir la cara definida por $\sum_{v\in V} x_{vn-1} \leq 2w_{n-1} - w_n$. Dado que $n \geq 5$ y $\chi_{eq} \leq n-2$, existen vértices $u_1, u_2, u_3, u_4, u_5 \in V$ tales que u_1 no es adyacente a u_2 y u_3 no es adyacente a u_4. Sea c^1 un n-eqcol tal que $c^1(u_1) = n$ y $c^1(u_2) = n-1$. Sean además $j_1 = c^1(u_3)$ y $j_2 = c^1(u_4)$. Consideramos los coloreos c^1, $c^2_{j,j'}$, c^3_j y c^4 generados por $PROC(c^1)$, junto con los siguientes:

- El $(n-2)$-eqcol $\hat{c}$ tal que $\hat{c}(u_1) = \hat{c}(u_2) = j_1$, $\hat{c}(u_3) = \hat{c}(u_4) = j_2$ y $\hat{c}(i) = c^1(i)\ \ \forall\, i \in V\backslash\{u_1,u_2,u_3,u_4\}$.
- $swap_{j,j_1}(\hat{c})$ para cada $j \in \{1,\dots,n-2\}\backslash\{j_1, j_2\}$. De esta forma, construimos $n-4$ coloreos más.
- $swap_{\hat{c}(u_5),j_2}(\hat{c})$.
- Un k-eqcol arbitrario de G para cada $k \in \{\chi_{eq},\dots,n-3\}\backslash\mathscr{S}$. Tenemos $n - \chi_{eq} - |\mathscr{S}| - 2$ coloreos más.

Se puede probar que estos coloreos son afínmente independientes siguiendo un razonamiento similar al del Lema 5.1.4. Por lo tanto, F_{n-1} es faceta de $\mathcal{ECP}$.

Cuando $j \neq n-1$, F_j no es faceta de $\mathcal{ECP}$. En efecto, si $j = n$ entonces la desigualdad (4.23) dada por igualdad es equivalente a la ecuación (5.4) del sistema minimal, concluyéndose así que $F_n = \mathcal{ECP}$. Si $j \leq n-2$, todo coloreo (x,w) que cae en F_j satisface $\sum_{v\in V} x_{vn-1} = w_{n-1}$ y F_j no puede ser faceta de $\mathcal{ECP}$. □

Más adelante veremos una familia de desigualdades que domina a las restricciones (4.23) para los casos en que $j \leq n-2$.

Respecto a las restricciones de adyacencia (4.13) y de vértices aislados (4.19) veremos que son dominadas por desigualdades que presentamos en la próxima sección.

5.3. Desigualdades válidas heredadas del poliedro de coloreo clásico

En [62] se estudia el siguiente poliedro asociado a una formulación del PCG.

Definición 5.3.1. *El poliedro $\mathcal{CP}$ es la cápsula convexa de las soluciones enteras de la formulación de asignación de colores a vértices con las restricciones (4.14) y (4.15):*

$$\mathcal{CP} = conv\{(x,w) \in \{0,1\}^{n^2+n} : (x,w)\ satisface\ (4.12), (4.13), (4.14), (4.15)\}$$

Como $\mathcal{ECP}$ está incluído en $\mathcal{CP}$, toda desigualdad válida para $\mathcal{CP}$ también lo es para $\mathcal{ECP}$. En esta sección analizaremos la dimensión de las caras que definen en $\mathcal{ECP}$ las desigualdades válidas estudiadas en [62] para $\mathcal{CP}$.

En algunos casos donde desigualdades que definen facetas de $\mathcal{CP}$ no definen faceta de $\mathcal{ECP}$, aplicaremos el procedimiento de *lifting secuencial* definido en [66] que permite incrementar la dimensión de la cara que define la desigualdad estudiada y, eventualmente, alcanzar la condición de faceta.

5.3.1. Desigualdades *block*

Estas desigualdades fueron introducidas en [62] con el nombre de desigualdades *Anula Color*. Dado un vértice v y un color j, la desigualdad (v,j)-*block* está definida como

$$\sum_{k=j}^{n} x_{vk} \leq w_j. \tag{5.5}$$

Esta desigualdad es válida en $\mathcal{CP}$ ya que, si $w_j = 0$, el vértice v no utiliza ningún color de $j,\dots,n$.

En $\mathcal{ECP}$, las desigualdades block dominan a las desigualdades de vértices aislados (4.19).

Teorema 5.3.2. *Sea $v \in V$ y $1 \leq j \leq n$. La desigualdad (5.5) define una faceta de $\mathcal{ECP}$ si y sólo si, o bien G admite un $(j-1)$-eqcol y $3 \leq j \leq n-1$, o bien $j = 2$.*

Demostración. Sea F la cara de $\mathcal{ECP}$ definida por la desigualdad (v, j)-block. Para efectuar la prueba de que F es faceta de $\mathcal{ECP}$ adaptamos las Proposiciones 5.6.1 y 5.6.2 de [62]. Sabiendo que G admite un $(j-1)$-eqcol y $2 \leq j \leq n-1$, vamos a proponer $n^2 - \chi_{eq} - |\mathscr{S}| - 1$ coloreos afínmente independientes pertenecientes a F. La independencia afín de ellos puede determinarse de manera similar a la presentada en Lema 5.1.4. Sea $\tilde{c}$ un $(j-1)$-eqcol de G y sea $q = \tilde{c}(v)$.

Caso $3 \leq j \leq n-2$. Sean $u, u' \in V \backslash \{v\}$ no adyacentes. Consideramos el siguiente conjunto de coloreos:

- Un n-eqcol c tal que $c(v) = j$, $c(u) = n$ y $c(u') = n-1$.
- $swap_{n,k,k'}(c)$ para cada $k, k' \in \{1, \ldots, n-1\}$ tal que $k' \neq k$ y, en el caso que $k' = j$ entonces $k > j$. Tenemos $(n-1)(n-2) - j + 1$ coloreos.
- $swap_{n,k}(c)$ para cada $k \in \{1, \ldots, n-1\}$. Con esto construimos $n-1$ coloreos más.
- El $(n-1)$-eqcol c' tal que $c'(u) = n-1$ y $c'(i) = c(i) \ \ \forall\, i \in V \backslash \{u\}$.
- $swap_{n-1,k}(c')$ para cada $k \in \{1, \ldots, n-2\}$. Aquí tenemos $n-2$ coloreos.
- El $(j-1)$-eqcol $\tilde{c}$.
- $swap_{q,k}(\tilde{c})$ para cada $k \in \{1, \ldots, j-1\} \backslash \{q\}$. Tenemos $j-2$ coloreos aquí.
- Un k-eqcol de G que coloree v con color j para cada $k \in \{j, \ldots, n-2\} \backslash \mathscr{S}$, y un k-eqcol arbitrario de G para cada $k \in \{\chi_{eq}, \ldots, j-2\} \backslash \mathscr{S}$. Obtenemos $n - \chi_{eq} - |\mathscr{S}| - 2$ coloreos más.

Caso $j = n-1$. Como G no tiene vértices universales, existe un vértice v' no adyacente a v. Además, dado que $\chi_{eq} \leq n-2$, existen $u, u' \in V \backslash \{v, v'\}$ no adyacentes entre sí. Consideramos el siguiente conjunto de coloreos:

- Un n-eqcol c tal que $c(v) = n-1$, $c(v') = n$ y $c(u) = n-2$.
- $swap_{n,k,k'}(c)$ para cada $k, k' \in \{1, \ldots, n-1\}$ tal que $k' \neq k$ y $k' \neq n-1$. Tenemos $(n-2)(n-2)$ coloreos.
- $swap_{n,k}(c)$ para cada $k \in \{1, \ldots, n-1\}$. Con esto construimos $n-1$ coloreos más.
- El $(n-1)$-eqcol c' tal que $c'(v') = n-1$ y $c'(i) = c(i) \ \ \forall\, i \in V \backslash \{v'\}$.
- El $(n-1)$-eqcol c'' tal que $c''(u') = n-2$, $c''(v') = c(u')$ y $c''(i) = c(i) \ \ \forall\, i \in V \backslash \{u', v'\}$.
- $swap_{n-2,k}(c'')$ para todo $k \in \{1, \ldots, n-3\}$. Aquí tenemos $n-3$ coloreos.
- El $(n-2)$-eqcol $\tilde{c}$.
- $swap_{q,k}(\tilde{c})$ para cada $k \in \{1, \ldots, n-2\} \backslash \{q\}$. Tenemos $n-3$ coloreos aquí.
- Un k-eqcol arbitrario de G para cada $k \in \{\chi_{eq}, \ldots, n-3\} \backslash \mathscr{S}$. Obtenemos $n - \chi_{eq} - |\mathscr{S}| - 2$ coloreos más.

Caso $j = 2$. Dado que $w_2 = 1$, la desigualdad $(v,2)$-block es $\sum_{k=2}^{n} x_{vk} \leq 1$. Restando la igualdad de asignación (5.1) obtenemos $x_{v1} \geq 0$, que define faceta de $\mathcal{ECP}$ por Teorema 5.2.1. Luego, la $(v,2)$-block también define faceta de $\mathcal{ECP}$.

Ahora revisamos los casos en donde F no es faceta de $\mathcal{ECP}$. La desigualdad $(v,1)$-block siempre es satisfecha por igualdad y entonces $F = \mathcal{ECP}$. En el caso de la desigualdad (v,n)-block, los puntos de la cara que la definen también satisfacen $x_{v'n} = 0$ para todo $v' \neq v$, no siendo entonces una desigualdad que defina faceta de $\mathcal{ECP}$. Finalmente, si G no admite un $(j-1)$-eqcol entonces toda solución de F satisfaría $x_{vj-1} = 0$. En efecto, sea (x,w) un k-eqcol de G tal que $\sum_{k=j}^{n} x_{vk} = w_j$. Si $k \leq j-2$, claramente $x_{vj-1} = 0$. Si no, $\sum_{k=j}^{n} x_{vk} = 1$ dado que $k \neq j-1$, y entonces $x_{vj-1} = 0$. De aquí que F no puede ser una faceta de $\mathcal{ECP}$. □

5.3.2. Desigualdades de rango

Dado un conjunto $S \subset V$ y un color j, llamamos *desigualdad de rango* (asociada a S y j) a la desigualdad $\sum_{v \in S} x_{vj} \leq \alpha(S) w_j$. Puede verse fácilmente que todas las desigualdades de rango son válidas para $\mathcal{CP}$. Más aún, si S no es α-maximal, es decir que existe un vértice v tal que $\alpha(S \cup \{v\}) = \alpha(S)$, entonces la desigualdad de rango asociada a S y j está dominada por la desigualdad de rango asociada a $S \cup \{v\}$ y j. Por este motivo, en esta sección asumimos que los conjuntos S siempre son α-maximales.

Observemos que si $\alpha(S) = 1$, S es una clique de G y las desigualdades de rango toman la forma

$$\sum_{v \in S} x_{vj} \leq w_j.$$

Estas son las *desigualdades de clique* presentadas en [62]. Dada una clique Q y un color j, nosotros llamaremos (Q,j)-clique a esa desigualdad, i.e. $\sum_{v \in Q} x_{vj} \leq w_j$.

Si $|Q| = 1$, i.e. $Q = \{v\}$ para algún v, entonces la desigualdad (Q,j)-clique es dominada por la (v,j)-block. Si $|Q| \geq 2$, esta desigualdad siempre define faceta de $\mathcal{CP}$ [62]. Esto vale aún para $\mathcal{ECP}$, como afirma el siguiente teorema.

Teorema 5.3.3. *Sea $j \leq n-1$ y Q una clique maximal de G tal que $|Q| \geq 2$. La desigualdad (Q,j)-clique define una faceta de $\mathcal{ECP}$.*

La prueba de este teorema puede encontrarse en el Capítulo 9.

En [62] se prueba que, para $\alpha(S) \geq 2$, las desigualdades de rango asociadas a un S α-maximal y un $j \leq n - \alpha(S)$ no definen facetas de $\mathcal{CP}$, y mediante la aplicación de un lifting secuencial se obtiene la siguiente desigualdad válida para $\mathcal{CP}$:

$$\sum_{v \in S} x_{vj} + \sum_{v \in V} \sum_{k=n-\alpha(S)+1}^{n-1} x_{vk} \leq \alpha(S) w_j + w_{n-\alpha(S)+1} - w_n.$$

Nos referiremos a esta última como la *desigualdad (S,j)-rango*. En la Proposición 5.9 de [62] se dan condiciones suficientes para que estas desigualdades definan facetas de $\mathcal{CP}$.

Analizamos ahora las desigualdades (S,j)-rango cuando $\alpha(S) = 2$. En este caso, la desigualdad toma la forma:

$$\sum_{v \in S} x_{vj} + \sum_{v \in V} x_{vn-1} \leq 2w_j + w_{n-1} - w_n, \tag{5.6}$$

y nosotros la llamaremos desigualdad (S,j)-*2-rango*.

El siguiente resultado da una condición necesaria para que esta desigualdad defina faceta de $\mathcal{ECP}$.

Lema 5.3.4. *Sea $j \leq \lceil n/2 \rceil - 1$ y $S \subset V$ tal que $\alpha(S) = 2$ y S es α-maximal. Si la desigualdad (S, j)-2-rango define faceta de $\mathcal{ECP}$ entonces para todo k tal que $k \in \{\text{máx}\{\chi_{eq}, j\}, \ldots, n-1\} \setminus \mathscr{S}$, existe un k-eqcol donde dos vértices de S comparten el mismo color.*

Demostración. Sea F la cara de $\mathcal{ECP}$ definida por (5.6). Si existiese un $k \in \{\text{máx}\{\chi_{eq}, j\}, \ldots, n-1\} \setminus \mathscr{S}$ en donde ningún k-eqcol de G pueda colorear dos vértices de S con el mismo color, entonces, en todo k-eqcol, el lado izquierdo de (5.6) sería menor que el lado derecho (cuyo valor es 2). Luego, no habría ningún k-eqcol que yace en F y, por lo tanto, todos los coloreos de F satisfarían $w_k = w_{k+1}$. Dado que $k \notin \mathscr{S}$, esta igualdad no podría ser obtenida como combinación lineal de las igualdades del sistema minimal de $\mathcal{ECP}$ y (5.6), y la cara F no podría ser una faceta de $\mathcal{ECP}$. □

Condiciones suficientes para que la desigualdad (S, j)-2-rango defina faceta de $\mathcal{ECP}$ son presentadas en el siguiente teorema. Su prueba se encuentra en el Capítulo 9.

Teorema 5.3.5. *Sea $j \leq \lceil n/2 \rceil - 1$ y $S \subset V$ tal que $\alpha(S) = 2$ y S es α-maximal. Si*

(i) existe un conjunto estable H de tamaño 3 en G tal que:

- *si n es impar, el complemento de $G - H$ tiene un matching perfecto M y ambos extremos de alguna arista de este matching pertenecen a S,*
- *si n es par, existe otro conjunto estable H' también de tamaño 3 en G tal que $H \cap H' = \varnothing$, el complemento de $G - (H \cup H')$ tiene un matching perfecto M, ambos extremos de alguna arista de este matching pertenecen a S y existen vértices $h \in H$, $h' \in H'$ no adyacentes entre sí,*

(ii) para todo $v \in V \setminus S$, existen vértices diferentes $s, s' \in S$ y un conjunto estable $H_v = \{v, s, s'\}$ en G tal que:

- *si n es impar, el complemento de $G - H_v$ tiene un matching perfecto,*
- *si n es par, existe otro conjunto estable H'_v de tamaño 3 en G tal que $H_v \cap H'_v = \varnothing$ y el complemento de $G - (H_v \cup H'_v)$ tiene un matching perfecto,*

(iii) para todo k tal que $k \in \{\text{máx}\{\chi_{eq}, j\}, \ldots, \lceil n/2 \rceil - 2\} \setminus \mathscr{S}$, existe un k-eqcol donde dos vértices de S comparten el mismo color

entonces la desigualdad (S, j)-2-rango define una faceta de $\mathcal{ECP}$.

Para el caso $j > \lceil n/2 \rceil - 1$, presentaremos más adelante una desigualdad válida que la domina.

Presentamos un ejemplo donde el teorema anterior se aplica para demostrar que una desigualdad (S, j)-2-rango define una faceta de $\mathcal{ECP}$.

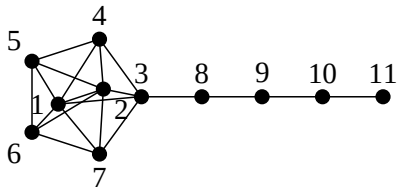


Figura 5.1: Grafo de los Ejemplos 5.3.6 y 5.3.11

Ejemplo 5.3.6. *Sea G el grafo presentado en la Figura 5.1. Su número cromático equitativo es 5. Si $S = \{1, 2, \ldots, 7\}$ entonces $\alpha(S) = 2$ y S es α-maximal. Si $H = \{4, 7, 8\}$ entonces H es un conjunto estable tal que $\overline{G - H}$ tiene un matching perfecto $\{(1,10), (2,11), (3,5), (6,9)\}$ con $\{3,5\} \subset S$. Más aún, para todo $v \in \{8, 9, 10, 11\}$ existe un conjunto estable $H_v = \{4, 7, v\}$ tal que $\overline{G - H_v}$ tiene un matching perfecto. Entonces, si $1 \leq j \leq 5 = \lceil 11/2 \rceil - 1$, la desigualdad (S, j)-2-rango define una faceta de $\mathcal{ECP}(G)$.*

A continuación presentamos otra forma de reforzar la desigualdad de rango clásica cuando $\alpha(S) = 2$.

Sea Q el conjunto de vértices de S que son universales en $G[S]$, i.e. $Q = \{q \in S \ : \ S \subset N[q]\}$. Si Q no es vacío, para cada $v \in Q$, el coeficiente de x_{vj} se puede incrementar en la desigualdad $\sum_{v \in S} x_{vj} \leq 2w_j$. Esto motiva a la siguiente definición.

Definición 5.3.7. *Sea $S \subset V$ tal que $\alpha(S) = 2$ y S es α-maximal. Sea $Q = \{q \in S \ : \ S \subset N[q]\} \neq \varnothing$ y $j \leq n-1$. Definimos la desigualdad (S,Q,j)-2-rango como*

$$\sum_{v \in S \setminus Q} x_{vj} + 2 \sum_{v \in Q} x_{vj} \leq 2w_j. \tag{5.7}$$

Lema 5.3.8. *La desigualdad (S,Q,j)-2-rango es válida para $\mathcal{CP}$.*

Demostración. Notemos que si algún vértice v de Q utiliza el color j, ninguno más en S puede pintarse de j dado que todo vértice de S es adyacente a v. Luego, el valor del lado izquierdo es a lo sumo 2 cuando el color j se usa. □

Si $|Q| = 1$, la desigualdad (S,Q,j)-2-rango es dominada por otra desigualdad que presentaremos más adelante.

Los siguientes resultados nos dan condiciones suficientes para que la (S,Q,j)-2-rango defina faceta de $\mathcal{ECP}$. Sus pruebas se hallan en el Capítulo 9.

Teorema 5.3.9. *Sea $S \subset V$ tal que $\alpha(S) = 2$ y S es α-maximal, y sea $Q = \{q \in S \ : \ S \subset N[q]\}$. Si $|Q| \geq 2$ y*

(i) ninguna componente conexa del complemento de $G[S \setminus Q]$ es bipartita,

(ii) para todo $v \in V \setminus S$ tal que $Q \subset N(v)$, existen vértices diferentes $s, s' \in S \setminus Q$ y un conjunto estable $H_v = \{v, s, s'\}$ en G que satisfacen:

- *Si n es impar, el complemento de $G - H_v$ tiene un matching perfecto,*
- *Si n es par, existe otro conjunto estable H'_v de tamaño 3 en G tal que $H_v \cap H'_v = \varnothing$ y el complemento de $G - (H_v \cup H'_v)$ tiene un matching perfecto,*

entonces, para todo $j \leq \lceil n/2 \rceil - 1$, la desigualdad (S,Q,j)-2-rango define una faceta de $\mathcal{ECP}$.

Corolario 5.3.10. *Sea $S \subset V$ tal que $\alpha(S) = 2$ y S es α-maximal, y sea $Q = \{q \in S \ : \ S \subset N[q]\}$. Si $|Q| \geq 2$, ninguna componente conexa del complemento de $G[S \setminus Q]$ es bipartita y para todo $v \in V \setminus S$, $Q \setminus N(v) \neq \varnothing$, entonces la desigualdad (S,Q,j)-2-rango define una faceta de $\mathcal{ECP}$ para todo $j \leq n-1$.*

El siguiente es un ejemplo en donde se emplea el resultado anterior.

Ejemplo 5.3.11. *Sea G el grafo de la Figura 5.1, $S = \{1,2,\ldots,7\}$ y $Q = \{1,2\}$. La desigualdad (S,Q,j)-2-rango define faceta de $\mathcal{ECP}(G)$ para $1 \leq j \leq 10$ pues se verifica el enunciado del corolario 5.3.10: los vértices $3,\ldots,7$ definen un agujero de longitud impar en $\overline{G}$ y para todo $v \in \{8,9,10,11\}$, $Q \setminus N(v) = \{1,2\}$.*

En lo que respecta a las desigualdades (S,j)-rango con $\alpha(S) \geq 3$, el siguiente resultado nos brinda condiciones necesarias para que las mismas definan facetas de $\mathcal{ECP}$.

Teorema 5.3.12. *Sea $S \subsetneq V$ tal que $\alpha(S) \geq 3$, y sea $j \leq n - \alpha(S)$. Si la desigualdad (S,j)-rango define faceta de $\mathcal{ECP}$ entonces se verifican las siguientes condiciones:*
(a) $\alpha(S) < \lceil n/\chi_{eq} \rceil$.
(b) G no admite un $(n - \alpha(S))$-eqcol.

Demostración. Sea F la cara de $\mathcal{ECP}$ definida por la desigualdad (S,j)-rango. Demostraremos por el absurdo, asumiendo que si no se cumple (a) o (b), siempre existe una igualdad que no puede escribirse como combinación lineal de las ecuaciones del sistema minimal de $\mathcal{ECP}$ y la (S,j)-rango, lo que lleva a probar que F no es faceta de $\mathcal{ECP}$.

Supongamos que no se verifique (a). Entonces $\alpha(S) \geq \lceil n/\chi_{eq} \rceil$. Si $\alpha(S) > \lceil n/\chi_{eq} \rceil$, ningún χ_{eq}-eqcol pertenece a F porque el lado derecho de la desigualdad vale $\alpha(S)$ mientras que el lado izquierdo a lo sumo puede valer $\lceil n/\chi_{eq} \rceil$. Luego, todos los coloreos de F satisfacen $w_{\chi_{eq}+1} = 1$.

Por lo tanto, sólo puede darse $\alpha(S) = \lceil n/\chi_{eq} \rceil$. Sea $v \in V \backslash S$ y sea c un coloreo equitativo arbitrario que yace en F. Observemos que la clase de color j de c tiene que estar contenida completamente en S, implicando que v no puede colorearse con j en c. Es decir que todos los coloreos de F satisfacen $x_{vj} = 0$.

Debido a que (a) está obligada a verificarse, supongamos ahora que no se verifique (b). Es decir que $n - \alpha(S) \geq \chi_{eq}$ y $n - \alpha(S) \notin \mathscr{S}$. Dado que $\chi_{eq} \geq 2$, resulta $3 \leq \alpha(S) \leq \lceil n/\chi_{eq} \rceil - 1 \leq \lceil n/2 \rceil - 1$, lo que obliga a G a tener al menos 7 vértices. Luego, $\alpha(S) \geq \lceil \frac{n}{n-\alpha(S)} \rceil + 1$ y, por lo tanto, ningún $(n-\alpha(S))$-eqcol satisface la desigualdad (S,j)-rango por igualdad. Entonces todo coloreo de F satisface $w_{n-\alpha(S)} = w_{n-\alpha(S)+1}$. Dado que $n - \alpha(S) \notin \mathscr{S}$, esta ecuación no puede escribirse como combinación lineal de las ecuaciones del sistema minimal y la (S,j)-rango. □

Las condiciones (a) y (b) del teorema anterior indican que las desigualdades (S,j)-rango definen facetas con *poca frecuencia*. En particular, es fácil ver que si G es monótono nunca se verifican (a) y (b) simultáneamente y, por lo tanto, estas desigualdades no definen facetas.

Respecto al caso $S = V$, veremos más adelante nuevas desigualdades válidas para $\mathcal{ECP}$ que dominan a la desigualdad (V,j)-rango.

5.4. Nuevas desigualdades válidas para $\mathcal{ECP}$

En esta sección presentaremos familias de desigualdades válidas para $\mathcal{ECP}$ que no son válidas para $\mathcal{CP}$. Daremos condiciones suficientes para que definan faceta y, en algunos casos, condiciones necesarias. También determinaremos una cota inferior de la dimensión de la cara que definen en $\mathcal{ECP}$, mostrando que todas ellas definen caras de alta dimensión.

5.4.1. Desigualdades *subvecindad*

Las desigualdades *de vecindad* fueron definidas en [64] de la siguiente manera. Dado un color j y un vértice $u \in V$, la desigualdad

$$\alpha(N(u))x_{uj} + \sum_{v \in N(u)} x_{vj} \leq \alpha(N(u))w_j$$

es válida para $\mathcal{CP}$. Estas desigualdades pueden verse como un caso particular de las desigualdades que proponemos a continuación.

Sea j un color, u un vértice de V y $S \subset N(u)$. Es fácil comprobar que la desigualdad

$$\alpha(S)x_{uj} + \sum_{v \in S} x_{vj} \leq \alpha(S)w_j$$

también es válida para $\mathcal{CP}$.

De hecho, si el color j no se usa, ambos lados valen cero, y si el color j se usa, entonces $\sum_{v \in S} x_{vj}$ puede contribuir hasta $\alpha(S)$ cuando u no se pinta de j.

Observemos que si $\alpha(S) = 1$, obtenemos la desigualdad clique, ya analizada en la Sección 5.3.2. Por esa razón, de ahora en más suponemos que $\alpha(S) \geq 2$.

En [62], la desigualdad de vecindad es ajustada mediante un lifting secuencial, para el caso $j \leq n - \alpha(N(u)) + 1$, obteniéndose la siguiente desigualdad válida para $\mathcal{CP}$:

$$\alpha(N(u))x_{uj} + \sum_{v \in N(u)} x_{vj} + \sum_{k=n-\alpha(N(u))+2}^{n} \big(\alpha(N(u)) + k - n - 1\big)x_{uk} \leq \alpha(N(u))w_j.$$

Nosotros realizamos un ajuste similar, sobre las desigualdades más generales presentadas anteriormente y en el contexto del poliedro $\mathcal{ECP}$. Con este mecanismo obtuvimos dos familias de desigualdades válidas que bautizamos como desigualdades de *subvecindad*.

Definición 5.4.1. *Dados $u \in V$, $S \subset N(u)$ tal que S no es una clique y $j \leq n-1$, la desigualdad* (u, j, S)-subvecindad *es definida como*

$$\gamma_{jS}x_{uj} + \sum_{v \in S} x_{vj} + \sum_{k=j+1}^{n} (\gamma_{jS} - \gamma_{kS})x_{uk} \leq \gamma_{jS}w_j, \tag{5.8}$$

donde $\gamma_{kS} = \min\{\lceil n/\chi_{eq} \rceil, \lceil n/k \rceil, \alpha(S)\}$ para todo k.

Lema 5.4.2. *La desigualdad (u, j, S)-subvecindad es válida para $\mathcal{ECP}$.*

Demostración. Sea (x, w) un r-eqcol. Si $r < j$, ambos lados de (5.8) valen cero. Si $r \geq j$ y $x_{uj} = 1$, el valor del lado izquierdo de (5.8) es exactamente γ_{jS}. En cambio, si $x_{uj} = 0$, el término $\sum_{v \in S} x_{vj}$ puede contribuir hasta γ_{rS} mientras que el término $\sum_{k=j+1}^{n} (\gamma_{jS} - \gamma_{kS})x_{uk}$ puede contribuir hasta $\gamma_{jS} - \gamma_{rS}$. Así que el lado izquierdo nunca excede de γ_{jS} y la desigualdad resulta válida. □

El siguiente resultado demuestra que estas desigualdades siempre definen caras de alta dimensión. Más específicamente, la dimensión de la cara definida por la desigualdad (u, j, S)-subvecindad es $o(n^2)$ como también lo es la dimensión de $\mathcal{ECP}$.

Teorema 5.4.3. *Sea F la cara de $\mathcal{ECP}$ definida por la desigualdad (u, j, S)-subvecindad. Entonces $dim(F) \geq dim(\mathcal{ECP}) - p$, donde:*

$$p = \begin{cases} \delta(u) + 1 - |S|, & si\ \gamma_{jS} = 2 \\ \lceil n/2 \rceil - 1 - |S| + \delta(u), & si\ \gamma_{jS} \geq 3 \end{cases}$$

Demostración. Sean $s_1, s_2 \in S$ vértices no adyacentes y sea $1 \leq r \leq \lceil n/2 \rceil - 1$ tal que $r \neq j$. Vamos a proponer $n^2 - \chi_{eq} - |\mathscr{S}| - p$ coloreos equitativos afínmente independientes que yacen en F:

- Un n-eqcol c tal que $c(u) = j$, $c(s_1) = n$ y $c(s_2) = r$.
- $swap_{n,j_1,j_2}(c)$ para cada $j_1, j_2 \in \{1, \ldots, n-1\} \backslash \{j\}$ tales que $j_1 \neq j_2$. Así son construidos $(n-2)(n-3)$ coloreos.
- $swap_{c(s),n,j}(c)$ para cada $s \in S \backslash \{s_1\}$. Tenemos $|S| - 1$ coloreos más.
- $swap_{n,j'}(c)$ para cada $j' \in \{1, \ldots, n-1\}$. Tenemos $n-1$ coloreos más.
- El $(n-1)$-eqcol c' tal que $c'(s_1) = r$ y $c'(i) = c(i)\ \ \forall\, i \in V \backslash \{s_1\}$.
- $swap_{j',r}(c')$ para cada $j' \in \{1, \ldots, n-1\} \backslash \{j, r\}$. Así fabricamos $n-3$ coloreos más.
- Consideramos 2 casos según el valor de γ_{jS}:
  Caso $\gamma_{jS} = 2$. $swap_{j,r,j'}(c')$ para cada $j' \in \{1, \ldots, n-1\} \backslash \{j, r\}$, y $swap_{j,r}(c')$. Así obtenemos $n-2$ coloreos.
  Caso $\gamma_{jS} \geq 3$. $swap_{j,r,j'}(c')$ para cada $j' \in \{\lceil n/2 \rceil, \ldots, n-1\}$ (observemos que $j' \neq j$). Así obtenemos $\lfloor n/2 \rfloor$ coloreos.

- Los $(n-1)$-eqcols c'' tales que $c''(s_1) = c(v)$, $c''(v) = j$ y $c''(i) = c(i) \quad \forall\, i \in V \backslash \{s_1, v\}$, para cada $v \in V \backslash N[u]$. De esta manera se construyen $n - \delta(u) - 1$ coloreos.
- Si $j \geq \chi_{eq}$, entonces proponemos un k-eqcol arbitrario de G para cada $k \in \{\chi_{eq}, \ldots, j-1\} \backslash \mathscr{S}$.
- $swap_{j,\hat{c}(u)}(\hat{c})$ donde $\hat{c}$ es un k-eqcol arbitrario de G, para cada $k \in \{\text{máx}\{j, \chi_{eq}\}, \ldots, n-2\} \backslash \mathscr{S}$. Entre este ítem y el anterior suman $n - \chi_{eq} - |\mathscr{S}| - 1$ coloreos más.

La independencia afín de estos coloreos puede ser verificada de manera análoga a la propuesta en el Lema 5.1.4. □

Corolario 5.4.4. *Si $\alpha(N(u)) = 2$, la desigualdad $(u, j, N(u))$-subvecindad define faceta de $\mathcal{ECP}$.*

Demostración. Observar que cuando $\alpha(N(u)) = 2$, $\gamma_{jN(u)} = 2$ y el teorema anterior nos asegura que la dimensión de la cara definida por la desigualdad $(u, j, N(u))$-subvecindad es al menos $dim(\mathcal{ECP}) - 1$. Como claramente existen coloreos equitativos que no satisfacen por igualdad esta desigualdad, sabemos que la dimensión de la cara es exactamente $dim(\mathcal{ECP}) - 1$ y, por lo tanto, la desigualdad siempre define faceta de $\mathcal{ECP}$. □

Veamos a continuación que no es necesario analizar la dimensión de las caras definidas por las desigualdades (u, j, S)-subvecindad para valores de j tales que $\lceil n/j \rceil > \lceil n/\chi_{eq} \rceil$, ya que la misma coincide con la dimensión de la cara definida por la desigualdad (u, χ_{eq}, S)-subvecindad.

Teorema 5.4.5. *Sea j tal que $\lceil n/j \rceil > \lceil n/\chi_{eq} \rceil$, F_j la cara de $\mathcal{ECP}$ definida por la desigualdad (u, j, S)-subvecindad y $F_{\chi_{eq}}$ la cara de $\mathcal{ECP}$ definida por la desigualdad (u, χ_{eq}, S)-subvecindad. Entonces, $dim(F_j) = dim(F_{\chi_{eq}})$.*

Demostración. Claramente, si $\alpha(S) < \lceil n/\chi_{eq} \rceil$, ambas desigualdades coinciden. Así que supondremos que $\alpha(S) \geq \lceil n/\chi_{eq} \rceil$. Debido a que $\lceil n/j \rceil > \lceil n/\chi_{eq} \rceil$, entonces $j < \chi_{eq}$ y $w_j = w_{\chi_{eq}} = 1$. Luego, ambas desigualdades sólo difieren en los coeficientes que acompañanan a x_{vj} y $x_{v\chi_{eq}}$, para todo $v \in V$. Más precísamente, el coeficiente de x_{vj} en la (u, j, S)-subvecindad es igual al coeficiente de $x_{v\chi_{eq}}$ en la (u, χ_{eq}, S)-subvecindad, y viceversa.
Sea $d = dim(F_{\chi_{eq}})$ y $d' = dim(F_j)$. Si $c^1, c^2, \ldots, c^{d+1}$ son coloreos afínmente independientes en $F_{\chi_{eq}}$, entonces $swap_{j,\chi_{eq}}(c^i)$ para todo $1 \leq i \leq d+1$ resultan coloreos bien definidos. Es evidente que estos $d+1$ *puntos* son también afínmente independientes, pues sólo hemos hecho intercambio de columnas entre ellos. Además yacen en F_j y, por lo tanto, $d \leq d'$.
Para probar que $d' \leq d$, seguimos el mismo razonamiento. □

El siguiente teorema presenta condiciones suficientes para que las desigualdades (5.8) definan facetas de $\mathcal{ECP}$, para el caso en que $\lceil n/j \rceil \leq \lceil n/\chi_{eq} \rceil$. Su prueba se encuentra en el Capítulo 9.

Teorema 5.4.6. *Sea G un grafo monótono, $u \in V$, $j \leq n-1$ tal que $\lceil n/j \rceil \leq \lceil n/\chi_{eq} \rceil$ y $S \subset N(u)$ tal que S no es una clique y además, si $S \neq N(u)$ entonces $\alpha(S) \leq \lceil n/j \rceil - 1$.*
Si se satisfacen las siguientes condiciones:

(i) para todo $3 \leq i \leq \min\{\lceil n/j \rceil, \alpha(S)\}$, existe un $\left(\left\lceil \frac{n}{i-1} \right\rceil - 1\right)$-eqcol cuya clase de color C_j satisface $|C_j \cap S| = i$,

(ii) para todo $v \in N(u) \backslash S$, existe un coloreo equitativo cuya clase de color C_j satisface $|C_j \cap S| = \alpha(S)$ y $(C_j \cap N(u)) \backslash S = \{v\}$ (i.e. que pinta a v y $\alpha(S)$ vértices de S de color j y a ningún vértice más en el vecindario de u).

entonces la desigualdad (u,j,S)-subvecindad define una faceta de $\mathcal{ECP}$.

Recordemos que en el análisis de las desigualdades (S,Q,j)-2-rango mencionamos que, cuando $|Q|=1$, la misma se encontraba dominada por otra desigualdad válida que presentaríamos más adelante. Veremos ahora que existe una desigualdad subvecindad que la domina.

Más específicamente, dados $j \leq n-1$, $S \subset V$ tal que $\alpha(S)=2$ y $Q=\{v \in S \ : \ S \subset N[v]\}=\{q\}$ (para algún q), la desigualdad (S,Q,j)-2-rango es dominada por la desigualdad $(q,j,S\backslash\{q\})$-subvecindad.

Como vimos en el Corolario 5.4.4 esta desigualdad siempre define faceta de $\mathcal{ECP}$ cuando $S\backslash\{q\}=N(q)$. Para los casos en que $S\backslash\{q\} \subsetneqq N(q)$, las condiciones suficientes del Teorema 5.4.6 pueden ser simplificadas como muestra el siguiente corolario.

Corolario 5.4.7. *Sea G un grafo monótono, $j \leq \lceil n/2 \rceil - 1$, $u \in V$ y $S \subsetneqq N(u)$ tal que $\alpha(S)=2$. Si para todo $v \in N(u)\backslash S$, existen vértices $s,s' \in S$ y un conjunto estable $H_v=\{v,s,s'\}$ en G tal que:*

- *Si n es impar, el complemento de $G-H_v$ tiene un matching perfecto,*
- *Si n es par, existe otro conjunto estable H'_v de tamaño 3 en G tal que $H_v \cap H'_v = \varnothing$ y el complemento de $G-(H_v \cup H'_v)$ tiene un matching perfecto,*

entonces la desigualdad (u,j,S)-subvecindad define una faceta de $\mathcal{ECP}$.

Demostración. **Caso** $\lceil n/j \rceil \leq \lceil n/\chi_{eq} \rceil$. Vamos a probar que las hipótesis del Teorema 5.4.6 se satisfacen. La hipótesis (i) se satisface trivialmente. La hipótesis $\alpha(S) \leq \lceil n/j \rceil - 1$ también pues $j \leq \lceil n/2 \rceil - 1$.
Sea $v \in N(u)\backslash S$ y M_v, H_v y H'_v (en caso de que n sea par) el matching y los conjuntos estables dado por hipótesis. Consideremos el $(\lceil n/2 \rceil - 1)$-eqcol cuya clase C_j es H_v y el resto de las clase de colores son H'_v (en caso de que n sea par) y los extremos de las aristas de M_v. Entonces, $|C_j \cap S|=2$, $(C_j \cap N(u))\backslash S=\{v\}$ y la hipótesis (ii) del Teorema 5.4.6 se cumple. Por lo tanto, la desigualdad (u,j,S)-subvecindad define una faceta de $\mathcal{ECP}$.

Caso $\lceil n/j \rceil > \lceil n/\chi_{eq} \rceil$. Según el caso probado anteriormente, sabemos que la desigualdad (u,χ_{eq},S)-subvecindad define faceta de $\mathcal{ECP}$ y, en virtud del Teorema 5.4.5, también define faceta la desigualdad (u,j,S)-subvecindad. □

Finalizamos esta sección presentando ejemplos en donde se aplican los Teoremas 5.4.5 y 5.4.6.

Ejemplo 5.4.8. *Sea G el grafo de la Figura 5.2(a). Tenemos que G es monótono y $\chi_{eq}(G)=3$. Consideremos la desigualdad (u,j,S)-subvecindad en los siguientes casos:*

1. *Si $u=1$, $S=N(1)$ y $j=3$, la desigualdad toma la forma*

$$4x_{1,3}+\sum_{v=2}^{6} x_{v,3}+x_{1,4}+x_{1,5}+2x_{1,6}+2x_{1,7}+2x_{1,8}+2x_{1,9}+2x_{1,10}+3x_{1,11} \leq 4w_3.$$

   Probaremos que define faceta de $\mathcal{ECP}$ exhibiendo un $(\lceil \frac{11}{2} \rceil - 1)$-eqcol que verifique $|C_3 \cap S|=3$ y un $(\lceil \frac{11}{3} \rceil - 1)$-eqcol que verifique $|C_3 \cap S|=4$. Ambos coloreos se muestran en la Figura 5.2 (b) y (c) respectivamente. Con estos coloreos podemos aplicar el Teorema 5.4.6.

2. *Si $u=1$, $S=N(1)$ y $4 \leq j \leq 10$, no es difícil de ver que, usando también el Teorema 5.4.6, la desigualdad (u,j,S)-subvecindad define faceta de $\mathcal{ECP}$.*

3. *Si $u=1$, $S=N(1)$ y $j \in \{1,2\}$, la desigualdad (u,j,S)-subvecindad define faceta de $\mathcal{ECP}$ por el Teorema 5.4.5.*

4. *Si* $u = 1$, $S = \{3,4,5\}$ *y* $j = 3$, *la desigualdad toma la forma*

$$3x_{1,3} + x_{3,3} + x_{4,3} + x_{5,3} + x_{1,6} + x_{1,7} + x_{1,8} + x_{1,9} + x_{1,10} + 2x_{1,11} \leq 3w_3,$$

y, por Teorema 5.4.6, define faceta de $\mathcal{ECP}$ *dado que* $\alpha(S) \leq \lceil \frac{11}{3} \rceil - 1$ *y existen los siguientes coloreos: un* $(\lceil \frac{11}{2} \rceil - 1)$*-eqcol tal que* $|C_3 \cap S| = 3$*, un coloreo equitativo tal que* $|C_3 \cap S| = 3$ *y* $(C_3 \cap N(1)) \backslash S = \{2\}$*, y finalmente un coloreo equitativo tal que* $|C_3 \cap S| = 3$ *y* $(C_3 \cap N(1)) \backslash S = \{6\}$*. Ellos se muestran en la Figura 5.2 (b), (c) y (d) respectivamente.*

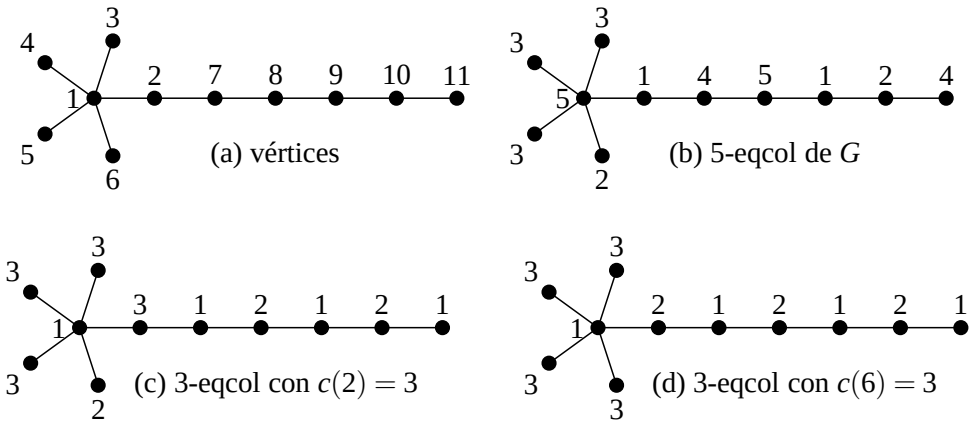


Figura 5.2: Grafo y coloreos del Ejemplo 5.4.8

5.4.2. Desigualdades *fuera-vecindad*

La siguiente familia de desigualdades surge de examinar la interacción entre un vértice u y los vértices que están fuera de su vecindario. Sea $j \geq \chi_{eq}$ y sea (x,w) un j-eqcol. Entonces se puede ver que (x,w) verifica la desigualdad

$$\left(\left\lfloor \frac{n}{j} \right\rfloor - 1 \right) x_{uj} - \sum_{v \in V \backslash N[u]} x_{vj} \leq 0.$$

Efectivamente, si $x_{uj} = 0$, la desigualdad se verifica claramente y si $x_{uj} = 1$, la clase de color j no puede tener menos de $\lfloor n/j \rfloor - 1$ vértices de $V \backslash N[u]$ pues el vecindario de u no puede pintarse de color j. Luego el lado izquierdo es cero o un valor negativo.

Consideremos ahora $j < \chi_{eq}$ y (x,w) un χ_{eq}-eqcol. También se puede ver que (x,w) satisface

$$\left(\left\lfloor \frac{n}{\chi_{eq}} \right\rfloor - 1 \right) x_{uj} - \sum_{v \in V \backslash N[u]} x_{vj} \leq 0.$$

Las desigualdades presentadas en la definición siguiente son una generalización de las mencionadas anteriormente.

Definición 5.4.9. *Dados* $j \leq \lfloor n/2 \rfloor$ *y* $u \in V$ *tal que* $N(u)$ *no es una clique, la desigualdad* (u,j)-fuera-vecindad *se define como*

$$\left(\left\lfloor \frac{n}{t_j} \right\rfloor - 1 \right) x_{uj} - \sum_{v \in V \backslash N[u]} x_{vj} + \sum_{k=t_j+1}^{n} b_{jk} x_{uk} \leq \sum_{k=t_j+1}^{n} b_{jk}(w_k - w_{k+1}), \tag{5.9}$$

donde $t_j = \max\{j, \chi_{eq}\}$ *y* $b_{jk} = \lfloor n/t_j \rfloor - \lfloor n/k \rfloor$.

Lema 5.4.10. *La desigualdad (u,j)-fuera-vecindad es válida en $\mathcal{ECP}$.*

Demostración. Sea (x,w) un r-eqcol de G. Si $r<j$, ambos lados de (5.9) son iguales a cero. Asumamos entonces que $r \geq j$ y denotemos con C_j a la clase de color j de (x,w).

Caso $r = t_j$. Los términos $\sum_{k=t_j+1}^{n} b_{jk}x_{uk}$ y $\sum_{k=t_j+1}^{n} b_{jk}(w_k - w_{k+1})$ desaparecen de la desigualdad así que sólo necesitamos chequear que $(\lfloor n/t_j \rfloor - 1)x_{uj} - \sum_{v\in V\setminus N[u]} x_{vj}$ es un valor no positivo. Si $x_{uj}=0$, claramente la desigualdad vale. Si en cambio $x_{uj}=1$,

$$\sum_{v\in V\setminus N[u]} x_{vj} = |C_j \setminus N[u]| \geq \lfloor n/t_j \rfloor - 1$$

y (5.9) se verifica.

Caso $r > t_j$. Solamente necesitamos chquear que el lado izquierdo de (5.9) sea a lo sumo b_{jr}. Si $x_{uj}=0$ entonces $\sum_{k=t_j+1}^{n} b_{jk}x_{uk} \leq \text{máx}\{b_{jk} : t_j+1 \leq k \leq r\} = b_{jr}$ y la desigualdad vale. Si en cambio $x_{uj}=1$, $\sum_{k=t_j+1}^{n} b_{jk}x_{uk} = 0$ y

$$\sum_{v\in V\setminus N[u]} x_{vj} = |C_j \setminus N[u]| \geq \lfloor n/r \rfloor - 1$$

y (5.9) se verifica nuevamente. □

Debido a que no tiene sentido considerar desigualdades fuera-vecindad sin la variable x_{uj} (i.e. el coeficiente $\lfloor n/t_j \rfloor - 1$ es cero), vamos a suponer que el número cromático equitativo de G es a lo sumo $\lfloor n/2 \rfloor$.

Para poder estudiar las caras de $\mathcal{ECP}$ definidas por desigualdades fuera-vecindad, caractericemos primero los coloreos equitativos que pertenecen a aquellas caras.

Observación 5.4.11. *Sea F la cara de $\mathcal{ECP}$ definida por la desigualdad (u,j)-fuera-vecindad y c un r-eqcol. Notemos que si $r<j$, c siempre cae en F. Para el caso $r \geq j$, sea C_j la clase de color j de c. Entonces c cae en F si y sólo si se satisfacen las siguientes condiciones:*

- *Si $c(u) = j$ entonces $|C_j| = \lfloor n/r \rfloor$.*
- *Si $c(u) \neq j$ entonces*
  - *$C_j \subset N(u)$ y además*
  - *si $\left\lfloor \frac{n}{r} \right\rfloor < \left\lfloor \frac{n}{\text{máx}\{j,\chi_{eq}\}} \right\rfloor$ entonces $c(u) \geq \left\lfloor \frac{n}{\lfloor n/r \rfloor + 1} \right\rfloor + 1$.*

Al igual que las desigualdades subvecindad, las fuera-vecindad definen caras de alta dimensión:

Teorema 5.4.12. *Sea F la cara definida por la desigualdad (u,j)-fuera-vecindad. Entonces:*

$$dim(F) \geq dim(\mathcal{ECP}) - \big(3n - \lceil n/2 \rceil - |\mathcal{S}| - \chi_{eq} - 4 - \delta(u)\big).$$

Demostración. Sea $v_1 \in V\setminus N[u]$, $v_2, v_3 \in N(u)$ tales que v_2 no es adyacente a v_3 y sea $1 \leq r \leq \lfloor n/2 \rfloor$ tal que $r \neq j$. A continuación proponemos $n^2 + \lceil n/2 \rceil - 3n + 4 + \delta(u)$ coloreos equitativos afínmente independientes que yacen en F:

- Un n-eqcol c tal que $c(u)=j$, $c(v_1)=n$, $c(v_2)=n-1$ y $c(v_3)=r$.
- $swap_{n,j_1,j_2}(c)$ para cada $j_1, j_2 \in \{1,\ldots,n-1\}\setminus\{j\}$ tales que $j_1 \neq j_2$. Con esto formamos $(n-2)(n-3)$ coloreos.

- $swap_{n,j,c(v)}(c)$ para cada $v \in N(u)$. Aquí tenemos $\delta(u)$ coloreos más.
- $swap_{j,r,j'}(c)$ para cada $j' \in \{\lfloor n/2 \rfloor + 1, \ldots, n-1\}$. Tenemos $\lceil n/2 \rceil - 1$ coloreos más.
- $swap_{n,j'}(c)$ para cada $j' \in \{1, \ldots, n-1\} \backslash \{j\}$. De esta manera obtenemos $n-2$ coloreos más.
- El $(n-1)$-eqcol c' tal que $c'(v_1) = r$, $c'(v_3) = n-1$ y $c'(i) = c(i) \ \ \forall\, i \in V \backslash \{v_1, v_3\}$.
- $swap_{j',n-1}(c')$ para cada $j' \in \{1, \ldots, n-2\}$. Se generan $n-2$ coloreos más.
- Un $(n-2)$-eqcol c'' tal que $c''(v_1) = c''(u) = n-2$ y $c''(v_2) = c''(v_3) = j$.

La independencia afín de estos coloreos puede ser verificada de manera análoga a la propuesta en el Lema 5.1.4. □

La cota inferior de $dim(F)$ presentada en el teorema anterior puede mejorarse un poco cuando $j > \chi_{eq}$:

Corolario 5.4.13. *Sea F la cara definida por la desigualdad (u,j)-fuera-vecindad. Si $\chi_{eq} \leq \lfloor n/2 \rfloor$ y $j > \chi_{eq}$ entonces:*

$$dim(F) \geq dim(\mathcal{ECP}) - \big(3n - \lceil n/2 \rceil - |\mathscr{S}| - 2\chi_{eq} - 4 - \delta(u)\big).$$

Demostración. Notemos que el vértice u solo recibe colores de $\{\lfloor n/2 \rfloor + 1, \ldots, n\} \cup \{j\}$ en el conjunto de coloreos propuestos en el teorema anterior. Entonces, si $\chi_{eq} \leq \lfloor n/2 \rfloor$ y $j > \chi_{eq}$, es posible incorporar un χ_{eq}-eqcol c_k de G tal que $c_k(u) = k$ para cada $k \in \{1, \ldots, \chi_{eq}\}$. Así, sumamos χ_{eq} coloreos al conjunto de coloreos del teorema anterior, y todos los coloreos del conjunto resultante son afínmente independientes. □

La cota de $dim(F)$ dada por el resultado anterior es ajustada en algunos casos. Por ejemplo, consideremos la desigualdad $(1,3)$-fuera-vecindad en el grafo $K_{3,3}$. La dimensión de la cara que define esta desigualdad es $dim(\mathcal{ECP}) - 3 = 29$, exactamente el mismo valor que propone el Corolario 5.4.13.

En el Capítulo 6 presentaremos una rutina de separación para las desigualdades fuera-vecindad que está basada en la siguiente condición necesaria para que las mismas definan faceta de $\mathcal{ECP}$.

Teorema 5.4.14. *Si la desigualdad (u,j)-fuera-vecindad define una faceta de $\mathcal{ECP}$ entonces*

$$\alpha(N(u)) \geq \left\lfloor \frac{n}{\text{máx}\{j, \chi_{eq}\}} \right\rfloor.$$

Demostración. Sea $t_j = \text{máx}\{j, \chi_{eq}\}$ y F la cara de $\mathcal{ECP}$ definida por la (u,j)-fuera-vecindad. Supongamos que $\alpha(N(u)) < \lfloor n/t_j \rfloor$. Vamos a probar que todo coloreo equitativo perteneciente a F satisface la igualdad:

$$\sum_{l=1}^{j-1} x_{ul} + w_j = 1 \tag{5.10}$$

Dado que esta igualdad no puede ser obtenida como combinación lineal del sistema minimal de $\mathcal{ECP}$ y la *igualdad* (u,j)-fuera-vecindad, F no es una faceta de $\mathcal{ECP}$.
Sea c un r-eqcol que cae en F. Claramente, si $r < j$ entonces $w_j = 0$ y $c(u) = l$ para algún $l \leq j-1$. Luego, se verifica (5.10). Si en cambio $r \geq j$, $w_j = 1$ y tenemos que probar que $\sum_{l=1}^{j-1} x_{ul} = 0$, o equivalentemente, que $c(u) \geq j$. Según la Observación 5.4.11, si $c(u) \neq j$ entonces $C_j \subset N(u)$ que implica que $\alpha(N(u)) \geq |C_j|$. A su vez este hecho implica que $\lfloor n/r \rfloor < \lfloor n/t_j \rfloor$ (si $\lfloor n/r \rfloor = \lfloor n/t_j \rfloor$ entonces $|C_j| \geq \lfloor n/t_j \rfloor$ y esto contradice la premisa $\alpha(N(u)) < \lfloor n/t_j \rfloor$). Entonces, por la Observación 5.4.11,

$$c(u) \geq \left\lfloor \frac{n}{\lfloor n/r \rfloor + 1} \right\rfloor + 1 > j,$$

y nuevamente se verifica 5.10. □

De la misma forma que con las desigualdades subvecindad, si $j < \chi_{eq}$ las desigualdades (u,j)-fuera-vecindad definen faceta si y sólo si la desigualdad (u,χ_{eq})-fuera-vecindad también define faceta:

Teorema 5.4.15. *Sea $j < \chi_{eq}$, F_j la cara definida por la desigualdad (u,j)-fuera-vecindad y $F_{\chi_{eq}}$ la cara definida por la desigualdad (u,χ_{eq})-fuera-vecindad. Entonces, $dim(F_j) = dim(F_{\chi_{eq}})$.*

El resultado anterior se puede demostrar siguiendo los mismos lineamientos presentes en la prueba del Teorema 5.4.5.

Ahora presentaremos condiciones suficientes para que las desigualdades fuera-vecindad definan facetas de $\mathcal{ECP}$, sólo considerando los casos donde $j \geq \chi_{eq}$. Su prueba se encuentra en el Capítulo 9.

Teorema 5.4.16. *Sea G un grafo monótono, $u \in V$ tal que $N(u)$ no es una clique y $\chi_{eq} \leq j \leq \lfloor n/2 \rfloor$. Si*

(i) existe un vértice $\hat{v} \in V \backslash N[u]$ que no es universal en $G-u$,

(ii) si n es impar, el complemento de $G-u$ tiene un matching perfecto,

(iii) para todo $v \in V \backslash N[u]$, ocurre lo siguiente:

- *si n es par, el complemento de $G-\{u,v\}$ tiene un matching perfecto,*
- *si n es impar, existe un conjunto estable $H_v \subset V \backslash \{u,v\}$ de tamaño 3 en G tal que el complemento de $G-(H_v \cup \{u,v\})$ tiene un matching perfecto,*

(iv) para todo r tal que $j \leq r \leq \lfloor n/2 \rfloor$, ocurre lo siguiente:

- *Si $\lfloor \frac{n}{r} \rfloor > \lfloor \frac{n}{r+1} \rfloor$ entonces existen dos r-eqcols cuya clase de color C_j satisface $C_j \subset N(u)$ en el primero de ellos y $u \in C_j$, $|C_j| = \lfloor n/r \rfloor$ en el segundo,*
- *Si $\lfloor \frac{n}{r} \rfloor = \lfloor \frac{n}{r+1} \rfloor$ entonces existe un r-eqcol que satisface las condiciones dadas en la Observación 5.4.11, i.e. que cae en la cara definida por la desigualdad,*

entonces la desigualdad (u,j)-fuera-vecindad define una faceta de $\mathcal{ECP}$.

A continuación exponemos ejemplos en donde se aplican los Teoremas 5.4.15 y 5.4.16.

Ejemplo 5.4.17. *Sea G el grafo introducido en la Figura 5.2(a). Recordemos que G es monótono y $\chi_{eq}(G) = 3$. Consideremos la desigualdad (u,j)-fuera-vecindad en los siguientes casos:*

1. *Si $u = 1$ y $j = 3$, la desigualdad es*

$$2x_{1,3} - \sum_{v=7}^{11} x_{v,3} + x_{1,4} + x_{1,5} + 2x_{1,6} + 2x_{1,7} + 2x_{1,8} + 2x_{1,9} + 2x_{1,10} + 2x_{1,11} \leq w_4 + w_6.$$

   No es difícil de ver que las premisas del Teorema 5.4.16 se satisfacen. Abajo mostramos algunos ejemplos de coloreos requeridos por la hipótesis (iv) de este teorema: en la Figura 5.3(a) hay un 5-eqcol de G tal que $C_3 \subset N(1)$, en la Figura 5.3(b) hay un 5-eqcol de G tal que $1 \in C_3$ y $|C_3| = 2$, en la Figura 5.3(c) hay un 3-eqcol de G tal que $C_3 \subset N(1)$ y, finalmente, en la Figura 5.3(d) hay un 3-eqcol de G tal que $1 \in C_3$ y $|C_3| = 3$. Luego, esta desigualdad define una faceta de $\mathcal{ECP}$.

2. *Si $u = 1$ y $j \in \{1,2\}$, el Teorema 5.4.15 nos permite deducir que la desigualdad $(1,j)$-fuera-vecindad también define faceta de $\mathcal{ECP}$.*

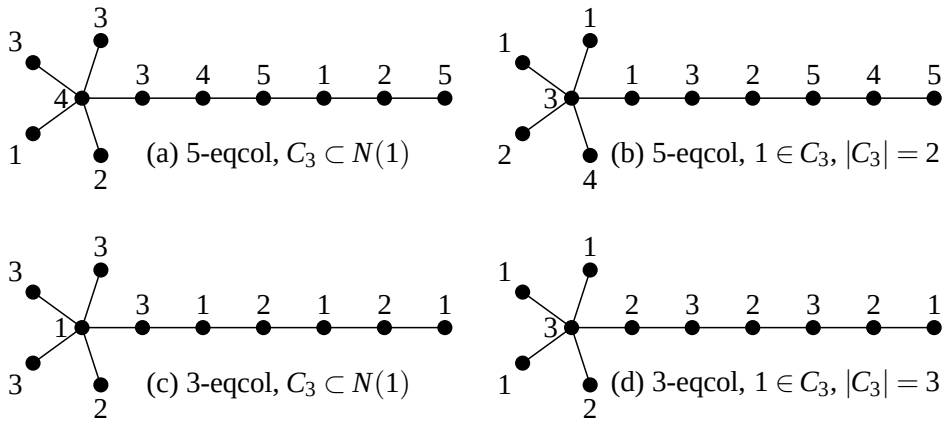


Figura 5.3: Coloreos del Ejemplo 5.4.17

5.4.3. Desigualdades *clique-vecindad*

Consideremos un vértice $u \in V$, un conjunto $S \subset V \setminus N[u]$ y números r, k, j tales que $k = \lceil n/r \rceil$ y $j \leq r$. Si (x,w) es un r-eqcol entonces el tamaño máximo de cualquier clase de color en (x,w) es k y la siguiente desigualdad

$$\big(k - \alpha(S)\big)x_{uj} + \sum_{v \in N(u)} x_{vj} + \sum_{v \in S} x_{vj} \leq k \tag{5.11}$$

es satisfecha por (x,w). En efecto, si $x_{uj} = 0$ entonces la desigualdad se cumple porque $|C_j| \leq k$. Si $x_{uj} = 1$, $\sum_{v \in N(u)} x_{vj} = 0$ y (5.11) se convierte en la desigualdad de rango $\sum_{v \in S} x_{vj} \leq \alpha(S)$, que también se cumple para (x,w).

La desigualdad planteada no depende de r. Entonces, para asegurar la existencia de r a partir de un k y j dados, podemos elegir cualquier k que satisfaga:

$$k = \left\lceil \frac{n}{\left\lceil \frac{n}{k-1} \right\rceil - 1} \right\rceil, \tag{5.12}$$

Luego, $r = \left\lceil \frac{n}{k-1} \right\rceil - 1$ y sólo resta verificar que $j \leq r$.

Si nos limitamos al caso en que S es una clique, la idea planteada anteriormente puede aprovecharse para obtener desigualdades que sean válidas en $\mathcal{ECP}$:

Definición 5.4.18. *Dados $u \in V$, una clique Q de G tal que $Q \cap N[u] = \varnothing$ y números j, k tales que se satisfaga (5.12), $3 \leq k \leq \alpha(N(u)) + 1$ y $1 \leq j \leq \left\lceil \frac{n}{k-1} \right\rceil - 1$, la desigualdad* (u,j,k,Q)-clique-vecindad *se define como*

$$(k-1)x_{uj} + \sum_{l=\lceil \frac{n}{k-1} \rceil}^{n-2} \left(k - \left\lceil \frac{n}{l} \right\rceil\right) x_{ul} + (k-1)\big(x_{un-1} + x_{un}\big) + \sum_{v \in N(u) \cup Q} x_{vj} + \sum_{v \in V \setminus \{u\}} (x_{vn-1} + x_{vn}) \leq \sum_{l=j}^{n} b_{ul}(w_l - w_{l+1}), \tag{5.13}$$

donde

$$b_{ul} = \begin{cases} \min\{\lceil n/l \rceil, \alpha(N(u)) + 1\}, & \text{si } j \leq l \leq \lceil n/k \rceil - 1 \\ k, & \text{si } \lceil n/k \rceil \leq l \leq n-2 \\ k+1, & \text{si } l \geq n-1 \end{cases}$$

Lema 5.4.19. *La desigualdad (u,j,k,Q)-clique-vecindad es válida en $\mathcal{ECP}$.*

Demostración. Sea (x,w) un r-eqcol of G. Si $r<j$, ambos lados de (5.13) son cero. Entonces asumamos que $r\geq j$ y observemos que el lado derecho de (5.13) siempre vale b_{ur}. Sean C_j, C_{n-1} y C_n las clases de color j, $n-1$ y n de (x,w) respectivamente.

Caso $r\leq\lceil n/k\rceil-1$. Hay que probar que (x,w) verifica

$$(k-1)x_{uj}+\sum_{v\in N(u)\cup Q}x_{vj}\leq b_{ur}=\min\left\{\left\lceil\frac{n}{r}\right\rceil,\alpha(N(u))+1\right\}.$$

Si $x_{uj}=1$, $\sum_{v\in N(u)}x_{vj}=0$ y $\sum_{v\in Q}x_{vj}\leq 1$. Debido a que $b_{ur}\geq k$, la desigualdad es válida. Si en cambio $x_{uj}=0$, $\sum_{v\in N(u)\cup Q}x_{vj}=|C_j\cap(N(u)\cup Q)|\leq\min\{\lceil n/r\rceil,\alpha(N(u)\cup Q)\}\leq\min\{\lceil n/r\rceil,\alpha(N(u))+1\}$.

Caso $\lceil n/k\rceil\leq r\leq n-2$. Hay que probar que (x,w) verifica

$$(k-1)x_{uj}+\sum_{l=\lceil\frac{n}{k-1}\rceil}^{n-2}\left(k-\left\lceil\frac{n}{l}\right\rceil\right)x_{ul}+\sum_{v\in N(u)\cup Q}x_{vj}\leq k.$$

Si $x_{uj}=1$, $\sum_{l=\lceil\frac{n}{k-1}\rceil}^{n-2}(k-\lceil n/l\rceil)x_{ul}=0$ y $\sum_{v\in N(u)\cup Q}x_{vj}\leq 1$. Por lo tanto la desigualdad es válida.
Si en cambio $x_{uj}=0$, $\sum_{l=\lceil\frac{n}{k-1}\rceil}^{n-2}(k-\lceil n/l\rceil)x_{ul}\leq k-\lceil n/r\rceil$ y $\sum_{v\in N(u)\cup Q}x_{vj}\leq|C_j|\leq\lceil n/r\rceil$ y la desigualdad es válida.

Caso $r\geq n-1$. Notemos primero que siempre se satisface $|C_j|+|C_{n-1}|+|C_n|\leq 3$. Al igual que en los casos anteriores, hay que probar que (x,w) satisface

$$L(x)+\sum_{v\in N(u)\cup Q}x_{vj}+\sum_{v\in V\setminus\{u\}}(x_{vn-1}+x_{vn})\leq k+1.$$

donde

$$L(x)=(k-1)x_{uj}+\sum_{l=\lceil\frac{n}{k-1}\rceil}^{n-2}\left(k-\left\lceil\frac{n}{l}\right\rceil\right)x_{ul}+(k-1)\big(x_{un-1}+x_{un}\big).$$

Observemos que $L(x)\leq k-1$ y, en particular, $L(x)=k-1$ si y sólo si $u\in C_j\cup C_{n-1}\cup C_n$. Entonces si $L(x)=k-1$, $u\in C_j\cup C_{n-1}\cup C_n$ y tenemos que $\sum_{v\in N(u)\cup Q}x_{vj}+\sum_{v\in V\setminus\{u\}}(x_{vn-1}+x_{vn})\leq|C_j|+|C_{n-1}|+|C_n|-1\leq 2$, y la desigualdad es válida.
Si, en cambio, $L(x)\leq k-2$, la desigualdad es válida pues $\sum_{v\in N(u)\cup Q}x_{vj}+\sum_{v\in V\setminus\{u\}}(x_{vn-1}+x_{vn})\leq|C_j|+|C_{n-1}|+|C_n|\leq 3$. □

Hay que destacar que si Q no fuese maximal en $G-N[u]$, la desigualdad (u,j,k,Q)-clique-vecindad estaría dominada por una (u,j,k,Q')-clique-vecindad, con una clique Q' tal que $Q\subsetneq Q'\subset G-N[u]$.

En la Definición 5.4.18 siempre consideramos k tales que $k\geq 3$. En realidad si k fuese 2, la desigualdad (u,j,k,Q)-clique-vecindad estaría dominada por otra que veremos en la siguiente sección.

Nos proponemos analizar las caras de $\mathcal{ECP}$ definidas por desigualdades clique-vecindad. Para ello es necesario caracterizar aquellos coloreos que pertenezcan a dichas caras:

Observación 5.4.20. *Sea F la cara de $\mathcal{ECP}$ definida por la desigualdad (u,j,k,Q)-clique-vecindad y c un r-eqcol. Si $r<j$ entonces c siempre yace en F. En cambio, si $r\geq j$, denotemos con C_j, C_{n-1} y C_n a las clases de color j, $n-1$ y n de c respectivamente. El coloreo c yace en F si y sólo si se satisfacen las siguientes condiciones:*

- *Si* $r \leq \lceil n/k \rceil - 1$ *entonces:*
  - *Si* $c(u) = j$ *entonces* $|C_j \cap Q| = 1$ *y* $k = \alpha(N(u)) + 1$.
    Si no, $|C_j \cap (N(u) \cup Q)| = \min\{\lceil n/r \rceil, \alpha(N(u)) + 1\}$.
- *Si* $\lceil n/k \rceil \leq r \leq n-2$ *entonces:*
  - *Si* $c(u) = j$ *entonces* $|C_j \cap Q| = 1$. *Si no,*
    - $|C_j \cap (N(u) \cup Q)| = \lceil n/r \rceil$ *y*
    - *si* $r \geq \left\lceil \frac{n}{k-1} \right\rceil$ *entonces* $c(u) \geq \left\lceil \frac{n}{\lceil n/r \rceil} \right\rceil$.
- *Si* $r \geq n-1$ *entonces:*
  - *Si* $c(u) \in \{j, n-1, n\}$ *entonces* $|C_j \cap Q| + |C_{n-1} \backslash \{u\}| + |C_n \backslash \{u\}| = 2$.
    Si no, $c(u) \geq \lceil n/2 \rceil$ *y* $|C_j \cap (N(u) \cup Q)| + |C_{n-1}| + |C_n| = 3$.

Al igual que las fuera-vecindad, las desigualdades clique-vecindad definen caras de dimensión alta:

Teorema 5.4.21. *Sea* F *la cara definida por la desigualdad* (u, j, k, Q)*-clique-vecindad. Entonces*

$$dim(F) \geq dim(\mathcal{ECP}) - \big(3n - |\mathscr{S}| - \chi_{eq} - \lfloor n/2 \rfloor - \delta(u) - |Q| - 4\big).$$

Demostración. Sean $v_1, v_2 \in N(u)$ vértices no adyacentes, y $q \in Q$. Abajo proponemos $n^2 + \lfloor n/2 \rfloor + 4 - 3n + \delta(u) + |Q|$ coloreos equitativos afínmente independientes que yacen en F:

- Un n-eqcol c tal que $c(u) = j$ y $c(q) = n$.
- $swap_{n,j_1,j_2}(c)$ para cada $j_1, j_2 \in \{1, \ldots, n-1\} \backslash \{j\}$ tales que $j_1 \neq j_2$. Tenemos $(n-2)(n-3)$ coloreos.
- $swap_{n,j,c(v)}(c)$ para cada $v \in (N(u) \cup Q) \backslash \{q\}$. Tenemos $\delta(u) + |Q| - 1$ coloreos más.
- $swap_{j',j,n}(c)$ para cada $j' \in \{\lceil n/2 \rceil, \ldots, n-1\}$. Tenemos $\lfloor n/2 \rfloor$ coloreos más.
- $swap_{j',n}(c)$ para cada $j' \in \{1, \ldots, n-1\}$. Tenemos $n-1$ coloreos más.
- Un $(n-1)$-eqcol c' tal que $c'(u) = j$ y $c'(v_1) = c'(v_2) = n-1$.
- $swap_{j,n-1}(c')$.
- Un $(n-2)$-eqcol c'' tal que $c''(u) = c''(q) = j$ y $c''(v_1) = c''(v_2) = n-2$.
- $swap_{j',n-2}(c'')$ para cada $j' \in \{1, \ldots, n-3\} \backslash \{j\}$. Tenemos $n-4$ coloreos más.

La independencia afín de estos coloreos puede ser verificada de manera análoga a la propuesta en el Lema 5.1.4. □

De la misma forma que el Corolario 5.4.13 se puede probar el siguiente:

Corolario 5.4.22. *Sea* F *la cara definida por la desigualdad* (u, j, k, Q)*-clique-vecindad. Si* $\chi_{eq} \leq \lceil n/2 \rceil - 1$ *y* $j > \chi_{eq}$ *entonces:*

$$dim(F) \geq dim(\mathcal{ECP}) - \big(3n - |\mathscr{S}| - 2\chi_{eq} - \lfloor n/2 \rfloor - \delta(u) - |Q| - 4\big).$$

La cota de $dim(F)$ que provee el Teorema 5.4.21 es ajustada en algunos casos, como el siguiente. Sea G un grafo de 5 vértices cuyo conjunto de aristas es $E = \{(1,2),(1,3),(2,3),(1,4),(4,5)\}$. La dimensión de la cara definida por la $(1,1,2,\{5\})$-clique-vecindad es $dim(\mathcal{ECP}) - 2 = 19$ que coincide con la cota que brinda el teorema.

El siguiente resultado da condiciones suficientes para que esta desigualdad defina faceta de $\mathcal{ECP}$, y cuya prueba se detalla en el Capítulo 9.

Teorema 5.4.23. *Sea G un grafo monótono. Dados $u \in V$, Q una clique de G tal que $Q \cap N[u] = \varnothing$ y números j,k que verifican $3 \leq k \leq \min\{\alpha(N(u))+1, \lceil n/\chi_{eq} \rceil\}$, $1 \leq j \leq \left\lceil \frac{n}{k-1} \right\rceil - 1$ y (5.12), si*

(i) para todo $v \in V \setminus (N[u] \cup Q)$, existen $\lceil n/3 \rceil \leq r \leq \lceil n/2 \rceil - 1$, $q_1, q_2 \in V$ y dos r-eqcols tales que, en uno de ellos $C_j = \{u, v, q_1\}$ y en el otro $C_j = \{u, q_2\}$ (aquí, q_1 y q_2 pueden tratarse del mismo vértice),

(ii) para todo t tal que $max\{j, \chi_{eq}\} \leq t \leq n-3$, tenemos que:

- *si $\lceil \frac{n}{t} \rceil > \lceil \frac{n}{t+1} \rceil$ entonces existen $q \in Q$ y un t-eqcol tales que $C_j \subset N(u)$, $|C_j| = \lceil n/t \rceil$ y $u, q \in C_t$,*
- *si $\lceil \frac{n}{t} \rceil = \lceil \frac{n}{t+1} \rceil$ enonces existe un t-eqcol que satisface las condiciones dadas en la Observación 5.4.20, i.e. que cae en la cara definida por (5.13),*

entonces la desigualdad (u, j, k, Q)-clique-vecindad define una faceta de $\mathcal{ECP}$.

Sería lógico presentar condiciones suficientes (para que la desigualdad clique-vecindad definiese faceta de $\mathcal{ECP}$) que no involucren la existencia de coloreos equitativos específicos, pues es claro que hallarlos es tan difícil como el mismo PCEG. Si bien no fuimos capaces de modificar la hipótesis (ii) del Teorema 5.4.23 sin tener que restringir drásticamente la familia de grafos en las que se aplican, si pudimos relajar la hipótesis (i) para no requerir la existencia de coloreos:

Corolario 5.4.24. *Sea G un grafo monótono y u, j, k, Q definidas como en el Teorema 5.4.23. Si la hipótesis (ii) del Teorema 5.4.23 se satisface y para todo $v \in V \setminus (N[u] \cup Q)$ tenemos que:*

- *en caso de ser n impar,*
  - *existe un vértice $q_1 \in Q$ y un conjunto estable $H_v^1 = \{u, v, q_1\}$ tal que el complemento de $G - H_v^1$ tiene un matching perfecto M_v,*
  - *existe un vértice $q_2 \in Q$ y dos conjuntos estables disjuntos $H_v^2 = \{u, q_2\}$, H_v^3 tales que $|H_v^3| = 3$ y el complemento de $G - (H_v^2 \cup H_v^3)$ tiene un matching perfecto M_v',*
- *en caso de ser n par,*
  - *existe un vértice $q_1 \in Q$ y dos conjuntos estables disjuntos $H_v^1 = \{u, v, q_1\}$, H_v^2 tales que $|H_v^2| = 3$ y el complemento de $G - (H_v^1 \cup H_v^2)$ tiene un matching perfecto M_v,*
  - *existe un vértice $q_2 \in Q$ y tres conjuntos estables disjuntos $H_v^3 = \{u, q_2\}$, H_v^4, H_v^5 tales que $|H_v^4| = |H_v^5| = 3$ y el complemento de $G - (H_v^3 \cup H_v^4 \cup H_v^5)$ tiene un matching perfecto M_v',*

entonces la desigualdad (u, j, k, Q)-clique-vecindad define faceta de $\mathcal{ECP}$.

Demostración. Consideremos primero el caso con n impar. Sea $v \in V \setminus (N[u] \cup Q)$ y sean M_v, M_v', H_v^1, H_v^2 y H_v^3 los matchings y los conjuntos estables dados por hipótesis. Por un lado, consideremos un $(\lceil n/2 \rceil - 1)$-eqcol tal que la clase de color j es H_v^1 y el resto de las clases de color se conforman con los extremos de las aristas de M_v, y por el otro, consideremos un $(\lceil n/2 \rceil - 1)$-eqcol tal que la clase de color j es H_v^2 y el resto

de las clases de color se conforman con H_v^3 y los extremos de las aristas de M'_v. Por lo tanto la hipótesis (i) del Teorema 5.4.23 se satisface y la desigualdad (u, j, k, Q)-clique-vecindad define faceta de $\mathcal{ECP}$.

La prueba para el caso con n par es análoga a la dada anteriormente. □

A continuación presentamos un ejemplo en donde se aplica el resultado anterior.

Ejemplo 5.4.25. *Sea G el grafo introducido en la Figura 5.2(a). Recordemos que G es monótono y $\chi_{eq}(G) = 3$. Vamos a aplicar el Corolario 5.4.24 considerando la desigualdad (u, j, k, Q)-clique-vecindad con $u = 1$, $j = 1$, $k = 4$ y $Q = \{7, 8\}$, i.e.*

$$3x_{1,1} + \sum_{v=2}^{8} x_{v,1} + x_{1,4} + x_{1,5} + 2x_{1,6} + 2x_{1,7} + 2x_{1,8} + 2x_{1,9} + 3x_{1,10} + 3x_{1,11} + \sum_{v=2}^{11} x_{v,10} + \sum_{v=2}^{11} x_{v,11} \leq 6w_1 - 2w_3 + w_{10}.$$

Se puede comprobar que las premisas del corolario son satisfechas. Abajo, presentamos algunos ejemplos de coloreos relacionados a la hipótesis (ii) del Teorema 5.4.23. La Figura 5.4(a) muestra un 3-eqcol de G tal que $1, 7 \in C_3$, $C_1 \subset N(1)$, $|C_1| = 4$ mientras que la Figura 5.4(b) muestra un 5-eqcol de G tal que $1, 7 \in C_5$, $C_1 \subset N(1)$, $|C_1| = 3$.

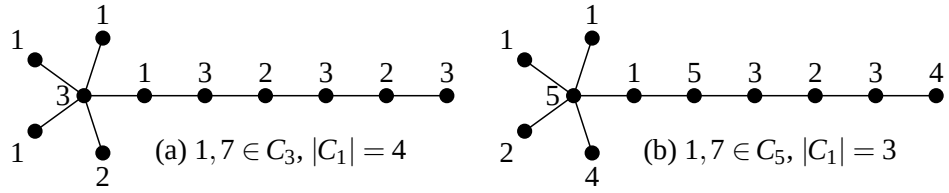


Figura 5.4: Coloreos del Ejemplo 5.4.25

5.4.4. Desigualdades J-color

Consideremos un conjunto de colores $J \subset \{1, \ldots, n\}$. Nos preguntamos cuál es el máximo valor que puede alcanzar la suma de los cardinales de todas las clases de color pertenecientes a J en un k-eqcol. Sea d_{Jk} la cantidad de colores *disponibles* de J, i.e. $d_{Jk} = |J \cap \{1, \ldots, k\}|$. Como ya vimos en la Observación 1.5.1, en un k-eqcol hay $n \bmod k = n - k\lfloor \frac{n}{k} \rfloor$ clases de tamaño $\lfloor \frac{n}{k} \rfloor + 1$ y $k - (n \bmod k)$ clases de tamaño $\lfloor \frac{n}{k} \rfloor$. Entonces el número de clases de color en J que tienen tamaño $\lfloor \frac{n}{k} \rfloor + 1$ es a lo sumo $\min\{d_{Jk}, n \bmod k\}$.

Si denotamos con $b_{Jk} = d_{Jk}\lfloor \frac{n}{k} \rfloor + \min\{d_{Jk}, n \bmod k\}$ entonces tenemos que $\sum_{j \in J} |C_j| \leq b_{Jk}$, lo que motiva a la siguiente definición.

Definición 5.4.26. *Sea $J \subset \{1, \ldots, n\}$ y, para todo $1 \leq k \leq n$, $d_{Jk} = |J \cap \{1, \ldots, k\}|$. La desigualdad J-color es definida como*

$$\sum_{j \in J} \sum_{v \in V} x_{vj} \leq \sum_{k=1}^{n} b_{Jk}(w_k - w_{k+1}), \tag{5.14}$$

donde

$$b_{Jk} = d_{Jk} \left\lfloor \frac{n}{k} \right\rfloor + \min\left\{ d_{Jk}, n - k \left\lfloor \frac{n}{k} \right\rfloor \right\}.$$

Lema 5.4.27. *La desigualdad J-color es válida para $\mathcal{ECP}$.*

Demostración. Sea (x, w) un r-eqcol. Si $r < j$, ambos lados de (5.14) son cero. Si en cambio $r \geq j$, el lado derecho de (5.14) es b_{Jr}, que a su vez es una cota superior de $\sum_{j \in J} |C_j| = \sum_{j \in J} \sum_{v \in V} x_{vj}$. □

Observación 5.4.28. *A continuación presentamos la relación entre las desigualdades J-color y las otras desigualdades válidas para $\mathcal{ECP}$ definidas anteriormente.*

1. *Sea $1 \leq j \leq n-1$. Las desigualdades J-color con $J = \{j\}$ son, en realidad, las restricciones (4.23). Si $j \leq n-2$, la desigualdad $\{j\}$-color está dominada por la desigualdad $\{j, n-1\}$-color. En cambio, si $j = n-1$, define faceta de $\mathcal{ECP}$ por Teorema 5.2.5.*
   Las caras que definen las desigualdades J-color con $J = \{1,\dots,n-1\}\backslash\{j\}$ son las mismas que definen las restricciones (4.22). Según el Teorema 5.2.4, son facetas de $\mathcal{ECP}$.

2. *Las desigualdades (V, j)-rango son, o están dominadas por, desigualdades J-color con $J = \{j, n-\alpha(G)+1,\dots,n-1\}$.*

3. *Sea $S \subsetneq V$ y $j \geq \lceil n/2 \rceil$. La desigualdad (S, j)-2-rango está dominada por la J-color con $J = \{j, n-1\}$.*

4. *No es difícil de ver que una desigualdad (u, j, k, Q)-clique-vecindad con $k = 2$ está dominada por la J-color con $J = \{j, n-1\}$.*

5. *Si para todo k tal que G admite un k-eqcol, tenemos que "k divide a n" o "n mod $k \geq d_{Jk}$", entonces la desiguadad J-color puede ser obtenida mediante la suma de restricciones (4.23), $\sum_{v\in V} x_{vj} \leq \sum_{k=j}^{n} \lceil n/k \rceil (w_k - w_{k+1})$, con $j \in J$. En efecto, una desigualdad J-color es capaz de cortar soluciones fraccionarias de la relajación lineal sólo si existe $k \in \{\chi_{eq},\dots,n-1\}\backslash\mathscr{S}$ tal que $1 \leq n \bmod k \leq d_{Jk} - 1$, y no todas las componentes del vector w son enteras.*

Sea $J \subset \{1,\dots,n-1\}$. Observemos que las desigualdades J-color y $(J\cup\{n\})$-color definen la misma cara, pues la segunda se puede obtener a partir de añadir la ecuación (5.4) a la primera. Es por eso que, a partir de ahora, supondremos sin pérdida de generalidad que $n \notin J$.

Estas desigualdades también definen caras de alta dimensión:

Teorema 5.4.29. *Sea $J \subset \{1,\dots,n-1\}$ tal que $|J| \geq 1$ y sea F la cara de $\mathcal{ECP}$ definida por la desigualdad J-color. Entonces,*

$$dim(F) \geq dim(\mathcal{ECP}) - (n - |J| - 1).$$

Demostración. Sean u, u' vértices no adyacentes y c^1 un n-eqcol tal que $c^1(u) = n$ y $c^1(u') = n-1$. Consideremos las soluciones de $PROC(c^1)$ dadas en la Definicion 5.1.3 y elijamos aquellos coloreos c^6_k de forma que caigan en F (dado un k-eqcol arbitrario de G siempre es posible intercambiar sus clases de color a fin de que las clases pertenecientes a J tengan tamaños maximales). Luego excluimos del conjunto anterior aquellos $(n-1)$-eqcols que asignan colores de $\{1,\dots,n-1\}\backslash J$ a u y u' simultáneamente, i.e. c^4 si $n-1 \notin J$ y c^5_j para los $j \notin J$. Los $n^2 - \chi_{eq} - |\mathscr{S}| - n + 1 + |J|$ coloreos propuestos son afínmente independientes y pertenecen a F. □

El teorema anterior da una cota ajustada de la dimensión para los casos en que $|J| = n-2$ pues, en estos casos, la desigualdad define faceta de $\mathcal{ECP}$.

El siguiente teorema da condiciones suficientes para que las desigualdades J-color definan facetas de $\mathcal{ECP}$. La prueba del teorema se halla en el Capítulo 9.

Teorema 5.4.30. *Sea M un matching del complemento de G tal que $2 \leq |M| \leq \lfloor \frac{n-1}{2} \rfloor$ y sea $J \subset \{1,\dots,n-1\}$ tal que $|J| = 2|M| - r - 1$ con $r \in \{1,2\}$ y J contiene todos los colores mayores que $n - |M|$, i.e. $\{n-|M|+1,\dots,n-1\} \subset J$. Entonces la desigualdad J-color define faceta de $\mathcal{ECP}$.*

Abajo presentamos un ejemplo en donde se aplica el Teorema 5.4.30.

Ejemplo 5.4.31. *Asumamos que G es el grafo presentado en la Figura 5.2(a). Observemos que $\overline{G}$ tiene el matching $\{(4,5),(3,6),(1,7),(2,8),(9,11)\}$. Así que para todo $J \subset \{1,\dots,10\}$ tal que $7 \leq |J| \leq 8$ y $\{7,\dots,10\} \subset J$, las premisas del Teorema 5.4.30 se satisfacen y la desigualdad J-color define una faceta de $\mathcal{ECP}$. Más aún, dado que $\overline{G}$ tiene matchings de tamaños entre 2 y 5, la desigualdad J-color define faceta para todo J tal que $2 \leq |J| \leq 8$ y $\{11 - \lceil \frac{|J|}{2} \rceil, \dots, 10\} \subset J$.*

5.4.5. Desigualdades (S,J)-color

Si nos limitamos a considerar un subconjunto de vértices en el soporte de la desigualdad J-color surge una nueva desigualdad que, bajo determinadas condiciones, puede definir faceta de $\mathcal{ECP}$. Sean los conjuntos $S \subset V$ y $J \subset \{1,\dots,n-2\}$ tales que $|S| = |J|+1$. Denominamos (S,J)-*color* a la desigualdad

$$\sum_{j\in J}\sum_{v\in S} x_{vj} + \sum_{v\in V} x_{vn-1} \leq \sum_{k=1}^{n} b_{SJk}(w_k - w_{k+1}) \tag{5.15}$$

donde $d_{Jk} = |J \cap \{1,\dots,k\}|$ y

$$b_{SJk} = \begin{cases} \min\{|S|, |J|\alpha(S), d_{Jk}\lfloor \frac{n}{k} \rfloor + \min\{d_{Jk}, n - k\lfloor \frac{n}{k} \rfloor\}\}, & \text{si } k \leq n-2 \\ |S|+1, & \text{si } k = n-1 \\ |S|, & \text{si } k = n \end{cases}$$

Lema 5.4.32. *La desigualdad (S,J)-color es válida en $\mathcal{ECP}$.*

Demostración. Sea (x,w) un r-eqcol. Debemos probar que el lado izquierdo de (5.15) no puede superar a b_{SJr}. Si $r \leq n-2$, es claro que $\sum_{j\in J}\sum_{v\in S} x_{vj} \leq |S|$ y que $\sum_{j\in J}\sum_{v\in S} x_{vj} \leq |J|\alpha(S)$. Pero también $\sum_{j\in J}\sum_{v\in S} x_{vj} \leq d_{Jr}\lfloor \frac{n}{r} \rfloor + \min\{d_{Jr}, n - r\lfloor \frac{n}{r} \rfloor$ por lo comentado en el Lema 5.4.27. Si $r = n-1$, una clase de color de (x,w) tiene dos vértices mientras que el resto sólo tiene uno y, por lo tanto, el lado izquierdo de (5.15) puede valer hasta $|J|+2 = |S|+1$. Si $r = n$, todas las clases de color de (x,w) tienen tamaño 1 y el lado izquierdo puede valer hasta $|J|+1 = |S|$. □

En los resultados que siguen damos condiciones suficientes para que una desigualdad (S,J)-color defina faceta de $\mathcal{ECP}$. Las pruebas se encuentran en el Capítulo 9.

Teorema 5.4.33. *Sean $S \subset V$ y $J \subset \{1,\dots,n-2\}$ tales que $|S| = |J|+1$. Si*

(i) el complemento de $G[S]$ tiene un matching de tamaño 2,

(ii) $V \backslash S$ no es una clique,

(iii) para todo $r \in \{\chi_{eq},\dots,n-3\}\backslash\mathcal{S}$ existe un r-eqcol en donde b_{SJr} vértices de S utilizan colores de J.

entonces la desigualdad (S,J)-color define faceta de $\mathcal{ECP}$.

Corolario 5.4.34. *Si se verifica las hipótesis (ii) y (iii) del Teorema 5.4.33 y, además, $|J| \geq 2$ y existen vértices $u_1, u_2 \in S$ no adyacentes entre sí y tales que todo $v \in V \backslash S$ no es universal en $G[(S \cup \{v\})\backslash\{u_1,u_2\}]$, entonces la desigualdad (S,J)-color define faceta de $\mathcal{ECP}$.*

Vale aclarar que la hipótesis (iii) del Teorema 5.4.33 también es una condición necesaria para que la desigualdad (5.15) defina faceta de $\mathcal{ECP}$. En efecto, si existiese un $r \in \{\chi_{eq},\dots,n-3\}\backslash\mathcal{S}$ en donde la cantidad de vértices de S que utilizan colores de J es inferior a b_{SJr} en todos los r-eqcols de G, los coloreos de la cara que definiría (5.15) también satisfarían $w_r = w_{r+1}$ por lo que esta cara no podría ser faceta. Otras condiciones necesarias son resumidas en el siguiente resultado.

Teorema 5.4.35. *Si la desigualdad (S,J)-color define faceta de $\mathcal{ECP}$ entonces:*

- *S no es una clique de G.*
- *Si $n-2 \notin J$ entonces $V\backslash S$ no es una clique de G.*

Demostración. Sea F la faceta de $\mathcal{ECP}$ definida por (5.15). Si S fuese una clique entonces, en un $(n-2)$-eqcol cualquiera, no todos los vértices de S pueden pintarse con colores de J. Luego, no existe ningún $(n-2)$-eqcol que pertenezca a F. Así que todos los coloreos que pertenecen a F también satisfacen la igualdad $w_{n-2} = w_{n-1}$ lo cual es un absurdo porque F es una faceta.
Asumamos entonces que S no es una clique y que $n-2$ no es un color de J. Si $V\backslash S$ fuese una clique entonces, en un $(n-2)$-eqcol que pertenece a F, el color $n-2$ sólo puede usarse en $V\backslash S$ y, al ser éste último una clique, el color $n-2$ se usa una vez. En un $(n-1)$-eqcol de F, el color $n-2$ se usa una vez también. Así que todos los coloreos que pertenecen a F también satisfacen la igualdad $\sum_{v\in V} x_{vn-2} = w_{n-2}$ lo cual también es absurdo por ser F una faceta. Por lo tanto, $V\backslash S$ no es una clique de G. □

Finalizamos esta sección con un ejemplo de aplicación del Teorema 5.4.33.

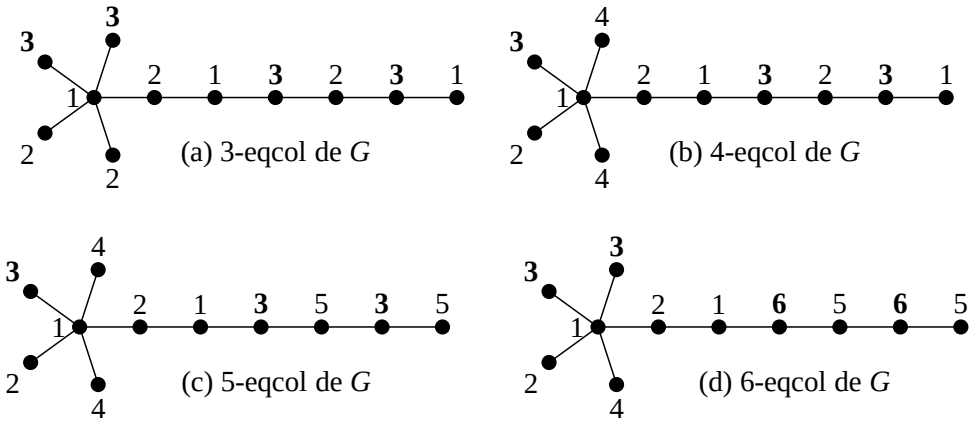


Figura 5.5: Coloreos del Ejemplo 5.4.36

Ejemplo 5.4.36. *Consideremos nuevamente el grafo dado en la Figura 5.2(a) y recordemos que $\chi_{eq}(G) = 3$. Vamos a aplicar el Teorema 5.4.33 para la desigualdad (S,J)-color con $S = \{3,4,8,10\}$ y $J = \{3,6,8\}$, i.e.*

$$\sum_{v\in S} x_{v3} + \sum_{v\in S} x_{v6} + \sum_{v\in S} x_{v8} + \sum_{v\in V} x_{v10} \leq 4w_3 - w_4 + w_6 + w_{10} - w_{11}.$$

Claramente, las hipótesis (i) y (ii) del teorema son satisfechas. En cambio, la hipótesis (iii) requiere de algunos coloreos que caigan en la cara dada por la desigualdad. Ejemplos de los coloreos necesarios para cumplir esta hipótesis son:

- *Un 3-eqcol que coloree los vértices de S con color 3, dado en Figura 5.5(a).*
- *Un 4-eqcol que coloree 3 vértices de S con color 3, dado en Figura 5.5(b).*
- *Un 5-eqcol que coloree 3 vértices de S con color 3, dado en Figura 5.5(c).*
- *Un 6-eqcol que coloree los vértices de S con colores 3 y 6, dado en Figura 5.5(c).*
- *Un 7-eqcol que coloree los vértices de S con colores 3 y 6, dado por intro$(c,9)$ donde c es el 6-eqcol anterior.*

- *Un 8-eqcol que coloree los vértices de S con colores 3, 6 y 8, dado por intro*$(c,10)$ *donde c es el 7-eqcol anterior.*

Entonces la desigualdad (S,J)*-color define una faceta de* $\mathcal{ECP}$.

5.5. Observaciones sobre $\mathcal{ECP}_1$ y $\mathcal{ECP}_2$

Si bien en [64] no pudo caracterizarse la dimension del poliedro asociado al modelo para el PCG que incluye las desigualdades (4.16), nosotros si pudimos caracterizar la dimensión del poliedro homólogo para el PCEG. En efecto, la dimension de $\mathcal{ECP}_1$ es

$$n^2 - (n + \chi_{eq} + |\mathscr{S}| - 1)$$

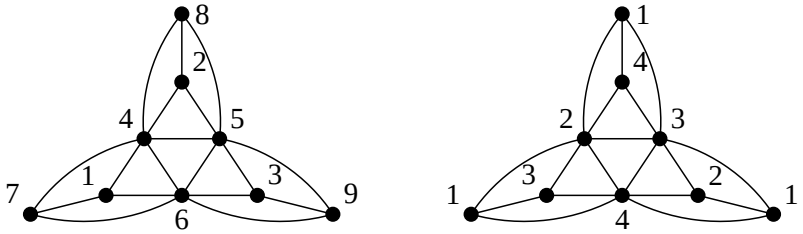
y un sistema minimal para $\mathcal{ECP}_1$ es el definido por las ecuaciones (5.1), (5.2), (5.3) y además, las igualdades (4.24) para $2 \leq j \leq n$ [63].

Aunque pudimos determinar la dimensión de $\mathcal{ECP}_1$, las igualdades (4.24) que aparecen en el sistema minimal hacen más complejas a las pruebas de facetitud. Un ejemplo de ello es el comportamiento de las desigualdades (Q,j)-clique. Si bien estas desigualdades siempre definen facetas de $\mathcal{ECP}$, esto no ocurre en $\mathcal{ECP}_1$. Esto es fácil ver considerando el caso en que $j > \lfloor n/2 \rfloor$, pues la desigualdad (Q,j)-clique es redundante debido a que la clase de color j está obligada a tener tamaño 1.

En [63] obtuvimos el siguiente resultado para las desigualdades (Q,j)-clique cuando $j \leq \lfloor n/2 \rfloor$.

Teorema 5.5.1. *Sea Q una clique maximal de G tal que* $|Q| \geq 2$ *y sea* $1 \leq j \leq \lfloor n/2 \rfloor$. *Consideremos el conjunto K de los* $k \geq j$ *tales que G admite un k-eqcol pero no existe ningún k-eqcol en la cara definida por la desigualdad* (Q,j)*-clique. Si* $K = \varnothing$ *y* $\{\lceil n/2 \rceil, \ldots, n-j\} \backslash \mathscr{S} \neq \varnothing$ *entonces la desigualdad* (Q,j)*-clique define una faceta de* $\mathcal{ECP}_1$.

El siguiente es un ejemplo de desigualdad clique con $j \leq \lfloor n/2 \rfloor$ que no define faceta de $\mathcal{ECP}_1$. Consideremos el grafo de la Figura 5.6(a) y la desigualdad $x_{41} + x_{51} + x_{61} \leq w_1$. Notemos que G admite un 4-eqcol, expuesto en la Figura 5.6(b), pero ningún 4-eqcol puede pintar un vértice de la clique de color 1 porque esta clase de color siempre tiene tamaño 3. Luego, todo coloreo de la cara de $\mathcal{ECP}_1$ definida por esta desigualdad satisface $w_4 = w_5$, y por lo tanto no es faceta de $\mathcal{ECP}_1$.



(a) vértices (b) 4-eqcol

Figura 5.6: Un 4-eqcol

En lo que respecta al poliedro $\mathcal{ECP}_2$, éste resulta particularmente difícil de estudiar pues su dimensión depende en gran medida del etiquetado del grafo.

Por ejemplo, considerando el grafo estrella $K_{1,7}$ presentado en el capítulo anterior tenemos que, para el etiquetado propuesto en la Figura 4.1(a), $dim(\mathcal{ECP}_2) = 23$. Sin embargo, para el etiquetado de la Figura 4.1(b), $dim(\mathcal{ECP}_2) = 28$.

En cambio, hemos obtenido una cota superior de $dim(\mathcal{ECP}_2)$ que comentamos brevemente a continuación.

Debido a la restricción (4.17) de la formulación ECF_2, ningún coloreo equitativo puede pintar un vértice v de algún color $v+1,\ldots,n$. Además, si existe un vértice u tal que $\{1,\ldots,u\}$ es una clique de G entonces, en todo coloreo equitativo, el vértice u se deberá pintar de *color u*. Luego, v no puede usar el color u si es adyacente a u. Entonces para cada $v \in V$,

$$\mathscr{Z}(v) = \{u : 1 \leq u \leq v-1,\ (u,v) \in E,\ \{1,\ldots,u\} \text{ es clique de } G\} \cup \{v+1,\ldots,n\}$$

es un conjunto de colores que no pueden ser asignados a v en ningún coloreo equitativo, y puede probarse que $dim(\mathcal{ECP}_2) \leq n^2 - (\chi_{eq} + |\mathscr{S}| + 1 + \sum_{v \in V} |\mathscr{Z}(v)|)$.

Por ejemplo, para el grafo $K_{1,7}$ con el etiquetado propuesto en la Figura 4.1(a), la cota superior es exactamente la dimensión de $\mathcal{ECP}_2$ mientras que con el etiquetado de la Figura 4.1(b), la cota superior es 29.

Capítulo 6

Algoritmo ECOPT - Un Branch-and-Cut para el Coloreo Equitativo

El propósito de este capítulo es describir los diversos elementos que componen nuestro algoritmo Branch-and-Cut, que llamaremos *ECOPT*. Especificamos cómo armar la relajación lineal inicial y qué criterios implementamos para generar y recorrer los nodos del árbol de búsqueda. Describimos además cómo mejorar las cotas superiores e inferiores. En el primer caso, elaboramos heurísticas que generan un coloreo equitativo a partir de la solución fraccionaria actual. En el segundo, y el más importante de nuestro trabajo dado que se desprende de la teoría vista en el capítulo anterior, daremos los procedimientos para separar desigualdades válidas que luego serán agregadas como cortes en las relajaciones asociadas a los nodos del árbol de búsqueda.

A lo largo de este capítulo nos encontraremos con algoritmos de diversas índoles que dependen de *parámetros*. Estos parámetros serán determinados en el siguiente capítulo para alcanzar un grado de eficiencia óptima en el desempeño de ECOPT.

6.1. Relajación lineal inicial

Con el objetivo de lograr una relajación más reducida en número de variables y restricciones, hemos tenido en cuenta diversas alternativas que detallaremos a continuación.

De ahora en más vamos a denotar con $\underline{\chi_{eq}}$ y $\overline{\chi_{eq}}$ a la cota inferior y superior inicial obtenidas por las heurísticas iniciales vistas en Sección 2.3 y Capítulo 3. También denotamos con q al tamaño de la mayor clique maximal obtenida.

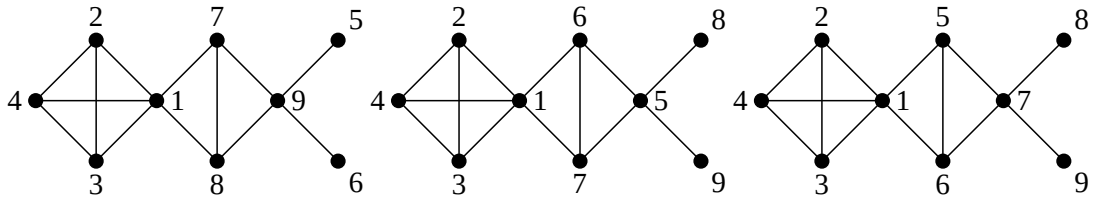
6.1.1. Enumeración de los vértices

En el caso de utilizarse el modelo ECF_2, hemos señalamos anteriormente que la estructura de éste depende de las etiquetas de los vértices y, en consecuencia, la perfomance del algoritmo está vinculada a ella. Hemos considerado diferentes criterios para asignar las etiquetas a los vértices. En primer lugar, a los vértices que constituyen la clique maximal hallada durante el cómputo de la cota inferior de χ_{eq}, les asignamos los primeros q números (es decir que, bajo esta enumeración, la clique maximal es $\{1,2,\ldots,q\}$). Los vértices restantes se ordenan de acuerdo a alguno de los 3 criterios siguientes:

- **Asc**: Orden ascendente de grado. Si $\delta(u) \leq \delta(v)$ entonces $u \leq v$. En caso de empate, se usa el grado de segundo orden $|N^2[v]|$, donde $N^2[v] = \{v'' \in N[v'] : v' \in N(v)\}$. Si $\delta(u) = \delta(v)$ y $|N^2[u]| \leq |N^2[v]|$ entonces $u \leq v$.

- **LF**: Orden descendente de grado. Si $\delta(u) \geq \delta(v)$ entonces $u \leq v$. En caso de empate, se usa el grado de segundo orden $|N^2[v]|$. Si $\delta(u) = \delta(v)$ y $|N^2[u]| \geq |N^2[v]|$ entonces $u \leq v$.
- **SL**: Cada vértice v es el de menor grado en el grafo $G - n - (n-1) - \ldots - (v+1)$. Para lograr esto, consideremos una lista L vacía. Elegimos el vértice con menor grado en el subgrafo inducido por los vértices que no están en L y lo colocamos en la lista. Repetimos el paso anterior y, una vez que L contenga todos los vértices del grafo, la invertimos. La lista resultante determina el orden de los vértices (este orden es aplicado en el algoritmo SL-COLOR de la Sección 3.2).

En la Figura 6.1 se muestran los distintos etiquetados que se aplican a los vértices $5, \ldots, 9$ si partimos de considerar la clique maximal $\{1,2,3,4\}$.



(a) Asc (b) LF (c) SL

Figura 6.1: Etiquetado de vértices

No hay ninguna razón aparente para elegir un ordenamiento u otro, siendo necesario testear la performance del algoritmo para todos ellos. Esta prueba computacional se desarrolla en el Capítulo 7.

6.1.2. Eliminación de variables y restricciones

A partir de los valores del tamaño de la clique maximal y las cotas inferior y superior que brindan las heurísticas iniciales, se pueden fijar ciertas variables. Por ejemplo, considerando que el modelo sea ECF_2 y la clique maximal sea $\{1, \ldots, q\}$ tenemos:

$$
\begin{aligned}
w_j &= 1, && \forall\ 1 \leq j \leq \underline{\chi_{eq}} \\
w_j &= 0, && \forall\ \overline{\chi_{eq}} + 1 \leq j \leq n \\
x_{ii} &= 1, && \forall\ 1 \leq i \leq q \\
x_{ij} &= 0, && \forall\ 1 \leq i \leq q,\ 1 \leq j \leq \overline{\chi_{eq}},\ i \neq j \\
x_{ij} &= 0, && \forall\ i \in N(j),\ i > q,\ 1 \leq j \leq q \\
x_{ij} &= 0, && \forall\ 1 \leq i \leq n,\ \overline{\chi_{eq}} + 1 \leq j \leq n
\end{aligned}
$$

Luego, toda restricción del modelo que involucre alguna de las variables fijadas se modifica para que resulte expresada solamente en términos de las variables que no están fijadas. En los casos en que la restricción quede redundante, se la elimina de la formulación. Con este tratamiento se eliminan las restricciones de asignación (4.12) para los vértices que conforman la clique maximal, las restricciones de adyacencia (4.13) para las aristas tales que un extremo es un vértices de la clique maximal y las restricciones (4.19) que involucran un vértice aislado y un color de $1, \ldots, \underline{\chi_{eq}}$.

6.1.3. Sustitución de restricciones en grafos de gran tamaño

Aún cuando el tamaño de la formulación disminuye considerablemente con los procedimiento descriptos anteriormente, la cantidad de restricciones de adyacencia todavía puede resultar inmanejable a la hora de

resolver las relajaciones lineales. En [62] se propone una alternativa a la formulación empleada, en donde se sustituyen las restricciones de adyacencia por restricciones de vecindad. Si bien la formulación resultante es más débil y está compuesta por una matríz más densa, la cantidad de restricciones total pasa de ser un valor del orden de n^3 a uno del orden de n^2.

En nuestro caso la experiencia nos indica que, en función de la cantidad de aristas del grafo, conviene o no adoptar la estrategia propuesta por [62], reemplazando las restricciones de adyacencia por la siguiente versión relajada de las desigualdades subvecindad (véase Sección 5.4.1):

$$\mu_j x_{uj} + \sum_{v \in N(u)} x_{vj} + \sum_{k=j+1}^{n} (\mu_j - \mu_k) x_{uk} \leq \mu_j w_j,$$

donde $\mu_k = \min\{\lceil n/\underline{\chi_{eq}} \rceil, \lceil n/k \rceil, \overline{\alpha}(N(u))\}$ para cualquier $1 \leq k \leq n$, El valor $\overline{\alpha}(N(u))$ representa una cota superior de $\alpha(N(u))$ y puede calcularse computando el cardinal de una partición en cliques del vecindario de u con el procedimiento CLIQUEPART descripto en la Sección 2.3. No es difícil ver que las desigualdades sustitutas no modifican el conjunto de soluciones enteras factibles, aunque debiliten la relajación de la formulación.

Experimentando con instancias de distintos tamaños pudimos concluir que efectuar la sustitución cuando el grafo tenga más de 3000 aristas es un criterio razonable.

6.1.4. Manejo de las restricciones (4.18)

En el modelo ECF_2, las restricciones (4.18) que eliminan soluciones simétricas son del orden de n^2. En grafos de tamaños medianos y grandes traen como consecuencia que la resolución de las relajaciones lineales insume mucho tiempo. Para lograr una mejor performance, decidimos considerarlas como una familia más de cortes, evitando así que formen parte explícita de la relajación inicial.

6.1.5. Algoritmo de resolución de la relajación lineal

Una de las funcionalidades de CPLEX, el software con el que realizamos la optimización, es que nos permite elegir con qué algoritmo resolver las relajaciones lineales asociadas a los nodos del árbol de búsqueda. Según nuestra experiencia, la mejor opción en términos de tiempo de proceso es emplear el *Método de Barrera* en la relajación inicial del nodo raíz, y *Simplex Primal* para el resto de las relajaciones.

6.2. Generación y recorrido del árbol de búsqueda

Las estrategias seguidas para la generación y recorrido del árbol de búsqueda son claves para explorar eficientemente el conjunto de soluciones del problema. El objetivo es evitar búsquedas innecesarias en subconjuntos de soluciones donde se pueda inferir que no se encuentra el óptimo y guiar la búsqueda de manera inteligente. Con este propósito en mente, experimentamos con diferentes criterios que presentamos a continuación.

Para unificar la notación de esta sección, vamos a llamar (x^*, w^*) a la solución óptima fraccionaria obtenida luego de evaluar un nodo determinado del árbol, y $z^* = \sum_{j=1}^{n} w_j^*$ al valor de su función objetivo. También denotamos con LB y UB a las mejores cotas inferior y superior obtenidas hasta el momento de evaluar el nodo. Se verifica, por lo tanto, $\underline{\chi_{eq}} \leq LB \leq \chi_{eq} \leq UB \leq \overline{\chi_{eq}}$ y $LB \leq z^* < UB$.

6.2.1. Estrategia de generación de nodos

Dentro de las capacidades que ofrece CPLEX existe la posibilidad de personalizar la forma en que se crean los nuevos nodos en la etapa de Bifurcación. Si bien tiene la limitación de que la estrategia debe ser

dicotómica (no se pueden generar más de dos nodos), nos otorga suficiente flexibilidad como para poder implementar criterios propuestos por nosotros y experimentar con ellos, descriptos a continuación:

- *Bifurcar por tamaño de la clase de color más fraccionaria*. Para cada color j tal que $w^*_j > 0$ se calcula el valor $D_j = \sum_{v \in V} x^*_{vj}$ y luego se elige el color con el valor D_j más fraccionario, i.e. aquel que maximiza $|\frac{1}{2} - (D_j - \lfloor D_j \rfloor)|$. Sea entonces $\hat{j}$ el color elegido. Si $|\frac{1}{2} - (D_{\hat{j}} - \lfloor D_{\hat{j}} \rfloor)| < 0.45$ se generan dos nuevos nodos agregando las restricciones $\sum_{v \in V} x_{v\hat{j}} \leq \lfloor D_{\hat{j}} \rfloor$ y $\sum_{v \in V} x_{v\hat{j}} \geq \lceil D_{\hat{j}} \rceil$ respectivamente. Esta estrategia se basa justamente en eliminar las soluciones fraccionarias tales que $\lfloor D_{\hat{j}} \rfloor < \sum_{v \in V} x^*_{v\hat{j}} < \lceil D_{\hat{j}} \rceil$. En caso de que $|\frac{1}{2} - (D_{\hat{j}} - \lfloor D_{\hat{j}} \rfloor)| \geq 0.45$, se recurre a la estrategia PseudoReducedCosts de CPLEX para efectuar la bifurcación.
  Lamentablemente no mostró un buen desempeño en la práctica.

- *Bifurcar por variables x^*_{vj}*. Dado que la integralidad de las variables x^*_{vj} implican la integralidad de las variables w^*_j, nos enfocaremos en una estrategia sobre las primeras. El criterio consiste en asignar en forma dinámica puntajes a las variables y elegir aquella con mayor puntaje para generar los dos nuevos subproblemas. En uno de ellos, la variable elegida es fijada en 0 y en el otro en 1.
  El puntaje dinámico se rige por las siguientes métricas:

  - *Métrica A (bifurcar por el vértice cuya vecindad es la más fraccionaria)*. Para cada variable x^*_{vj} fraccionaria calculamos la cantidad de variables fraccionarias en el vecindario de v con el color j:
    $$A_{vj} = |\{u \in N(v) \ : \ 0 < x^*_{uj} < 1\}|.$$
  - *Métrica B (bifurcar por el vértice cuya vecindad es la más colorida)*. Para cada variable x^*_{vj} fraccionaria calculamos la cantidad de colores distintos que usa la vecindad de v:
    $$B_v = |\{k \in \{1, \ldots, n\} \ : \ x^*_{uk} = 1, \ \forall \ u \in N(v)\}|.$$

  Ambas métricas resultaron buenas en la práctica. Sin embargo se producen muchos casos en donde una métrica no es capaz de definir entre algunas variables (por ejemplo, a dos variables fraccionarias $x^*_{vj_1}$ y $x^*_{vj_2}$ le corresponden el mismo puntaje B_v). Por eso optamos por probar con una métrica y, en caso de que varias variables presenten el mismo puntaje, desempatar con la otra métrica. Llamamos **AB-Rule** a la estrategia que aplica la métrica A y en caso de empate la métrica B, y **BA-Rule** a la estrategia que aplica la métrica B y en caso de empate la métrica A. Ante empate en ambas métricas elegimos aquella variable x_{vj} cuyo v es el de mayor grado y cuyo j es el menor, y en el caso de un nuevo empate simplemente elegimos la primera posibilidad considerada.

CPLEX cuenta con algunas estrategias de branching incorporadas. De todas ellas destacamos las que presentaron mejor rendimiento sobre nuestra formulación:

- **Default**. Estrategia por defecto de CPLEX. No está reportado su funcionamiento interno.

- **PseudoReducedCosts**. Es una variante de la estrategia *Reduced Costs* [59] que consiste en estimar, para cada variable fraccionaria, el crecimiento de la función objetivo si se fija aquella variable en 0 o en 1, y utilizar esta estimación para determinar la variable con que se bifurcará.

En el Capítulo 7 experimentamos con las estrategias de propósito general Default y PseudoReducedCosts, y con las estrategias AB-rule y BA-rule propuestas por nosotros.

Fijado de variables por implicaciones lógicas

En el proceso de bifurcación, el fijado de cierta variable en 1 brinda la oportunidad de fijar más variables. Si bien sus valores se deducirían lógicamente de las restricciones, setearlas explícitamente ayudan a mejorar la performance.

En el nodo en donde fijamos $x_{vj} = 1$ es claro que ningún vértice adyacente a v podrá usar el color j en la relajación que surja de dicho nodo, así que podemos fijar todas las variables $x_{uj} = 0$ para $u \in N(v)$. Por otra parte, el vértice v no podrá usar otro color que no sea j, así que podemos fijar todas las variables $x_{vk} = 0$ para $k \neq j$. Finalmente, se puede fijar $w_j = 1$.

Otras variables que podemos fijar, independientemente del valor de x_{vj}, son:

$$\begin{aligned} w_k &= 1, && \forall\ 1 \leq k \leq \lceil z^* \rceil \\ w_k &= 0, && \forall\ UB \leq k \leq \overline{\chi_{eq}} \\ x_{uk} &= 0, && \forall\ u \in V,\ UB \leq k \leq \overline{\chi_{eq}} \end{aligned}$$

La primera restricción se debe a que una relajación con función objetivo z^* no puede tener como solución entera a un coloreo que use menos de $\lceil z^* \rceil$ colores, mientras que las restricciones restantes se deben a que no nos interesa explorar relajaciones que generen soluciones con UB o más colores.

Se puede habilitar un mecanismo en CPLEX para realizar un preprocesamiento de la relajación en cada nodo del árbol antes de evaluarlo (llamado *node presolve switch*). Hemos notado que la combinación de habilitar este mecanismo y fijar variables *limpia* la relajación asociada al nodo y permite evaluar éste con mayor velocidad.

6.2.2. Estrategia de recorrido

En nuestro algoritmo experimentamos solamente con criterios provistos por CPLEX: **DFS** (*primero profundo*), **BFS** (*primero a lo ancho*) y **BestF** (*mejor cota*). En caso de elegirse la estrategia BestF, CPLEX proporciona un parámetro adicional llamado *backtracking tolerance* que indica con qué frecuencia CPLEX retrocede en el proceso de búsqueda (en el sentido de que no continúa con el nodo hijo del nodo actual). Valores cercanos a 0 en este parámetro hacen que la búsqueda tienda a comportarse como un proceso BFS mientras que valores cercanos a 1 en este parámetro hacen que se comporte más como un proceso DFS.

Hemos probado con todas las estrategias anteriores y notamos que una combinación de parámetros balanceada es aplicar la estrategia BestF con el parámetro backtracking tolerance en 0,9999. Le atribuimos este comportamiento al hecho de que la búsqueda DFS da buenos resultados en problemas de coloreo, aunque no suela poner énfasis en mejorar la cota *LB*. Esto último se logra realizando backtracking eventualmente.

6.3. Rutina de enumeración implícita

Para grafos pequeños, la enumeración implícita es más rápida que un algoritmo Branch-and-Cut. Esto sucede porque el porcentaje de tiempo invertido en la resolución de las relajaciones lineales en los nodos del árbol es muy significativo. En cambio, una enumeración directa de las soluciones enteras (como la que utiliza DSATUR) brinda en estos casos una mejor performance.

Durante la ejecución del algoritmo Branch-and-Cut, el mejoramiento de la cota inferior de χ_{eq} a partir de los valores óptimos de las relajaciones es más pronunciado en la base del árbol, pero disminuye a medida que aumenta la profundidad. Es por esto que, a una cierta profundidad, puede ser conveniente realizar una enumeración implícita de las soluciones enteras en vez de resolver relajaciones lineales que no son capaces de mejorar la cota inferior.

La situación planteada anteriormente es considerada en ECOPT. A partir de una cierta profundidad, ponemos en marcha un algoritmo enumerativo basado en DSATUR [18].

En cualquier nodo abierto del árbol disponemos de la siguiente información:

- Un coloreo parcial del grafo definido por las variables fijadas en 1 en la rama del árbol.
- Para cada vértice $v \in V$, un conjunto no vacío $D_v \subset \{1,\dots,UB-1\}$ de colores factibles que pueden ser asignados a v.
- Las clases del coloreo parcial $C_j = \{v : D_v = \{j\}\}$ para todo $1 \leq j \leq UB-1$.
- El conjunto de vértices $U = \{u : |D_u| \geq 2\}$ no coloreados en el coloreo parcial.
- Cota inferior del número cromático equitativo $lb = \lceil z^* \rceil$.

El algoritmo enumerativo deberá determinar si es posible extender el coloreo parcial disponible de manera que se obtenga un coloreo equitativo que pinte cada vértice v con los colores disponibles en D_v y que use al menos lb colores. Para lograr ese objetivo aplicamos iterativamente los siguientes pasos:

1. Para cada color $j \in \{1,\dots,UB-1\}$, sea f_j una cota inferior de la cantidad de vértices que le falta a la clase de color C_j actual para que sea posible formar un coloreo equitativo:

$$f_j = \text{máx}\left\{0, \left\lfloor \frac{n}{UB-1} \right\rfloor - |C_j| \right\}.$$

   Si $|U| < \sum_{j=1}^{UB-1} f_j$ entonces no existen suficientes vértices no coloreados y el nodo se poda por *infactibilidad*.

2. Se elige un vértice $u \in U$ para colorear.

3. Para cada color $j \in D_u$, sea g_j una cota superior de la cantidad de vértices que puede llegar a tener la clase de color C_j actual:

$$g_j = \left\lceil \frac{n}{\text{máx}\{j, lb\}} \right\rceil.$$

   Si $|C_j| + 1 \leq g_j$, se evalúa un nuevo nodo con los siguientes argumentos C'_j, U', D' y lb':

   - El coloreo parcial pinta a u de color j: $C'_j = C_j \cup \{u\}$, $U' = U \backslash \{u\}$.
   - Se aumenta la cota inferior si es necesario: $lb' = \text{máx}\{j, lb\}$.
   - Los vértices adyacentes a u no pueden pintarse de j: para todo $v \in N(u) \cap U$, $D'_v = D_v \backslash \{j\}$.
   - Ningún vértice fuera del vecindario de u puede pintarse de j en caso de que la clase de color j esté llena: si $|C'_j| = g_j$ entonces para todo $v \in U \backslash N[u]$, $D'_v = D_v \backslash \{j\}$; caso contrario, para todo $v \in U \backslash N[u]$, $D'_v = D_v$.

En el paso 2 no está especificado el criterio para elegir el vértice u. Con respecto a esto, decidimos aplicar la estrategia de branching CELIM sugerida en [71]. Se elige el vértice no coloreado que tenga la vecindad más colorida (i.e. el vértice con mayor cantidad de colores distintos usados por los vértices adyacentes a él que ya tienen fijado su color). Ante un empate, se elige aquel vértice u que presenta la mayor suma $\sum_{j \in D_u} |\{v \in N(u) : j \in D_v\}|$, y en el caso de un nuevo empate simplemente elegimos el primer vértice considerado.

Durante el desarrollo del algoritmo de enumeración hemos considerado dos esquemas para la selección de los colores que le son asignados a los vértices no coloreados aún:

- *Selección estática*. Para cada vértice $u \in U$, se define inicialmente

$$D_u = \{j : x_{uj} \text{ no está fijada en 0},\ 1 \leq j \leq UB-1\}.$$

- *Selección dinámica*. Para cada vértice $u \in U$, se define inicialmente

$$D_u = \{j : u \text{ no es adyacente a un vértice coloreado con } j,\ 1 \leq j \leq UB-1\}.$$

  Luego, en el paso 3, calculamos el mayor color utilizado por el coloreo parcial actual. Sea k este valor. Para el u elegido, en vez de considerar todos colores de D_u sólo vamos a considerar aquellos de $D_u \cap \{1,\ldots,\text{mín}\{UB-1,k+1\}\}$. En otras palabras, se introduce a lo sumo un nuevo color (el color $k+1$ justamente) en cada nivel del árbol.

El segundo enfoque es el utilizado por DSATUR [18] y permite evitar senderos que inducen soluciones simétricas. Sin embargo, no puede aprovechar la información de nodo actual respecto a las variables x_{vj} fijadas en cero al inicializar D_u.

En ambos esquemas, la enumeración puede resultar ineficiente si el tamaño inicial de los conjuntos D_u es grande. Por ejemplo, la selección estática en el peor caso enumera $\prod_{u \in U} |D_u|$ soluciones. Sólo la experimentación nos permitirá decidir en qué momento conviene comenzar la enumeración implícita. Trataremos esto en el Capítulo 7.

Un parámetro que aún no tuvimos en cuenta es el orden en el que se evalúan los nodos. Más precisamente, qué ordenamiento podemos aplicar a los colores de D_u en el paso 3, donde ese ordenamiento representa la forma en que después se evaluarán los nodos. Hemos probado ordenando los tamaños de las clases de color de mayor a menor y de menor a mayor, y resulta muy superior en cantidad de nodos evaluados y tiempo de proceso el ordenar de *menor a mayor*.

En el caso en que la selección de colores sea dinámica, en [62] se proponen dos alternativas para recorrer los nodos. Sea k el mayor color utilizado por el coloreo parcial y supongamos que, para el vértice en cuestión $u \in U$, tenemos que $k+1$ es uno de los colores a evaluar. La primera alternativa consiste en evaluar aquellos que asignan algún color de D_u inferior a $k+1$ al vértice u y luego el nodo que asigna el color $k+1$ al vértice u. La segunda es invertida: se evalúa primero el nodo con el vértice u pintado de $k+1$ y luego con los colores restantes de D_u. Nosotros probamos ambas opciones, ordenando los colores de D_u que no son $k+1$ según los tamaños de sus clases de menor a mayor, como comentamos en el párrafo anterior. La estrategia que resultó ser mejor en la práctica fue la segunda. Le atribuimos esto al hecho de que, al evaluar primero el nodo que asigna $k+1$ a u, permite tempranamente alcanzar coloreos factibles, reduciendo la brecha entre cota inferior y superior al comienzo de la enumeración.

6.4. Heurísticas primales

Una *heurística primal* es un algoritmo heurístico que genera, o intenta generar, una solución factible entera de un problema a partir de conocer una solución óptima fraccionaria de una relajación lineal de aquel problema. Por ejemplo, algunas heurísticas primales (como *Dive-and-Fix*) asignan valores enteros a las componentes fraccionarias siguiendo un criterio para que, en lo posible, la solución entera que se obtenga sea factible [78].

CPLEX nos brinda la facultad de ejecutar una heurística primal cada vez que dispone de una solución óptima fraccionaria. Si la heurística es capaz de encontrar una solución entera mejor que la actual, entonces la cota superior puede ser actualizada al igual que cuando el proceso de Branch-and-Bound poda un nodo por optimalidad.

Además de tener disponible la heurística de propósito general que ofrece CPLEX, nosotros implementamos dos heurísticas diseñada específicamente para el PCEG que explotan las características del problema. Ambos procedimientos parten de un coloreo parcial inicial y van llenando las clases de color iterativamente hasta alcanzar un coloreo equitativo o fallar en el intento, según la aplicación de criterios preestablecidos sobre la solución (x^*, w^*). Para un k dado, vamos a decir que una clase de color está *llena* cuando alcanza el cardinal $\lceil n/k \rceil$ y, entonces, no se pueden agregar más vértices a ella. Además, si ya tenemos $n \bmod k$ clases de color con tamaño $\lceil n/k \rceil$ entonces, en las clases de color que no están aún llenas, sólo se pueden incorporar hasta $\lfloor n/k \rfloor$ vértices y se considerará llena cuando alcance este umbral.

Describimos a continuación las dos heurísticas. Al final de la sección se encuentran los pseudocódigos correspondientes a ellas.

El procedimiento PRIMHEUR$_1$ corresponde a la primera heurística y consiste en asignar un color a un vértice en cada iteración. En esta heurística se asigna el color j a un vértice v cuyo valor x^*_{vj} es máximo. El vértice v es elegido de entre los vértices no coloreados aún y el color j es elegido de entre los colores disponibles no usados por los vértices adyacentes a v.

El procedimiento PRIMHEUR$_2$ corresponde a la segunda heurística que, en cada iteración, elige el color j cuya clase de color no está llena aún y contiene la menor cantidad de vértices v tales que x^*_{vj} es fraccionaria. Luego elige el vértice v que maximice x^*_{vj} y que pueda ser coloreado por j.

Al experimentar con estas heurísticas observamos que, en reiteradas ocasiones, existen muchas variables x^*_{vj} con el mismo valor. En tal caso, consideramos los siguientes criterios entre el conjunto de vértices y colores que debe desempatar:

- *Vértice de mayor grado*. Se elige el vértice v que maximiza $\delta(v)$.
- *Vértice con la vecindad más colorida*. Se elige el vértice v que maximiza $|\{j \in \{1,\dots,k\} \ : \ u \in C_j, \ \forall\, u \in N(v)\}|$.
- *Clase de color más grande/pequeña*. Se elige el color j que maximiza/minimiza $|C_j|$.
- *Clase de color con la mayor/menor cantidad de variables fraccionarias*. Se elige el color j que maximiza/minimiza $|\{v \in C_j : 0 < x^*_{vj} < 1\}|$.
- *Clase de color más/menos fraccionaria*. Se elige el color j que minimiza/maximiza $|0{,}5 - (\sum_{v \in C_j} x^*_{vj} - \lfloor \sum_{v \in C_j} x^*_{vj} \rfloor)|$.
- *Clase de color que más/menos vértices pueda pintar*. Se elige el color j que maximiza/minimiza $|\{v \in V : v$ no está coloreado y j no es usado por ningún vecino de $v\}|$.

Estos criterios tienen poco costo computacional y pueden ser implementados fácilmente.

De probar con diferentes combinaciones de los criterios anteriores, la experiencia indica para la primera heurística elegir el *vértice con la vecindad más colorida* y el criterio que elige la *clase de color más pequeña*. Para la segunda heurística se busca la *clase de color con la menor cantidad de variables fraccionarias* y, en caso de empate, se elige la *clase de color que menos vértices pueda pintar*. Una vez fijado el color j, se busca el vértice v que maximice x^*_{vj} y, en caso de empate, se elige el *vértice con la vecindad más colorida*. Tanto en la primera como en la segunda heurística, cuando todos los criterios definidos no pudieron desempatar, simplemente elegimos la primera posibilidad considerada.

En las heurísticas presentadas no hemos mencionado cómo obtener el coloreo parcial inicial que debemos suministrarles. Este coloreo parcial puede ser generado a partir de la solución fraccionaria (x^*, w^*): si $x^*_{vj} = 1$ y la clase de j no está llena entonces se pinta el vértice v de color j. Detallamos este procedimiento al final de la sección, con el nombre PARTIALCOLINIT.

Observemos que PARTIALCOLINIT no especifica qué vértice elegir en caso de que haya más de uno. Un criterio adecuado es usar el ordenamiento SL explicado en la Sección 6.1.1.

Cada vez que disponemos de una solución fraccionaria (x^*, w^*) ejecutamos, para cada valor de $k \in \{LB, \ldots, UB-1\}$, PARTIALCOLINIT para obtener el coloreo parcial y luego una o las dos heurísticas primales. Es claro que si encontramos un k-eqcol durante este proceso, no seguimos explorando entre $k+1$ y $UB-1$.

En el Capítulo 7 desarrollamos algunas pruebas computacionales para saber si existe un impacto positivo en la performance del algoritmo Branch-and-Cut cuando embebemos la primera, la segunda o ambas heurísticas primales.

A continuación presentamos los pseudocódigos correspondientes a PRIMHEUR_1, PRIMHEUR_2 y PARTIALCOLINIT.

Algoritmo 7: PRIMHEUR$_1$

Data: coloreo parcial con clases $C_1, C_2, \ldots, C_k$
Result: k-eqcol con clases $C_1, C_2, \ldots, C_k$

begin
  while *quede algún vértice del coloreo parcial sin pintar* **do**
    $value_{best} \leftarrow -1$
    for $v \in V$ *tal que v no está coloreado aún* **do**
      sea DISP el conjunto de colores cuyas clases de color no están llenas aún
      $\mathsf{DISP} \leftarrow \mathsf{DISP} \backslash \{j : u \in C_j \ \ \forall\, u \in N(v)\}$
      if $\mathsf{DISP} = \varnothing$ **then**
        no hay colores disponibles para asignar al vértice v, **fail**
      end
      for $j \in \mathsf{DISP}$ **do**
        if $x^*_{vj} > value_{best}$ **then**
          $v_{best} \leftarrow v$
          $j_{best} \leftarrow j$
          $value_{best} \leftarrow x^*_{vj}$
        end
      end
    end
    $C_{j_{best}} \leftarrow C_{j_{best}} \cup \{v_{best}\}$
  end
end

Algoritmo 8: PRIMHEUR$_2$

Data: coloreo parcial con clases $C_1, C_2, \ldots, C_k$
Result: k-eqcol con clases $C_1, C_2, \ldots, C_k$

begin
  while *quede algún vértice del coloreo parcial sin pintar* **do**
    para cada clase C_j aún no llena, sea $F_j = \{v \in C_j : 0 < x^*_{vj} < 1\}$
    sea j_{best} el color j que minimiza $|F_j|$
    $value_{best} \leftarrow -1$
    for $v \in V$ *tal que v no está coloreado aún* **do**
      if $j_{best} \notin \{j : u \in C_j \ \ \forall\, u \in N(v)\}$ **then**
        if $x^*_{vj} > value_{best}$ **then**
          $v_{best} \leftarrow v$
          $value_{best} \leftarrow x^*_{vj}$
        end
      end
    end
    if $value_{best} = -1$ **then**
      no se puede colorear ningún vértice de j_{best}, **fail**
    end
    $C_{j_{best}} \leftarrow C_{j_{best}} \cup \{v_{best}\}$
  end
end

Algoritmo 9: PARTIALCOLINIT

Data: valor k
Result: coloreo parcial con clases $C_1, C_2, \ldots, C_k$

begin
  $C_j = \varnothing \ \ \forall\, 1 \leq j \leq k$
  for $v \in V$ *tal que existe un* $j \leq k$ *para el cual* $x^*_{vj} = 1$ **do**
    sea j el color para el cual $x^*_{vj} = 1$
    if *hay más de n mod k clases de colores llenas* **then**
      if $|C_j| < \lfloor n/k \rfloor$ **then**
        $C_j \leftarrow C_j \cup \{v\}$
      end
    else
      if $|C_j| < \lceil n/k \rceil$ **then**
        $C_j \leftarrow C_j \cup \{v\}$
      end
    end
  end
end

6.5. Algoritmo de planos de corte

Como dijimos en el Capítulo 4, el algoritmo de planos de corte es el encargado de incorporar nuevas restricciones a una relajación lineal y está conformado por diversas rutinas de separación que producen estas restricciones durante el proceso de optimización. En nuestro caso, cada rutina de separación tiene asociada una familia de desigualdades válidas analizada en el Capítulo 5.

Para lograr una performance aceptable, las rutinas de separación deben ser rápidas en términos de tiempo. En algunos casos, este objetivo puede cumplirse con algoritmos exactos, pero en otros es necesario implementar heurísticas. Estas últimas no aseguran que siempre se pueda detectar desigualdades violadas, pero es una solución de compromiso ante el alto costo computacional de algunos problemas de separación.

Para implementar el algoritmo de planos de corte, aprovechamos la interfaz ofrecida por CPLEX. Una de las prestaciones de CPLEX es la de permitir introducir cortes en cualquier nodo del árbol. Los cortes pueden ser restricciones tanto de desigualdad como de igualdad y, al momento de agregarlos, debemos informarle a CPLEX en cuáles de las siguientes categorías cae:

- *Cortes locales*. Es una restricción que corta a la solución fraccionaria actual pero es válida para toda solución entera perteneciente a la relajación actual. Una vez que se agrega a la relajación de un nodo, queda para siempre en aquel nodo y sus descendientes.
- *Cortes globales*. Es una restricción que corta a la solución fraccionaria actual pero es válida para toda solución entera del problema. La restricción puede ser incorporada en cualquier nodo del árbol (según el criterio de CPLEX) además del nodo actual. Una vez que se agrega, queda para siempre en aquel nodo y sus descendientes.
- *Cortes globales purgables*. Es un corte global que puede ser eliminado de cualquier nodo (y, por lo tanto, de su descendencia) si CPLEX considera que es ineficaz mantenerlo.

6.5.1. Separación de Desigualdades Clique

El problema de separación de estas desigualdades consiste en lo siguiente:

dada una solución fraccionaria (x^*, w^*) *y un color* j *tal que* $w^*_j > 0$, *se debe decidir si existe una clique maximal* Q *en* G *tal que* $\sum_{v \in Q} x^*_{vj} > w^*_j$.

Debido a que este problema es NP-completo, lo usual es utilizar procedimientos heurísticos. Se conocen dos enfoques para encarar este problema [10]:

- Antes de comenzar la optimización, se genera un conjunto de cliques maximales y se las almacena en una lista. Luego, durante la optimización, se chequea en la lista la existencia de desigualdades clique violadas. Este enfoque es el utilizado por CPLEX en sus cortes clique.
- Durante el proceso de optimización, buscar cliques maximales a partir del conocimiento de la solución fraccionaria que se quiere cortar.

En el Branch-and-Cut para el PCG propuesto en [62] se reporta que la experiencia con la primera estrategia no fue alentadora, inclinando a los autores a optar por la segunda. Es por eso que decidimos elegir esta última opción.

Para separar desigualdades clique utilizamos una heurística de tipo goloso. Se arma una lista ordenada en forma decreciente de las variables x^*_{vj} fraccionarias o nulas y, en forma iterativa y por cada uno de los

vértices de estas variables, se buscan clique maximales. Si la desigualdad clique corta a la solución entonces se agrega como corte y se prohibe el último vértice añadido a dicha clique. Además se prohibe el segundo vértice de una clique, independientemente de si se agregó o no como corte. Los vértices *prohibidos* no pueden formar parte de una clique en las iteraciones sucesivas. Este procedimiento se repite hasta 3 veces por cada vértice. Tanto los vértices v que satisfacen $x^*_{vj} = 1$ como los vértices aislados son prohibidos al comienzo del procedimiento, pues estos vértices no pueden formar parte de desigualdades cliques que puedan ser violadas. Este procedimiento se detalla al final del capítulo con el nombre de CLIQUESEP.

Debido a que los cortes generados son válidos en cualquier nodo del árbol, los agregamos como *cortes globales purgables*.

6.5.2. Separación de Desigualdades 2-rango

El problema de separación de estas desigualdades es el siguiente:

dada una solución fraccionaria (x^*, w^*) y un color j tal que $w^*_j > 0$, se debe decidir si existe una clique maximal Q y un conjunto $R \subset V$ tal que $\alpha(R) = 2$, todos los vértices de Q son adyacentes a todos los vértices de R y $\sum_{v\in R} x^*_{vj} + 2\sum_{v\in Q} x^*_{vj} > 2w^*_j$.

Para separar estas desigualdades consideramos un criterio goloso. Se arma una lista ordenada en forma decreciente de las variables x^*_{vj} fraccionarias o nulas. Luego buscamos un par de vértices no adyacentes v_1 y v_2 tal que $x^*_{v_1 j} + x^*_{v_2 j}$ sea máximo e inicializamos los conjuntos $R = \{v_1, v_2\}$ y $Q = \varnothing$. A partir de aquí, vamos llenando los conjuntos R y Q agregando vértices candidatos v con el mayor valor posible de x^*_{vj} hasta que no haya más candidatos por agregar. En este sentido, consideramos que un vértice v es *candidato* para incorporarse en R cuando es adyacente a todos los vértices de Q y $\alpha(R \cup \{s\}) = 2$ (basta con chequear que para todo par de vértices de R que no sean adyacentes a v, sí sean adyacentes entre si, a fin de evitar un triángulo en el complemento de $G[R \cup \{s\}]$). También consideramos que un vértice v es *candidato* para incorporarse en Q cuando es adyacente a todos los vértices de Q y R. Como medida adicional, si un vértice es candidato para ingresar a ambos conjuntos, se incorpora a Q. Una vez que no es posible agregar más vértices a Q y R se procede a verificar si $\sum_{v\in R} x^*_{vj} + 2\sum_{v\in Q} x^*_{vj} > 2w^*_j$, y en caso afirmativo, se agrega la desigualdad $\sum_{v\in R} x_{vj} + 2\sum_{v\in Q} x_{vj} \leq 2w_j$ como *corte global purgable*.

Al igual que en CLIQUESEP, mantenemos una lista de vértices prohibidos que no pueden ingresar a R y Q en iteraciones posteriores. El criterio que aplicamos es siempre prohibir los vértices de R independientemente de si se agregó o no el corte constituido por R y Q.

Tuvimos dos inconvenientes al utilizar esta rutina de separación. El primero es que la rutina genera, en reiterados casos, desigualdades con soporte similar a las cliques. Por ejemplo, si tenemos una clique Q cuyos vértices son adyacentes a dos vértices v_1 y v_2 no adyacentes entre sí, entonces puede ocurrir que CLIQUESEP genere $\sum_{v\in Q\cup\{v_1\}} x_{vj} \leq w_j$ mientras que esta rutina genera $x_{v_1 j} + x_{v_2 j} + 2\sum_{v\in Q} x_{vj} \leq 2w_j$. Si bien son desigualdades distintas, son muy paralelas entre sí y no vale la pena incorporar ambas. Para resolver este problema, impedimos que se formen desigualdades 2-rango con vértices ya utilizados en cortes cliques, para el mismo color j. Esto es porque las cliques suelen ser cortes de mejor calidad, inclinándonos por agregarlos antes que las 2-rango.

La otra dificultad que encontramos fue que la rutina consume un tiempo considerable intentando separar desigualdades infructuosas, por lo que implementamos un *contador de intentos*. Luego de encontrar consecutivamente 5 desigualdades que no cortan la solución fraccionaria, el proceso se detiene. Esto surgió de observar que si el algoritmo falla las primeras 5 veces, es inusual que en los próximos intentos encuentre desigualdades violadas.

6.5.3. Separación de Desigualdades Clique-vecindad

Sea $u \in V$ tal que $N(u)$ no es una clique, Q una clique de G tal que $Q \cap N[u] = \varnothing$ y sean dos números j, k tales que

$$k = \left\lceil \frac{n}{\left\lceil \frac{n}{k-1} \right\rceil - 1} \right\rceil, \;\; 3 \leq k \leq \alpha(N(u)) + 1 \;\; \text{y} \;\; 1 \leq j \leq \left\lceil \frac{n}{k-1} \right\rceil - 1.$$

Dado que trabajamos con grafos donde $\chi_{eq} \leq n-2$ tenemos que $w_{n-1} = 0$ y la desigualdad (u, j, k, Q)-clique-vecindad se puede simplificar a:

$$(k-1)x_{uj} + \sum_{l=\lceil \frac{n}{k-1} \rceil}^{n-2} \left(k - \left\lceil \frac{n}{l} \right\rceil\right) x_{ul} + \sum_{v \in N(u) \cup Q} x_{vj} \leq \sum_{l=j}^{n} b_{ul}(w_l - w_{l+1}), \tag{6.1}$$

donde

$$b_{ul} = \begin{cases} \min\{\lceil n/l \rceil, \alpha(N(u)) + 1\}, & \text{si } j \leq l \leq \lceil n/k \rceil - 1 \\ k, & \text{si } l \geq \lceil n/k \rceil \end{cases}$$

Como determinar $\alpha(N(u))$ es un problema NP-difícil, reemplazamos en la desigualdad (6.1) $\alpha(N(u))$ por una cota superior a la cual denotamos $\overline{\alpha}(N(u))$. Esta cota superior se calcula para todo $u \in V$ antes de comenzar la optimización con el procedimiento CLIQUEPART visto en el Capítulo 2. Se puede ver que la desigualdad resultante es válida para $\mathcal{ECP}$ aún para valores de k tales que $\alpha(N(u)) + 2 \leq k \leq \overline{\alpha}(N(u)) + 1$. Aunque la misma sea más débil que la desigualdad (6.1), será de utilidad para cortar soluciones fraccionarias.

El problema de separación de estas nuevas desigualdades es:

dada una solución fraccionaria (x^*, w^*), un color j tal que $w^*_j > 0$, y un vértice u tal que $N(u)$ no es una clique y $0 < x^*_{uj} < 1$, se debe decidir si existe una clique Q de G y un número k tales que

$$Q \cap N[u] = \varnothing, \;\; k = \left\lceil \frac{n}{\left\lceil \frac{n}{k-1} \right\rceil - 1} \right\rceil, \;\; 3 \leq k \leq \overline{\alpha}(N(u)) + 1, \;\; j \leq \left\lceil \frac{n}{k-1} \right\rceil - 1, \quad y$$

$$(k-1)x^*_{uj} + \sum_{l=\lceil \frac{n}{k-1} \rceil}^{n-2} \left(k - \left\lceil \frac{n}{l} \right\rceil\right) x^*_{ul} + \sum_{v \in N(u) \cup Q} x^*_{vj} > \sum_{l=j}^{n} \hat{b}_{ul}(w^*_l - w^*_{l+1}),$$

$$\textit{donde} \quad \hat{b}_{ul} = \begin{cases} \min\{\lceil n/l \rceil, \overline{\alpha}(N(u)) + 1\}, & \text{si } j \leq l \leq \lceil n/k \rceil - 1 \\ k, & \text{si } l \geq \lceil n/k \rceil \end{cases}$$

Para generar estas desigualdades implementamos un algoritmo enumerativo que aprovecha las cliques generadas durante la ejecución de CLIQUESEP. Sea entonces Q una clique maximal. Para cada u tal que x^*_{uj} es fraccionario, chequeamos que u no sea adyacente a la clique Q y que su vecindario tenga al menos 3 vértices y no sea una clique. En tal caso, elegimos valores de k en el intervalo

$$\max\left\{3, \left\lceil \frac{n}{UB} \right\rceil\right\} \leq k \leq \min\left\{\left\lceil \frac{n}{j} \right\rceil, \left\lceil \frac{n}{\lceil z^* \rceil} \right\rceil, \overline{\alpha}(N(u)) + 1\right\},$$

y en caso de que $k = \lceil \frac{n}{\lceil \frac{n}{k-1} \rceil - 1} \rceil$, chequeamos que la desigualdad sea violada.

Estas desigualdades no pueden ser añadidas como cortes globales pues dependen de z^* que es una cota inferior de χ_{eq} válida sólo en el nodo actual y sus descendientes. Por lo tanto, las consideramos como *cortes locales*.

6.5.4. Separación de Desigualdades J-color

El problema de separación de las desigualdades J-color es el siguiente:

dada una solución fraccionaria (x^*, w^*), *debemos decidir si existe un conjunto* $J \subset \{1,\dots,n-1\}$ *tal que*

$$\sum_{j\in J}\sum_{v\in V} x^*_{vj} > \sum_{k=1}^{n} b_{Jk}(w^*_k - w^*_{k+1}),$$

con b_{Jk} definido en la Sección 5.4.4.

Para identificar desigualdades J-color violadas implementamos un algoritmo goloso, siguiendo la Observación 5.4.28.5. Primero verificamos que los w^* no sean todos enteros. En tal caso debe existir un w^*_t fraccionario tal que $w^*_{t+1} = 0$. Luego ordenamos en forma decreciente las clases de color $j \in \{1,\dots,t\}$ según la cantidad de variables fraccionarias x^* que presentan, i.e. el cardinal de $\{v : x^*_{vj} \notin \mathbb{Z} \ \forall v \in V\}$. Es decir que, si las clases de color ordenadas son C_{i_1}, C_{i_2}, …, C_{i_t}, entonces C_{i_1} es la que se compone de más variables fraccionarias. Luego para cada $2 \le s \le t-2$ tal que $s \ge 1 + \min\{n \bmod k : k \in \{1,\dots,t\} \wedge\ k \text{ no divide a } n\}$, probamos las desigualdades J-color donde J incluye las clases más fraccionarias y el cardinal de J es justamente s, i.e. $J = \{C_{i_1}, C_{i_2}, \dots, C_{i_s}\}$. Si la desigualdad J-color no es satisfecha por (x^*, w^*), entonces se añade como un *corte global purgable*.

En el procedimiento anterior sólo consideramos J tales que $2 \le |J| \le n-3$ pues, como dijimos en la Observación 5.4.28.1, desigualdades J-color con $|J| = 1$ son justamente las restricciones (4.23), y con $|J| = n-2$ definen la misma cara que (4.22).

6.5.5. Separación de Desigualdades Block

El problema de separación de estas desigualdades es:

a partir de una solución fraccionaria (x^*, w^*) *y un color* j *tal que* $0 < w^*_j < 1$, *decidir si existe un vértice* v *tal que* $\sum_{k=j}^{n} x^*_{vk} > w^*_j$.

La cantidad total de desigualdades block es del orden de n^2 por lo que podemos enumerarlas y probarlas, una a una, agregándolas como cortes cuando sean violadas por la solución fraccionaria. Este es el método utilizado para separarlas en el contexto del PCG [62].

En nuestra implementación manejamos estas desigualdades como *cortes globales purgables*.

6.5.6. Separación de Desigualdades Subvecindad

Para las desigualdades (u, j, S)-subvecindad sólo nos restringimos al caso en que $S = N(u)$. Además, vértices u tales que $\alpha(N(u)) \le 2$ dan lugar a desigualdades clique o 2-rango, por lo que debemos imponer que $\alpha(N(u)) \ge 3$. El problema de separación es, entonces:

decidir si para una solución fraccionaria (x^*, w^*) *y un color* j *tal que* $w^*_j > 0$ *existe un vértice* u *tal que* $\alpha(N(u)) \ge 3$ *y*

$$\gamma_{jN(u)} x^*_{uj} + \sum_{v\in N(u)} x^*_{vj} + \sum_{k=j+1}^{n} (\gamma_{jN(u)} - \gamma_{kN(u)}) x^*_{uk} > \gamma_{jN(u)} w^*_j,$$

donde $\gamma_{kN(u)} = \min\{\lceil n/\chi_{eq} \rceil, \lceil n/k \rceil, \alpha(N(u))\}$ para todo k.

En el problema de separación requerimos conocer el número de estabilidad del vecindario de cada vértice. Este tema ya fue discutido al tratar las desigualdades clique-vecindad, de manera que vamos a

asumir la disponibilidad de cotas inferiores y superiores de este parámetro, a saber $\underline{\alpha}(N(u))$ y $\overline{\alpha}(N(u))$ para todo vértice $u \in V$. Al igual que en la separación de desigualdades clique-vecindad, vamos a considerar desigualdades más débiles que las subvecindad.

Para todo $u \in V$ tal que $\underline{\alpha}(N(u)) \geq 3$, identificamos desigualdades violadas de la forma:

$$\xi_j x_{uj} + \sum_{v \in N(u)} x_{vj} + \sum_{k=j+1}^{n} (\xi_j - \xi_k) x_{uk} \leq \xi_j w_j,$$

donde $\xi_k = \min\{\lceil \frac{n}{\lfloor z^* \rfloor} \rceil, \lceil \frac{n}{k} \rceil, \overline{\alpha}(N(u))\}$. Debido a que $\xi_k \geq \gamma_{kN(u)}$ esta desigualdad es una versión más débil de la $(u, j, N(u))$-subvecindad (5.8).

Estas desigualdades deben ser agregadas como *cortes locales* pues dependen de z^* que es una cota inferior de χ_{eq} válida sólo en el nodo actual y sus descendientes.

6.5.7. Separación de Desigualdades Fuera-vecindad

El problema de separación de las desigualdades fuera-vecindad es:

decidir si para una solución fraccionaria (x^*, w^*) y un color j tal que $w_j^* > 0$, existe un vértice u tal que

$$\left(\left\lfloor \frac{n}{t_j} \right\rfloor - 1 \right) x_{uj}^* - \sum_{v \in V \setminus N[u]} x_{vj}^* + \sum_{k=t_j+1}^{n} b_{jk} x_{uk}^* > \sum_{k=t_j+1}^{n} b_{jk} (w_k^* - w_{k+1}^*),$$

donde $t_j = \max\{j, \chi_{eq}\}$ y $b_{jk} = \lfloor n/t_j \rfloor - \lfloor n/k \rfloor$.

Similarmente a lo que ocurre para las desigualdades subvecindad, asumimos la disponibilidad de cotas inferiores y superiores de $\alpha(N(u))$ para todo vértice $u \in V$, y consideramos desigualdades más débiles que las fuera-vecindad.

Para identificar desigualdades violadas computamos el valor $\tau_j = \max\{j, \lceil z^* \rceil\}$ y, para cada vértice $u \in V$ tal que $\underline{\alpha}(N(u)) \geq 3$, verificamos si $\underline{\alpha}(N(u)) \geq \lfloor n/\tau_j \rfloor$ según el resultado del Teorema 5.4.14. En caso afirmativo, consideramos la desigualdad que resulta de reemplazar t_j con τ_j en (5.9), i.e.

$$\left(\left\lfloor \frac{n}{\tau_j} \right\rfloor - 1 \right) x_{uj} - \sum_{v \in V \setminus N[u]} x_{vj} + \sum_{k=\tau_j+1}^{n} \beta_{jk} x_{uk} \leq \sum_{k=\tau_j+1}^{n} \beta_{jk} (w_k - w_{k+1}),$$

donde $\beta_{jk} = \lfloor n/\tau_j \rfloor - \lfloor n/k \rfloor$.

Al igual que las desigualdades subvecindad, estas desigualdades deben ser agregadas como *cortes locales* pues dependen de z^*.

6.5.8. Separación de Desigualdades (S, J)-color

El problema de separación asociado a estas desigualdades consiste en:

decidir si para una solución fraccionaria (x^*, w^*) existen conjuntos $S \subset V$ y $J \subset \{1, \ldots, n-2\}$ tales que $|S| = |J| + 1$ y

$$\sum_{j \in J} \sum_{v \in S} x_{vj}^* + \sum_{v \in V} x_{vn-1}^* > \sum_{k=1}^{n} b_{SJk} (w_k^* - w_{k+1}^*),$$

donde b_{SJk} es un valor definido en la Sección 5.4.5 que depende del número de estabilidad de S.

Como hemos hecho en casos anteriores, utilizamos una versión más débil de estas desigualdades, que se diferencian de las originales en que a cada ocurrencia de $\alpha(S)$ la reemplazamos por una cota superior de $\alpha(S)$ calculada con el procedimiento CLIQUEPART.

Hemos probado con varios criterios golosos para elegir los conjuntos S y J, pero en ningún caso estos procedimientos generaron una cantidad razonable de desigualdades violadas. Lo más frecuente era que no se generara ningún corte a lo largo de la optimización. Por esta razón, no las consideramos para que formaran parte del algoritmo de planos de corte.

6.5.9. Separación de Restricciones (4.18)

En la Sección 6.1.4 planteamos la posibilidad de no incorporar las desigualdades (4.18) en la relajación inicial y de separarlas a medida que son violadas por la solución fraccionaria actual.

Al igual que las block, la cantidad total de desigualdades (4.18) es polinomial respecto de n por lo que las generamos con una rutina de separación enumerativa. Para todo vértice $2 \leq v \leq n$ y para todo color $2 \leq j \leq v-1$ tal que $x^*_{vj} > 0$, se evalúa si es posible agregar $x_{vj} \leq \sum_{u=j-1}^{v-1} x_{uj-1}$ como corte.

Como estás desigualdades pueden eliminar soluciones enteras, las añadimos como *corte local* acorde a lo especificado en el Manual de CPLEX.

6.5.10. Parámetros del algoritmo de planos de corte

Experiencias previas nos indicaron que las desigualdades block, subvecindad, fuera-vecindad y J-color no son violadas tan frecuentemente como en el caso de las desigualdades clique y 2-rango, ni producen cortes que eleven significativamente la cota inferior. Sin embargo, permiten explorar soluciones fraccionarias que proporcionan una retroalimentación a la separación de las desigualdades clique y 2-rango, haciendo que estas últimas se generen en mayor cantidad durante la siguiente iteración del algoritmo de planos de corte. Dicho de otro modo, las desigualdades block, subvecindad, fuera-vecindad y J-color estimulan la producción de desigualdades clique y 2-rango. Este hecho hace que sea necesario realizar un experimento que considere distintas combinaciones de rutinas de separación a fin de poder conocer aquella que minimice los tiempos de ejecución total; tarea que realizamos en el Capítulo 7.

Una vez definida la estrategia del algoritmo de planos de corte, deben tomarse ciertas decisiones relacionadas al funcionamiento de éste. Por ejemplo, después de resolver la relajación lineal de un nodo del árbol, se debe decidir si se generan cortes o se procede a realizar la bifurcación del nodo. Es de esperar que los cortes introducidos ayuden a mejorar las cotas y esto permita podar ramas del árbol. Sin embargo, el proceso de búsqueda de desigualdades violadas y la posterior resolución de la relajación con las desigualdades añadidas tienen un costo. Debe entonces lograrse un equilibrio entre estas dos posibilidades.

Las decisiones que intervienen en el comportamiento del algoritmo de planos de corte pueden ser traducidas como parámetros a determinar:

- $MAXROUNDS_{root}$. Máxima cantidad de iteraciones del algoritmo de planos de corte que pueden ser ejecutadas en el nodo raíz.
- $MAXROUNDS_{node}$. Máxima cantidad de iteraciones del algoritmo de planos de corte que pueden ser ejecutadas en cualquier nodo interior del árbol de búsqueda.
- $MAXDEPTHTREE$. Profundidad del árbol en donde se deja de ejecutar el algoritmo de planos de corte.
- $SKIPFACTOR_{cut}$. Parámetro que indica cada cuántos nodos evaluados se debe habilitar el algoritmo de planos de corte.
- $VIOL_{threshold}$. Márgen mínimo de violación para que un corte sea añadido. Si (x^*, w^*) es una solución fraccionaria y $\pi^X x + \pi^W w \leq \pi_0$ es el corte propuesto, entonces $\pi^X x^* + \pi^W w^* - \pi_0 > VIOL_{threshold}$ para que el corte sea considerado.

- $MAXCUTS$. Máxima cantidad de cortes que pueden ser agregados en una iteración del algoritmo de planos de corte para una determinada familia.

La fijación de estos parámetros conducen al siguiente comportamiento del algoritmo Branch-and-Cut:

1. Se resuelve la relajación inicial.
2. Se ejecutan hasta $MAXROUNDS_{root}$ rounds que introducen cortes en la relajación. En cada round se generan desigualdades válidas violadas de las cuales sólo se agregan a la relajación las primeras $MAXCUTS$ generadas por familia, y se reoptimiza la relajación. Si en un round determinado no se generan cortes, entonces no se ejecutan los rounds restantes.
3. Se comienza el proceso de Branch-and-Bound.
4. Cada $SKIPFACTOR_{cut}$ nodos se realiza lo siguiente. Si la profundidad es inferior a $MAXDEPTHTREE$ niveles, se ejecutan $MAXROUNDS_{node}$ rounds del algoritmo de planos de corte antes de bifurcar.

El esquema que elegimos es el mismo que el Branch-and-Cut implementado en [62] para el PCG, con la salvedad de que no ejecutamos el algoritmo de planos de corte a partir de un cierto nivel en el árbol de búsqueda (en el B&C para el PCG, $MAXDEPTHTREE = \infty$). Esta decisión se debe a que los cortes generados a ese nivel se vuelven ineficaces por la cantidad de variables fijadas.

El comportamiento del algoritmo también es sensible al parámetro $VIOL_{threshold}$. Este parámetro es uno de los más importantes pues establece la agresividad con que se producen cortes y la calidad de ellos. Un valor muy grande limita la cantidad de cortes generados y un valor muy pequeño hace que se produzcan una cantidad considerable de cortes, de los cuales muchos resultan improductivos. En nuestro caso decidimos elegirlo según la familia: para las desigualdades clique usamos $VIOL_{threshold} = 0{,}1$, para las desigualdades 2-rango usamos $VIOL_{threshold} = 0{,}2$ y para el resto usamos $VIOL_{threshold} = 0{,}01$.

En lo que respecta a la cantidad de cortes generados, fijamos $MAXCUTS = 200$ para todas las familias de desigualdades.

En el Capítulo 7 experimentamos con distintos valores para los parámetros $MAXROUNDS_{root}$, $MAXROUNDS_{node}$, $MAXDEPTHTREE$ y $SKIPFACTOR_{cut}$.

En la próxima página mostramos un pseudocódigo de la rutina de separación de desigualdades Clique CLIQUESEP.

Este capítulo fue destinado a describir las principales características de ECOPT y, a lo largo de él, fueron surgiendo parámetros que modelan su comportamiento. Es difícil establecer a priori cuál es la elección más adecuada para estos parámetros y sólo a través de la experimentación podemos estimarlos. El propósito del próximo capítulo será ajustar empíricamente los valores de estos parámetros.

Algoritmo 10: CLIQUESEP

Data: grafo G, color j, solución fraccionaria (x^*, w^*)

begin
  si $w_j^* = 0$, salir
  ordenar los vértices de G según el valor de x_{vj}^* de mayor a menor
  prohibidos $\leftarrow \{v : x_{vj}^* = 1 \vee v \text{ es aislado}\}$
  while *existan vértices* $v \in V \backslash$prohibidos *tales que* $x_{vj}^* > 0$ **do**
    elegir un vértice $v \in V \backslash$prohibidos tal que $x_{vj}^* > 0$
    contador $\leftarrow 0$
    while contador $< 3 \wedge N(v) \backslash$prohibidos $\neq \varnothing$ **do**
      clique $\leftarrow \{v\}$
      candidatos $\leftarrow N(v) \backslash$prohibidos
      while candidatos $\neq \varnothing$ **do**
        $v' \leftarrow$ vértice de mayor valor x_{vj}^* en candidatos
        clique $\leftarrow$ clique $\cup \{v'\}$
        candidatos $\leftarrow$ candidatos $\cap N(v')$
      end
      if clique *es maximal en* G **then**
        if $\sum_{v \in \text{clique}} x_{vj}^* > w_j^*$ **then**
          se agrega el corte clique $\sum_{v \in \text{clique}} x_{vj} \leq w_j$
          se agrega el último vértice de clique a prohibidos
        end
      end
      se agrega el segundo vértice de clique a prohibidos
      contador $\leftarrow$ contador $+ 1$
    end
    se quitan de prohibidos los vértices agregados en el bucle anterior
    prohibidos $\leftarrow$ prohibidos $\cup \{v\}$
  end
end

Capítulo 7

Experiencia Computacional

Este capítulo está dedicado a realizar experimentos computacionales con nuestro algoritmo Branch-and-Cut ECOPT. Se evalúa la performance del mismo con distintas elecciones de parámetros. Una vez obtenida aquella combinación de parámetros que mejor se desempeñe, se compara ECOPT con otros algoritmos de la literatura.

Los experimentos se llevaron a cabo en un servidor equipado con un procesador AMD Phenom 2.2Ghz, sistema operativo Suse Linux y el optimizador de propósito general CPLEX 12.1 [4]. El algoritmo ECOPT está implementado en el lenguaje C utilizando la interfaz avanzada de CPLEX.

Para permitir una futura comparación válida aunque aproximada entre los resultados relacionados al tiempo, es común en la literatura reportar el tiempo consumido por el programa *dfmax* [1], el cual es utilizado para propósitos de *benchmarking* justamente. En nuestro caso, éste arrojó 6.31 segundos (user time) para la instancia *r500.5.b*.

7.1. Selección del modelo de programación lineal entera

En esta sección vamos a comparar las formulaciones ECF, ECF_1 y ECF_2 introducidas en el Capítulo 4. Para cada formulación corremos un Branch-and-Bound puro sobre 50 instancias de 40 vértices generadas aleatoriamente (10 instancias por grado de densidad), con un tiempo máximo de 2 horas por instancia. Luego tomamos ciertos indicadores que nos permitan interpretar los resultados:

(i) *Cantidad de instancias resueltas*: Es la cantidad de instancias (por grado de densidad) en las cuales la optimización alcanza la optimalidad dentro del tiempo máximo establecido. Este factor es determinante a la hora de elegir un modelo ya que se pretende que el mismo resuelva la mayor cantidad de instancias posibles.

(ii) *Promedio de la relajación inicial*: Es el promedio sobre todas las instancias (de la misma densidad) del valor de la función objetivo correspondiente a la formulación sin las restricciones de integralidad. En general se espera que un modelo arroje el mayor valor posible, ya que este hecho mejorará la cota inferior de χ_{eq} durante el proceso de Branch-and-Bound.

(iii) *Promedio de gap relativo*: Es el promedio sobre todas las instancias (de la misma densidad) del gap relativo entre la cota inferior y superior del número cromático equitativo al finalizar la optimización. Si LB y UB son las cotas inferior y superior de χ_{eq} entonces el gap relativo se calcula con $100(UB - LB)/UB$. Se espera que este indicador sea lo más cercano a cero posible.

(iv) *Promedio de nodos evaluados*: Es el promedio sobre las instancias *resueltas* (de la misma densidad) de la cantidad de nodos evaluados para alcanzar la optimalidad. Un valor bajo en la cantidad de nodos evaluados indica que hubieron pocas bifurcaciones durante el proceso de Branch-and-Bound.

(v) *Promedio de tiempo transcurrido*: Es el promedio sobre las instancias *resueltas* (de la misma densidad) de la cantidad de tiempo transcurrido para alcanzar la optimalidad. Este factor también es determinante a la hora de elegir un modelo ya que es importante que éste consuma la menor cantidad de tiempo posible.

En cada ejecución del algoritmo Branch-and-Bound se utiliza la cota superior inicial $\overline{\chi_{eq}} = \Delta(G) + 1$. Para el caso de ECF_2, los vértices se ordenan según su grado en forma decreciente.

El cuadro 7.1 resume los resultados de la prueba. En cada fila se muestran los indicadores correspondientes a un grado de densidad particular. Las columnas 2-4 reportan la cantidad de instancias resueltas para cada formulación. Las columnas 5-7 muestran el promedio de la relajación lineal, las columnas 8-10 muestran el promedio de gap relativo, las columnas 11-13 muestran el promedio de nodos evaluados y las columnas 14-16 muestran el tiempo transcurrido en segundos. Los mejores resultados por fila son reportados en **negrita**. Un guión "–" significa que no es posible calcular el promedio por que ninguna instancia fue resuelta.

Densidad	Instancias resueltas			Prom. rel. lineal			Prom. gap relativo			Prom. nodos eval.			Prom. tiempo		
	ECF	ECF_1	ECF_2	*ECF*	ECF_1	ECF_2	*ECF*	ECF_1	ECF_2	*ECF*	ECF_1	ECF_2	*ECF*	ECF_1	ECF_2
10%	10	10	10	2	2	**2.85**	0%	0%	0%	62	80	**20**	1.1	1.4	**0.5**
30%	5	0	**10**	2	2	**3.85**	8%	16%	**0 %**	35790	–	**1878**	4112	–	**162**
50%	0	0	**9**	2	2	**4.63**	36.6%	26.1%	**1.1 %**	–	–	**9216**	–	–	**1919**
70%	0	0	**3**	2	2	**8.34**	63.2%	27.9%	**7.6 %**	–	–	**9507**	–	–	**3813**
90%	0	1	**5**	2	2	**17.30**	78%	23.6%	**4.6 %**	–	86397	**2304**	–	6769	**821**

Cuadro 7.1: Comparación de las formulaciones en grafos de 40 vértices.

Estos resultados reflejan una marcada diferencia entre la formulación ECF_2 y las otras. Observemos que con ECF_2 se pudieron resolver todas las instancias con 30% de densidad y casi todas las de 50%. Más aún, el valor de la relajación inicial de ECF_2 crece paralelamente con la densidad a diferencia de las otras formulaciones. En lo que respecta al gap relativo, para instancias de media a alta densidad, los valores dados por ECF_2 se alejan un mínimo de 20 puntos porcentuales comparados con los dados por ECF_1, y esta diferencia aumenta con los valores dados por *ECF*. Con el resto de los indicadores también se puede alcanzar una conclusión positiva sobre ECF_2.

7.1.1. Enumeración de los vértices

En la Sección 4.3.3 hemos dicho que el rendimiento de ECF_2 es suceptible a la forma en que etiquetamos los vértices del grafo. El test que presentamos aquí consiste en comparar el comportamiento de ECF_2 con los distintos ordenamientos propuestos en la Sección 6.1.1. Para cada caso corremos un Branch-and-Bound puro sobre 50 instancias aleatorias de 50 vértices (10 instancias por grado de densidad), con un tiempo máximo de 2 horas. Antes de comenzar la optimización se computa una cota inferior $\underline{\chi_{eq}}$ como se describe en la Sección 2.3. Los colores de la clique obtenida durante esta etapa serán fijados durante la generación del modelo. Como cota superior se usa $\Delta(G) + 1$.

El cuadro 7.2 reporta los resultados obtenidos, en donde se emplea letra **negrita** para enfatizar el mejor valor. Las columnas 2-4 muestran la cantidad de instancias resueltas por la formulación ECF_2 con los ordenamientos ascendente (*Asc*), descendente (*LF*) y *Smallest Last* (*SL*). Las columnas 5-7 y 8-10 muestran el promedio de gap relativo y tiempo transcurrido respectivamente.

Densidad	Instancias resueltas			Prom. gap relativo			Prom. tiempo (en segundos)		
	Asc	*LF*	*SL*	*Asc*	*LF*	*SL*	*Asc*	*LF*	*SL*
10%	10	10	10	0%	0%	0%	2.6	1.7	1.5
30%	**9**	**9**	8	**1.4%**	**1.4%**	1.8%	1086	**415**	1334
50%	3	**4**	2	14.2%	**6.7%**	12.1%	4649	**1949**	2830
70%	0	2	**5**	14.5%	6.2%	**4.2%**	–	5307	**4897**
90%	10	10	10	0%	0%	0%	**32**	60	433

Cuadro 7.2: Comparación de los ordenamientos en grafos de 50 vértices.

En términos generales, el ordenamiento ascendente no produce buenos resultados debiendo volcar nuestra decisión en los ordenamientos *LF* o *SL*. El primero de ellos presenta mejor comportamiento en los grafos con densidades 30%, 50% y 90%. En cambio, para grafos de 70% de densidad, *SL* es mejor en promedio. Para poder analizar con más profundidad lo que ocurre en estas instancias, reportamos en el Cuadro 7.3 la cantidad de nodos evaluados y el tiempo transcurrido para LF y SL en aquellas instancias de 50% y 70% de densidad que alguno de los dos resuelve.

Dens.	Nodos eval.		Tiempo	
	LF	SL	LF	SL
50%	**2137**	5684	**1184**	3787
	876	1718	**603**	1873
	1756	–	**1630**	–
	5132	–	**4378**	–
70%	**3328**	6850	**3726**	5690
	7281	**4419**	6888	**3375**
	–	**9867**	–	**4754**
	–	**17689**	–	**6615**
	–	**7491**	–	**4055**

Cuadro 7.3: Comparación entre LF y SL en instancias de 50% y 70% de densidad.

Observemos que *LF* presenta menores tiempos en 5 de las 9 instancias que figuran en la tabla y, en particular, en 3 de las 4 instancias que ambos resuelven simultáneamente. Teniendo en cuenta, además, que las instancias de 50% de densidad son las más difíciles de resolver en términos generales, elegimos a *LF* como el mejor ordenamiento.

Vale remarcar que la formulación, más allá del ordenamiento que se aplique, se comporta extremadamente bien para los casos con 10% de densidad con respecto a los casos con 30% de densidad o más, por lo que adoptaremos mayor cantidad de vértices para grafos con 10% de densidad en algunas de las pruebas posteriores.

7.2. Estrategias de recorrido del árbol de búsqueda

En la Sección 6.2.1 propusimos dos estrategias de branching, que llamamos AB-rule (*AB*) y BA-rule (*BA*), y comentamos que CPLEX cuenta con otras dos estrategias, llamadas Default (*Def*) y PseudoReducedCosts (*PRC*), que se comportan bien con nuestro modelo, entre otras estrategias disponibles. La siguiente prueba compara cada una de las estrategias mencionadas, corriendo un Branch-and-Bound puro sobre 50 instancias aleatorias (10 instancias por grado de densidad), con un tiempo límite de 2 horas.

Vért.	Dens.	Prom. nodos evaluados				Prom. tiempo (en segundos)			
		Def	*PRC*	*AB*	*BA*	*Def*	*PRC*	*AB*	*BA*
90	10%	1470	847	556	**263**	18.0	7.4	8.2	**4.2**
60	30%	2940	2179	1573	**379**	45	30	24	**7**
60	50%	28209	9590	23516	**4912**	1039	285	613	**171**
60	70%	9159	1909	4503	**1326**	356	82	137	**43**
60	90%	57	87	100	**38**	1.4	1.4	0.9	**0.6**

Cuadro 7.4: Comparación de las estrategias de branching.

El cuadro 7.4 muestra los resultados obtenidos. Dado que todas las instancias fueron resueltas, sólo se reporta promedio de nodos evaluados y promedio de tiempo transcurrido. De lo que se puede apreciar, la estrategia BA-rule es incuestionablemente mejor que las demás.

También hemos probado con instancias de tamaño mayor y, aunque el Branch-and-Bound ya no es capaz de resolver todas las instancias, en los casos en donde usamos la estrategia BA-rule, se han resuelto mayor cantidad de instancias y el promedio de gap relativo resultó menor.

Vale la pena aclarar también que Default, i.e. la estrategia por defecto de CPLEX, tuvo la peor performance entre las 4 estrategias evaluadas.

7.2.1. Rutina de enumeración

El propósito de esta sección es determinar cómo y cuándo se debe ejecutar la rutina de enumeración implícita presentada en la Sección 6.3.

En primer lugar nuestro algoritmo se comporta muy bien para grafos de baja densidad y observamos que una enumeración implícita no produce una mejora en estos casos. Debido a ello, sólo la utilizaremos cuando el grado de densidad sea superior a 20%.

Comenzamos analizando la diferentes estrategias de selección. El cuadro 7.5 reporta (con el mismo formato que el Cuadro 7.3) la ejecución de un Branch-and-Bound que utiliza la estrategia de branching BA-rule y la rutina de enumeración sobre 50 instancias aleatorias de 70 vértices con un tiempo límite de 2 horas. La enumeración implícita se lleva cabo cuando a lo sumo 50 vértices están sin colorear en el coloreo parcial asociado a la rama del árbol actual. La estrategias a evaluar son *selección estática* (*SE*), *selección dinámica* en donde el nodo que asigna el color $k+1$ se evalúa *último* (SD_1) y *selección dinámica* en donde el nodo que asigna el color $k+1$ se evalúa *primero* (SD_2), donde k es el mayor color utilizado por el coloreo parcial.

Densidad	Instancias resueltas			Prom. gap relativo			Prom. tiempo (en segundos)		
	SE	SD_1	SD_2	*SE*	SD_1	SD_2	*SE*	SD_1	SD_2
30%	**10**	9	**10**	**0%**	2.2%	**0%**	335	**300**	335
50%	**7**	4	**7**	4.9%	8.2%	**2.4%**	3734	4023	**3629**
70%	3	4	**5**	8.1%	17.2%	**5.6%**	3619	3805	**3468**
90%	**10**	9	**10**	**0%**	1.2%	**0%**	741	1741	**1**

Cuadro 7.5: Comparación de las diferentes estrategias de selección

Claramente, la estrategia SD_2 es la que presenta el mejor desempeño en grafos de densidad media y alta. Para hacer una comparación más precisa en grafos de 30% de densidad, calculamos el promedio de tiempo de *SE* y SD_2 en las 9 instancias que resuelve SD_1. Estos promedios son 302 y 303 respectivamente, por lo que no existe una mejora apreciable al usar SD_1. Optamos entonces por SD_2.

Para poder determinar en qué momento debe ejecutarse la rutina de enumeración, consideramos los siguientes criterios basados en la cantidad de vértices sin colorear en el coloreo parcial asociado a la rama del árbol actual, que denotamos con $|U|$, y en la cota superior del nodo actual, que denotamos con UB.

- Criterio A_p: la enumeración se ejecuta cuando $|U| \leq p$.
- Criterio B_p: la enumeración se ejecuta cuando $|U|.UB \leq p$.

El cuadro 7.6 reporta la ejecución de un Branch-and-Bound que utiliza la estrategia de branching BA-rule y la rutina de enumeración sobre las mismas 50 instancias aleatorias con un tiempo máximo de 2 horas. Se evalúan los criterios A_{40}, A_{50}, A_{60} y B_{800}. También se reportan los resultados correspondientes a un Branch-and-Bound que *nunca* llama a la rutina de enumeración (NO).

Densidad	Instancias resueltas					Prom. gap relativo					Prom. tiempo (en segundos)				
	NO	A_{40}	A_{50}	A_{60}	B_{800}	NO	A_{40}	A_{50}	A_{60}	B_{800}	NO	A_{40}	A_{50}	A_{60}	B_{800}
30%	10	10	10	10	10	0%	0%	0%	0%	0%	338	340	335	190	**9**
50%	7	7	7	7	**10**	2.4%	2.4%	2.4%	4.5%	**0%**	4056	4126	3629	1459	**317**
70%	6	5	5	**7**	6	**2.6%**	4.6%	5.6%	8.6%	3.1%	3133	4347	3468	**1037**	3339
90%	10	10	10	10	10	0%	0%	0%	0%	0%	10	15	**1**	**1**	10

Cuadro 7.6: Comparación con distintos criterios para ejecutar la enumeración implícita

En grafos de alta densidad, la mejor estrategia resulta ser A_{60}. La ganancia principal se produce en grafos de 70% de densidad, donde A_{60} sólo necesita alrededor de un tercio del tiempo que utilizan las estrategias restantes. Sin embargo, en las 3 instancias que no se resuelven aplicando la estrategia A_{60}, la rutina de enumeración consume todo el tiempo disponible mientras que las estrategias restantes priorizan más la resolución de las relajaciones y, en consecuencia, la elevación de la cota inferior. Esto queda reflejado en el promedio de gap relativo.

Por otra parte, la estrategia B_{800} tiene un rendimiento excelente en grafos de 30% y 50% de densidad, y podemos considerarla *aceptable* en grafos de mayor densidad. Consideramos esta estrategia para ECOPT, esperando que la introducción de cortes en las relajaciones ayuden a mejorar el tiempo de proceso en grafos de 70% de densidad.

7.3. Incorporación de Heurísticas Primales

La siguiente prueba está destinada a evaluar distintas combinaciones de heurísticas primales cuando éstas son utilizadas en un Branch-and-Bound. Las combinaciones que consideramos son: ninguna heurística (NO), sólo heurística 1 (H_1), sólo heurística 2 (H_2), ambas heurísticas (H_1H_2) y sólo heurística de CPLEX (H_{CPX}). Para la prueba utilizamos 25 instancias (5 por cada grado de densidad). Las instancias de 10% de densidad son de 90 vértices mientras que el resto son de 70. El Cuadro 7.7 contiene la cantidad de nodos evaluados y el tiempo de ejecución para cada combinación. Además se reporta la mejor cota obtenida por la combinación en cuestión (esta cota no incluye las soluciones enteras que fueron encontradas al podar por optimalidad) y se marcan en **negrita** a los mejores resultados. Un guión "$-$" indica que la heurística evaluada no fue capaz de encontrar ninguna solución que mejore la cota superior actual al momento de ser ejecutada. Por cada instancia también se reporta el número cromático equitativo.

Las columnas correspondientes a las mejores cotas obtenidas por H_1 y H_2 nos pueden dar una idea de la capacidad de las heurísticas primales para generar soluciones. La heurística 1 es la que obtiene los coloreos con menor cantidad de colores en la mayoría de los casos aunque, eventualmente, la heurística 2 logra una solución de mejor calidad como en el caso de las instancias 6 y 13. El hecho de que la mejor solución sea obtenida a veces por una heurística y a veces por otra justifica considerar la combinación H_1H_2 como alternativa.

Inst.	Dens.	χ_{eq}	Nodos evaluados					Tiempo (en segundos)					Mejor cota obtenida			
			NO	H_1	H_2	H_1H_2	H_{CPX}	NO	H_1	H_2	H_1H_2	H_{CPX}	H_1	H_2	H_1H_2	H_{CPX}
1		5	129	129	129	129	129	**1**	**1**	**1**	**1**	3	–	–	–	–
2		4	531	531	543	543	**445**	7	7	7	7	**6**	–	6	6	**5**
3	10	5	**154**	**154**	**154**	**154**	161	**1**	**1**	**1**	**1**	3	–	–	–	**5**
4		5	1	1	1	1	**0**	0	0	0	0	0	–	–	–	**5**
5		5	151	149	149	149	**135**	3	3	3	3	3	5	5	5	5
6		8	1684	457	1673	457	**287**	80	28	81	29	**22**	12	11	12	**8**
7		8	845	842	845	842	**823**	**42**	43	43	45	45	**8**	–	**8**	**8**
8	30	8	4113	4113	4113	4113	4113	**154**	157	161	162	158	–	–	–	–
9		8	27949	27945	27949	27945	**25223**	855	918	898	897	**813**	**8**	–	**8**	**8**
10		8	7146	**2295**	7142	**2295**	7100	279	116	289	**108**	324	12	12	12	**8**
11		12	101321	**96465**	101321	**96465**	100022	5557	5609	5757	**5517**	5823	**13**	–	**13**	14
12		12	42263	41994	24419	42156	**19191**	3145	3212	1801	3243	**1453**	**13**	18	**13**	**13**
13	50	12	59588	57470	63189	**55909**	60028	4080	3984	4480	**3942**	4240	16	**14**	17	–
14		12	46735	42344	**40632**	**40632**	43670	2817	2589	**2455**	2480	2705	15	16	16	**12**
15		12	32264	32263	32264	32263	**31669**	2273	2303	2297	2320	**2265**	**13**	–	**13**	16
16		17	57177	**40819**	66147	41503	59635	3454	**3056**	3950	3088	3657	20	22	**19**	20
17		17	70417	70291	**45541**	**45541**	64732	4230	4448	**2767**	2799	4090	21	23	23	**19**
18	70	17	4558	**4557**	4558	**4557**	4569	338	314	**305**	310	323	**18**	–	**18**	–
19		18	4120	1374	5783	**1363**	3109	165	89	228	**87**	150	**18**	24	**18**	19
20		18	29748	**29724**	34609	**29724**	29751	1712	1575	1811	**1543**	1585	**19**	25	**19**	**19**
21		29	686	**583**	625	639	620	10	8	**7**	**7**	12	**29**	36	**29**	**29**
22		27	4011	**2664**	4003	**2664**	3998	54	**38**	54	**38**	60	**27**	31	**27**	–
23	90	36	17	**9**	17	**9**	10	0	0	0	0	0	**36**	39	**36**	**36**
24		27	118	**44**	119	**44**	567	4	**1**	4	**1**	12	**27**	33	**27**	–
25		28	562	**535**	651	648	558	**7**	**7**	**7**	8	9	**28**	34	**28**	–

Cuadro 7.7: Evaluación de las heurísticas primales en grafos de 70 y 90 vértices

Al comparar las combinaciones H_1, H_2 y H_1H_2, podemos concluir que la última hereda lo mejor de las anteriores: la instancia 20 se resuelve en menor tiempo debido a la primer heuristica, y lo mismo ocurre con la instancia 17 y la segunda heurística. La tabla también pone en evidencia que es preferible usar alguna de estas estrategias que no usar ninguna. La combinación H_1H_2 logra el mejor promedio con 1771 segundos en las instancias de densidad media (6 a 20) mientras que para NO, H_1 y H_2 este promedio es de 1945, 1896 y 1822 segundos respectivamente. Lo mismo ocurre al comparar promedio de nodos evaluados.

Por otra parte, la heurística de propósito general que brinda CPLEX logra, en algunos caso, cotas de buena calidad como se observa en la última columna de la tabla. Sin embargo, a medida que la densidad del grafo aumenta, esta heurística va perdiendo su eficiencia. En términos de tiempo, H_{CPX} obtiene un promedio de 1843 segundos para las instancias de densidad media contra los 1771 de H_1H_2. Para grafos de 90% de densidad, H_{CPX} es la que presenta el peor comportamiento entre todas las estrategias evaluadas.

Si bien las conclusiones anteriores sugieren que H_1H_2 es la mejor combinación, esta situación podría cambiar en grafos de mayor tamaño, ya que las heurísticas podrían insumir suficiente tiempo como para que resulte desfavorable incorporarlas. Aún en los casos en que las instancias no sean resueltas por ninguna combinación propuesta en un tiempo preestablecido, es deseable que la combinación elegida disminuya el intervalo donde se encuentra el número cromático equitativo.

El propósito de la próxima prueba es evaluar las combinaciones NO, H_1H_2 y H_{CPX} sobre 25 instancias de 150 vértices. Se ejecuta un Branch-and-Bound imponiendo un tiempo límite de 2 horas y se reportan los resultados en el Cuadro 7.8, organizados de la siguiente manera. Las columnas 3-5 muestran el gap relativo alcanzado durante la optimización. Las columnas 6-8 muestran la mejor cota superior encontrada y el tiempo transcurrido desde que comenzó la optimización hasta el momento en que se encuentra dicha cota, separados por un guión. Finalmente, las columnas 9 y 10 muestran la mejor cota superior obtenida por las

heurísticas que se están evaluando. Estos últimos valores se diferencian de los dados en las columnas 7 y 8 en que sólo corresponden a soluciones generadas por las heurísticas, mientras que los de las columnas 7 y 8 pueden corresponden también a soluciones provenientes de la integralidad de la relajación de un nodo.

En el cuadro se marcan en **negrita** a los mejores resultados. La forma de reconocer cuál es el mejor resultado en las columnas 6-8 es eligiendo la menor cota superior reportada y, en caso de empate, el menor tiempo para alcanzar dicha cota.

Inst.	Dens.	Gap relativo			Mejor cota superior - Tiempo			Mejor cota obtenida	
		NO	H_1H_2	H_{CPX}	NO	H_1H_2	H_{CPX}	H_1H_2	H_{CPX}
1		16	16	16	**6-188**	6-189	6-853	**6**	7
2		28	28	28	7-43	7-43	**7-9**	–	**7**
3	10	16	16	16	**6-129**	6-132	6-136	**7**	–
4		28	28	28	7-43	7-44	**7-36**	–	**7**
5		16	16	16	**6-35**	6-36	6-141	–	**7**
6		50	50	**46**	16-162	16-179	**15-5934**	17	**15**
7		50	50	50	16-456	**16-175**	16-516	20	**18**
8	30	50	50	**46**	16-369	16-246	**15-2311**	17	**15**
9		50	50	**46**	16-363	16-376	**15-2840**	16	16
10		46	46	46	15-1782	**15-1164**	15-1670	19	**16**
11		53	53	**52**	26-1052	26-454	**25-3162**	31	**26**
12		56	**54**	56	25-6455	**24-5896**	25-5869	31	**26**
13	50	52	**50**	57	25-3375	**24-2189**	26-5466	**27**	–
14		**52**	**52**	53	**25-2539**	25-5246	26-307	27	27
15		53	**52**	53	26-1711	**25-1170**	26-695	27	**26**
16		52	52	52	**38-4843**	38-5667	38-6949	**39**	–
17		69	**51**	57	55-0	**37-6319**	40-6571	**39**	–
18	70	66	**51**	**51**	56-0	39-6506	**39-3969**	40	**39**
19		51	51	51	39-6116	**39-6114**	39-6522	**39**	–
20		56	56	56	39-4131	**39-2423**	39-4865	**39**	56
21		19	**18**	**18**	56-6755	**55-2342**	55-5408	**55**	56
22		48	**21**	48	84-0	**55-4637**	84-0	**55**	–
23	90	7	**5**	7	55-2970	**54-1079**	55-2052	**54**	58
24		5	**3**	5	54-3178	**53-5911**	54-3419	**53**	–
25		10	**7**	10	57-5375	**55-4701**	57-6786	**55**	57

Cuadro 7.8: Evaluación de las heurísticas primales en grafos de 150 vértices

Para la combinación H_1H_2 impusimos un límite de 300 segundos en el tiempo que utilizan las heurísticas. Una vez superado este límite, ambas heurísticas se deshabilitan durante el resto de la optimización.

Puede verse en la tabla que la heurística de CPLEX es la más eficaz en las primeras instancias pero, al igual que en la prueba anterior, pierde su efectividad al aumentar la densidad del grafo. Esta relación se invierte para la combinación H_1H_2.

En particular, H_{CPX} encontró dificultades al intentar obtener soluciones en 7 instancias mientras que H_1H_2 no pudo hallar soluciones sólo en 3 instancias (todas ellas de 10% de densidad).

En términos de gap relativo, existió un beneficio al considerar H_{CPX} en las instancias de 30% de densidad. Para instancias de mayor densidad es preferible considerar H_1H_2. Más aún, el promedio de gap relativo sobre todas las instancias es de 40%, 37% y 38.5% para NO, H_1H_2 y H_{CPX} respectivamente, lo que supone una mejora para el Branch-and-Bound el considerar la combinación H_1H_2.

También hemos notado que el tiempo que consumen las heurísticas a lo largo de la optimización se incrementa notablemente con el tamaño del grafo, lo que nos llevó a imponer un límite de tiempo en su ejecución. En el caso de la heurística de CPLEX, no es posible controlar este límite.

En la primera prueba (Cuadro 7.7) registramos el mayor tiempo en la instancia 11, donde la combinación

H_1H_2 utilizó 36 segundos del tiempo total mientras que en la segunda (Cuadro 7.8) se alcanza el límite de 300 segundos en la mayoría de las instancias.

Debido a una limitación en la interfaz de CPLEX, no es posible conocer el tiempo preciso que insume la heurística primal de CPLEX a lo largo de la optimización. Aún así notamos que, en general, la heurística de CPLEX dedica una porción de tiempo considerable antes de que el algoritmo comience con el proceso de bifurcación. Por ejemplo, en la instancia 22 de la segunda prueba, utilizó alrededor de 196 segundos sólo en ese momento.

7.4. Evaluación de las desigualdades válidas

En esta sección realizamos experimentos relacionados al algoritmo de planos de corte a fín de poder ajustar los parámetros que lo gobiernan.

7.4.1. Evolución de la cota inferior en el nodo raíz

Una forma de evaluar la calidad de un algoritmo de planos de corte es analizando el crecimiento de la cota inferior a medida que los cortes generados son agregados a la relajación lineal inicial. En este experimento, comparamos la performance de ocho estrategias dadas en el Cuadro 7.9, donde cada una es una combinación de rutinas de separación que determinan el comportamiento del algoritmo de planos de corte. Estas combinaciones fueron elegidas cuidadosamente con el siguiente criterio. En primer lugar, las desigualdades cliques son fundamentales para el buen comportamiento general del algoritmo. Con respecto al resto de las familias, éstas se incorporan siguiendo un orden dado por el comportamiento que fuimos observando en experimentos preliminares en lo que respecta a costo de separación y beneficio.

Estrategia Nombre	Clique	2-rango	Block	J-color	Sub-vecindad	Fuera-vecindad	Clique-vecindad
S1	•						
S2	•	•					
S3	•	•	•				
S4	•	•	•	•			
S5	•	•	•	•	•		
S6	•	•	•	•	•	•	
S7	•	•	•	•			•
S8	•	•	•	•	•	•	•

Cuadro 7.9: Combinaciones de rutinas de separación

Vamos a ejecutar 30 iteraciones del algoritmo de planos de corte (para cada una de las estrategias propuestas anteriormente) sobre 40 instancias de 150 vértices, 8 instancias por cada grado de densidad. En cada ejecución calculamos los siguientes indicadores, en donde LB_i se define como el valor de la función objetivo de la relajación lineal luego de la iteración i-ésima:

- *Incr*: Incremento absoluto en la cota inferior, i.e. $Incr = \lceil LB_{30} \rceil - \lceil LB_0 \rceil$.
- *Iter*: Mínimo número de iteraciones necesarias para alcanzar la mejor cota inferior, i.e. $Iter = \text{mín}\{i : \lceil LB_i \rceil = \lceil LB_{30} \rceil\}$.
- *Tiempo*: Tiempo transcurrido en segundos hasta alcanzar la mejor cota inferior.

- *Cortes*: Cantidad de cortes generados hasta alcanzar la mejor cota inferior.

El indicador *Incr* es el más importante. Cuando no podamos comparar con *Incr*, analizaremos *Tiempo* o *Cortes*. Es decir que, entre dos estrategias que alcancen la misma cota inferior, optaremos por aquella que lo hace en el menor tiempo posible. Por otra parte, a mayor cantidad de cortes generados, la relajación resultante será más pesada de resolver y es un buen criterio elegir aquella estrategia en donde se generen menos cortes.

El indicador *Iter* será útil para determinar un promedio de iteraciones necesarias para alcanzar la mejor cota inferior.

También reportaremos, por cada instancia, el *gap absoluto inicial* (la diferencia entre $\lceil LB_0 \rceil$ y una cota superior inicial suministrada por TABUEQCOL). Este valor nos permite evaluar si el incremento absoluto en la cota inferior es o no significativo, pues nos da una idea de cuánto es el margen para poder elevar la cota inferior.

Para grafos con 10% de densidad, ninguna estrategia dió señales de mejora en la cota inferior ($Incr = 0$ en todos los casos). Esto no presenta una dificultad ya que el gap absoluto resultó ser de 1 unidad.

Estrategia	Instancia 1 ($gap = 7$)				Instancia 2 ($gap = 7$)				Instancia 3 ($gap = 7$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	1	7	32.61	1400	1	11	68.24	2200	2	22	211.01	4400
S2	1	5	63.19	1371	1	5	58.65	1407	2	11	188.76	3071
S3	1	5	47.83	1380	1	5	63.21	1442	2	11	162.66	3107
S4	1	5	47.75	1380	1	5	63.14	1442	2	11	162.7	3107
S5	1	5	47.72	1380	1	5	59.49	1455	2	12	226.8	3355
S7	1	5	67.25	2010	1	5	81.34	2099	2	11	250.94	3990
S8	1	5	65.56	2012	1	5	78.98	2102	2	11	234.52	3990
	Instancia 4 ($gap = 7$)				Instancia 5 ($gap = 7$)				Instancia 6 ($gap = 6$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	1	20	59.53	4000	2	27	203.9	5400	1	16	159.45	3200
S2	1	5	20.12	1299	2	13	245.61	3564	1	7	132.91	1928
S3	1	5	20.09	1299	2	13	268.9	3602	1	7	132.32	1963
S4	1	5	20.33	1299	2	13	268.62	3602	1	7	118.26	1959
S5	1	5	20.16	1299	2	13	232.25	3579	1	7	144.2	1968
S7	1	6	28.91	2355	2	13	313.91	4506	1	7	153.39	2685
S8	1	6	28.86	2355	2	13	275.81	4491	1	7	160.21	2714
	Instancia 7 ($gap = 6$)				Instancia 8 ($gap = 7$)				**Promedio**			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	1	15	137.32	3000	2	22	187.97	4400	1.38	17.50	132.50	3500.00
S2	1	8	111.67	2232	2	9	167.43	2484	1.38	7.88	123.54	**2169.50**
S3	1	8	112.98	2234	2	9	151.18	2504	1.38	7.88	119.90	2191.38
S4	1	8	112.87	2234	2	9	151.5	2504	1.38	7.88	**118.15**	2190.88
S5	1	8	114.15	2242	2	9	153.17	2497	1.38	8.00	124.74	2221.88
S7	1	8	144.78	2960	2	9	212.55	3091	1.38	8.00	156.63	2962.00
S8	1	8	137.88	2959	2	9	192.17	3122	1.38	8.00	146.75	2968.13

Cuadro 7.10: Comparación para grafos de 30% de densidad.

Los resultados correspondientes a grafos con 30% de densidad o más se resumen a continuación:

- *Grafos con 30% de densidad* (Cuadro 7.10). No mostramos los resultados correspondientes a la estrategia S6 puesto que no se ha generado ningún corte fuera-vecindad. Estos resultados son prácticamente iguales a lo de S5 debido a que el tiempo usado para separar dichas desigualdades es despreciable.
  En el cuadro se puede observar que todas las estrategias alcanzan el mismo incremento en la cota

inferior, por lo que debemos desempatarlas con *Tiempo* o *Cortes*. Las estrategias S2-S4 tienen un comportamiento bastante similar aquí. Podemos inferir que se da un salto cuantitativo al emplear las desigualdades 2-rango debido a que se requieren alrededor de 1300 cortes menos para alcanzar la misma cota inferior, y con 10% menos de tiempo. Las desigualdades subvecindad no parecen mejorar esta situación y, definitivamente, las clique-vecindad no tienen buen desempeño.
Cabe mencionar que sólo en la Instancia 6 se generaron desigualdades J-color. Ellas lograron mejorar el tiempo de 132 segundos a 118 segundos.

Estrategia	Instancia 1 ($gap = 12$)				Instancia 2 ($gap = 12$)				Instancia 3 ($gap = 11$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	4	9	119.89	1800	3	16	213.42	3200	3	24	263.72	4800
S2	4	7	195.49	2025	3	9	396.63	2684	3	17	273.11	5068
S3	4	7	120.88	2070	3	9	406.63	2679	3	18	379.87	5396
S5	4	9	190.61	2955	3	9	554.25	2904	3	13	236.73	3992
S7	4	7	106.47	2026	3	9	420.59	2693	3	23	488.85	6926
S8	4	9	177.74	2933	3	9	413.06	2926	3	13	293.27	4066
	Instancia 4 ($gap = 12$)				Instancia 5 ($gap = 11$)				Instancia 6 ($gap = 12$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	4	16	131.07	3200	2	7	89.58	1400	4	19	171.16	3800
S2	4	13	212.14	3824	3	15	407.13	4460	4	17	261.94	5105
S3	4	11	171.76	3283	3	15	395.29	4460	4	16	259.69	4790
S5	4	13	287.77	4116	3	15	408.04	4722	4	18	255.91	5669
S7	4	11	176.51	3284	3	15	461.63	4490	4	13	182.63	3941
S8	4	13	386.03	4182	3	15	360.95	4789	4	13	176.89	4161
	Instancia 7 ($gap = 12$)				Instancia 8 ($gap = 11$)				**Promedio**			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	2	11	262.04	2200	1	17	389.51	3400	2.88	14.88	205.05	2975.00
S2	4	15	239.06	4497	2	11	331.13	3175	3.38	13.00	**289.58**	3854.75
S3	4	15	240.56	4509	2	11	361.74	3178	3.38	12.75	292.05	**3795.63**
S5	4	14	230.84	4362	2	11	360.92	3247	3.38	12.75	315.63	3995.88
S7	4	13	275.89	3926	2	11	436.17	3362	3.38	12.75	318.59	3831.00
S8	4	15	318.79	4723	2	11	374.75	3408	3.38	12.25	312.69	3898.50

Cuadro 7.11: Comparación para grafos de 50% de densidad.

- *Grafos con 50% de densidad* (Cuadro 7.11). No mostramos los resultados correspondientes a la estrategia S4 puesto que en sólo una instancia se ha generado un único corte J-color el cual no delató ningún impacto. Tampoco mostramos los resultados correspondientes a la estrategia S6 por el mismo motivo que para el caso 30% de densidad.
  En el cuadro hay 3 instancias en donde la estrategia compuesta únicamente de desigualdades clique no logra la misma cota que el resto de las estrategias, concluyendo así que es necesaria la incorporación de desigualdades 2-rango. Para comparar las estrategias S2-S8 consideramos los indicadores *Tiempo* y *Cortes*, y nuevamente las estrategias S2-S4 revelan el mejor desempeño y se vuelve a poner en evidencia que los cortes subvecindad y clique-vecindad no realizan ningún aporte.

Estrategia	Instancia 1 ($gap = 13$)				Instancia 2 ($gap = 13$)				Instancia 3 ($gap = 13$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	5	7	277.32	1400	5	26	1282.9	5200	5	16	817.64	3200
S2	6	13	786.72	3778	5	14	878.63	4143	6	20	1644.61	5982
S3	6	13	742.05	3772	5	14	882.32	4143	6	19	1489.64	5682
S4	6	13	765.19	3842	5	16	1086.1	4690	6	19	1465.47	5636
S5	6	14	935.4	4472	5	17	1065.09	5437	6	18	1591.68	5720
S6	6	14	1004	4565	5	14	964.75	4585	6	21	1881.16	6677
	Instancia 4 ($gap = 13$)				Instancia 5 ($gap = 15$)				Instancia 6 ($gap = 13$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	4	19	901.67	3800	7	11	555.82	2200	5	12	478.1	2400
S2	5	20	1333.54	5969	8	25	1612.74	7280	6	17	1077.1	4992
S3	5	20	1336.36	5969	8	20	1192.56	5833	6	21	1398.81	6107
S4	5	22	1692.29	6623	8	23	1386.36	6740	6	20	1349.84	5851
S5	5	20	1630.36	6265	8	19	1247.73	6096	6	16	1263.32	5148
S6	5	22	1665.65	6874	8	19	1286.17	6066	6	23	1626.17	7249
	Instancia 7 ($gap = 14$)				Instancia 8 ($gap = 13$)				**Promedio**			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	6	12	719.54	2400	3	9	449.54	1800	5.00	14.00	685.32	2800.00
S2	7	26	1646.18	7667	5	15	997.88	4501	6.00	18.75	1247.18	5539.00
S3	7	22	1404.78	6511	5	15	1182.61	4498	6.00	18.00	**1203.64**	**5314.38**
S4	7	20	1291.56	5914	5	14	958.19	4167	6.00	18.38	1249.38	5432.88
S5	7	20	1535.59	6513	5	15	1121.83	4832	6.00	17.38	1298.88	5560.38
S6	7	19	1462.05	6364	5	13	1001.53	4245	6.00	18.13	1361.44	5828.13

Cuadro 7.12: Comparación para grafos de 70% de densidad.

- *Grafos con 70% de densidad* (Cuadro 7.12). No mostramos los resultados correspondientes a las estrategias S7-S8 puesto que en sólo una instancia se ha generado un único corte clique-vecindad que no hizo ninguna modificación significativa.
  En el cuadro hay sólo una instancia en donde la estrategia S1 logra la misma cota que el resto de las estrategias, mostrándonos una vez más la mejora que produce incorporar las desigualdades 2-rango Entre las estrategias restantes, S3 arroja los mejores resultados seguidas de cerca por S2 y S4.
  En estas instancias aparecen por primera vez las desigualdades fuera-vecindad. Estas desigualdades producen una reducción sustancial de cortes en las instancias 2 y 8, pero esta situación se invierte en las instancias 3, 4 y 6. Con los tiempos ocurre algo similar por lo que no hay razón hasta ahora para considerar a estas desigualdades en el algoritmo de planos de corte.

Estrategia	Instancia 1 ($gap = 9$)				Instancia 2 ($gap = 8$)				Instancia 3 ($gap = 9$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	2	6	424.67	1200	1	2	151.64	400	2	10	761.04	1971
S2	3	25	2283.65	6803	2	22	2025.91	5857	3	22	1806.85	6069
S3	3	27	2710.34	7441	2	21	1914.75	5664	3	22	1921.39	6110
S4	3	25	884.8	6588	2	22	678.19	5549	3	22	811.53	5550
S5	3	25	957.98	6588	2	22	725.26	5549	3	20	772.28	5196
S6	3	24	858.01	6283	2	20	573.71	5009	3	21	981.42	5689
	Instancia 4 ($gap = 9$)				Instancia 5 ($gap = 8$)				Instancia 6 ($gap = 9$)			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	2	22	1687.65	3930	2	16	1215.02	3200	2	2	155.06	400
S2	2	13	1167.45	3144	3	24	2225.88	6556	3	20	1568.17	5390
S3	2	13	1272.69	3110	3	22	2166.81	6143	3	19	1851.64	5105
S4	2	13	339.34	2964	3	23	862.92	6015	3	19	679.43	5058
S5	2	13	360.45	2964	3	22	911.61	5757	3	19	739.5	5058
S6	2	12	312.45	2759	3	21	820.67	5558	3	19	683.94	5058
	Instancia 7 ($gap = 8$)				Instancia 8 ($gap = 8$)				**Promedio**			
	Incr	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*	*Incr*	*Iter*	*Tiempo*	*Cortes*
S1	1	3	150.34	576	2	10	536.95	1974	1.75	8.88	635.30	1706.38
S2	2	29	2449.53	7747	3	25	1750.08	6370	2.63	22.50	1909.69	5992.00
S3	2	29	2640.59	8044	3	26	1828.54	6685	2.63	22.38	2038.34	6037.75
S4	2	28	982.11	6980	3	26	762.16	6374	2.63	22.25	**750.06**	5634.75
S5	2	28	1071.3	6980	3	26	867.82	6436	2.63	21.88	800.78	5566.00
S6	2	27	1037.32	6778	3	26	832.15	6527	2.63	21.25	762.46	**5457.63**

Cuadro 7.13: Comparación para grafos de 90% de densidad.

- *Grafos con 90% de densidad* (Cuadro 7.13). No mostramos los resultados correspondientes a las estrategias S7-S8 puesto que no se han generado cortes clique-vecindad en ninguna instancia.
  Aquí, una vez más, se pone en evidencia el impacto de las desigualdades 2-rango sobre *Incr*. De las estrategias que utilizan estas desigualdades, S6 es la que genera en promedio menos *Cortes*. Esto muestra que podría ser apropiado tener en cuenta a las desigualdades subvecindad y fuera-vecindad en grafos de alta densidad. No obstante, consideramos que el *Tiempo* es el criterio que debe predominar y, en este sentido, elegimos a S4.

En resumen, las estrategias que contemplan las desigualdades 2-rango pero no las clique-vecindad tienen una performance similar, optando por S4 como la mejor estrategia que balancea *Tiempo* y *Cortes*. Sin embargo, la estrategia que considera solamente desigualdades clique genera mucho menos cortes, haciendo que las relajaciones sean más livianas. De manera que para poder saber qué estrategia se comportará mejor a largo plazo debemos realizar otro experimento que las compare.

Respecto a la cantidad de iteraciones necesarias para alcanzar el máximo incremento (*Iter*), tenemos que para las estrategias S2-S6 se requieren 0, 8, 13, 19 y 23 iteraciones aproximadamente, para grafos con grados de densidad 10%, 30%, 50%, 70% y 90% respectivamente. Consideramos entonces que aplicar 20 iteraciones del algoritmo de planos de corte es razonable y, por lo tanto, fijamos $MAXROUNDS_{root} = 20$ para las próximas pruebas.

7.4.2. Evaluación del algoritmo de planos de corte a largo plazo

En el siguiente experimento vamos a comparar un Branch-and-Bound puro con un *Cut-and-Branch* que realiza 20 iteraciones del algoritmo de planos de corte con las estrategias S1 o S4 seguido de un Branch-and-Bound, sobre un total de 50 instancias (10 instancias por cada grado de densidad).

El Cuadro 7.14 reporta los resultados organizados de la siguiente manera. Las columnas 3-5 muestran la cantidad de instancias resueltas por cada estrategia y las columnas 6-8, 9-11 y 12-14 muestran el promedio de gap relativo, nodos evaluados y tiempo transucurrido respectivamente. Un guión "–" significa que no es posible calcular el promedio por que ninguna instancia fue resuelta.

Vért.	Dens.	Instancias resueltas			Prom. gap relativo			Prom. nodos evaluados			Prom. tiempo (en segundos)		
		B&B	S1	S4	B&B	S1	S4	B&B	S1	S4	B&B	S1	S4
90	10%	10	10	10	0%	0%	0%	3399	**1162**	2211	30.9	**13.6**	35.4
70	30%	10	10	10	0%	0%	0%	55080	43053	**42927**	2405	1943	**1776**
70	50%	0	0	**2**	11.5%	15.3%	**9.8%**	–	–	**51621**	–	–	**6192**
70	70%	1	2	**6**	10.6%	8.7%	**3.6%**	**19079**	24177	19237	**1497**	2495	3233
70	90%	10	10	10	0%	0%	0%	39382	3852	**1267**	1113	199	**68**

Cuadro 7.14: Comparación entre Branch-and-Bound y Cut-and-Branch con S1 o S4.

En grafos de 10% de densidad, la mejor opción es usar sólo cliques como cortes. Sin embargo, todas las instancias son resueltas en menos de un minuto por cualquiera de los algoritmos, por lo que no es tan relevante la elección de uno u otro. No ocurre lo mismo en grafos de mayor densidad, en donde el promedio de tiempo de CPU ya supera el minuto. En particular, en grafos de 30% de densidad, el Branch-and-Bound presentó la peor performance mientras que los otros dos están casi empatados. El Cut-and-Branch con la estrategia S4 evalúa apenas menos nodos y, en términos de tiempo, hay una reducción del 9%.

Nodos evaluados			Tiempo (en segundos)		
B&B	S1	S4	B&B	S1	S4
19079	20903	**6928**	1497	1715	**841**
–	27451	**18106**	–	3274	**2566**
–	–	**17757**	–	–	**5157**
–	–	**23872**	–	–	**3735**
–	–	**32042**	–	–	**4469**
–	–	**16717**	–	–	**2631**

Cuadro 7.15: Comparación entre Branch-and-Bound y Cut-and-Branch con S1 o S4, en instancias de 70%.

En grafos de densidad media en adelante, se hace indispensable utilizar un Cut-and-Branch para poder resolver más instancias. En particular, con la estrategia S4 se logran los mejores resultados en cantidad de instancias resueltas y gap relativo. Sin embargo, no podemos concluir lo mismo al analizar cantidad de nodos evaluados y tiempo de CPU para grafos con 70% de densidad. Para poder determinar la estrategia vencedora, mostramos en el Cuadro 7.15 los resultados para las instancias resueltas. Aquí, un guión "–" indica que la instancia no fue resuelta en el tiempo límite establecido. Considerando aquellas instancias que se resuelven simultáneamente por S1 y S4, obtenemos un promedio de 2495 segundos para la primera estrategia y 1705 para la segunda por lo que, de ahora en más, elegimos a S4 como la estrategia de nuestro algoritmo de planos de corte.

7.4.3. Determinación de los parámetros del algoritmo de planos de corte

Recién vimos que utilizar un algoritmo de planos de corte en el nodo raíz con la estrategia S4 es conveniente. Sin embargo, la aplicación del algoritmo de planos de corte en nodos del árbol de enumeración puede mejorar aún más el rendimiento. Para lograr esto, el siguiente paso es determinar los parámetros: $SKIPFACTOR_{cut}$, $MAXROUNDS_{node}$ y $MAXDEPTHTREE$ definidos en la Sección 6.5.10.

Debido a la gran cantidad de combinaciones que hay que evaluar, optamos por comparar algunos parámetros teniendo fijados los restantes. En la primera prueba corremos el algoritmo *Branch-and-Cut* con los

parámetros $MAXROUNDS_{node} = 1$ y $MAXDEPTHTREE = \infty$, y variamos el parámetro $SKIPFACTOR_{cut}$ con los valores 1, 2 y 4. Recordemos que este parámetro determina la frecuencia con que se ejecuta el algoritmo de planos de corte en los nodos del árbol. Los resultados son reportados en el Cuadro 7.16 con el mismo formato que el Cuadro 7.14.

Vért.	Dens.	Inst. resueltas			Prom. gap relativo			Prom. nodos evaluados			Prom. tiempo (en segundos)		
		1	2	4	1	2	4	1	2	4	1	2	4
90	10%	10	10	10	0%	0%	0%	1247	2422	**754**	31	62.9	**15.4**
70	30%	10	10	10	0%	0%	0%	**6264**	7611	9779	**1950**	2065	2133
70	50%	1	**3**	0	**9.8%**	11%	11.9%	**5986**	9342	–	**3769**	5182	–
70	70%	**5**	4	4	5.5%	5.6%	**3.5%**	2400	**2378**	3377	2685	**1293**	1994
70	90%	10	10	10	0%	0%	0%	**142**	165	236	49.5	**44.3**	51.9

Cuadro 7.16: Comparación de Branch-and-Cut con distintos valores de $SKIPFACTOR_{cut}$.

Salvo en grafos de 10% de densidad, usar $SKIPFACTOR_{cut} = 4$ no es lo más conveniente, por lo que nuestra decisión se centrará en elegir entre valores 1 y 2 para $SKIPFACTOR_{cut}$. En grafos de 30%, usar $SKIPFACTOR_{cut} = 1$ alcanza el mejor promedio de tiempo seguido muy de cerca por $SKIPFACTOR_{cut} = 2$. En grafos de 90%, la situación se invierte.

Para grafos con densidades entre 50% y 70%, los resultados son dispares: la estrategia que presenta el menor tiempo también resuelve menos instancias. Por ello, hacer un análisis de lo que ocurre con cada instancia nos puede ayudar a desempatar entre ambas estrategias. En el Cuadro 7.17 mostramos la cantidad de nodos evaluados y tiempo transcurrido para cada instancia resuelta (por al menos una estrategia). Aquí, todas las instancias se resuelven más rápido con $SKIPFACTOR_{cut} = 2$ respecto de $SKIPFACTOR_{cut} = 1$. Optamos entonces por fijar $SKIPFACTOR_{cut} = 2$.

Dens.	Nodos evaluados			Tiempo (en segundos)		
	1	2	4	1	2	4
50%	**5986**	8175	–	3769	**3486**	–
	–	**9210**	–	–	**6331**	–
	–	**10642**	–	–	**5728**	–
70%	1510	1068	**405**	816	255	**152**
	3680	**3665**	7641	2676	**1968**	3640
	2493	2904	2960	4036	**1852**	2323
	1245	1874	2502	1267	**1095**	1862
	3073	–	–	**4631**	–	–

Cuadro 7.17: Comparación con distintos valores de $SKIPFACTOR_{cut}$, en instancias de 50% y 70%.

El Cuadro 7.18 realiza una comparación semejante a la anterior, pero esta vez evaluando distintos valores para $MAXROUND_{node}$, el parámetro que determina la cantidad de iteraciones del algoritmo de planos de corte en un nodo del árbol.

Para grafos con 10% y 90% de densidad, la mejor elección es usar el valor 1 o 4 en $MAXROUNDS_{node}$, y para grafos con 30% de densidad, el mejor valor para este parámetro es 2.

Sin embargo, en grafos de media-alta densidad (que suelen ser los más difíciles de resolver), no está muy claro cuál es la mejor estrategia pues, si bien $MAXROUNDS_{node} = 2$ utiliza menos tiempo de CPU, también resuelve menos instancias y empeora el promedio de gap relativo. Como hicimos en casos anteriores, reportamos en el Cuadro 7.19 el promedio de nodos evaluados y tiempo transcurrido para las instancias resueltas de 50% y 70% de densidad.

Vért.	Dens.	Inst. resueltas			Prom. gap relativo			Prom. nodos evaluados			Prom. tiempo (en segundos)		
		1	2	4	1	2	4	1	2	4	1	2	4
90	10%	10	10	10	0%	0%	0%	2422	2684	**2340**	**62.9**	75.6	**63.2**
70	30%	10	10	10	0%	0%	0%	7611	**5389**	6369	2065	**1397**	1782
70	50%	**3**	1	1	**11%**	12.3%	11.6%	9342	**3825**	4839	5182	**1709**	2675
70	70%	4	4	**6**	5.6%	5.6%	5.6%	2378	**1860**	2228	1293	**1274**	1936
70	90%	10	10	10	0%	0%	0%	165	153	**126**	**44.3**	50.9	**44.7**

Cuadro 7.18: Comparación de Branch-and-Cut con distintos valores de $MAXROUNDS_{node}$.

Dens.	Nodos evaluados			Tiempo (en segundos)		
	1	2	4	1	2	4
50%	10642	**3825**	4839	5728	**1709**	2675
	8175	–	–	**3486**	–	–
	9210	–	–	**6331**	–	–
70%	**1068**	1407	1651	**255**	462	728
	3665	–	3819	**1968**	–	2850
	–	**2602**	3597	–	**1527**	3164
	2904	2429	**1225**	1852	2483	**1349**
	1874	1003	**838**	1095	**624**	665
	–	–	**2237**	–	–	**2861**

Cuadro 7.19: Comparación con distintos valores de $MAXROUNDS_{node}$, en instancias de 50% y 70%.

Si bien la estrategia con $MAXROUNDS_{node} = 1$ no se desempeño bien en la instancia de la primera fila del Cuadro 7.19, consideramos que esta estrategia es la mejor opción para grafos de 50% de densidad pues resuelve 3 de las 10 instancias y su promedio de gap relativo es el menor de entre las 3 estrategias evaluadas. Además no hay suficiente evidencia que avale la elección de otro valor para $MAXROUNDS_{node}$ en vez del valor 1.

Para grafos de 70% de densidad, las estrategias tienen un desempeño bastante similar entre ellas por lo que las desempatamos mediante el cálculo de promedios. En el Cuadro 7.19 se puede observar que las instancias resueltas tanto por $MAXROUNDS_{node} = 1$ como por $MAXROUNDS_{node} = 2$ son las reportadas en la cuarta, séptima y octava fila. Si calculamos el promedio de tiempo sobre estas instancias obtenemos 1067 segundos para $MAXROUNDS_{node} = 1$ y 1190 segundos para $MAXROUNDS_{node} = 2$. Un análisis similar producto de promediar sobre la cuarta, quinta, séptima y octava fila arroja 1293 segundos para $MAXROUNDS_{node} = 1$ y 1398 segundos para $MAXROUNDS_{node} = 4$. Elegimos, por lo tanto, $MAXROUNDS_{node} = 1$.

En las pruebas anteriores hemos observado que, a medida que el nivel de profundidad del nodo aumenta, menos cortes son generados. Para evitar correr las rutinas de separación innecesariamente así como las posteriores reoptimizaciones de las relajaciones, utilizamos el parámetro $MAXDEPTHTREE$ que fija un nivel de profundidad límite para el algoritmo de planos de corte.

Para determinar $MAXDEPTHTREE$, realizamos una medición de la cantidad de cortes generados por nivel de profundidad, con datos extraídos de la estrategia ganadora en la prueba anterior. Para cada nivel de profundidad, sumamos los cortes generados por todos los nodos en dicho nivel sobre todas las instancias. La Figura 7.1 registra esta suma total para profundidades entre 5 y 100.

Si bien se generan cortes en casi todos los niveles del árbol, en su mayoría se encuentran comprendidos entre el nodo raíz y el nivel 50 de profundidad. Entonces resulta apropiado fijar $MAXTREEDEPTH$ en 50.

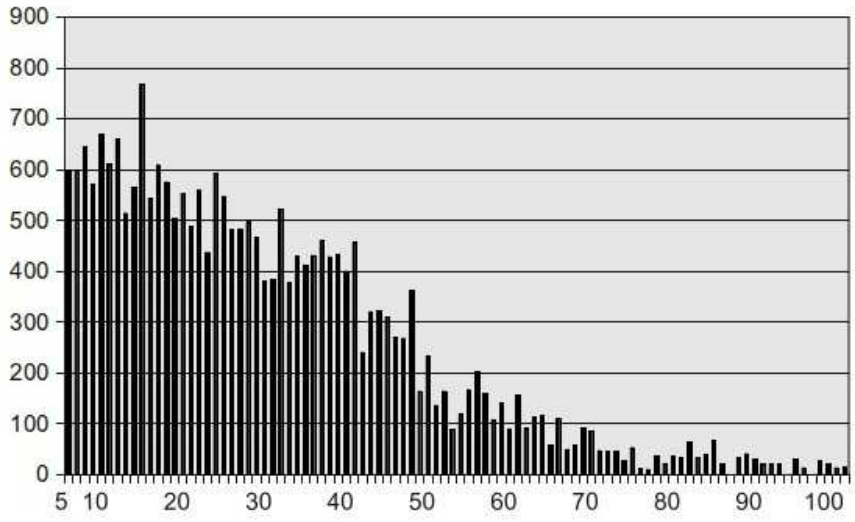


Figura 7.1: Cortes que se agregan a un nivel determinado de profundidad

7.4.4. Tiempo de separación

El tiempo insumido por los algoritmos de separación en todos los casos considerados está por debajo del 2% del tiempo total. Concluimos entonces que este factor no altera de manera apreciable el comportamiento de nuestro algoritmo.

7.4.5. Inclusión de cortes de CPLEX

Evaluamos la posibilidad de incluir cortes de propósito general además de los nuestros. CPLEX proporciona varias familias que pueden ser habilitadas mediante parámetros, quedando la estrategia de incorporación en manos de CPLEX y no pudiendo ser controlada por nuestro algoritmo.

Salvo para algunos casos aislados, no se produjo ninguna mejora en términos de tiempo de CPU por lo que decidimos no incluirlos.

7.4.6. Comparación entre Cut-and-Branch y Branch-and-Cut

Para estar seguros que la introducción de cortes en los nodos interiores del árbol de búsqueda resulte beneficiosa para nuestro algoritmo, realizamos un experimento que se resume en el Cuadro 7.20.

Vért.	Dens.	Inst. resueltas		Prom. gap rel.		Prom. nodos eval.		Prom. tiempo	
		C&B	B&C	C&B	B&C	C&B	B&C	C&B	B&C
90	10%	10	10	0%	0%	2211	**2166**	**35.4**	56.6
70	30%	10	10	0%	0%	32329	**6141**	**1578**	1819
70	50%	3	3	8.9%	**8.2%**	66733	**8540**	6372	**4962**
70	70%	**5**	4	6.5%	**4.5%**	21370	**4155**	3535	**2872**
70	90%	10	10	0%	0%	1267	**130**	67.8	**37**

Cuadro 7.20: Comparación entre Cut-and-Branch y Branch-and-Cut.

Lo primero que se observa es una disminución sustanciosa entre la cantidad de nodos evaluados por

uno y otro algoritmo, y no menos importante, una disminución en el promedio de gap relativo a favor del Branch-and-Cut. Por otro lado, el Cut-and-Branch logra tiempos inferiores en instancias de baja densidad y resuelve una instancia más que el Branch-and-Cut.

En el Cuadro 7.21 detallamos los resultados obtenidos para las instancias de 50% y 70% de densidad que fueron resueltas por uno u otro algoritmo. Se puede ver claramente la superioridad del Branch-and-Cut tanto en términos de nodos evaluados como de tiempo de CPU.

Dens.	Nodos eval.		Tiempo	
	C&B	B&C	C&B	B&C
50%	96957	**10248**	6732	**4996**
	52708	**7151**	5599	**4945**
	–	**8222**	–	**4944**
	50533	–	**6784**	–
70%	**17757**	–	**5157**	–
	–	**5220**	–	**3189**
	33408	**3469**	4579	**1994**
	6928	**353**	841	**229**
	–	**7579**	–	**6075**
	32042	–	**4469**	–
	16717	–	**2631**	–

Cuadro 7.21: Comparación entre Cut-and-Branch y Branch-and-Cut, en instancias de 50% y 70%.

Ya que para instancias de media a alta densidad resultó favorable introducir cortes en los nodos del árbol, además del nodo raíz, y que la performance puede considerarse aceptable en instancias de baja densidad, adoptamos esta estrategia para nuestra versión final de ECOPT.

7.5. Resultados finales

En esta sección comparamos ECOPT con el Branch-and-Cut genérico de CPLEX y el algoritmo B&C-LF_2 específico para el PCEG propuesto en [11, 12].

Hay que tener en cuenta que ECOPT utiliza el framework proporcionado por CPLEX pero con algunas funcionalidades de éste deshabilitadas (*búsqueda dinámica* y *reducciones* en la etapa de preprocesamiento) debido a consideraciones técnicas.

Las cotas iniciales suministradas tanto a CPLEX como ECOPT son computadas con los procedimientos vistos en los Capítulos 2 y 3 (TABUEQCOL). La ejecución de CPLEX se lleva a cabo con sus parámetros por defecto sobre la formulación ECF_2 con la enumeración LF.

En cuanto al algoritmo B&C-LF_2 sólo mostramos los resultados reportados en [11] y en el Congreso LAGOS celebrado en Noviembre de 2009. Hay que tener en cuenta que estos resultados se obtuvieron en un entorno diferente al nuestro, tanto hardware como software, y deben ser vistos sólo como referencias.

7.5.1. Comparación entre distintos algoritmos Branch-and-Cut en grafos aleatorios

En el Cuadro 7.22 presentamos los resultados correspondientes a instancias aleatorias de 60, 70 y 80 vértices con diferentes grados de densidad. Cada fila de la tabla corresponde a 10 instancias evaluadas del mismo grado de densidad, excepto para B&C-LF_2 que son 5 (estas últimas además son diferentes a

las que nosotros consideramos para CPLEX y ECOPT). Se reporta un guión "−" cuando el algoritmo no resolvió ninguna de las instancias con el mismo grado de densidad. La leyenda "?" que aparece en las columnas correspondientes a B&C-LF_2 significa que desconocemos el comportamiento del algoritmo en el caso contemplado.

Vért.	Dens.	Inst. resueltas			Prom. gap relativo			Prom. nodos			Prom. tiempo		
		CPX	ECO	LF_2	CPX	ECO	LF_2	CPX	ECO	LF_2	CPX	ECO	LF_2
	10%	100%	100%	100%	0%	0%	0%	**0.3**	1.6	12.6	**1.6**	**1.6**	21.2
	30%	**100%**	**100%**	40%	**0%**	**0%**	7.5%	1118	1002	**89**	53	**9.5**	86
60	50%	90%	**100%**	60%	0.9%	**0%**	3.6%	4811	113158	**338**	502	**16.5**	206
	70%	100%	100%	100%	0%	0%	0%	226	276189	**200**	72	**21.4**	71
	90%	100%	100%	100%	0%	0%	0%	24	6.8	**1.2**	5.7	**5.2**	9
	10%	100%	100%	100%	0%	0%	0%	**0**	**0**	57	**3.7**	**3.7**	109
	30%	60%	**100%**	0%	4.8%	**0%**	18.3%	**15383**	145670	–	1932	**16.5**	–
70	50%	0%	**100%**	0%	8%	**0%**	8.5%	–	**3902278**	–	–	**124**	–
	70%	50%	90%	**100%**	3.7%	0.5%	**0%**	3243	891	**678**	1447	329	**273**
	90%	100%	100%	100%	0%	0%	0%	**1.3**	3.3	9.4	**4.7**	5.2	11
	10%	100%	100%	?	0%	0%	?	36	**15.5**	?	6.9	**6.5**	?
	30%	10%	**100%**	?	12.1%	**0%**	?	**21319**	7576529	?	4200	**204**	?
80	50%	0%	0%	?	11.6%	**7.7%**	?	–	–	?	–	–	?
	70%	0%	**40%**	?	7%	**3.5%**	?	–	**4354**	?	–	**3227**	?
	90%	100%	100%	?	0%	0%	?	497	**124**	?	366	**41**	?

Cuadro 7.22: Comparación entre CPLEX (CPX), ECOPT (ECO) y B&C-LF_2 (LF_2) sobre grafos aleatorios.

En lo que respecta a cantidad de instancias resueltas y promedio de gap relativo, ECOPT fue el más eficiente entre los algoritmos evaluados. La excepción ocurre en los grafos con 70 vértices y 70% de densidad en donde B&C-LF_2 demuestra ser competitivo.

Las instancias más difíciles resultaron ser las de 80 vértices con 50% de densidad. Esto era lo esperado ya que es bien sabido que las instancias de densidad media presentan esta característica. Tanto ECOPT como la estrategia por defecto de CPLEX no fueron capaces de lidiar con estas instancias. No obstante, ECOPT logró disminuir el intervalo donde se encuentra el número cromático equitativo.

Las instancias con 70 vértices y 50% de densidad tampoco fueron resueltas por CPLEX y B&C-LF_2. Estas instancias no resultaron ser un obstáculo para ECOPT ya que pudo resolver todas en un tiempo promedio de 124 segundos.

B&C-LF_2 presenta la menor cantidad de nodos evaluados. Por el contrario, ECOPT evalúa una cantidad cuantiosa de nodos en algunos casos, poniendo en evidencia un uso más intensivo de la rutina de enumeración. Aún así, ECOPT obtiene los mejores resultados en términos de tiempo total de ejecución, lo que finalmente demuestra su robustez y competitividad frente a los otros algoritmos.

7.5.2. Comparación entre distintos algoritmos Branch-and-Cut en instancias de prueba

Aunque ECOPT demuestre ser competitivo en instancias aleatorias, aún nos queda por probar si esta tendencia se mantiene cuando comparamos a los algoritmos sobre instancias de prueba como las utilizadas en el Capítulo 3.

En el Cuadro 7.23 presentamos los resultados correspondientes a las instancias de prueba Kneser y COLORLIB de DIMACS que no fueron resueltas por optimalidad durante el cómputo de las cotas iniciales. La leyenda "?" que aparece en las columnas correspondientes a B&C-LF_2 significa que desconocemos el comportamiento del algoritmo en la instancia contemplada. Cuando "?" aparece en la cuarta columna indica que desconocemos el número cromático equitativo en esa instancia. En el caso de CPLEX y ECOPT, se asignó un tiempo límite de 4 horas. Cuando este tiempo es superado, se reporta un guión "−".

Inst.	$\|V\|$	$\|E\|$	χ_{eq}	Gap relativo			Nodos evaluados			Tiempo		
				CPX	ECO	LF_2	CPX	ECO	LF_2	CPX	ECO	LF_2
miles750	128	2113	31	0%	0%	0%	**0**	**0**	6	**7**	**7**	171
miles1000	128	3216	42	0%	0%	0%	**0**	**0**	13	**11**	12	267
miles1500	128	5198	73	0%	0%	0%	**0**	**0**	1	14	14	**13**
zeroin.i.1	211	4100	49	0%	0%	0%	**0**	1	1	**19**	20	50
zeroin.i.2	211	3541	36	0%	0%	0%	**0**	3	23	28	**24**	510
zeroin.i.3	206	3540	36	0%	0%	0%	4	**3**	28	28	**23**	491
queen6_6	36	290	7	0%	0%	0%	**0**	**0**	1	3	3	**1**
queen8_8	64	728	9	0%	0%	0%	2235	566379	**297**	166	**27**	441
queen9_9	81	1056	10	0%	0%	?	**40252**	412297132	?	**4670**	10255	?
queen10_10	100	1470	?	16%	16%	?	–	–	?	–	–	?
1-FullIns_3	30	100	4	0%	0%	0%	**0**	**0**	34	**1**	**1**	2
2-FullIns_3	52	201	5	0%	0%	0%	**0**	**0**	84	**1**	**1**	25
3-FullIns_3	80	346	6	0%	0%	0%	**0**	**0**	38	**1**	**1**	85
4-FullIns_3	114	541	7	0%	0%	0%	**0**	**0**	3	**1**	2	72
5-FullIns_3	154	792	8	0%	0%	0%	**0**	**0**	5	**2**	3	268
1-FullIns_4	93	593	5	0%	0%	?	**3170**	4133	?	**55**	60	?
mug88_1	88	146	4	0%	0%	?	1092	**281**	?	1	**0**	?
mug88_25	88	146	4	0%	0%	?	521	**183**	?	1	**0**	?
mug100_1	100	166	4	0%	0%	?	770	**579**	?	1	1	?
mug100_25	100	166	4	0%	0%	?	908	**755**	?	1	1	?
mulsol.i.1	197	3925	49	0%	0%	?	**0**	1	?	**16**	17	?
mulsol.i.2	188	3885	?	**5%**	29%	?	–	–	?	–	–	?
school1	385	19095	15	0%	0%	?	0	0	?	109	**40**	?
fpsol2.i.1	496	11654	65	0%	0%	?	0	0	?	**37**	38	?
fpsol2.i.2	451	8691	47	0%	0%	?	27	**24**	?	5204	**48**	?
fpsol2.i.3	425	8688	55	24%	**0%**	?	–	**41**	?	–	**79**	?
1-Insertions_4	67	232	5	20%	**0%**	?	–	**1887885**	?	–	**3968**	?
2-Insertions_3	37	72	4	0%	0%	?	1203	**353**	?	0	0	?
3-Insertions_3	56	110	4	0%	0%	?	21215	**17049**	?	**11**	14	?
4-Insertions_3	79	156	4	0%	0%	?	1587888	**294593**	?	**1131**	1300	?
DSJC125.1	125	736	5	0%	0%	?	0	0	?	30	30	?
DSJC125.5	125	3891	?	31%	31%	?	–	–	?	–	–	?
DSJC125.9	125	6961	?	6%	**2%**	?	–	–	?	–	–	?
DSJC250.1	250	3218	?	44%	44%	?	–	–	?	–	–	?
DSJC250.5	250	15668	?	67%	**52%**	?	–	–	?	–	–	?
DSJC250.9	250	27897	?	32%	**30%**	?	–	–	?	–	–	?
le450_15a	450	8168	15	6%	0%	?	–	**1121**	?	–	**1935**	?
le450_15b	450	8169	15	6%	0%	?	–	**459**	?	–	**854**	?
le450_5a	450	5714	5	0%	0%	?	**0**	313	?	**267**	8804	?
le450_5b	450	5734	5	0%	0%	?	**0**	226	?	**4624**	10048	?
inithx.i.1	864	18707	54	0%	0%	?	0	0	?	138	**116**	?
inithx.i.2	645	13979	?	75%	**53%**	?	–	–	?	–	–	?
myciel4	23	71	5	0%	0%	0%	**118**	555	237	**0**	**0**	5
myciel5	47	236	6	0%	0%	?	**45055**	160549	?	150	**2**	?
myciel6	95	755	?	**14%**	28%	?	–	–	?	–	–	?
flat300_20_0	300	21375	?	71%	**60%**	?	–	–	?	–	–	?
will199GPIA	701	6772	7	0%	0%	?	0	0	?	95	**46**	?
kneser7_2	21	105	6	0%	0%	0%	**333**	3211	357	**0**	**0**	6
kneser7_3	35	70	3	0%	0%	0%	**0**	**0**	4	**0**	**0**	2
kneser9_4	126	315	3	0%	0%	0%	**0**	**0**	4	**0**	**0**	809
kneser11_5	462	1386	3	0%	0%	?	**0**	113	?	229	**36**	?

Cuadro 7.23: Comparación entre CPLEX, ECOPT y B&C-LF_2 sobre instancias de prueba.

En primer lugar, las instancias resueltas por el algoritmo B&C-LF_2 no presentaron ningún obstáculo para CPLEX y ECOPT. En particular, en `miles750`, `miles1000`, `zeroin.i.2`, `zeroin.i.3`, `queen8_8`, `2-FullIns_3`, `3-FullIns_3`, `4-FullIns_3`, `5-FullIns_3` y `kneser9_4`, B&C-LF_2 tardó entre uno y dos órdenes de magnitud más que ECOPT.

Puede observarse en la tabla que la performance entre CPLEX y ECOPT fue bastante similar, aunque ECOPT logra resolver todas las instancias que resuelve CPLEX (un total de 37) y 4 más: `fpsol2.i.3`, `1-Insertions_4`, `le450_15a` y `le450_15b`.

Si consideramos que hubo un empate cuando la diferencia entre los tiempos que utilizan CPLEX y ECOPT para alcanzar la optimalidad es de a lo sumo 1 segundo, entonces de las 37 instancias que ambos resuelven, tenemos que en 22 instancias ocurrió un empate entre ambos Branch-and-Cut. Respecto a las instancias restantes, CPLEX logra mejores tiempos en 6 de ellas (`queen9_9`, `1-FullIns_4`, `3-Insertions_3`, `4-Insertions_3`, `le450_5a` y `le450_5b`) mientras que ECOPT resulta ser mejor en las otras 9 instancias (`zeroin.i.2`, `zeroin.i.3`, `queen8_8`, `school1`, `fpsol2.i.2`, `inithx.i.1`, `myciel5`, `will199GPIA` y `kneser11_5`).

Existen 10 instancias para las cuales ningún Branch-and-Cut fue capaz de alcanzar la optimalidad. En el Cuadro 7.24 se detallan las cotas inferiores y superiores que obtienen cada Branch-and-Cut después de las 4 horas de ejecución asignadas en estas instancias. Podemos notar que CPLEX obtiene mejores cotas en `mulsol.i.2` y `myciel6` mientras que ECOPT lo hace en `DSJC125.9`, `DSJC250.5`, `DSJC250.9`, `inithx.i.2` y `flat300_20_0`.

Instancia	Cota inferior			Cota superior		
	Inicial	CPLEX	ECOPT	Inicial	CPLEX	ECOPT
queen10_10	10	10	10	12	12	12
mulsol.i.2	31	34	34	51	**36**	48
DSJC125.5	9	13	13	19	19	19
DSJC125.9	42	43	43	47	46	**44**
DSJC250.1	4	5	5	9	9	9
DSJC250.5	11	11	**16**	34	34	34
DSJC250.9	63	63	**65**	93	93	93
inithx.i.2	30	30	**33**	122	122	**71**
myciel6	3	**6**	5	7	7	7
flat300_20_0	11	11	**15**	38	38	38

Cuadro 7.24: Mejoramiento de cotas con CPLEX y ECOPT sobre instancias no resueltas.

Resumimos en el Cuadro 7.25 la cantidad de instancias de prueba en donde un algoritmo u otro presentó mejor performance según los criterios tenidos en cuenta anteriormente.

	CPLEX gana	ECOPT gana	Hay empate	TOTAL
Instancias que ninguno resuelve	2	5	3	10
Instancias que ambos resuelven	6	9	22	37
Instancias restantes	0	4	0	4
Total de instancias evaluadas	8	15	25	51

Cuadro 7.25: Resumen de la comparación entre CPLEX y ECOPT sobre instancias de prueba.

Capítulo 8

Conclusiones

En esta tesis se desarrolló un algoritmo Branch-and-Cut para el Problema del Coloreo Equitativo de Grafos que denominamos ECOPT.

En [11] se presenta otro algoritmo Branch-and-Cut para el PCEG conocido como B&C-LF_2. Las instancias reportadas allí no presentan ninguna dificultad para ECOPT y generalmente este último supera a B&C-LF_2 en términos de tiempo de CPU. ECOPT resuelve, además, instancias que pueden considerarse de mayor dificultad, tanto en lo que se refiere al tamaño de la misma como a características estructurales tales como densidad o, en caso de las instancias DIMACS, aquellas que ya habían sido un desafío para el PCG.

Por otro lado, al comparar a ECOPT con el algoritmo Branch-and-Cut de propósito general de CPLEX, observamos que ambos presentan una performance similar sobre aquellas instancias que ambos resuelven. Sin embargo, ECOPT resuelve otras instancias que CPLEX no. Sobre aquellas que ambos no resuelven, ECOPT es capaz de reducir el gap de CPLEX en la mayoría de los casos.

Dentro de los diversos aspectos del problema que han sido abordados para el desarrollo de ECOPT, creemos importante destacar el diseño de una heurística inicial (TABUEQCOL) y el estudio poliedral de una formulación de Programación Lineal Entera para el PCEG.

En lo que respecta a la heurística inicial, además de significar un componente importante de ECOPT por la calidad de las cotas que brinda, constituye independientemente una herramienta para generar buenas soluciones del PCEG cuando el objetivo no es la optimalidad. Esto se desprende de la experiencia computacional reportada en el Capítulo 3, donde puede verse que esta heurística resuelve por optimalidad un porcentaje considerable de instancias de prueba y, en el caso en que no las resuelve, el gap es pequeño.

Por otra parte, el estudio realizado en el Capitulo 5 muestra claramente la dificultad, desde el punto de vista de un enfoque poliedral, de incorporar las restricciones de equidad al modelo para el PCG que, de por sí, ya es un problema muy complejo.

La primera dificultad la introducen las ecuaciones asociadas al conjunto $\mathscr{S}(G)$ en el sistema minimal del poliedro $\mathcal{ECP}$. Esto nos obliga a imponer, en algunos casos, la monotonía del grafo. También encontramos otros inconvenientes de tipo *aritméticos* asociados a la divisibilidad del número de vértices del grafo.

Sin embargo, la eficiencia como cortes de algunas de las desigualdades válidas presentadas muestran la importancia de este tipo de estudios teóricos. Esto se puso en evidencia en la Sección 7.4.2 cuando observamos un aumento significativo de la performance al comparar un Cut-and-Branch basado en estas desigualdades contra un Branch-and-Bound.

En conclusión, podemos afirmar que los resultados de esta tesis refuerzan la importancia de los algoritmos Branch-and-Cut que aprovechan la estructura particular de un problema de Optimización Combinatoria. Con el estudio realizado hasta este momento, el algoritmo resultante es competitivo contra otros existentes

y los supera en términos de instancias que cada uno de ellos puede resolver.

Creemos que aún queda bastante por recorrer en esta línea de investigación. En primer lugar, estamos lejos de haber dado un caracterización completa del poliedro de coloreo equitativo. Durante nuestra investigación hemos utilizado las herramientas PORTA [2] y y ZERONE [3]. Con estos programas hemos obtenido descripciones completas del poliedro sobre grafos de hasta 6 vértices y hemos observado que las familias de desigualdades válidas que enunciamos en el Capítulo 5 son sólo una pequeña fracción de la cantidad total de desigualdades válidas que definen facetas. Incluso grafos con estructuras tan simples como caminos y estrellas revelan una estructura poliedral muy complicada.

A modo de ejemplo, presentamos algunas familias de desigualdades válidas para $\mathcal{ECP}$ que describen facetas en las instancias pequeñas analizadas por los programas mencionados, pero cuyo estudio general queda pendiente:

- Sea un conjunto $S \subset V$ tal que S es α-maximal y $2 \leq \alpha(S) < \lceil n/\chi_{eq} \rceil$. Para todo los colores $1 \leq j \leq \lceil n/\alpha(S) \rceil - 1$, la siguiente desigualdad es válida para $\mathcal{ECP}$:

$$\sum_{v \in S} x_{vj} + \sum_{v \in V} x_{vn+1} \leq \sum_{k=j}^{n} \text{mín} \left\{ \alpha(S), \left\lceil \frac{n}{k} \right\rceil \right\} (w_k - w_{k+1}) + w_{n-1}.$$

- Sean $u_1, u_2, u_3 \in V$ tales que $(u_1,u_2),(u_2,u_3) \in E$ y $Q \subset V \backslash \{u_1,u_3\}$ una clique de G. Para todos los colores $j_1 \in \{1,\ldots,n-3\}$ y $j_2 \in \{\lceil n/2 \rceil,\ldots,n-2\}$ tales que $j_1 < j_2$, la siguiente desigualdad es válida para $\mathcal{ECP}$:

$$x_{u_1 j_1} + x_{u_2 j_1} + x_{u_3 j_1} + x_{u_1 j_2} + x_{u_3 j_2} + \sum_{v \in Q} x_{v j_2} + \sum_{v \in V} x_{v n-1} \leq 2w_{j_1} + w_{j_2} + w_{n-1} - w_n.$$

- Sea $m \geq 1$, $P = \{v_1, v_2, \ldots, v_{3m+1}\}$ un camino inducido en G y $P' = \{v_1, v_4, v_7, \ldots, v_{3m+1}\}$. Para todos los colores j_1 y j_2 tales que $j_1 < j_2 \leq \left\lceil \frac{n}{\lceil \frac{3m+1}{2} \rceil} \right\rceil$, la siguiente desigualdad es válida para $\mathcal{ECP}$:

$$\sum_{v \in P} x_{v j_1} + \sum_{v \in P'} x_{v j_2} \leq \left\lceil \frac{3m+1}{2} \right\rceil w_{j_1} + \left\lceil \frac{m}{2} \right\rceil w_{j_2}.$$

El estudio de éstas y otras desigualdades válidas no analizadas en esta tesis, con el objeto de evaluar su performance como nuevos cortes para ECOPT, es una de las líneas de investigación abiertas.

También resultaría interesante obtener la descripción completa de $\mathcal{ECP}$ por desigualdades lineales para familias de grafos con alguna estructura simple y evaluar la posibilidad de derivar un algoritmo polinomial para el PCEG.

Finalmente, la experiencia computacional también ha abierto algunas líneas a ser exploradas con mayor profundidad y que podrían significar mejoras importantes en el rendimiento de ECOPT. En particular, creemos que el algoritmo de enumeración implícita tiene mucho más potencial que el que hoy está aportando a ECOPT. Esta suposición se basa en la buena performance que el mismo ha demostrado como algoritmo de enumeración puro sobre algunas instancias de tamaños de hasta 80 vértices.

Con el avance del hardware, en los últimos años se ha puesto énfasis en mejorar los algoritmos de optimización para arquitecturas multiprocesadores. Ejemplos claros de este avance son *Gurobi* y el mismo CPLEX. Otra línea de investigación pendiente consistiría en adaptar ECOPT a una arquitectura de este tipo.

Por último, el estudio realizado en esta tesis puede ser adaptado sencillamente para el diseño de un algoritmo Branch-and-Cut para el *Problema de Coloreo Acotado*.

En su versión como problema de decisión, el Problema de Coloreo Acotado consiste en, dado un grafo G y enteros k y h, determinar si G admite un k-coloreo tal que el tamaño de todas las clases de color sea a lo sumo h. Este problema es NP-completo [17] puesto que puede ser convertido al problema de decisión del coloreo equitativo (y viceversa) mediante una simple transformación de G: un grafo G admite un k-coloreo con sus clases de color acotadas por h si y sólo si el grafo G' obtenido de G al agregar un conjunto estable de tamaño $hk-n$ admite un k-eqcol, y recíprocamente, un grafo G admite un k-eqcol si y sólo si el grafo G'' obtenido de G al agregar una clique de tamaño $\lceil n/k \rceil k - n$ admite un k-coloreo con sus clases de color acotadas por $\lceil n/k \rceil$.

Entre las aplicaciones del Problema de Coloreo Acotado podemos mencionar la planificación de tareas con exclusión mutua, en donde las tareas se deben ejecutar en a lo sumo h procesadores y dos tareas no pueden correr sobre el mismo procesador cuando están asociadas a intervalos de tiempo que no sean disjuntos, lo cual es modelado con un grafo de conflictos. Al resolver el Problema de Coloreo Acotado se obtiene una asignación compatible de tareas a procesadores. Estos problemas de planificación se utilizan para evitar la sobrecarga de un procesador al resolver ecuaciones diferenciales [14, 72]. Otras aplicaciones del Problema de Coloreo Acotado involucran planificación de sistemas de comunicación [45] y planificación de horarios [52]. Hasta lo que conocemos, no existe aún en la literatura algoritmos exactos para la resolución del Problema de Coloreo Acotado.

Claramente, un modelo de Programación Lineal Entera para el Problema de Coloreo Acotado puede obtenerse al reemplazar las restricciones de equidad de la formulación ECF, por las siguientes:

$$\sum_{v \in V} x_{vj} \leq \sum_{k=j}^{n} h\left(w_k - w_{k+1}\right), \qquad \forall\, 1 \leq j \leq n$$

y muchas de las desigualdades válidas encontradas en el Capítulo 5 de esta tesis que involucran la cota superior del cardinal de las clases de colores en los k-eqcols, pasan a convertirse en desigualdades válidas para este nuevo problema, reemplazando esta cota superior por h. Heurísticas, rutinas de enumeración y otros componentes de ECOPT pueden también ser adaptados sin gran esfuerzo.

Una generalización del Problema de Coloreo Acotado es el *Problema de Coloreo Capacitado*. En este caso, dado un grafo G, un entero k y una lista de enteros llamados *capacidades*, $\alpha_1, \ldots, \alpha_k$, se debe decidir si G admite un k-coloreo de manera que el tamaño de la clase de color j esté acotado superiormente por el valor de α_j, para todo j. Es claro que también en este caso el estudio realizado para el PCEG puede ser adaptado para este problema lo cual provee otra línea de continuidad en los temas abordados en esta tesis.

Es nuestro deseo haber contribuido a despertar el interés para trabajar en estas áreas y estimular la discusión de las líneas de investigación presentes y futuras.

Capítulo 9

Apéndice

En este capítulo se presentan las demostraciones de teoremas presentados en el Capítulo 5, principalmente aquellas que son muy largas para ser expuestas en ese capítulo.

Demostración del Lema 5.1.4. Sean (x^1,w^1), $(x^2,w^2)_{j,j'}$, $(x^3,w^3)_j$, (x^4,w^4), $(x^5,w^5)_j$ y $(x^6,w^6)_k$ los vectores binarios asociados a los coloreos c^1, $c^2_{j,j'}$, c^3_j, c^4, c^5_j y c^6_k definidos en 5.1.3 respectivamente. Consideremos la combinación afín con ciertos coeficientes α por cada coloreo de $PROC(c^1)$:

$$\alpha^1(x^1,w^1)+\sum_{j=1}^{n-1}\sum_{\substack{j'=1\\ j'\neq j}}^{n-1}\alpha^2_{j,j'}(x^2,w^2)_{j,j'}+\sum_{j=1}^{n-1}\alpha^3_j(x^3,w^3)_j+$$
$$\alpha^4(x^4,w^4)+\sum_{j=1}^{n-2}\alpha^5_j(x^5,w^5)_j+\sum_{\substack{k=\chi_{eq}\\ k\notin\mathscr{S}}}^{n-2}\alpha^6_k(x^6,w^6)_k=0 \quad (9.1)$$

y

$$\alpha^1+\sum_{j=1}^{n-1}\sum_{\substack{j'=1\\ j'\neq j}}^{n-1}\alpha^2_{j,j'}+\sum_{j=1}^{n-1}\alpha^3_j+\alpha^4+\sum_{j=1}^{n-2}\alpha^5_j+\sum_{\substack{k=\chi_{eq}\\ k\notin\mathscr{S}}}^{n-2}\alpha^6_k=0.$$

Para verificar la independencia afín de los coloreos hay que demostrar que todos los coeficientes α deben ser nulos. A esto lo demostramos sistemáticamente aprovechando que, una vez que un coeficiente se fija en cero, no es necesario tenerlo en cuenta en los pasos siguientes. También vamos a considerar, sin pérdida de generalidad, que el vértice que se pinta con el color i en el coloreo c^1 es justamente el vértice i, i.e. $c^1(i)=i$ para todo $1\leq i\leq n$.

En primer lugar, observemos que c^1 es el único coloreo tal que $x_{nn}=1$. La combinación (9.1) correspondiente a la componente x_{nn} es entonces $\alpha^1=0$.

Sea $k\in\{\chi_{eq},\ldots,n-2\}\cap\mathscr{S}$ y sea k' el entero más pequeño mayor a k tal que $k'\notin\mathscr{S}$. Las combinaciones (9.1) correspondientes a las componentes de w_k y $w_{k'}$ son

$$\sum_{j=1}^{n-1}\sum_{\substack{j'=1\\ j'\neq j}}^{n-1}\alpha^2_{j,j'}+\sum_{j=1}^{n-1}\alpha^3_j+\alpha^4+\sum_{j=1}^{n-2}\alpha^5_j+\sum_{\substack{t=k\\ t\notin\mathscr{S}}}^{n-2}\alpha^6_t=0$$

y

$$\sum_{j=1}^{n-1}\sum_{\substack{j'=1\\ j'\neq j}}^{n-1}\alpha^2_{j,j'}+\sum_{j=1}^{n-1}\alpha^3_j+\alpha^4+\sum_{j=1}^{n-2}\alpha^5_j+\sum_{\substack{t=k'\\ t\notin\mathscr{S}}}^{n-2}\alpha^6_t=0$$

respectivamente (si $k' = n-1$ entonces la última sumatoria desaparece). De aquí se deduce que $\alpha_k^6 = 0$.

Transcribimos a continuación las combinaciones (9.1) correspondientes a algunas componentes de x asumiendo que $\alpha^1 = \alpha_k^6 = 0$:

1. Componente $x_{j,j'}$ con $j, j' \in \{1,\dots,n-2\}$ y $j \neq j'$: $\alpha_{j,j'}^2 = 0$.
2. Componente $x_{j,n-1}$ con $j \in \{1,\dots,n-2\}$: $\alpha_{n-1,j}^2 + \alpha_j^5 = 0$.
3. Componente $x_{j,n}$ con $j \in \{1,\dots,n-2\}$: $\sum_{\substack{j'=1 \\ j'\neq j}}^{n-2} \alpha_{j,j'}^2 + \alpha_{j,n-1}^2 + \alpha_j^3 = 0$.
4. Componente $x_{n-1,j}$ con $j \in \{1,\dots,n-2\}$: $\alpha_{j,n-1}^2 + \alpha_j^5 = 0$.
5. Componente $x_{n-1,n-1}$: $\sum_{j=1}^{n-2} \sum_{\substack{j'=1 \\ j'\neq j}}^{n-2} \alpha_{j,j'}^2 + \sum_{j=1}^{n-2} \alpha_j^3 + \alpha^4 = 0$.
6. Componente $x_{n,j}$ con $j \in \{1,\dots,n-2\}$: $\alpha_{n-1,j}^2 + \alpha_j^3 + \alpha_j^5 = 0$.
7. Componente $x_{n-1,n}$: $\sum_{j=1}^{n-2} \alpha_{n-1,j}^2 + \alpha_{n-1}^3 = 0$.

De los ítems 6 y 2 se deduce que $\alpha_j^3 = 0$ para $1 \leq j \leq n-2$. Luego, por los ítems 1 y 3 se deduce que $\alpha_{j,n-1}^2 = 0$ para $1 \leq j \leq n-2$. Por lo tanto, por el ítem 4, $\alpha_j^5 = 0$, y por el ítem 2, $\alpha_{n-1,j}^2 = 0$ para $1 \leq j \leq n-2$. Finalmente deducimos del ítem 5 que $\alpha^4 = 0$ y del ítem 7 que $\alpha_{n-1}^3 = 0$. □

El resto de las demostraciones prueban que las desigualdades válidas de una determinada familia definen faceta de $\mathcal{ECP}$ bajo ciertas condiciones y están basadas en la misma técnica, frecuentemente utilizada en la literatura para esta clase de resultados. Esta técnica resulta apropiada en aquellos casos en donde no es trivial reconocer puntos afínmente independientes que caigan en la cara definida por la desigualdad en cuestión. Se basa en el siguiente resultado:

Lema 9.0.1. *[66] Sea $P \subset \mathbb{R}^n$ y $A^= x = b^=$ un sistema minimal de ecuaciones de éste. Sea $\pi x \leq \pi_0$ una desigualdad que define una cara propia F de P. Entonces F es una faceta de P si y sólo si para cualquier $(\lambda, \lambda_0) \in \mathbb{R}^{n+1}$ que satisfaga $\lambda x = \lambda_0 \ \ \forall\, x \in F$ existen $u \in \mathbb{R}^n$ y $\alpha \in \mathbb{R}$ tal que $\alpha > 0$ y $(\lambda, \lambda_0) = (\alpha\pi + uA^=, \alpha\pi_0 + ub^=)$.*

Sea $\pi^X x + \pi^W w \leq \pi_0$ una desigualdad que define una cara propia F' de $\mathcal{ECP}$. Para probar que F' es una faceta de $\mathcal{ECP}$ asumimos la existencia de una cara $F = \{(x,w) \in \mathcal{ECP} \ : \ \lambda^X x + \lambda^W w = \lambda_0\}$ tal que $F' \subset F$ y mostramos que $\lambda^X x + \lambda^W w = \lambda_0$ puede ser obtenida como combinación lineal de $\pi^X x + \pi^W w = \pi_0$ y el sistema minimal de ecuaciones de $\mathcal{ECP}$ dado en el Teorema 5.1.5. Esta última condición es equivalente a probar que (λ^X, λ^W) verifica un cierto sistema de ecuaciones de $dim(\mathcal{ECP}) - 1$ igualdades. La validez de cada una de esas igualdades es derivada de $\lambda^X x^1 + \lambda^W w^1 = \lambda_0 = \lambda^X x^2 + \lambda^W w^2$ aplicada sobre ciertos pares de coloreos equitativos $(x^1, w^1), (x^2, w^2)$ que caen en F'.

A continuación detallamos cómo construir la combinación lineal del sistema minimal y la desigualdad estudiada. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_j los asociados a (5.2) para $j \in \{1,\dots,\chi_{eq}\}$, γ_j los asociados a (5.3) para $j \in \mathscr{S}$, δ el asociado a (5.4) y finalmente π el asociado a $\pi^X x + \pi^W w \leq \pi_0$. Entonces:

- $\lambda_{vj}^X = \alpha_v + \pi\pi_{vj}^X, \quad \forall\, v \in V,\ 1 \leq j \leq n-1.$
- $\lambda_{vn}^X = \alpha_v + \pi\pi_{vn}^X + \delta, \quad \forall\, v \in V.$
- $\lambda_j^W = \beta_j + \pi\pi_j^W, \quad \forall\, 1 \leq j \leq \chi_{eq}.$

- $\lambda_j^W = \pi\pi_j^W, \quad \forall\, \chi_{eq}+1 \le j \le n-1 : j \notin \mathscr{S} \wedge j-1 \notin \mathscr{S}.$
- $\lambda_j^W = \pi\pi_j^W + \gamma_j, \quad \forall\, \chi_{eq}+1 \le j \le n-1 : j \in \mathscr{S} \wedge j-1 \notin \mathscr{S}.$
- $\lambda_j^W = \pi\pi_j^W - \gamma_{j-1}, \quad \forall\, \chi_{eq}+1 \le j \le n-1 : j \notin \mathscr{S} \wedge j-1 \in \mathscr{S}.$
- $\lambda_j^W = \pi\pi_j^W + \gamma_j - \gamma_{j-1}, \quad \forall\, \chi_{eq}+1 \le j \le n-1 : j \in \mathscr{S} \wedge j-1 \in \mathscr{S}.$
- $\lambda_n^W = \pi\pi_n^W - \delta.$
- $\lambda_0 = \sum_{v\in V} \alpha_v + \sum_{j=1}^{\chi_{eq}} \beta_j + \pi_0.$

A veces resulta difícil lidiar con las ecuaciones donde ocurren los coeficientes γ_j. En aquellos casos, imponemos que G sea un grafo monótono así dichas ecuaciones desaparecen de la combinación lineal.

Demostración del Teorema 5.3.3. Dado que Q es maximal, existen vértices no adyacentes $u \in Q$ y $u' \in V\backslash Q$. Estos vértices distinguidos tomarán partido a lo largo de la prueba.

La desigualdad (Q, j)-clique puede ser escrita de la siguiente forma:

$$\sum_{v\in Q} x_{vj} + \sum_{k=1}^{n-1} \pi_k^W w_k \le 0$$

donde $\pi_k^W = 0$ para todo $1 \le k \le n-1$ tal que $k \ne j$, y $\pi_j^W = -1$.
Sea F' la cara definida por esta desigualdad y $F = \{(x,w) \in \mathcal{ECP} : \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad clique. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_k los asociados a (5.2) para $k \in \{1,\dots,\chi_{eq}\}$, γ_k los asociados a (5.3) para $k \in \mathscr{S}$, δ el asociado a (5.4) y π el asociado a la desigualdad clique. De esta combinación lineal se pueden despejar los coeficientes de la siguiente manera: $\delta = -\lambda_n^W$, $\alpha_v = \lambda_{vn}^X + \lambda_n^W$ para todo $v \in V$, $\pi = \lambda_{uj}^X - \lambda_{un}^X - \lambda_n^W$, $\beta_k = \lambda_k^W - \pi_k^W(\lambda_{uj}^X - \lambda_{un}^X - \lambda_n^W)$ para todo $1 \le k \le \chi_{eq}$ y $\gamma_k = \sum_{t=\theta(k)+1}^{k}(\lambda_t^W - \pi_t^W(\lambda_{uj}^X - \lambda_{un}^X - \lambda_n^W))$ para todo $k \in \mathscr{S}$, donde $\theta(k) = \max\{i \in \mathbb{Z} : i \le k,\ i \notin \mathscr{S}\}$ (i.e. $\theta(k) \notin \mathscr{S}$ pero $\theta(k)+1, \theta(k)+2, \dots, k \in \mathscr{S}$), dando lugar al siguiente sistema de ecuaciones:

(a) $\lambda_{vj}^X + \lambda_{un}^X = \lambda_{vn}^X + \lambda_{uj}^X, \quad \forall\, v \in Q\backslash\{u\}.$

(b) $\lambda_{vj}^X = \lambda_{vn}^X + \lambda_n^W, \quad \forall\, v \in V\backslash Q.$

(c) $\lambda_{vk}^X = \lambda_{vn}^X + \lambda_n^W, \quad \forall\, v \in V,\ k \in \{1,\dots,n-1\}\backslash\{j\}.$

(d) $\sum_{t=\theta(k-1)+1}^{k} \left(\lambda_t^W - \pi_t^W(\lambda_{uj}^X - \lambda_{un}^X - \lambda_n^W)\right) = 0, \quad \forall\, k \in \{\chi_{eq}+1,\dots,n-1\}\backslash\mathscr{S}.$

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior.

(a) Sea c^1 un n-eqcol tal que $c^1(v) = j$, $c^1(u) = n$ y sea $c^2 = swap_{j,n}(c^1)$. Obtenemos $\lambda_{vj}^X + \lambda_{un}^X = \lambda_{vn}^X + \lambda_{uj}^X$.

(b) Dado que Q es maximal, existe un vértice $v' \in Q$ que no es adyacente a v. Sea c^1 un $(n-1)$-eqcol tal que $c^1(v) = c^1(v') = j$ y sea $c^2 = intro(c^1, v)$. Así obtenemos $\lambda_{vj}^X = \lambda_{vn}^X + \lambda_n^W$.

(c) **Caso** $v \in V\backslash Q$. Sea $v' \in Q$ un vértice no adyacente a v, c^1 un $(n-1)$-eqcol tal que $c^1(v) = c^1(v') = k$, $c^1(w) = j$ donde $w \in Q\backslash\{v'\}$, y sea $c^2 = intro(c^1, v)$. Luego, $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n$.
Caso $v = u$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(u) = c^1(u') = k$ y $c^1(w) = j$ donde $w \in Q\backslash\{u\}$, y sea $c^2 = intro(c^1, u)$. Luego, $\lambda^X_{uk} = \lambda^X_{un} + \lambda^W_n$.
Caso $v \in Q\backslash\{u\}$. Sea c^1 un n-eqcol tal que $c^1(v) = k$, $c^1(u) = j$, y sea $c^2 = swap_{j,k}(c^1)$. Así obtenemos $\lambda^X_{vj} + \lambda^X_{uk} = \lambda^X_{vk} + \lambda^X_{uj}$. Aplicando la condición (a) y la condición recientemente probada, i.e. $\lambda^X_{uk} = \lambda^X_{un} + \lambda^W_n$, concluimos que $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n$.

(d) Sean c y c' un k-eqcol y $\theta(k-1)$-eqcol arbitrarios de G respectivamente. Ambos coloreos existen porque $k, \theta(k-1) \notin \mathscr{S}$. Definamos $c^1 = (x^1, w^1)$ como c si $k < j$, y $swap_{c(u),j}(c)$ si $k \geq j$. Análogamente, definamos $c^2 = (x^2, w^2)$ como c' si $\theta(k-1) < j$, y $swap_{c'(u),j}(c')$ si $\theta(k-1) \geq j$. Entonces $\lambda^X x^1 + \sum_{t=\theta(k-1)+1}^{k} \lambda^W_t - \lambda^X x^2 = 0$ pues las variables w_t con $\theta(k-1)+1 \leq t \leq k$ valen 1 en c^1 y 0 en c^2.
Caso $k = j$. Sea C_j la clase de color de j en c^1. De $\lambda^X x^1 + \sum_{t=\theta(j-1)+1}^{j} \lambda^W_t - \lambda^X x^2 = 0$ se desprende que

$$\lambda^X_{uj} + \sum_{v \in C_j\backslash\{u\}} \lambda^X_{vj} + \sum_{v \in V\backslash C_j} \lambda^X_{vc^1(v)} + \sum_{t=\theta(j-1)+1}^{j} \lambda^W_t - \sum_{v \in V} \lambda^X_{vc^2(v)} = 0.$$

Por la condición (b), $\sum_{v \in C_j\backslash\{u\}} \lambda^X_{vj} = \sum_{v \in C_j\backslash\{u\}} \lambda^X_{vn} + (|C_j| - 1)\lambda^W_n$. Por la condición (c), $\sum_{v \in V\backslash C_j} \lambda^X_{vc(v)}$ $= \sum_{v \in V\backslash C_j} \lambda^X_{vn} + (n - |C_j|)\lambda^W_n$ y $\sum_{v \in V} \lambda^X_{vc^2(v)} = \sum_{v \in V} \lambda^X_{vn} + n\lambda^W_n$. Obtenemos

$$\sum_{t=\theta(j-1)+1}^{j} \lambda^W_t + \lambda^X_{uj} - \lambda^X_{un} - \lambda^W_n = 0,$$

y, por lo tanto, la ecuación $\sum_{t=\theta(k-1)+1}^{k} \left(\lambda^W_t - \pi^W_t(\lambda^X_{uj} - \lambda^X_{un} - \lambda^W_n)\right) = 0$ es satisfecha.
Caso $k \neq j$. Se procede de manera similar al caso anterior.

□

Demostración del Teorema 5.3.5. La desigualdad (S, j)-2-rango puede ser escrita de la siguiente forma:

$$\sum_{v \in S} x_{vj} + \sum_{v \in V} x_{vn-1} + \sum_{k=1}^{n-2} \pi^W_k w_k - w_{n-1} + w_n \leq 0$$

donde $\pi^W_k = 0$ para todo $1 \leq k \leq n-2$ tal que $k \neq j$, y $\pi^W_j = -2$. Sea F' la cara definida por esta desigualdad y $F = \{(x, w) \in \mathcal{ECP} : \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad 2-rango. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_k los asociados a (5.2) para $k \in \{1, \ldots, \chi_{eq}\}$, γ_k los asociados a (5.3) para $k \in \mathscr{S}$, δ el asociado a (5.4) y π el asociado a la desigualdad 2-rango. De esta combinación lineal se pueden despejar los coeficientes de la siguiente manera: $\pi = -\lambda^W_{n-1}$, $\beta_k = \lambda^W_k + \pi^W_k \lambda^W_{n-1}$ para todo $1 \leq k \leq \chi_{eq}$, $\delta = -\lambda^W_{n-1} - \lambda^W_n$, $\alpha_v = \lambda^X_{vn} + \lambda^W_{n-1} + \lambda^W_n$ para todo $v \in V$ y $\gamma_k = \sum_{t=\theta(k)+1}^{k} (\lambda^W_t + \pi^W_t \lambda^W_{n-1})$ para todo $k \in \mathscr{S}$, donde $\theta(k) = \max\{i \in \mathbb{Z} : i \leq k,\ i \notin \mathscr{S}\}$ (i.e. $\theta(k) \notin \mathscr{S}$ pero $\theta(k)+1, \theta(k)+2, \ldots, k \in \mathscr{S}$), dando lugar al siguiente sistema de ecuaciones:

(a) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in S.$

(b) $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in V.$

(c) $\lambda^X_{vk} = \lambda^X_{vn-1} + \lambda^W_{n-1}, \quad \forall\, v \in V,\ k \in \{1, \ldots, n-2\}\backslash\{j\}.$

(d) $\lambda^X_{vj} = \lambda^X_{vn-1} + \lambda^W_{n-1}, \quad \forall\, v \in V \backslash S.$

(e) $\sum_{t=\theta(k-1)+1}^{k} \left(\lambda^W_t + \pi^W_t \lambda^W_{n-1}\right) = 0, \quad \forall\, k \in \{\chi_{eq}+1, \ldots, n-2\} \backslash \mathscr{S}.$

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior.

(a) Sean $s, s' \in S$ vértices no adyacentes.
Caso $v = s$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(s) = c^1(s') = j$ y $c^2 = intro(c^1, s)$. Es inmediato que $\lambda^X_{sj} = \lambda^X_{sn} + \lambda^W_n$.
Caso $v \neq s$. Sea c^1 un n-eqcol tal que $c^1(v) = j$, $c^1(s) = n$ y $c^2 = swap_{j,n}(c^1)$. Entonces $\lambda^X_{vj} + \lambda^X_{sn} = \lambda^X_{vn} + \lambda^X_{sj}$ y, dado que $\lambda^X_{sj} = \lambda^X_{sn} + \lambda^W_n$, obtenemos $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$.

(b) **Caso** $v \notin S$. Por la hipótesis (ii), existe un conjunto estable $\{v, s, s'\}$ en G con vértices $s, s' \in S$. Sea entonces c^1 un $(n-1)$-eqcol tal que $c^1(v) = c^1(s) = n-1$, $c^1(s') = j$ y $c^2 = intro(c^1, v)$. Por lo tanto, $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$.
Caso $v \in S$ **y** $|S| = 2$. Entonces $S = \{v, v'\}$ para algún v' y, por la hipótesis (ii), existe un conjunto estable $\{u, v, v'\}$. Sea entonces c^1 un $(n-1)$-eqcol tal que $c^1(u) = c^1(v) = n-1$, $c^1(v') = j$ y $c^2 = intro(c^1, v)$. Por lo tanto, $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$.
Caso $v \in S$ **y** $|S| \geq 3$. Sean $s, s' \in S$ vértices no adyacentes. Consideremos c^1 un $(n-1)$-eqcol tal que $c^1(s) = c^1(s') = n-1$ y otro vértice de S es pintado con el color j, y sea $c^2 = intro(c^1, s)$. Entonces, $\lambda^X_{sn-1} = \lambda^X_{sn} + \lambda^W_n$ probando así la ecuacion para el caso $v = s$. Si en cambio $v \neq s$, consideremos c^1 un n-eqcol tal que $c^1(v) = n-1$, $c^1(s) = n$ y otro vértice de S es pintado con el color j, y sea $c^2 = swap_{n,n-1}(c^1)$. Es inmediato que $\lambda^X_{vn-1} + \lambda^X_{sn} = \lambda^X_{vn} + \lambda^X_{sn-1}$. Pero, dado que $\lambda^X_{sn-1} = \lambda^X_{sn} + \lambda^W_n$, obtenemos $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$.

(c) Sean H y M el conjunto estable y el matching dados por hipótesis (i). Sean $s, s' \in S$ los extremos de alguna arista de M y sean $u, u' \in H$.
Caso $v = u$. Sea c^1 un $(n-2)$-eqcol ta que $c^1(u) = c^1(u') = k$, $c^1(s) = c^1(s') = j$ y $c^2 = intro(c^1, u)$. Concluimos que $\lambda^X_{uk} = \lambda^X_{un-1} + \lambda^W_{n-1}$.
Caso $v \neq u$. Sea c^1 un n-eqcol tal que $c^1(u) = k$, $c^1(v) = n-1$ y un vértices de S es coloreado con j, y sea $c^2 = swap_{k,n-1}(c^1)$. Entonces $\lambda^X_{uk} + \lambda^X_{vn-1} = \lambda^X_{un-1} + \lambda^X_{vk}$. Dado que $\lambda^X_{uk} = \lambda^X_{un-1} + \lambda^W_{n-1}$, tenemos $\lambda^X_{vk} = \lambda^X_{vn-1} + \lambda^W_{n-1}$.

(d) Sean $H_v = \{v, s, s'\}$, H'_v (en caso de que n sea par) y M_v los conjuntos estables y el matching dados por hipótesis (ii), y sean H, H' (en caso de que n sea par) y M los conjuntos estables y el matching dados por hipótesis (i). Sea c^1 un $(\lceil n/2 \rceil - 1)$-eqcol tal que la clase de color j es H_v y las clases de color restantes son H'_v (en caso de que n sea par) y los extremos de las aristas de M_v. Ahora sea $\hat{s}, \hat{s}' \in S$ los extremos de alguna arista de M y c^2 un $(\lceil n/2 \rceil - 1)$-eqcol tal que la clase de color j es $\{\hat{s}, \hat{s}'\}$ y las clases de color restantes son H, H' (en caso de que n sea par) y los extremos de las aristas de M excepto $(\hat{s}, \hat{s}')$. Estos coloreos implican

$$\lambda^X_{vj} + \lambda^X_{sj} + \lambda^X_{s'j} + \sum_{w \in V \backslash \{v,s,s'\}} \lambda^X_{wc^1(w)} = \lambda^X_{\hat{s}j} + \lambda^X_{\hat{s}'j} + \sum_{w \in V \backslash \{\hat{s}, \hat{s}'\}} \lambda^X_{wc^2(w)}.$$

Al aplicar las condiciones (a)-(c), la última ecuación se vuelve

$$\lambda^X_{vj} + \sum_{w \in V \backslash \{v\}} \lambda^X_{wn} + (n-3)\lambda^W_{n-1} + (n-1)\lambda^W_n = \sum_{w \in V} \lambda^X_{wn} + (n-2)\lambda^W_{n-1} + n\lambda^W_n.$$

Simplificando obtenemos $\lambda^X_{vj} = \lambda^X_{vn-1} + \lambda^W_{n-1}$.

(e) Esta condición puede ser verificada conociendo un k-eqcol (x^1, w^1) y un $\theta(k-1)$-eqcol (x^2, w^2) pertenecientes a F': se deben aplicar las condiciones (a)-(d) a la ecuación $\lambda^X x^1 + \sum_{t=\theta(k-1)+1}^{k} \lambda_t^W = \lambda^X x^2$ como se hizo en la condición (d) del Teorema 5.3.3.

Es decir que sólo necesitamos probar que, para cualquier $r \in \{\chi_{eq}, \dots, n-2\} \backslash \mathscr{S}$, existe un r-eqcol c que cae en la cara F'.
Caso $r < j$. Elegimos un r-eqcol arbitrario de G.
Caso $j \leq r \leq \lceil n/2 \rceil - 2$. La hipótesis (iii) garantiza la existencia de un r-eqcol c' donde dos vértices $s, s' \in S$ satisfacen $c'(s) = c'(s')$. Entonces $c = swap_{c'(s),j}(c')$ es un r-eqcol que cae en F'.
Caso $r = \lceil n/2 \rceil - 1$. c puede ser uno de los coloreos dados en la condición (d).
Caso $r = \lceil n/2 \rceil$. Sean H, H' (en caso de que n sea par) y M los conjuntos estables y el matching dado por hipótesis (i). Sean $s, s' \in S$ los extremos de alguna arista de M y $h \in H$, $h' \in H'$ (en caso de que n sea par) vértices no adyacentes.
Si n es impar, las clases de color de c son $\{h\}$, $H \backslash \{h\}$ y los extremos de las aristas de M donde, en particular, $C_j = \{s, s'\}$. Si en cambio n es par, las clases de color de c son $\{h, h'\}$, $H \backslash \{h\}$, $H' \backslash \{h'\}$ y los extremos de las aristas de M donde, en particular, $C_j = \{s, s'\}$.
Case $r \geq \lceil n/2 \rceil + 1$. Consideremos el $\lceil n/2 \rceil$-eqcol producido en el caso anterior y sean v_1, v_2 vértices que comparten un color distinto de j. Para generar un $(\lceil n/2 \rceil + 1)$-eqcol c, introducimos un nuevo color en v_1, i.e. $c = intro(c', v_1)$ donde c' es el $\lceil n/2 \rceil$-eqcol. Repitiendo este procedimiento podemos generar un $(\lceil n/2 \rceil + 2)$-eqcol y así sucesivamente.

□

Demostración del Teorema 5.3.9. Sean q, q' vértices de Q. Estos vértices distinguidos serán usados a lo largo de la prueba.

La desigualdad (S, Q, j)-2-rango puede ser escrita de la siguiente forma:

$$\sum_{v \in S \backslash Q} x_{vj} + 2 \sum_{v \in Q} x_{vj} + \sum_{k=1}^{n-1} \pi_k^W w_k \leq 0$$

donde $\pi_k^W = 0$ para todo $1 \leq k \leq n-1$ tal que $k \neq j$, y $\pi_j^W = -2$. Sea F' la cara definida por esta desigualdad y $F = \{(x, w) \in \mathcal{ECP} : \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad 2-rango. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_k los asociados a (5.2) para $k \in \{1, \dots, \chi_{eq}\}$, γ_k los asociados a (5.3) para $k \in \mathscr{S}$, δ el asociado a (5.4) y π el asociado a la desigualdad 2-rango. De esta combinación lineal se pueden despejar los coeficientes de la siguiente manera: $\pi = \frac{1}{2}(\lambda_{qj}^X - \lambda_{qn}^X - \lambda_n^W)$, $\delta = -\lambda_n^W$, $\beta_k = \lambda_k^W - \frac{1}{2}\pi_k^W(\lambda_{qj}^X - \lambda_{qn}^X - \lambda_n^W)$ para todo $1 \leq k \leq \chi_{eq}$, $\alpha_v = \lambda_{vn}^X + \lambda_n^W$ para todo $v \in V$ y $\gamma_k = \sum_{t=\theta(k)+1}^{k}(\lambda_t^W - \frac{1}{2}\pi_t^W(\lambda_{qj}^X - \lambda_{qn}^X - \lambda_n^W))$ para todo $k \in \mathscr{S}$, donde $\theta(k) = \text{máx}\{i \in \mathbb{Z} : i \leq k,\ i \notin \mathscr{S}\}$ (i.e. $\theta(k) \notin \mathscr{S}$ pero $\theta(k)+1, \theta(k)+2, \dots, k \in \mathscr{S}$), dando lugar al siguiente sistema de ecuaciones:

(a) $\lambda_{vj}^X = \lambda_{vn}^X + \lambda_n^W, \quad \forall\, v \in V \backslash S$ tal que $Q \backslash N(v) \neq \varnothing$.

(b) $\lambda_{vk}^X = \lambda_{vn}^X + \lambda_n^W, \quad \forall\, v \in V,\ k \in \{1, \dots, n-1\} \backslash \{j\}$.

(c) $\lambda_{qn}^X + \lambda_{vj}^X = \lambda_{qj}^X + \lambda_{vn}^X, \quad \forall\, v \in Q \backslash \{q\}$.

(d) $\lambda_{qn}^X + 2\lambda_{vj}^X = 2\lambda_{vn}^X + \lambda_{qj}^X + \lambda_n^W, \quad \forall\, v \in S \backslash Q$.

(e) $\lambda_{vj}^X = \lambda_{vn}^X + \lambda_n^W, \quad \forall\, v \in V \backslash S$ tal que $Q \subset N(v)$.

(f) $\sum_{t=\theta(k-1)+1}^{k}\left(\lambda_t^W - \frac{1}{2}\pi_t^W(\lambda_{qj}^X - \lambda_{qn}^X - \lambda_n^W)\right) = 0, \quad \forall\, k \in \{\chi_{eq}+1, \dots, n-1\} \backslash \mathscr{S}$.

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior.

(a) Sea $\hat{q} \in Q \backslash N(v)$ y sea c^1 un $(n-1)$-eqcol tal que $c^1(\hat{q}) = c^1(v) = j$, y $c^2 = intro(c^1, v)$. Concluimos que $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$.

(b) Sean $s, s' \in S \backslash Q$ vértices no adyacentes entre sí.
Caso $v = s$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(s) = c^1(s') = k$, $c^1(q) = j$ y sea $c^2 = intro(c^1, s)$. Entonces $\lambda^X_{sk} = \lambda^X_{sn} + \lambda^W_n$.
Caso $v \neq s$. Sea c^1 un n-eqcol tal que $c^1(v) = k$ y $c^1(s) = n$. En caso de que $v = q$ hacemos $c^1(q') = j$ y, en caso contrario, hacemos $c^1(q) = j$. Sea además $c^2 = swap_{k,n}(c^1)$. Entonces $\lambda^X_{vk} + \lambda^X_{sn} = \lambda^X_{vn} + \lambda^X_{sk}$. Pero, en virtud de $\lambda^X_{sk} = \lambda^X_{sn} + \lambda^W_n$, concluimos que $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n$.

(c) Sea c^1 un n-eqcol tal que $c^1(q) = n$ y $c^1(v) = j$, y sea $c^2 = swap_{j,n}(c^1)$. Por lo tanto, $\lambda^X_{qn} + \lambda^X_{vj} = \lambda^X_{qj} + \lambda^X_{vn}$.

(d) Sea J la componente conexa en el complemento de $G[S \backslash Q]$ que contiene el vértice v. Dado que $\alpha(S) = 2$, J no contiene triángulos. Por hipótesis (i), J no es bipartita y, por lo tanto, debe existir al menos un agujero de longitud p impar (en adelante, *agujero impar*) en J con $p \geq 5$.
Ahora, dado un vértice w y un agujero impar $C \subset V$ de J, definimos la *distancia de w a C* como la mínima distancia entre w y los vértices de C. Sea entonces $d(v)$ la mínima distancia entre v y todos los agujeros impares en J. Probaremos la condición (d) por inducción sobre $d(v)$.
Caso $d(v) = 0$. Aquí estamos en el caso en que v pertenece a un agujero impar de tamaño $p \geq 5$ en J. Sean $v_1 = v, v_2, \ldots, v_p \in S \backslash Q$ los vértices de aquel agujero impar, y sean $k_1, k_2, \ldots, k_{p+1}$ colores distintos entre sí y distintos de j.
Vamos a denotar con $\oplus$ a la suma de dos enteros módulo p. Sea entonces c^1, c^2, …, c^p $(n-1)$-eqcols tales que, para cada $1 \leq i \leq p$, tenemos que $c^i(v_i) = j$, $c^i(v_{i\oplus 1}) = j$, $c^i(v_r) = k_r \ \forall\, r \in \{1, \ldots, p\} \backslash \{i, i \oplus 1\}$, $c^i(q) = k_{p+1}$, y sea c^{p+1} un n-eqcol tal que $c^{p+1}(v_1) = n$, $c^{p+1}(v_r) = k_r \ \forall\, r \in \{2, \ldots, p\}$, $c^{p+1}(q) = j$. Por ejemplo, si $p = 5$, los colores de v_1, …, v_5 y q serían:

c^i con i impar		c^i con i par	
tamaño	$v_1\ v_2\ v_3\ v_4\ v_5\ \ q$	tamaño	$v_1\ v_2\ v_3\ v_4\ v_5\ \ q$
$n-1$	$j\ \ j\ k_3\ k_4\ k_5\ k_6$	$n-1$	$k_1\ \ j\ \ j\ k_4\ k_5\ k_6$
$n-1$	$k_1\ k_2\ \ j\ \ j\ k_5\ k_6$	$n-1$	$k_1\ k_2\ k_3\ \ j\ \ j\ k_6$
$n-1$	$j\ \ k_2\ k_3\ k_4\ \ j\ \ k_6$	n	$n\ \ k_2\ k_3\ k_4\ k_5\ \ j$

Asumimos que los vértices restantes tienen el mismo color en todos los coloreos propuestos. Luego,

$$\sum_{\substack{i=1 \\ i \text{ impar}}}^{p+1} \sum_{v \in V} \lambda^X_{vc^i(v)} = \sum_{\substack{i=1 \\ i \text{ par}}}^{p+1} \sum_{v \in V} \lambda^X_{vc^i(v)} + \lambda^W_n.$$

Pero, según la condición (b), obtenemos $\lambda^X_{qn} + 2\lambda^X_{vj} = 2\lambda^X_{vn} + \lambda^X_{qj} + \lambda^W_n$.
Caso $d(v) \geq 1$. Sea $v' \in J$ un vértice adyacente a v en J para el cual $d(v') = d(v) - 1$. Por hipótesis inductiva, $\lambda^X_{qn} + 2\lambda^X_{v'j} = 2\lambda^X_{v'n} + \lambda^X_{qj} + \lambda^W_n$.
Entonces, sea c^1 un $(n-1)$-eqcol tal que $c^1(v) = c^1(v') = j$ y $c^1(q) = k$, donde $k \neq j$. Sea c^2 un n-eqcol tal que $c^2(v) = k$, $c^2(v') = n$, $c^2(q) = j$ y $c^2(i) = c^1(i) \ \ \forall\, i \in V \backslash \{v, v', q\}$. Entonces, $\lambda^X_{vj} + \lambda^X_{v'j} + \lambda^X_{qk} = \lambda^X_{vk} + \lambda^X_{v'n} + \lambda^X_{qj} + \lambda^W_n$. Multiplicando esta igualdad por 2, luego sustrayendo $\lambda^X_{qn} + 2\lambda^X_{v'j} = 2\lambda^X_{v'n} + \lambda^X_{qj} + \lambda^W_n$ y finalmente aplicando la condición (b) llegamos a $\lambda^X_{qn} + 2\lambda^X_{vj} = 2\lambda^X_{vn} + \lambda^X_{qj} + \lambda^W_n$.

(e) Por la hipótesis (ii), podemos establecer un $(\lceil n/2\rceil-1)$-eqcol c^1 tal que la clase de color j es $\{v,s,s'\}$ donde $s,s'\in S$ (como lo hicimos en la condición (d) del Teorema 5.3.5). Sea $c^2=swap_{j,c^1(q)}(c^1)$. Mediante la aplicación de las condiciones probadas anteriormente obtenemos $\lambda^X_{vj}=\lambda^X_{vn}+\lambda^W_n$.

(f) Esta condición puede ser verificada conociendo un k-eqcol (x^1,w^1) y un $\theta(k-1)$-eqcol (x^2,w^2) pertenecientes a F': se deben aplicar las condiciones (a)-(e) a la ecuación $\lambda^X x^1+\sum_{t=\theta(k-1)+1}^{k}\lambda^W_t=\lambda^X x^2$ como se hizo en la condición (d) del Teorema 5.3.3.
Vale aclarar que cualquier r-eqcol c con $r<j$ pertenece a F'. Si $r\geq j$, $swap_{c(q),j}(c)$ pertenece a F'.

□

Demostración del Corolario 5.3.10. El Teorema 5.3.9 manifiesta, entre otras cosas, que $j\leq\lceil n/2\rceil-1$ para que la desigualdad (S,Q,j)-2-rango defina faceta. En realidad esta condición es utilizada por el teorema únicamente al probar las ecuaciones dadas en el ítem (e), i.e. $\lambda^X_{vj}=\lambda^X_{vn}+\lambda^W_n$, para todo $v\in V\backslash S$ tal que $Q\subset N(v)$. Entonces, si todo vértice $v\in V\backslash S$ verifica $Q\backslash N(v)\neq\varnothing$, aquellas ecuaciones desaparecen del sistema de ecuaciones en (λ^X,λ^W), dando lugar a que la desigualdad (S,Q,j)-2-rango defina faceta aún cuando $j>\lceil n/2\rceil-1$. □

Demostración del Teorema 5.4.6. Sea F' la cara definida por (5.8) y $F=\{(x,w)\in\mathcal{ECP}:\lambda^X x+\lambda^W w=\lambda_0\}$ una cara tal que $F'\subset F$. Consideremos la combinación lineal del sistema minimal y (5.8). Sean α_v los coeficientes asociados a las igualdades (5.1) para $v\in V$, β_k los asociados a (5.2) para $k\in\{1,\dots,\chi_{eq}\}$, δ el asociado a (5.4) y π el asociado a la desigualdad subvecindad. De esta combinación lineal se pueden despejar primero los coeficientes β_k para todo $1\leq k\leq\chi_{eq}$, y luego el resto de los coeficientes de la siguiente manera: $\delta=-\lambda^W_n$, $\pi=\lambda^X_{uj}-\lambda^X_{un}-\lambda^W_n$, $\alpha_u=\gamma_{jS}\lambda^X_{un}+\gamma_{jS}\lambda^W_n-(\gamma_{jS}-1)\lambda^X_{uj}$ y $\alpha_v=\lambda^X_{vn}+\lambda^W_n$ para todo $v\in V\backslash\{u\}$. Así obtenemos el siguiente sistema de ecuaciones que debe ser verificado:

(a) $\lambda^X_{vj}=\lambda^X_{vn}+\lambda^W_n,\quad\forall\, v\in V\backslash N[u]$.

(b) $\lambda^X_{vk}=\lambda^X_{vn}+\lambda^W_n,\quad\forall\, v\in V\backslash\{u\},\ k\in\{1,\dots,n-1\}\backslash\{j\}$.

(c) $\lambda^X_{vj}+\lambda^X_{un}=\lambda^X_{vn}+\lambda^X_{uj},\quad\forall\, v\in S$.

(d) $\lambda^X_{uk}+(\gamma_{jS}-1)\lambda^X_{uj}=\gamma_{jS}\lambda^X_{un}+\gamma_{jS}\lambda^W_n,\quad\forall\, k\in\{1,\dots,j-1\}$.

(e) $\lambda^X_{uk}+(\gamma_{kS}-1)\lambda^X_{uj}=\gamma_{kS}\lambda^X_{un}+\gamma_{kS}\lambda^W_n,\quad\forall\, k\in\{j+1,\dots,n-1\}$.

(f) $\lambda^X_{vj}=\lambda^X_{vn}+\lambda^W_n,\quad\forall\, v\in N(u)\backslash S$.

(g) $\lambda^W_k=0,\quad\forall\, k\in\{\chi_{eq}+1,\dots,n-1\}\backslash\{j\}$.

(h) Si $j\geq\chi_{eq}+1$ entonces $\gamma_{jS}\lambda^X_{un}+\gamma_{jS}\lambda^W_n=\gamma_{jS}\lambda^X_{uj}+\lambda^W_j$.

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior:

(a) Sea c^1 un $(n-1)$-eqcol tal que $c^1(u)=c^1(v)=j$ y sea $c^2=intro(c^1,v)$. Concluimos que $\lambda^X_{vj}=\lambda^X_{vn}+\lambda^W_n$.

(b) Sean $s,s'\in S$ no adyacentes.
Caso $v=s$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(s)=c^1(s')=k$, $c^1(u)=j$ y sea $c^2=intro(c^1,s)$. Luego, $\lambda^X_{sk}=\lambda^X_{sn}+\lambda^W_n$.
Caso $v\neq s$. Sea c^1 un n-eqcol tal que $c^1(v)=k$, $c^1(s)=n$, $c^1(u)=j$ y sea $c^2=swap_{k,n}(c^1)$. Entonces $\lambda^X_{vk}+\lambda^X_{sn}=\lambda^X_{vn}+\lambda^X_{sk}$ y, debido a que $\lambda^X_{sk}=\lambda^X_{sn}+\lambda^W_n$, concluimos que $\lambda^X_{vk}=\lambda^X_{vn}+\lambda^W_n$.

(c) Sea c^1 un n-eqcol tal que $c^1(v) = j$, $c^1(u) = n$ y sea $c^2 = swap_{j,n}(c^1)$. Por lo tanto, $\lambda^X_{vj} + \lambda^X_{un} = \lambda^X_{vn} + \lambda^X_{uj}$.

(d) **Caso** $\gamma_{jS} = 2$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(s) = c^1(s') = j$ donde s y s' son vértices no adyacentes de S.
Caso $\gamma_{jS} \geq 3$. Sea c el $(\lceil \frac{n}{\gamma_{jS}-1} \rceil - 1)$-eqcol dado en la hipótesis (i) y $c^1 = swap_{c(u),k}(c)$.
En ambos casos, $c^1(u) = k$. Ahora, sean C_j y C_k las clases de color j y k de c^1 respectivamente. Considerando $c^2 = swap_{j,k}(c^1)$ da lugar a la ecuación

$$\lambda^X_{uk} + \sum_{v \in C_j} \lambda^X_{vj} + \sum_{v \in C_k \setminus \{u\}} \lambda^X_{vk} = \lambda^X_{uj} + \sum_{v \in C_j} \lambda^X_{vk} + \sum_{v \in C_k \setminus \{u\}} \lambda^X_{vj}.$$

Pero, dado que $|C_j \cap S| = \gamma_{jS}$, tenemos $C_j \subset S$ y entonces podemos aplicar las condiciones (a)-(c) para obtener $\lambda^X_{uk} + (\gamma_{jS} - 1)\lambda^X_{uj} = \gamma_{jS}\lambda^X_{un} + \gamma_{jS}\lambda^W_n$.

(e) Procedemos de la misma forma que en (d) con la excepción de que, para el caso $\gamma_{jS} \geq 3$, utilizamos el $(\lceil \frac{n}{\gamma_{kS}-1} \rceil - 1)$-eqcol que provee la hipótesis (i) en vez del $(\lceil \frac{n}{\gamma_{jS}-1} \rceil - 1)$-eqcol.

(f) En primer lugar, como $v \in N(u) \setminus S$ entonces $S \subsetneq N(u)$ y, por hipótesis, $\alpha(S) \leq \lceil n/j \rceil - 1$. Luego, por hipótesis (ii), existe un coloreo c^1 que colorea a v y $\alpha(S)$ vértices de S con j pero los vértices restantes de $N(u)$ no usan j. Sea $k = c^1(u)$ y sean C_j, C_k las clases de color j y k en c^1 respectivamente. Considerando $c^2 = swap_{j,k}(c^1)$ tenemos:

$$\lambda^X_{uk} + \lambda^X_{vj} + \sum_{w \in C_j \setminus \{v\}} \lambda^X_{wj} + \sum_{w \in C_k \setminus \{u\}} \lambda^X_{wk} = \lambda^X_{uj} + \lambda^X_{vk} + \sum_{w \in C_j \setminus \{v\}} \lambda^X_{wk} + \sum_{w \in C_k \setminus \{u\}} \lambda^X_{wj}.$$

En virtud de las condiciones (a)-(e) ya probadas, obtenemos $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$.

(g) Dado que G es monótono, existen un k-eqcol c y un $(k-1)$-eqcol c' en G. Definamos $c^1 = (x^1, w^1)$ como c si $k < j$, y $swap_{c(u),j}(c)$ si $k \geq j$. Análogamente, definamos $c^2 = (x^2, w^2)$ como c' si $k - 1 < j$, y $swap_{c'(u),j}(c')$ si $k - 1 \geq j$. Luego se deben aplicar las condiciones (a)-(f) a la ecuación $\lambda^X x^1 + \lambda^W_k = \lambda^X x^2$ como se hizo en la condición (d) del Teorema 5.3.3.

(h) Sea c un j-eqcol y c' un $(j-1)$-eqcol. Luego procedemos como en (g).

□

Demostración del Teorema 5.4.16. La desigualdad (5.9) se puede reescribir así:

$$\left(\left\lfloor \frac{n}{j} \right\rfloor - 1\right)x_{uj} - \sum_{v \in V \setminus N[u]} x_{vj} + \sum_{k=j+1}^{n} \left(\left\lfloor \frac{n}{j} \right\rfloor - \left\lfloor \frac{n}{k} \right\rfloor\right)x_{uk} + \sum_{k=j+1}^{n} \left(\left\lfloor \frac{n}{k} \right\rfloor - \left\lfloor \frac{n}{k-1} \right\rfloor\right)w_k \leq 0.$$

Sea F' la cara definida por esta desigualdad y $F = \{(x,w) \in \mathcal{ECP} : \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad fuera-vecindad. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_k los asociados a (5.2) para $k \in \{1, \ldots, \chi_{eq}\}$, δ el asociado a (5.4) y π el asociado a la desigualdad fuera-vecindad. De esta combinación lineal se pueden despejar los coeficientes de la siguiente manera: $\beta_k = \lambda^W_k$ para todo $1 \leq k \leq \chi_{eq}$, $\delta = -\lambda^W_n$, $\pi = -\lambda^W_{\lfloor n/2 \rfloor + 1}$ y $\alpha_v = \lambda^X_{vn} + \lambda^W_n$ para todo $v \in V$. Así obtenemos el siguiente sistema de ecuaciones que debe ser verificado:

(a) $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in V \setminus N[u],\ k \in \{1, \ldots, n-1\} \setminus \{j\}$

(b) $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in N(u),\ k \in \{1, \ldots, n-1\}$

(c) $\lambda^X_{uj} = \lambda^X_{un} + \lambda^W_n$,

(d) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{\lfloor n/2\rfloor+1}, \quad \forall\, v \in V\backslash N[u]$

(e) $\lambda^X_{uk} = \lambda^X_{un} + \lambda^W_n + (\lfloor n/j\rfloor - 1)\lambda^W_{\lfloor n/2\rfloor+1}, \quad \forall\, k \in \{1,\ldots,j-1\}$

(f) $\lambda^X_{uk} = \lambda^X_{un} + \lambda^W_n + (\lfloor n/k\rfloor - 1)\lambda^W_{\lfloor n/2\rfloor+1}, \quad \forall\, k \in \{j+1,\ldots,n-1\}$

(g) Si $j \neq \chi_{eq}$ entonces $\lambda^W_k = 0, \quad \forall\, k \in \{\chi_{eq}+1,\ldots,j\}$

(h) $\lambda^W_k = \left(\left\lfloor \frac{n}{k-1}\right\rfloor - \left\lfloor \frac{n}{k}\right\rfloor\right)\lambda^W_{\lfloor n/2\rfloor+1}, \quad \forall\, k \in \{j+1,\ldots,n-1\}\backslash\{\lfloor n/2\rfloor+1$

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior:

(a) Por la hipótesis (i) existen vértices $\hat{v} \in V\backslash N[u]$ y $\hat{v}' \in V\backslash\{u,\hat{v}\}$ no adyacentes entre sí.
Caso $v = \hat{v}$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(u) = j$, $c^1(\hat{v}) = c^1(\hat{v}') = k$ y sea $c^2 = intro(c^1, \hat{v})$. Luego, $\lambda^X_{\hat{v}k} = \lambda^X_{\hat{v}n} + \lambda^W_n$.
Caso $v \neq \hat{v}$. Sea c^1 un n-eqcol tal que $c^1(u) = j$, $c^1(v) = k$, $c^1(\hat{v}) = n$ y sea $c^2 = swap_{k,n}(c^1)$. Tenemos que $\lambda^X_{vk} + \lambda^X_{\hat{v}n} = \lambda^X_{vn} + \lambda^X_{\hat{v}k}$ y, por lo tanto, $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n$.

(b) Sean $u_1, u_2 \in N(u)$ vértices no adyacentes entre sí.
Caso $v = u_1$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(u_1) = c^1(u_2) = k$. En caso de que $k = j$ pedimos también que $c^1(u) = n-1$, si no pedimos que $c^1(u) = j$. Sea también $c^2 = intro(c^1, u_1)$. Entonces, $\lambda^X_{u_1k} = \lambda^X_{u_1n} + \lambda^W_n$.
Caso $v \neq u_1$. Sea c^1 un n-eqcol tal que $c^1(v) = k$ y $c^1(u_1) = n$. En caso de que $k = j$ pedimos que $c^1(u) = n-1$, si no pedimos que $c^1(u) = j$. Sea también $c^2 = swap_{k,n}(c^1)$. Tenemos que $\lambda^X_{vk} + \lambda^X_{u_1n} = \lambda^X_{vn} + \lambda^X_{u_1k}$ y, por lo tanto, $\lambda^X_{vk} = \lambda^X_{vn} + \lambda^W_n$.

(c) Sea $v \in N(u)$, c^1 un n-eqcol tal que $c^1(u) = j$, $c^1(v) = n$ y $c^2 = swap_{j,n}(c^1)$. En virtud de la condición (b), i.e. $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$, obtenemos $\lambda^X_{uj} = \lambda^X_{un} + \lambda^W_n$.

(d) **Caso** n **par**. Sea M_v el matching dado por hipótesis (iii). Sea c^1 el $\lfloor n/2\rfloor$-eqcol cuyas clases de color son los extremos de M_v y $C_j = \{u,v\}$. De considerar $c^2 = intro(c^1, v)$, deducimos que $\lambda^X_{vj} = \lambda^X_{v\lfloor n/2\rfloor+1} + \lambda^W_{\lfloor n/2\rfloor+1} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{\lfloor n/2\rfloor+1}$.
Caso n **impar**. Sea M_v y H_v el matching y el conjunto estable dados por hipótesis (iii). Sea c^1 el $\lfloor n/2\rfloor$-eqcol cuyas clases de color son H_v, los extremos de M_v y $C_j = \{u,v\}$. Ahora, sea M el matching dado por hipótesis (ii) y v' aquel vértice tal que (v,v') pertenece a M. Consideremos c^2 como el $(\lfloor n/2\rfloor+1)$-eqcol cuyas clases de color son los extremos de $M\backslash(v,v')$, $C_j = \{u\}$ y $C_{\lfloor n/2\rfloor+1} = \{v,v'\}$. Así obtenemos

$$\lambda^X_{vj} + \sum_{i\in V\backslash\{u,v\}} \lambda^X_{ic^1(i)} = \lambda^X_{v\lfloor n/2\rfloor+1} + \sum_{i\in V\backslash\{u,v\}} \lambda^X_{ic^2(i)} + \lambda^W_{\lfloor n/2\rfloor+1}.$$

Luego, aplicando las condiciones (a) y (b) llegamos a $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{\lfloor n/2\rfloor+1}$.

(e) Observemos que si $r = \lfloor n/\lfloor n/j\rfloor\rfloor$ entonces $\lfloor n/j\rfloor = \lfloor n/r\rfloor$, $j \leq r \leq \lfloor n/2\rfloor$ y $\lfloor \frac{n}{r}\rfloor > \lfloor \frac{n}{r+1}\rfloor$. Entonces por hipótesis (iv), existe un r-eqcol c tal que $N(u)$ contiene todos los vértices con color j. Sea $c^1 = swap_{c(u),k}(c)$ y c^2 el r-eqcol que pinta al vértice u y $\lfloor n/j\rfloor - 1$ vértices de $V\backslash N[u]$ con color j, también proporcionada por la hipótesis (iv). Aplicando la condición (c) sobre $\lambda^X x^1 = \lambda^X x^2$, llegamos a $\lambda^X_{uk} + \sum_{v\in V\backslash\{u\}} \lambda^X_{vc^1(v)} = \lambda^X_{un} + \lambda^W_n + \sum_{v\in V\backslash\{u\}} \lambda^X_{vc^2(v)}$. Luego, aplicando las condiciones (a), (b) y (d), obtenemos $\lambda^X_{uk} = \lambda^X_{un} + \lambda^W_n + (\lfloor n/j\rfloor - 1)\lambda^W_{\lfloor n/2\rfloor+1}$.

(f) **Caso** $k \leq \lfloor n/2 \rfloor$. Procedemos de la misma manera que en (e), salvo que esta vez usamos $r = \lfloor n/\lfloor n/k \rfloor \rfloor$ en vez de $\lfloor n/\lfloor n/j \rfloor \rfloor$.
Caso $k \geq \lfloor n/2 \rfloor + 1$. Entonces, $\lfloor n/k \rfloor = 1$. Sean $v \in N(u)$, c^1 un n-eqcol tal que $c^1(u) = k$, $c^1(v) = j$ y sea $c^2 = swap_{k,j}(c^1)$. Las condiciones (b) y (c) aplicadas a $\lambda^X x^1 = \lambda^X x^2$ nos permiten obtener $\lambda^X_{uk} = \lambda^X_{un} + \lambda^W_n$.

(g)-(h) Esta condición puede ser verificada conociendo un k-eqcol (x^1, w^1) y un $(k-1)$-eqcol (x^2, w^2) que caigan en F': se deben aplicar las condiciones (a)-(f) a la ecuación $\lambda^X x^1 + \lambda^W_k = \lambda^X x^2$ como se hizo en la condición (d) del Teorema 5.3.3.

Es decir que sólo necesitamos probar que, para cualquier $r \in \{\chi_{eq}, \ldots, n-1\}$, existe un r-eqcol c que cae en la cara F'.
Caso $r < j$. La existencia de c es garantizada por la monotonicidad de G.
Caso $j \leq r \leq \lfloor n/2 \rfloor$. La existencia de c es garantizada por la hipótesis (iv).
Caso $r = \lfloor n/2 \rfloor + 1$. c puede ser el $(\lfloor n/2 \rfloor + 1)$-eqcol producido en el ítem (d).
Caso $\lfloor n/2 \rfloor + 2 \leq r \leq n-1$. Consideremos el $(\lfloor n/2 \rfloor + 1)$-eqcol del caso anterior y sean v_1, v_2 vértices que comparten un color distinto de j. Para generar un $(\lfloor n/2 \rfloor + 2)$-eqcol c, introducimos un nuevo color en v_1, i.e. $c = intro(c', v_1)$ donde c' es el $(\lfloor n/2 \rfloor + 1)$-eqcol. Repitiendo este procedimiento podemos generar un $(\lfloor n/2 \rfloor + 3)$-eqcol y así sucesivamente.

□

Demostración del Teorema 5.4.23. Sea F' la cara de $\mathcal{ECP}$ definida por (5.13) y $F = \{(x,w) \in \mathcal{ECP} : \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad clique-vecindad. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_r los asociados a (5.2) para $r \in \{1, \ldots, \chi_{eq}\}$, δ el asociado a (5.4) y π el asociado a la desigualdad clique-vecindad. De esta combinación lineal se puede despejar $\pi = -\lambda^W_{n-1}$ y luego el resto de los coeficientes α_v, β_r y δ, obteniéndose el siguiente sistema de ecuaciones:

(a) $\lambda^X_{uj} = \lambda^X_{un} + \lambda^W_n$.

(b) $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in V$.

(c) $\lambda^X_{vr} = \lambda^X_{vn-1} + \lambda^W_{n-1}, \quad \forall\, v \in V \backslash \{u\},\ r \in \{1, \ldots, n-2\} \backslash \{j\}$.

(d) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in N(u) \cup Q$.

(e) $\lambda^X_{vj} = \lambda^X_{vn-1} + \lambda^W_{n-1}, \quad \forall\, v \in V \backslash (N[u] \cup Q)$.

(f) $\lambda^X_{ur} = \lambda^X_{un} + (k-1)\lambda^W_{n-1} + \lambda^W_n, \quad \forall\, r \in \{1, \ldots, \lceil \frac{n}{k-1} \rceil - 1\} \backslash \{j\}$.

(g) $\lambda^X_{ur} = \lambda^X_{un} + (\lceil n/r \rceil - 1)\lambda^W_{n-1} + \lambda^W_n, \quad \forall\, r \in \{\lceil \frac{n}{k-1} \rceil, \ldots, n-2\}$.

(h) $\lambda^W_r = (b_{ur} - b_{ur-1})\lambda^W_{n-1}, \quad \forall\, r \in \{\chi_{eq}+1, \ldots, n-2\}$, donde

$$b_{ur} = \begin{cases} 0, & \text{si } r < j \\ \min\{\lceil n/r \rceil, \alpha(N(u)) + 1\}, & \text{si } j \leq r \leq \lceil n/k \rceil - 1 \\ k, & \text{si } \lceil n/k \rceil \leq r \leq n-2 \end{cases}$$

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior:

(a) Sea $q \in Q$, c^1 un $(n-1)$-eqcol tal que $c^1(u) = c^1(q) = j$ y $c^2 = intro(c^1, u)$. Entonces, $\lambda^X_{uj} = \lambda^X_{un} + \lambda^W_n$.

(b) **Caso** $v = u$. Sea $q \in Q$, $w \in N(u) \cup Q \setminus \{q\}$, c^1 un $(n-1)$-eqcol tal que $c^1(u) = c^1(q) = n-1$, $c^1(w) = j$ y $c^2 = intro(c^1, u)$. Entonces, $\lambda^X_{un-1} = \lambda^X_{un} + \lambda^W_n$.
Para los casos restantes, sean $v_1, v_2 \in N(u)$ vértices no adyacentes.
Caso $v = v_1$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(v_1) = c^1(v_2) = n-1$, $c^1(u) = j$ y $c^2 = intro(c^1, v_1)$. Concluimos que $\lambda^X_{v_1 n-1} = \lambda^X_{v_1 n} + \lambda^W_n$.
Caso $v \in V \setminus \{u, v_1\}$. Sea c^1 un n-eqcol tal que $c^1(u) = j$, $c^1(v) = n-1$, $c^1(v_1) = n$ y $c^2 = swap_{n-1,n}(c^1)$. Entonces $\lambda^X_{vn-1} + \lambda^X_{v_1 n} = \lambda^X_{vn} + \lambda^X_{v_1 n-1}$ y, dado que $\lambda^X_{v_1 n-1} = \lambda^X_{v_1 n} + \lambda^W_n$, obtenemos $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$.

(c) Sea $q \in Q$, y sean $v_1, v_2 \in N(u)$ vértices no adyacentes.
Caso $v = v_1$. Sea c^1 un $(n-2)$-eqcol tal que $c^1(v_1) = c^1(v_2) = r$, $c^1(u) = c^1(q) = j$ y $c^2 = intro(c^1, v_1)$. Por lo tanto, $\lambda^X_{v_1 r} = \lambda^X_{v_1 n-1} + \lambda^W_{n-1}$.
Caso $v \neq v_1$. Sea c^1 un n-eqcol tal que $c^1(u) = j$, $c^1(v) = r$, $c^1(v_1) = n-1$ y $c^2 = swap_{r,n-1}(c^1)$. Tenemos que $\lambda^X_{vr} + \lambda^X_{v_1 n-1} = \lambda^X_{vn-1} + \lambda^X_{v_1 r}$ y, dado que $\lambda^X_{v_1 r} = \lambda^X_{v_1 n-1} + \lambda^W_{n-1}$, obtenemos $\lambda^X_{vr} = \lambda^X_{vn-1} + \lambda^W_{n-1}$.

(d) Sea $q \in Q$.
Caso $v = q$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(u) = c^1(q) = j$ y $c^2 = intro(c^1, q)$. Luego, $\lambda^X_{qj} = \lambda^X_{qn} + \lambda^W_n$.
Caso $v \neq q$. Sea c^1 un n-eqcol tal que $c^1(u) = n-1$, $c^1(q) = n$, $c^1(v) = j$ y $c^2 = swap_{j,n}(c^1)$. Tenemos entonces que $\lambda^X_{vj} + \lambda^X_{qn} = \lambda^X_{vn} + \lambda^X_{qj}$ y, dado que $\lambda^X_{qj} = \lambda^X_{qn} + \lambda^W_n$, obtenemos $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$.

(e) La hipótesis (i) nos asegura que existe un coloreo equitativo c^1 tal que la clase de color j es $\{u, v, q_1\}$ y existe otro coloreo equitativo c^2 (con la misma cantidad de colores) tal que la clase de color j es $\{u, q_2\}$, donde $q_1, q_2 \in Q$. Tenemos entonces

$$\sum_{w \in V \setminus \{u,v,q_1\}} \lambda^X_{wc^1(w)} + \lambda^X_{q_1 j} + \lambda^X_{vj} = \sum_{w \in V \setminus \{u,v,q_2\}} \lambda^X_{wc^2(w)} + \lambda^X_{q_2 j} + \lambda^X_{vc^2(v)}$$

y, por las condiciones (b)-(d), podemos derivar $\lambda^X_{vj} = \lambda^X_{vc^2(v)} = \lambda^X_{vn} + \lambda^W_{n-1}$.

(f) Sea $t = \lceil \frac{n}{k-1} \rceil - 1$. Es claro que $max\{j, \chi_{eq}\} \leq t \leq n-3$ y $\lceil \frac{n}{t} \rceil > \lceil \frac{n}{t+1} \rceil$. Por hipótesis (ii), existe un t-eqcol c cuya clase de color C_j satisface $C_j \subset N(u)$ y $|C_j| = \lceil n/t \rceil$, y además u y un vértice de Q usan el color t. Sea $c^1 = swap_{j,t}(c)$ y $c^2 = swap_{r,t}(c)$ (puesto que $t \geq j$ y $t \geq r$, ambos coloreos están bien definidos). Entonces, $c^1(u) = j$ y $c^2(u) = r$. Aplicando las condiciones probadas anteriormente a $\lambda^X x^1 = \lambda^X x^2$ (donde x^1 y x^2 son los vectores binarios x que representan a c^1 y c^2 respectivamente) nos permite derivar $\lambda^X_{ur} = \lambda^X_{un} + (k-1)\lambda^W_{n-1} + \lambda^W_n$.

(g) **Caso** $r \leq \lceil \frac{n}{2} \rceil - 1$. Procedemos como en (f), pero usando $t = \lceil \frac{n}{\lceil n/r \rceil - 1} \rceil - 1$ en vez de $\lceil \frac{n}{k-1} \rceil - 1$.
Caso $r \geq \lceil \frac{n}{2} \rceil$. Sea $q \in Q$, y sean $v_1, v_2 \in N(u)$ vértices no adyacentes. Consideremos el $(n-2)$-eqcol c^1 tal que $c^1(v_1) = c^1(v_2) = r$, $c^1(u) = c^1(q) = j$ y $c^2 = swap_{j,r}(c^1)$. Aplicando las condiciones probadas anteriormente a $\lambda^X x^1 = \lambda^X x^2$ (donde x^1 y x^2 son los vectores binarios x que representan a c^1 y c^2 respectivamente) nos permite derivar $\lambda^X_{ur} = \lambda^X_{un} + \lambda^W_{n-1} + \lambda^W_n$.

(h) Esta condición puede ser verificada conociendo un r-eqcol (x^1, w^1) y un $(r-1)$-eqcol (x^2, w^2) que caigan en F': se deben aplicar las condiciones (a)-(g) a la ecuación $\lambda^X x^1 + \lambda^W_r = \lambda^X x^2$ como se hizo en la condición (d) del Teorema 5.3.3.

Es decir que sólo necesitamos probar que, para cualquier $t \in \{\chi_{eq}, \ldots, n-2\}$, existe un t-eqcol c que cae en la cara F'.

Caso $t < j$. La existencia de c es garantizada por la monotonicidad de G.
Caso $j \leq t \leq n-3$. La existencia de c es garantizada por hipótesis (ii).
Caso $t = n-2$. c puede ser el $(n-2)$-eqcol dado en el ítem (c).

□

Demostración del Teorema 5.4.30. Observemos que $|J| \geq 1$. En particular, si $|J| = 1$ sólo puede ser $J = \{n-1\}$ y, según la Observación 5.4.28.1, define faceta de $\mathcal{ECP}$. Por eso, de ahora en más asumimos que $|J| \geq 2$.
Definamos $p = n - |M|$ y consideremos F' la cara de $\mathcal{ECP}$ definida por la desigualdad J-color y $F = \{(x,w) \in \mathcal{ECP} : \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad J-color. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_k los asociados a (5.2) para $k \in \{1,\dots,\chi_{eq}\}$, γ_k los asociados a (5.3) para $k \in \mathscr{S}$, δ el asociado a (5.4) y π el asociado a la desigualdad J-color. Dado que el coeficiente que acompaña a w_{p+1} en la desigualdad es $b_{Jp} - b_{Jp+1} = r$, de la combinación lineal se puede despejar $\pi = -\frac{1}{r}\lambda^W_{p+1}$, y luego δ, β_k, α_v y γ_k como se hizo en ocasiones anteriores. Sea $\theta(k) = \text{máx}\{i \in \mathbb{Z} : i \leq k,\ i \notin \mathscr{S}\}$ (i.e. $\theta(k) \notin \mathscr{S}$ pero $\theta(k)+1, \theta(k)+2, \dots, k \in \mathscr{S}$). Entonces obtenemos el siguiente sistema de ecuaciones:

(a) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in V,\ j \in J.$

(b) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \frac{1}{r}\lambda^W_{p+1}, \quad \forall\, v \in V,\ j \in \{1,\dots,n-1\}\backslash J.$

(c) $\sum_{t=\theta(k-1)+1}^{k} \lambda^W_t = (b_{Jk} - b_{J\theta(k-1)})\frac{1}{r}\lambda^W_{p+1}, \quad \forall\, k \in \{\chi_{eq}+1,\dots,n-1\}\backslash(\mathscr{S} \cup \{p+1\}).$

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior:

(a) Sea v' un vértice no adyacente a v (recordemos que G no tiene vértices universales) y sea c^1 un $(n-1)$-eqcol tal que $c^1(v) = c^1(v') = j$ y $c^2 = intro(c^1, v)$. Concluimos que $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$.

(b) Dado que $j \notin J$, entonces sólo puede ser $j \leq p$ así que un p-coloreo tiene una clase de color j. Sea $\{(u_1,u'_1),(u_2,u'_2),\dots,(u_{|M|},u'_{|M|})\}$ el matching M del complemento de G y $T = J\backslash\{p+1,\dots,n-1\}$. Dado que $\{p+1,\dots,n-1\} \subset J$ y $|J| = 2|M| - r - 1$, entonces $|T| = |J| - (n-p-1) = |M| - r$. Más aún, $T \neq \varnothing$.
Para probar $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \frac{1}{r}\lambda^W_{p+1}$, consideramos tres casos:
Caso $v = u_1$ **y** $r = 1$. Sea $T = \{t_1,t_2,\dots,t_{|M|-1}\}$, c^1 un p-eqcol tal que $c^1(u_{i+1}) = c^1(u'_{i+1}) = t_i$ para $1 \leq i \leq |M|-1$, $c^1(u_1) = c^1(u'_1) = j$ y $c^2 = intro(c^1, u_1)$. Por lo tanto, $\lambda^X_{u_1 j} = \lambda^X_{u_1 p+1} + \lambda^W_{p+1}$. Puesto que la condición (a) dice que $\lambda^X_{u_1 p+1} = \lambda^X_{u_1 n} + \lambda^W_n$, concluimos que $\lambda^X_{u_1 j} = \lambda^X_{u_1 n} + \lambda^W_n + \lambda^W_{p+1}$.
Caso $v = u_1$ **y** $r = 2$. Dado que $|M| \leq \lfloor \frac{n-1}{2} \rfloor$, tenemos $|\{1,\dots,p\}\backslash T| = p - |M| + 2 \geq 3$ y entonces podemos asegurar que existen diferentes colores $k,l \in \{1,\dots,p\}\backslash(T \cup \{j\})$. Además, existe un vértice $w \in V\backslash\{u_1,u'_1,\dots,u_{|M|},u'_{|M|}\}$ por que M no es un matching perfecto.
Ahora vamos a proponer pares de coloreos equitativos en donde cada par deriva una igualdad. Sea $T = \{t_1,t_2,\dots,t_{|M|-2}\}$ y c^1, c^2 coloreos equitativos tales que $c^1(u_{i+2}) = c^1(u'_{i+2}) = t_i$ para $1 \leq i \leq |M|-2$, $c^2(i) = c^1(i)$ para $i \in V\backslash\{u_1,u'_1,u_2,u'_2,w\}$ y los colores de los vértices u_1, u'_1, u_2, u'_2 y w son:

c^1						c^2					
tamaño	u_1	u'_1	u_2	u'_2	w	tamaño	u_1	u'_1	u_2	u'_2	w
p	j	j	k	k	l	$p+1$	$p+1$	$p+1$	k	j	l
p	l	l	j	j	k	p	l	l	k	k	j
n	j	$p+1$	l	k	n	n	$p+1$	j	l	k	n
n	l	$p+1$	k	n	j	n	l	$p+1$	j	n	k

Cada combinación nos da una igualdad diferente de la forma $\lambda^X x_1 + \lambda^W w_1 = \lambda^X x_2 + \lambda^W w_2$:

1. $\lambda^X_{u_1 j} + \lambda^X_{u'_1 j} + \lambda^X_{u'_2 k} = \lambda^X_{u_1 p+1} + \lambda^X_{u'_1 p+1} + \lambda^X_{u'_2 j} + \lambda^W_{p+1}$
2. $\lambda^X_{u_2 j} + \lambda^X_{u'_2 j} + \lambda^X_{wk} = \lambda^X_{u_2 k} + \lambda^X_{u'_2 k} + \lambda^X_{wj}$
3. $\lambda^X_{u_1 j} + \lambda^X_{u'_1 p+1} = \lambda^X_{u_1 p+1} + \lambda^X_{u'_1 j}$
4. $\lambda^X_{u_2 k} + \lambda^X_{wj} = \lambda^X_{u_2 j} + \lambda^X_{wk}$

La adición de estas igualdades resulta $2\lambda^X_{u_1 j} = 2\lambda^X_{u_1 p+1} + \lambda^W_{p+1}$. Puesto que la condición (a) afirma que $\lambda^X_{u_1 p+1} = \lambda^X_{u_1 n} + \lambda^W_n$, concluimos que $2\lambda^X_{u_1 j} = 2\lambda^X_{u_1 n} + 2\lambda^W_n + \lambda^W_{p+1}$.
Caso $v \neq u_1$. Sea c^1 un n-eqcol tal que $c^1(v) = j$, $c^1(u_1) = n$ y $c^2 = swap_{j,n}(c^1)$. Las condiciones probadas recientemente aplicadas a $\lambda^X x_1 = \lambda^X x_2$ nos permite obtener $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \frac{1}{r}\lambda^W_{p+1}$.

(c) Sean (x^1, w^1) y (x^2, w^2) un k-eqcol y $\theta(k-1)$-eqcol arbitrarios de G respectivamente. Si cualquiera de ellos no pertenece a F', siempre es posible intercambiar sus clases de color para que sí pertenezca. Por lo tanto, $\lambda^X x^1 + \sum_{t=\theta(k-1)+1}^{k} \lambda^W_t = \lambda^X x^2$. En virtud de las condiciones (a) y (b), y siguiendo pasos análogos a los de la prueba de (d) en 5.3.3, obtenemos $\sum_{t=\theta(k-1)+1}^{k} \lambda^W_t = (b_{Jk} - b_{J\theta(k-1)})\frac{1}{r}\lambda^W_{p+1}$.

□

Demostración del Teorema 5.4.33. Las hipótesis (i) y (ii) nos aseguran que existen vértices $u_1, u_2, u_3, u_4 \in S$ y $u_5, u_6 \in V \backslash S$ tales que G no tiene las aristas (u_1, u_2), (u_3, u_4) y (u_5, u_6). Estos vértices distinguidos serán empleados a lo largo de la prueba.

Sea F' la cara definida por esta desigualdad y $F = \{(x,w) \in \mathcal{ECP} \ : \ \lambda^X x + \lambda^W w = \lambda_0\}$ una cara tal que $F' \subset F$. Consideremos la combinación lineal del sistema minimal y la desigualdad (S,J)-color. Sean α_v los coeficientes asociados a las igualdades (5.1) para $v \in V$, β_k los asociados a (5.2) para $k \in \{1, \ldots, \chi_{eq}\}$, γ_k los asociados a (5.3) para $k \in \mathscr{S}$, δ el asociado a (5.4) y π el asociado a la desigualdad (S,J)-color. De la combinación lineal se puede despejar $\pi = -\lambda^W_{n-1}$, y luego δ, β_k, α_v y γ_k como se hizo en ocasiones anteriores. Sea $\theta(k) = \max\{i \in \mathbb{Z} : i \leq k, \ i \notin \mathscr{S}\}$ (i.e. $\theta(k) \notin \mathscr{S}$ pero $\theta(k)+1, \theta(k)+2, \ldots, k \in \mathscr{S}$). Entonces obtenemos el siguiente sistema de ecuaciones:

(a) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in V, \ j \in \{1, \ldots, n-2\}, \ (v,j) \in S \times J.$

(b) $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n, \quad \forall\, v \in V.$

(c) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{n-1}, \quad \forall\, v \in V, \ j \in \{1, \ldots, n-2\} \backslash J.$

(d) $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{n-1}, \quad \forall\, v \in V \backslash S, \ j \in J.$

(e) $\sum_{t=\theta(k-1)+1}^{k} \lambda^W_t = \left(b_{SJk} - b_{SJ\theta(k-1)}\right)\lambda^W_{n-1}, \quad \forall\, k \in \{\chi_{eq}+1, \ldots n-2\} \backslash \mathscr{S}.$

A continuación presentamos pares de coloreos equitativos de F' que nos permiten probar la validez de cada ecuación del sistema anterior:

(a) **Caso** $v = u_1$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(u_1) = c^1(u_2) = j$ y todos los colores de J son usados por vértices de S, y sea $c^2 = intro(c^1, u_1)$. Así obtenemos $\lambda^X_{u_1 j} = \lambda^X_{u_1 n} + \lambda^W_n$.
Caso $v \in S \backslash \{u_1\}$. Sea c^1 un n-eqcol tal que $c^1(v) = j$, $c^1(u_1) = n$ y todos los colores de J son usados por vértices de S, y sea $c^2 = swap_{j,n}(c^1)$. Así obtenemos $\lambda^X_{vj} + \lambda^X_{u_1 n} = \lambda^X_{vn} + \lambda^X_{u_1 j}$. Aplicando la condición recientemente probada $\lambda^X_{u_1 j} = \lambda^X_{u_1 n} + \lambda^W_n$ concluimos que $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n$.

(b) **Caso** $v = u_5$. Sea c^1 un $(n-1)$-eqcol tal que $c^1(u_5) = c^1(u_6) = n-1$ y todos los colores de J son usados por vértices de S, y sea $c^2 = intro(c^1, u_5)$. Así obtenemos $\lambda^X_{u_5 n-1} = \lambda^X_{u_5 n} + \lambda^W_n$.
Caso $v \in V \backslash \{u_5\}$. Sea c^1 un n-eqcol tal que $c^1(v) = n-1$, $c^1(u_5) = n$ y todos los colores de J son usados por vértices de S, y sea $c^2 = swap_{n-1,n}(c^1)$. Así obtenemos $\lambda^X_{vn-1} + \lambda^X_{u_5 n} = \lambda^X_{vn} + \lambda^X_{u_5 n-1}$. Aplicando la condición recientemente probada $\lambda^X_{u_5 n-1} = \lambda^X_{u_5 n} + \lambda^W_n$ concluimos que $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$.

(c) **Caso** $v = u_5$. Sea c^1 un $(n-2)$-eqcol tal que $c^1(u_5) = c^1(u_6) = j$, $c^1(u_1) = c^1(u_2) \in J$ y todos los colores de J son usados por vértices de S, y sea $c^2 = intro(c^1, u_5)$. Así obtenemos $\lambda^X_{u_5 j} = \lambda^X_{u_5 n-1} + \lambda^W_{n-1}$. Pero debido a que $\lambda^X_{u_5 n-1} = \lambda^X_{u_5 n} + \lambda^W_n$ podemos concluir que $\lambda^X_{u_5 j} = \lambda^X_{u_5 n} + \lambda^W_n + \lambda^W_{n-1}$.
Caso $v \in V \backslash \{u_5\}$. Sea c^1 un n-eqcol tal que $c^1(v) = j$, $c^1(u_5) = n$ y todos los colores de J son usados por vértices de S, y sea $c^2 = swap_{j,n}(c^1)$. Así obtenemos $\lambda^X_{vj} + \lambda^X_{u_5 n} = \lambda^X_{vn} + \lambda^X_{u_5 j}$. Aplicando la condición recientemente probada $\lambda^X_{u_5 j} = \lambda^X_{u_5 n} + \lambda^W_n + \lambda^W_{n-1}$ concluimos que $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{n-1}$.

(d) Observemos que S tiene al menos 4 vértices así que $|J| \geq 3$. Entonces podemos asumir que existen 2 colores $k, l \in J$ diferentes de j. Sea c^1 un $(n-2)$-eqcol tal que $c^1(v) = j$, $c^1(u_1) = c^1(u_2) = k$, $c^1(u_3) = c^1(u_4) = l$, todos los colores de $J \backslash \{j\}$ son usados por vértices de S, y sea $c^2 = swap_{j,n-1}(intro(c^1, u_1))$. Entonces c^2 es un $(n-1)$-eqcol que satisface $c^2(v) = n-1$ y $c^2(u_1) = j$. Luego, $\lambda^X_{vj} + \lambda^X_{u_1 k} = \lambda^X_{vn-1} + \lambda^X_{u_1 j} + \lambda^W_{n-1}$ y, en virtud de $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$ y $\lambda^X_{u_1 k} = \lambda^X_{u_1 n} + \lambda^W_n = \lambda^X_{u_1 j}$, obtenemos $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{n-1}$.

(e) Esta condición puede ser verificada conociendo un k-eqcol (x^1, w^1) y un $\theta(k-1)$-eqcol (x^2, w^2) pertenecientes a F': se deben aplicar las condiciones (a)-(d) a la ecuación $\lambda^X x^1 + \sum_{t=\theta(k-1)+1}^{k} \lambda^W_t = \lambda^X x^2$ como se hizo en la condición (d) del Teorema 5.3.3.

Es decir que sólo necesitamos probar que, para cualquier $r \in \{\chi_{eq}, \ldots, n-2\} \backslash \mathscr{S}$, existe un r-eqcol c que cae en la cara F'.
Caso $r \leq n-3$. La hipótesis (iii) garantiza la existencia de tal coloreo.
Caso $r = n-2$. c puede ser el coloreo dado en la condición (c).

□

Demostración del Corolario 5.4.34. El teorema 5.4.33 requiere, entre otras cosas, que el complemento de $G[S]$ tenga un matching de tamaño 2. Si esta hipótesis no se cumple entonces la condición (d) del teorema, i.e.

$$\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{n-1}, \quad \forall\, v \in V \backslash S,\ j \in J$$

puede probarse como sigue:
Dado que v no es universal en $G[(S \cup \{v\}) \backslash \{u_1, u_2\}]$, existe un vértice $u \in S \backslash \{u_1, u_2\}$ no adyacente a v. Sea c^1 un $(n-2)$-eqcol tal que $c^1(v) = c^1(u) = j$, $c^1(u_1) = c^1(u_2) \in J \backslash \{j\}$ y todos los colores de J son usados por vértices de S, y sea $c^2 = intro(c^1, v)$. Por lo tanto, $\lambda^X_{vj} = \lambda^X_{vn-1} + \lambda^W_{n-1}$. Pero la condición (b) del teorema prueba $\lambda^X_{vn-1} = \lambda^X_{vn} + \lambda^W_n$ así que podemos concluir que $\lambda^X_{vj} = \lambda^X_{vn} + \lambda^W_n + \lambda^W_{n-1}$. □

Bibliografía

[1] Aplicación dfmax, http://mat.gsia.cmu.edu/COLOR03.

[2] Herramienta PORTA, http://www2.iwr.uni-heidelberg.de/groups/comopt/software/PORTA.

[3] Herramienta zerOne, http://www.math.tu-berlin.de/∼luebbeck/zerone.html.

[4] IBM ILOG CPLEX 12.1, http://www-01.ibm.com/software/integration/optimization/cplex-optimizer.

[5] Librería COLORLIB de DIMACS, http://mat.gsia.cmu.edu/COLOR/instances.html.

[6] K. I. Aardal, A. Hipolito, C. P. M. van Hoesel, and B. Jansen. A branch-and-cut algorithm for the frequency assignment problem. Tech. Rep., Maastricht University, 1996.

[7] Kenneth Appel and Wolfgang Haken. Every planar map is four colorable. *Illinois J. Math.*, 21:429–567, 1977.

[8] David Applegate, Robert Bixby, Vašek Chvátal, and William Cook. *The Traveling Salesman Problem: A computational study*. Princeton University Press, 2006.

[9] David L. Applegate, Robert E. Bixby, Vašek Chvátal, William J. Cook, Daniel G. Espinoza, Marcos Goycoolea, and Keld Helsgaun. Certification of an optimal TSP tour through 85900 cities. *Operations Research Letters*, 37(1):11–15, 2009.

[10] Alper Atamtürk, George L. Nemhauser, and Martin W.P. Savelsbergh. Conflict graphs in solving integer programming problems. *European Journal of Operational Research*, 121(1):40–55, 2000.

[11] L. Bahiense, Y. Frota, N. Maculan, T. Noronha, and C. Ribeiro. A Branch-and-Cut Algorithm for Equitable Coloring based on a Formulation by Representatives. *Electr. Notes Discrete Math.*, 35(1):347–352, 2009.

[12] L. Bahiense, Y. Frota, N. Maculan, T. Noronha, and C. Ribeiro. Branch-and-Cut for Equitable Coloring. *International Network Optimization Conference (INOC), Pisa, Italy*, 2009.

[13] Laura Bahiense, Clicia Friedman, Samuel Jurkiewicz, Abel Lozano, Milene Pimenta, and Christina Waga. An integer programming approach to equitable coloring problems. Tech. Rep. RT EP 001/07, COPPE-Produção/UFRJ, May 2007.

[14] B. S. Baker and E. G. Coffman. Mutual exclusion scheduling. *Theoretical Computer Science*, 162:225–243, 1996.

[15] Ivo Blöchliger and Nicolas Zufferey. A graph coloring heuristic using partial solutions and a reactive tabu scheme. *Comput. Oper. Res.*, 35(3):960–975, 2008.

[16] C. Blum and A. Roli. Metaheuristics in Combinatorial Optimization: Overview and Conceptual Comparison. *ACM Computing Surveys*, 35(3):268–308, 2003.

[17] H. L. Bodlaender and K. Jansen. Restrictions of graph partition problems, part i. *Theoretical Computer Science*, 148:93–109, 1995.

[18] Daniel Brélaz. New methods to color vertices of a graph. *Comunications of the ACM*, 22(4):251–256, 1979.

[19] Rowland Leonard Brooks. On colouring the nodes of a network. *Proc. Cambridge Philosophical Society, Math. Phys. Sci.*, 37:194–197, 1941.

[20] Edmund K. Burke, Jakub Mareček, Andrew J. Parkes, and Hana Rudová. On a Clique-Based Integer Programming Formulation of Vertex Colouring with Applications in Course Timetabling. Tech. Rep. NOTTCS-TR-2007-10, The University of Nottingham, 2007.

[21] Manoel Campêlo, Ricardo Corrêa, and Victor Campos. On the asymmetric representatives formulation for the vertex coloring problem. *Discrete Applied Mathematics*, 156(7):1097–1111, 2008.

[22] Manoel Campêlo, Ricardo Corrêa, and Yuri Frota. Cliques, holes and the vertex coloring polytope. *Information Processing Letters*, 89:159–164, 2004.

[23] Gregory J. Chaitin, Marc A. Auslander, Ashok K. Chandra, John Cocke, Martin E. Hopkins, and Peter W. Markstein. Register allocation via coloring. *Computer Languages*, 6(1):47–57, 1981.

[24] Bor-Liang Chen and Kuo-Ching Huang. On the equitable colorings of kneser graphs. *India-Taiwan Conference on Discrete Mathematics, NTU*, 2009.

[25] Bor-Liang Chen, Ming-Tat Ko, and Ko-Wei Lih. Equitable and m-Bounded Coloring of Split Graphs. *Lecture Notes in Computer Science*, 1120:1–5, 1996.

[26] Bor-Liang Chen, Jing-Ho Yan, and Ko-Wei Lih. Equitable coloring of interval graphs and products of graphs. *manuscript*, 1998.

[27] Vašek Chvátal. *Linear Programming*. W. H. Freeman, 1983.

[28] François Margot. *Symmetry in Integer Linear Programming, 50 Years of Integer Programming*. Springer, 2009.

[29] P. Coll, J. Marenco, I. Méndez-Díaz, and P. Zabala. Facets of the Graph Coloring Polytope. *Annals of Operations Research*, 116(1):79–90, 2002.

[30] Harlan Crowder, Ellis L. Johnson, and Manfred Padberg. Solving Large-Scale Zero-One Linear Programming Problems. *Operations Research*, 31(5):803–834, 1983.

[31] Sajal K. Das, Irene Finocchi, and Rossella Petreschi. Conflict-free star-access in parallel memory systems. *J. Parallel Distrib. Comput.*, 66(11):1431–1441, 2006.

[32] Dominique de Werra. An introduction to timetabling. *European Journal of Operations Research*, 19:151–162, 1985.

[33] Marek Kubale et al. *Graph Colorings*. American Mathematical Society, Providence, Rhode Island, 2004.

[34] Leonhard Euler. Solutio problematis ad geometriam situs pertinentis. *Commentarii academiae scientiarum Petropolitanae*, 8:128–140, 1741.

[35] R. Figueiredo, V. Barbosa, N. Maculan, and C. de Souza. Acyclic Orientations with Path Constraints. *RAIRO - Operations Research*, 42(4):455–467, 2008.

[36] Hannah Furmańczyk and Marek Kubale. The complexity of equitable vertex coloring of graphs. *Journal of Applied Computer Science*, 13:97–107, 2005.

[37] Phillippe Galinier and Jin-Kao Hao. Hybrid evolutionary algorithms for graph coloring. *Journal of Combinatorial Optimization*, 3(4):379–397, 1999.

[38] Fred Glover. Tabu Search: A Tutorial. *Interfaces*, 20(1):74–94, 1990.

[39] Ralph E. Gomory. An algorithm for integer solutions to linear programs. In *Recent Advances in Mathematical Programming*, pages 269–302. McGraw-Hill, New York, 1963.

[40] M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1:169–197, 1981.

[41] András Hajnal and Endre Szemerédi. Proof of a conjecture of P. Erdös. *Combinatorial theory and its applications*, 2:601–623, 1970. Proc. Colloq., Balatonfüred, North-Holland, Amsterdam.

[42] Pierre Hansen, Martine Labbé, and David Schindl. Set covering and packing formulations of graph coloring: Algorithms and first polyhedral results. *Discrete Optimization*, 6(2):135–147, 2009.

[43] Heinrich Heesch. *Untersuchungen zum Vierfarbenproblem*. Mannheim: Bibliographisches Institut, 1969.

[44] Alain Hertz and Dominique de Werra. Using tabu search techniques for graph coloring. *Computing*, 39(4):345–351, 1987.

[45] S. Irani and V. Leung. Scheduling with conflicts, and applications to traffic signal control. *Proceedings of the Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 85–94, 1996.

[46] J. D. Isaacson, G. Marble, and D. W. Matula. Graph coloring algorithms. In *Graph Theory and Computing*, pages 109–122. Academic Press, New York, 1972.

[47] Svante Janson and Andrzej Ruciński. The infamous upper tail. *Random Structures & Algorithms*, 20(3):317–342, 2002.

[48] Richard M. Karp. Reducibility among combinatorial problems. *Complexity of Computer Computations*, pages 85–103, 1972.

[49] H.A. Kierstead and A.V. Kostochka. A short proof of the Hajnal-Szemerédi Theorem on equitable coloring. *Combin. Probab. Comput.*, 17:265–270, 2008.

[50] Hal A. Kierstead and Alexandr V. Kostochka. An Ore-type theorem on equitable coloring. *J. Combin. Theory, Series B*, 98(1):226–234, 2008.

[51] Henry Kierstead, Alexandr Kostochka, Marcelo Mydlarz, and Endre Szemerédi. A fast algorithm for equitable coloring. *Combinatorica*, 30(2):217–224, 2010.

[52] F. Kitagawa and H. Ikeda. An existential problem of a weight-controlled subset and its application to school timetable construction. *Discrete Mathematics*, 72:195–211, 1988.

[53] Arie Koster. *Frequency Assignment*. Phd. thesis, Universiteit Maastricht, 1999.

[54] Michael Krivelevich and Balázs Patkós. Equitable coloring of random graphs. *Random Struct. Algorithms*, 35(1):83–99, 2009.

[55] J. Lee and F. Margot. On a binary-encoded ILP coloring formulation. *INFORMS J. Comput.*, 19(3):406–415.

[56] Ko-Wei Lih. *The Equitable Coloring of Graphs, Handbook of Combinatorial Optimization Vol. 3*. Kluwer Academic, 1998.

[57] Ko-Wei Lih and Bor-Liang Chen. Equitable coloring of trees. *J. Combin. Theory, Series B*, 61(1):83–87, 1994.

[58] Ko-Wei Lih and Pou-Lin Wu. On equitable coloring of bipartite graphs. *Discrete Math.*, 151(1):155–160, 1996.

[59] J. T. Linderoth and M. W. P. Savelsbergh. A computational study of search strategies for mixed integer programming. *INFORMS J. Comput.*, 11:173–187, 1999.

[60] Enrico Malaguti, Michele Monaci, and Paolo Toth. An Exact Approach for the Vertex Coloring Problem. *Discrete Optimization*, 8(2):174–190, 2011.

[61] Anuj Mehrotra and Michael Trick. A column generation approach for graph coloring. *INFORMS J. Comput.*, 8(4):344–353, 1996.

[62] Isabel Méndez-Díaz. *Problema de Coloreo de Grafos, Un Estudio Poliedral y un Algoritmo Branch-and-Cut*. Tesis doctoral, DC FCEN Universidad de Buenos Aires, 2003.

[63] Isabel Méndez-Díaz, Graciela Nasini, and Daniel Severín. A polyhedral approach for the graph equitable coloring problem. *VI ALIO/EURO Workshop on Applied Combinatorial Optimization, Buenos Aires*, 2008.

[64] Isabel Méndez-Díaz and Paula Zabala. A cutting plane algorithm for graph coloring. *Discrete Applied Mathematics*, 156(2):159–179, 2008.

[65] Walter Meyer. Equitable coloring. *Amer. Math. Monthly*, 80:920–922, 1973.

[66] George Nemhauser and Laurence Wolsey. *Integer and Combinatorial Optimization*. John Wiley & Sons, 1988.

[67] Panos Pardalos, Thelma Mavridou, and Jue Xue. *The Graph Coloring Problem: A Bibliographic Survey, Handbook of Combinatorial Optimization Vol. 2*. Kluwer Academic, 1998.

[68] Sriram V. Pemmaraju. Equitable colorings extend Chernoff-Hoeffding bounds. In *Approximation, Randomization, and Combinatorial Optimization: Algorithms and Techniques*, volume 2129 of *Lecture Notes in Computer Science*, pages 285–296. 2001.

[69] Steffen Rebennack, Marcus Oswald, Dirk Theis, Hanna Seitz, Gerhard Reinelt, and Panos Pardalos. A Branch and Cut solver for the maximum stable set problem. *Journal of Combinatorial Optimization*, 21(4):434–457, 2011.

[70] Neil Robertson, Daniel Sanders, Paul Seymourc, and Robin Thomas. The Four-Colour Theorem. *Journal of Combinatorial Theory, Series B*, 70(1):2–44, 1997.

[71] E. C. Sewell. An Improved Algorithm for Exact Graph Coloring. In *Cliques, Coloring and Satisfiability*, volume 26, pages 359–373. DIMACS - Series in Discrete Mathematics and Theoretical Computer Science.

[72] B. F. Smith, P. E. Bjørstad, and W. D. Gropp. *Domain Decomposition. Parallel multilevel methods for elliptic partial differential equations*. Cambridge University Press, 1996.

[73] James Joseph Sylvester. Chemistry and algebra. *Nature*, 17:284–284, 1878.

[74] Alan Tucker. Perfect graphs and an application to optimizing municipal services. *SIAM Review*, 15:585–590, 1973.

[75] Vijay V. Vazirani. *Approximation Algorithms*. Springer-Verlag, 2003.

[76] Weifan Wang and Kemin Zhang. Equitable colorings of line graphs and complete r-partite graphs. *Systems Science and Mathematical Sciences*, 13(2):190–194, 2000.

[77] Douglas West. *Introduction to Graph Theory*. Prentice Hall, 2000.

[78] Laurence Wolsey. *Integer Programming*. Wiley-Interscience, 1998.

[79] H. P. Yap and Y. Zhang. The Equitable Δ-Coloring Conjecture holds for outerplanar graphs. *Bulletin of the Inst. of Math. Academia Sinica*, 25:143–149, 1997.