UNIVERSITAT DE VALÈNCIA

Departament d'Estadística i Investigació Operativa

Algunos problemas de rutas por arcos

Tesis doctoral

Thais Ávila Valverde

2014

Dirigida por:

Ángel Corberán Salvador

Isaac Plana Andani

José María Sanchis LLopis

D. Ángel Corberán Salvador, catedrático del Departament d'Estadística i Investigació Operativa de la Universitat de València, D. Isaac Plana Andani, profesor titular del Departamento de Matemáticas para la Economía y la Empresa de la Universitat de València y D. José María Sanchis Llopis, profesor titular del Departamento de Matemática Aplicada de la Universidad Politécnica de Valencia,

CERTIFICAN:

que la presente memoria de investigación **"Algunos problemas de rutas por arcos"** ha sido realizada bajo su dirección por Thais Ávila Valverde y constituye su tesis para optar al grado de *Doctor en Estadística y Optimización*.

Y para que así conste, en cumplimiento de la normativa vigente, autorizan su presentación ante la Facultat de Matemátiques de la Universitat de València para que puedan ser tramitadas su lectura y defensa públicas.

Burjassot, 19 de Diciembre de 2013

Ángel Corberán Salvador

José María Sanchis Llopis

Isaac Plana Andani

A mis Padres.

Agradecimientos

En primer lugar, quiero dar las gracias a la Generalitat Valenciana por concederme una beca predoctoral VALi+d y financiar el proyecto GVPROMETEO2013-049, así como a los Ministerios de Educación y Ciencia y de Economía y Competitividad, que han financiado los proyectos MTM2009-14039-C06-02 y MTM2012-36163-C06-02, bajo los cuales se ha realizado la labor investigadora recogida en esta memoria.
Durante estos tres años muchas personas han puesto su grano de arena para que esta tesis se pudiera realizar. Quiero dar las gracias a todas ellas pero en especial a las siguientes personas: a Ángel Corberán por creer en mí y darme la oportunidad de trabajar con él, con José Maria Sanchis y con Isaac Plana. Tres investigadores de prestigio dentro del campo de los problemas de rutas por arcos. Sin su experiencia y su capacidad de desarrollo como grupo de investigación, en el cual cada uno es imprescindible, no hubiera sido posible el desarrollo de esta tesis. También quiero dar las gracias a toda la gente del Departamento de Estadística e Investigación Operativa, en especial a las personas que durante este tiempo han compartido despacho conmigo, como Antonio Martínez Sykora, quien ha sido un apoyo imprescindible cuando el trabajo no fluía, Guillermo Vinué y Karen Cecilia Florez, por hacerme más ameno cada día. Sin olvidarme también del grupo de predocs de la Facultad de Matemáticas. Gracias a ellos hemos conseguido tener un buen ambiente de trabajo, así como la capacidad de motivarnos entre nosotros. Finalmente, a mis padres y a mi hermano, quienes siempre me han dado un amor incondicional y apoyo a todos los proyectos que he querido realizar, y que durante estos tres últimos años han sido la motivación para levantarme todas las mañanas e ir a trabajar.

Índice general

Lista de Acrónimos

ARP Arc Routing Problem (Problema de Rutas por Arcos)

ATSP Asymetric Traveling Salesman Problem (Problema del Viajante Asimétrico)

CPP Chinese Postman Problem (Problema del Cartero Chino)

DCPP Directed Chinese Postman Problem (Problema del Cartero Chino en un grafo dirigido)

DCVRP Distance Constrained Capacited VRP (Problema de Rutas de Vehículos con restricciones de distancia)

DGRP Directed General Routing Problem (Problema General de Rutas en un grafo dirigido)

DRPP Directed Rural Postman Problem (Problema del Cartero Rural en un grafo dirigido)

GDRPP Generalized Directed Rural Postman Problem (Problema del Cartero Rural Generalizado en un grafo dirigido)

GTSP Graphical TSP (Problema Gráfico del Viajante)

HC Desigualdades Honeycomb

ILS Iterated Local Search (Búsqueda Local Iterada)

K-C Desigualdades K-C

MDVRP VRP with Multiple Depots (Problema de Rutas de Vehículos con Múltiples Depósitos)

MCPP Mixed CPP (Problema del Cartero Chino en un grafo mixto)

MGRP Mixed GRP (Problema General de Rutas en un grafo mixto)

MRPP Mixed RPP (Problema del Cartero Rural en un grafo mixto)

MS Multi-Start Algorithm (Algoritmo Multi-Arranque)

OVRP Open Vehicle Routing Problem (Problema Abierto de Rutas de Vehículos)

PB Desigualdades Path-Bridge

PCVRP Precedence Constrained VRP (Problema de Rutas de Vehículos con Restricciones de Precedencia)

PL Programación Lineal o Problema Lineal

PLE Programación Lineal Entera o Problema de Programación Lineal Entera

POC Problema de Optimización Combinatoria

RPP Rural Postman Problem (Problema del Cartero Rural)

SDVRP Split Delivery Vehicle Routing Problem (Problema de Rutas de Vehículos con Demandas Compartidas)

SCP Stacker Crane Problem (Problema de la Grúa)

SCP Set Covering Problem (Problema de Cubrimiento de Conjuntos)

TSP Traveling Salesman Problem (Problema del Viajante)

VND Variable Neighborhood Descent (Búsqueda Descendente por Entornos Variables)

VRP Vehicle Routing Problem (Problema de Rutas de Vehículos)

VRPTW VRP with Time Windows (Problema de Rutas de Vehículos con Ventanas de Tiempo)

WPP Windy Postman Problem (Problema del Cartero Chino con Viento)

WRPP Windy RPP (Problema del Cartero Rural con Viento)

Índice de figuras

Introducción

Los problemas de rutas por arcos (Arc Routing Problems, ARP's) se modelizan en un grafo $G = (V, A)$ donde V es un conjunto de vértices y A es un conjunto de arcos o/y aristas (i, j) con costes c_{ij} no negativos. El objetivo de estos problemas es encontrar rutas óptimas que recorran, parcialmente o en su totalidad, los arcos o/y aristas del grafo.

El origen de los problemas de rutas por arcos está en el famoso *Problema de los Puentes de Königsberg* que resumimos a continuación. La ciudad de Königsberg estaba atravesada por el río Pregel e incluía dos grandes islas que estaban conectadas entre ellas y a su vez con las dos riberas del río mediante siete puentes (Figura 1). El problema consistía en encontrar un camino que cruzase todos los puentes de la ciudad una única vez.

Figura 1: *Los siete puentes de Königsberg*

Euler demostró en 1736 que no era posible tal camino. La demostración se encuentra en su publicación *"Solutio problematis ad geometriam situs pertonentis"*. La razón es

que el grafo que representa las cuatro partes de la ciudad y los siete puentes, tiene sus cuatro vértices "de grado impar". Euler señaló que si hay dos vértices de grado impar en un grafo, es posible encontrar un camino que atraviese todos los puentes exactamente una vez, si empezamos y acabamos en esos vértices. Si no hay vértices de grado impar, tal camino existe partiendo desde cualquier vértice (y volviendo al mismo). En honor de Euler, un camino que comience en uno de los vértices, atraviese una única vez cada arista del grafo y acabe en el punto de partida, se conoce como tour Euleriano y el grafo que lo contiene es un grafo Euleriano.

Pero el primer problema propiamente de rutas por arcos (donde aparece ya la optimización) es el problema del Cartero Chino (Chinese Postman Problem, CPP), introducido por Guan (Kwan) en 1960 en un artículo en chino y en 1962 en su traducción inglesa (*"Graphic Programming using odd and even points"*). En este trabajo, Guan planteaba el problema, al que se enfrenta un cartero, de encontrar un camino de longitud mínima para el reparto de la correspondencia. El CPP tiene como objetivo encontrar un camino cerrado de mínimo coste que pase por todos los arcos y/o aristas del grafo al menos una vez. Una generalización del CPP es el problema del Cartero Rural (Rural Postman Problem, RPP), cuyo objetivo es encontrar un camino cerrado de mínimo coste que pase al menos una vez por cada arco y/o arista de un subconjunto $R \subseteq A$.

La versión del RPP definido en un grafo mixto, en la que R es un conjunto de arcos y $A \backslash R$ es un conjunto de aristas, es conocida como el problema de la Grúa (Stacker Crane Problem, SCP), uno de los problemas que estudiaremos en esta tesis. El SCP fue introducido por Frederickson, Hecht and Kim en 1978.

Aunque las versiones dirigida y no dirigida del CPP se pueden resolver en tiempo polinómico, la versión mixta y windy son NP-difíciles. En el caso del RPP, tanto el caso dirigido como no dirigido son NP-difíciles, con lo que se concluye que el SCP es también un problema NP-difícil.

Los problemas de rutas por arcos tienen aplicación en la organización de tareas tales como la recogida de basura, el reparto de la leche o correo, la inspección de sistemas de distribución (redes eléctricas, de teléfono o de ferrocarril), la limpieza y riego de calles, etc. Todos los años se gastan millones de euros en estas operaciones y el ahorro que se puede conseguir con la optimización de éstas es enorme. El gran reto de estos problemas es que no se pueden modelizar como problemas sencillos de rutas por arcos. Cada problema tiene sus características propias, por lo que la metodología utilizada para su resolución ha de ser específica y ha de tener en cuenta el contexto del problema.

Los problemas de rutas por arcos, como hemos comentado antes, pueden ser definidos en grafos dirigidos, no dirigidos o mixtos, según las características de las carreteras, la topología de la calles de una ciudad, etc. Por ejemplo, si tenemos calles de dirección única, las representaremos a través de arcos (obteniendo un grafo dirigido) y si son de doble sentido se representan por aristas (dando lugar a un grafo no dirigido). Los grafos mixtos se utilizan en situaciones en las que existen calles de sentido único y de doble sentido. Muchos de estos problemas se pueden considerar como problemas del cartero rural porque, normalmente, no en todas las calles debe realizarse el servicio, y servir o recorrer una calle suele tener un coste. Generalmente, recorrer las calles sin realizar ningún servicio recibe el nombre de *"deadheading"*.

Los problemas que estudiamos en esta tesis son el problema General de Rutas por Arcos en un grafo dirigido (Directed General Routing Problem, DGRP), su caso particular, el problema de la Grúa (Stacker Crane Problem, SCP) y el problema del Cartero Rural Generalizado en un grafo dirigido (Generalized Directed Rural Postman Problem, GDRPP). Veamos la definición matemática de estos tres problemas.

El problema de la Grúa se define en un grafo mixto $G = (V, E, A)$, donde cada arista o arco, (i, j), tiene un coste asociado $c_{ij} \geq 0$, y tiene como objetivo hallar una ruta de coste mínimo que recorra al menos una vez cada arco del grafo. El problema General

de Rutas por Arcos en un grafo dirigido se define en un grafo $G = (V, A)$ con costes no negativos asociados a los arcos donde, dados un subconjunto de arcos requeridos $A_R \subseteq A$ y un subconjunto de vertices requeridos $V_R \subseteq V$, se trata de encontrar una ruta de mínimo coste que recorra todos los arcos requeridos y visite todos los vértices requeridos. Por último, el problema del Cartero Rural Generalizado en un grafo dirigido se define en un grafo $G = (V, A)$ con costes no negativos asociados a los arcos en el que, dados unos conjuntos de arcos $H_1, ..., H_n$, el objetivo es encontrar una ruta de coste mínimo que recorra al menos un arco de cada conjunto H_i.

La motivación que nos ha llevado a estudiar estos tres problemas es la siguiente. Inicialmente comenzamos a estudiar el SCP por la simplicidad de su planteamiento, la dificultad de su resolución y su aplicación a la resolución de problemas reales, como puede ser el caso de la optimización de los movimientos de las grúas de carga en puertos o en almacenes. Así, comenzamos desarrollando una metodología heurística, que se describe en el capítulo 3, para su resolución aproximada. También deseábamos implementar un algoritmo exacto para el SCP. A partir de una formulación para el SCP, empezamos a estudiar el poliedro de sus soluciones, tanto la dimensión como algunas posibles facetas. Al poco tiempo nos dimos cuenta de que el estudio que estábamos haciendo para el SCP podía extenderse sin mucha dificultad a un problema más general, y más importante, que es el DGRP. Además del SCP, el DGRP incluye como casos particulares a muchos otros problemas conocidos dentro del los problemas de rutas por arcos, como el problema del Cartero Chino Dirigido (DCPP), el problema del Cartero Rural Dirigido (DRPP) o el problema Gráfico del Viajante asimétrico (Graphical ATSP). Ésto nos llevó a estudiar directamente el DGRP y tratar al SCP como un caso particular. Por otro lado, leyendo trabajos recientes sobre los anteriores problemas, encontramos el GDRPP. Éste es un problema con un gran interés teórico y práctico que, a pesar de que en él el conjunto de arcos requeridos no es fijo y por lo tanto es una variable de decisión, tiene importantes similitudes con el DGRP. Puesto que estábamos trabajando ya en el DGRP, pensamos que podría ser una buena idea estudiar el GDRPP y aplicar todo aquello

que ya sabíamos del DGRP al GDRPP.

El estudio y resolución de estos problemas ha sido abordado en esta tesis desde el punto de vista de la Combinatoria Poliédrica, uno de los enfoques más útiles para la resolución de muchos problemas NP-difíciles de optimización combinatoria. También presentamos aquí un heurístico, basado en una búsqueda local iterada, para la resolución aproximada del SCP.

Una novedad en esta tesis que queremos resaltar es que el estudio poliédrico del DGRP, y del SCP, se ha realizado sobre el grafo original. A diferencia de la mayor parte de trabajos sobre problemas de rutas por arcos, nosotros no hemos supuesto aquí ningún tipo de transformación previa para que todos los vértices del grafo sean requeridos o incidentes con arcos requeridos.

Por otro lado, puesto que estudiamos problemas NP-difíciles, es altamente improbable que lleguemos a conocer la descripción completa de un poliedro y, por lo tanto, nos conformaremos con obtener descripciones parciales lo más ajustadas posible.

En esta tesis presentamos las siguientes familias de desigualdades válidas para el DGRP que, bajo ciertas condiciones, definen facetas de su poliedro asociado.

- Desigualdades triviales.
- Desigualdades de obligatoriedad.
- Desigualdades de conectividad.
- Desigualdades K-C.
- Desigualdades Honeycomb.
- Desigualdades Path-Bridge y 2-Path-Bridge asimétricas.

Algunas de estas desigualdades están basadas en otras que ya han sido propuestas para otros problemas de rutas por arcos. Otras han sido obtenidas expresamente para el DGRP. Muchas de estas desigualdades tienen en común la propiedad de que son desigualdades de configuración. Así, pueden ser asociadas al grafo resultante de particionar el grafo original en subconjuntos de vértices y representar los arcos no requeridos paralelos (entre los mismos pares de subconjuntos) por un solo arco. Los coeficientes de las variables asociadas a los arcos entre vértices pertenecientes a un mismo subconjunto serán 0, mientras que los coeficientes de los arcos requeridos y paralelos serán iguales. Esto permite ignorar "lo que ocurre" dentro de esos subconjuntos de vértices y tratar de la misma manera a los arcos que van entre los mismos subconjuntos. Se trata de un grafo comprimido al que se denomina grafo de configuración. Se demuestra en la tesis que, si una desigualdad define faceta del poliedro de soluciones en el grafo de configuración, también define faceta del poliedro de soluciones en el grafo original.

Para el estudio del poliedro del GDRPP, ha sido crucial el hecho de que se puede considerar a este problema como la combinación de dos problemas conocidos (y estudiados), el Set Covering Problem y el DGRP. Usando los resultados ya conocidos para el poliedro de soluciones del Set Covering y del DGRP, hemos podido demostrar que las siguientes desigualdades son válidas para el GDRPP:

- Desigualdades triviales.
- Desigualdades de obligatoriedad.
- Desigualdades de servicio.
- Desigualdades de conectividad.
- Desigualdades de imparidad.
- Desigualdades K-C.

También hemos probado que las desigualdades triviales, de obligatoriedad, de servicio, de conectividad y un caso particular de las K-C definen faceta del poliedro del GDRPP. Lamentablemente, no hemos sido capaces de demostrar ese mismo resultado en el caso de las desigualdades de imparidad ni en el de las K-C generales.

Puesto que es imposible incorporar explícitamente a la formulación todas desigualdades válidas que hemos encontrado (su número es exponencial), algunas de éstas pueden ser añadidas de manera iterativa mediante un algoritmo de planos de corte. En este tipo de algoritmos, se parte de un problema lineal que suele ser una parte de la relajación lineal del problema original. Se resuelve este PL inicial. Usualmente, la solución hallada estará fuera del poliedro de soluciones del problema, por lo que habrá que buscar alguna desigualdad válida que sea violada por esa solución y, en caso de encontrarla, añadirla al problema lineal para cortar la solución no factible. Para ello se utilizan los llamados algoritmos de separación, que buscan esa desigualdad dentro de una determinada familia de desigualdades. Este proceso se repite hasta que se obtiene una solución factible del problema, que será óptima, o hasta que no encontremos más desigualdades violadas, en cuyo caso lo que tenemos es una (buena) cota inferior al valor óptimo de nuestro problema. Otra posibilidad es usar algoritmos de ramificación y acotación. Éstos se basan en la construcción de un árbol de enumeración para explorar el espacio de soluciones y en la utilización de cotas inferiores obtenidas, generalmente, mediante la resolución de problemas lineales. También se suelen utilizar cotas superiores asociadas a soluciones posibles conocidas para limitar la búsqueda. Si en cada nodo del árbol de enumeración se ejecuta un algoritmo de planos de corte, tenemos los conocidos como algoritmos de ramificación y corte (Branch & Cut).

En esta tesis presentamos un algoritmo de ramificación y corte para el DGRP y otro para el GDRPP. Para comprobar la eficiencia de dichos algoritmos se han utilizado instancias obtenidas de la literatura y otras que han sido generadas ex profeso. Los tamaños de las instancias del DGRP y del SCP utilizadas varían entre 100 y 3000

arcos requeridos y 500 y 5000 vertices. En el caso de las instancias del GDRPP, los tamaños van desde 1000 a 1500 arcos, de 500 a 800 vértices y de 500 a 15000 clientes. Como se verá, los resultados obtenidos mejoran, a veces de forma sustancial, los presentados por otros autores para estos o similares problemas.

La tesis finaliza con unas conclusiones y las líneas de trabajo a seguir en el futuro.

Capítulo 1

Preliminares Matemáticos

Este capítulo recoge algunos conceptos básicos de teoría de grafos, álgebra lineal, programación lineal, teoría de poliedros y teoría de la complejidad algorítmica que son utilizados en capítulos posteriores.

1.1. Teoría de grafos

Los grafos suelen utilizarse para modelizar muchos problemas del mundo real, entre ellos los problemas de rutas por arcos. Gracias a los grafos, estos problemas se pueden representar de manera clara y precisa, aunque a priori el problema pueda parecer complicado y ambiguo. En esta sección, introduciremos algunos conceptos y resultados de la Teoría de Grafos que hemos empleado en el presente trabajo. Algunas referencias clásicas en esta área son Harary (1969), Berge (1973), Bondy y Murty (1976) y Christofides (1975).

Definición 1.1.1 (Grafo)

*Un **grafo** G se define como un par (V, E), donde V es un conjunto cuyos elementos son denominados vértices o nodos y E es un conjunto de pares de vértices que reciben el nombre de arista si el par es no ordenado y arco si el par es ordenado.*

Existen tres tipos de grafos según sus aristas y arcos:

- **No dirigidos:** E es un conjunto de pares no ordenados, es decir, está formado únicamente por aristas. Denotamos $G = (V, E)$.
- **Dirigidos:** E es un conjunto de pares ordenados, es decir, está formado únicamente por arcos. Normalmente en este caso se suele denotar a E por A. Denotamos $G = (V, A)$.
- **Mixto:** E está formado por pares ordenados y por pares no ordenados. En este caso se denota por E al conjunto de los pares no ordenados (aristas) y por A al conjunto de pares ordenados (arcos). Denotamos $G = (V, E, A)$.

A los vértices los denotaremos por $i, j, ...$, y por h_i, t_i si nuestra intención es resaltar que es el vértice final o inicial de un arco i. En general, denotaremos a las aristas como $e = (i, j)$ y a los arcos como $a = (i, j)$ o $a =< t_i, h_i >$. Representaremos por $n = |V|$ al número de vértices, por $m_E = |E|$ al número de aristas y por $m_A = |A|$ al número de arcos. Si el grafo sólo contiene arcos o aristas, su número será representado simplemente por m.

Definición 1.1.2 (Subgrafo)

*Diremos que G'=(V',E',A') es un **subgrafo** de G si $V' \subseteq V$, $E' \subseteq E$, $A' \subseteq A$.*

En muchos problemas de grafos es interesante definir una función que asigne un coste no negativo a cada arista o arco del grafo, es decir, dado un grafo $G = (V, E, A)$, definimos:

$$\begin{array}{rccc} C: & E \cup A & \longrightarrow & \mathbb{R} \\ & (i,j) & \longrightarrow & c_{ij} \geqslant 0 \end{array}$$

Definición 1.1.3 (Grafo completo)

Decimos que un grafo no dirigido G con n vértices es un ***grafo completo*** *si cada par de vértices está unido por una arista. A este tipo de grafo se le denota por K_n. Un grafo dirigido es completo si desde cada vértice sale un arco a todos los demás vértices.*

Definición 1.1.4 (Grafo bipartido)

Un grafo G es un ***grafo bipartido*** *si existe una bipartición de $V = X \cup Y$, con X, Y no vacíos, de manera que toda arista o arco del grafo tenga un extremo en X y otro en Y.*

Definición 1.1.5 (Grado de un vértice)

Podemos definir tres tipos de grados para un vértice:

- ***Grado del vértice*** *v: Número de aristas y arcos incidentes con v. Se representará por $d(v)$. Diremos que un vértice es par cuando su grado sea par. Impar en caso contrario.*
- ***Grado de entrada del vértice*** *v: Número de arcos que entran en v. Lo representaremos por $d^-(v)$.*
- ***Grado de salida del vértice*** *v: Número de arcos que salen de v. Lo representaremos por $d^+(v)$.*

Teorema 1.1.1

El número de vértices de grado impar en un grafo es siempre par.

Si S_1 y S_2 son dos subconjuntos disjuntos de V, denotaremos por $(S_1 : S_2)$ al conjunto de todas las aristas y arcos de G con un vértice en S_1 y el otro en S_2. En particular, si representamos por $\overline{S}$ a $V \setminus S$, el conjunto $(S : \overline{S})$ se denota como $\delta(S)$

y se llama cortadura del grafo:

Definición 1.1.6 (Cortadura)

Llamaremos ***cortadura*** *de G a $\delta(S) = (S : \overline{S})$ para cualquier $S \subseteq V$ no vacío, y diremos que una cortadura es par (resp. impar) si el número de aristas y arcos que contiene es par (resp. impar). En particular, $\delta(\{i\})$ (abreviadamente $\delta(i)$), $i \in V$, es una cortadura de G.*

Teorema 1.1.2

$\delta(S) = (S : \overline{S})$ es una cortadura impar de G si y sólo si S (y $\overline{S}$) contiene un número impar de vértices impares.

Definición 1.1.7 (Camino)

Un ***camino*** *es una secuencia $w = (i_0, e_1, i_1, e_2, ..., e_k, i_k)$ en la que los vértices y las aristas (o arcos) aparecen alternativamente y tales que los vértices finales de cualquier arista (o arco) e_r son i_{r-1} e i_r. Si tratamos con grafos simples, es decir, aquellos que no contienen aristas o arcos en paralelo, el camino puede ser representada exclusivamente por la secuencia de vértices $i_0, ..., i_k$. Un camino en la que todos los vértices son distintos recibe el nombre de* ***camino simple****. Definimos el coste de un camino como la suma de los costes asociados a las aristas (o arcos) que lo forman.*

Dado un grafo G no dirigido, diremos que dos vértices están conectados si y sólo si existe un camino entre ellos. En el caso de un grafo dirigido hemos de exigir que, además de un camino de i a j, exista otro de j a i.

Definición 1.1.8 (Conexión)

Dado un grafo G no dirigido, decimos que es ***conexo*** *si dado cualquier par de vértices del grafo existe al menos un camino entre ellos. Un grafo mixto, o dirigido, es* ***fuertemente conexo*** *si, para cualquier par de vértices i y j, existe un camino desde i a j y otro de j a i. Diremos que es* ***débilmente conexo*** *si el grafo no*

dirigido subyacente es conexo.

Definición 1.1.9 (Componente conexa)

Dado un grafo no dirigido, llamamos ***componente conexa*** *de G a cada subgrafo conexo maximal de G. Las componentes conexas de G forman una partición de sus vértices V.*

Definición 1.1.10 (Tour)

Dado un grafo G, llamaremos ***tour*** *(o closed walk) a un camino que acabe y empiece en el mismo vértice.*

Teorema 1.1.3

Si w es un tour sobre las aristas de G, entonces w pasa un número par de veces (quizás cero) por las aristas o/y arcos de cualquier cortadura de G.

Definición 1.1.11 (Grafo Euleriano)

Decimos que un grafo G es ***Euleriano*** *si podemos encontrar un tour que pase por todas sus aristas y arcos exactamente una vez.*

Teorema 1.1.4

Un grafo G conexo y no dirigido es Euleriano si, y sólo si, todos los vértices son de grado par.

Un grafo G dirigido fuertemente conexo es Euleriano si, y sólo si, para todos los vértices se tiene que el grado de salida y el grado de entrada son iguales.

Definición 1.1.12 (Grafo Hamiltoniano)

Decimos que un grafo G es ***Hamiltoniano*** *si existe un tour que pase por todos los vértices de G una única vez.*

Definición 1.1.13 (Árbol)

Un ***árbol*** *es un grafo no dirigido conexo que no contiene ciclos.*

Teorema 1.1.5

En un árbol, cualquier par de vértices está unido por un único camino.

Teorema 1.1.6

Si G es un árbol, entonces $m = n - 1$.

Definición 1.1.14 (Árbol generador)

Dado un grafo G no dirigido, un ***árbol generador*** *de G es un subgrafo de G que contiene a todos los vértices y además es árbol.*

Dado un grafo conexo, podemos encontrar siempre un árbol generador, pero éste no tiene porqué ser único. El coste de un árbol generador se define como la suma de los costes de sus aristas. Llamaremos **árbol de mínimo coste** a aquél en el que la suma de los costes de sus aristas sea menor.

Definición 1.1.15 (Acoplamiento)

Sea G un grafo no dirigido. Un subconjunto de aristas se dice que es un ***acoplamiento*** *de G si no tiene bucles y no hay dos aristas incidentes con el mismo vértice. Diremos que M es un acoplamiento perfecto si, para cualquier vértice del grafo, hay una arista de M incidente con él.*

Dado un grafo G no dirigido, con un número par de vértices y cuyas aristas tienen asociadas un coste, el problema del **Acoplamiento Perfecto de Coste Mínimo** consiste en encontrar un acoplamiento perfecto en G (si existe) que tenga la suma de los costes de sus aristas mínimo.

1.2. Álgebra lineal

Denotaremos por $\mathbb{R}^n$ el espacio de todos los vectores columna de n componentes reales. Sean $x_i \in \mathbb{R}^n$, $i = 1, ..., m$.

Definición 1.2.1 (Combinación lineal)

Diremos que un vector $y \in \mathbb{R}^n$ *es una* ***combinación lineal*** *de los* x_i *si y sólo si* y *se puede expresar como* $y = \sum_{i=1}^{m} \lambda_i x_i$ *, con* $\lambda_i \in R$ *cualesquiera.*

Definición 1.2.2 (Combinación afín)

Diremos que un vector $y \in \mathbb{R}^n$ *es una* ***combinación afín*** *de los* x_i *si, además de ser combinación lineal se cumple que* $\sum_{i=1}^{m} \lambda_i = 1$.

Definición 1.2.3 (Combinación convexa)

Llamaremos combinación convexa a una combinación afín que cumpla $\lambda_i \geq 0$ *para cada* i.

Definición 1.2.4 (Combinación cónica)

Si una combinación lineal cumple $\lambda_i \geq 0$ *para cada* i*, entonces diremos que es una* ***combinación cónica****.*

A partir de estas definiciones y dado un conjunto de vectores $S \subseteq \mathbb{R}^n$, llamaremos **envoltura lineal** (resp. **afín**, **convexa** y **cónica**) de S al conjunto de todas las combinaciones lineales (resp. afines, convexas y cónicas) que se pueden construir con vectores de S, y la representaremos por $span(S)$ (resp. $aff(S)$, $conv(S)$ y $cone(S)$). Diremos que S es un **subespacio lineal** (resp. **subespacio afín**, **conjunto convexo** y **cono**) de $\mathbb{R}^n$ si $S = span(S)$ (resp. $aff(S)$, $conv(S)$, $cone(S)$).

Definición 1.2.5 (Independencia lineal)

Diremos que los vectores $x_i \in \mathbb{R}^n$*,* $1 \leq i \leq m$*, son* ***linealmente independientes*** *si ninguno de ellos es combinación lineal de los demás o, equivalentemente, si* $\sum_{i=0}^{m} \lambda_i x_i = 0$ *implica que* $\lambda_i = 0$ *para todo* i *. En caso contrario diremos que son* ***linealmente dependientes****.*

Definición 1.2.6 (Independencia afín)

Diremos que los vectores $x_i \in \mathbb{R}^n$, $1 \leq i \leq m$ son ***afínmente independientes*** *si ninguno de ellos es combinación afín de los demás o, equivalentemente, si $\sum_{i=0}^{m} \lambda_i x_i = 0$, con $\sum_{i=0}^{m} \lambda_i = 0$, implica que $\lambda_i = 0$ para todo i . En otro caso, diremos que los vectores son* ***afínmente dependientes****.*

Teorema 1.2.1

Sea $S \subseteq \mathbb{R}^n$. Son equivalentes:

1. *S es afínmente independiente.*
2. *Dado $y \in S$, $\{x - y : x \in S, x \neq y\}$ es linealmente independiente.*
3. *Dado $y \in \mathbb{R}^n$, $\{x - y : x \in S\}$ es afínmente independiente.*

Definición 1.2.7 (Rango)

Sea $S \subseteq \mathbb{R}^n$, llamaremos ***rango*** *de S al número máximo de vectores de S linealmente independientes, y lo denotaremos por $rg(S)$. De forma similar, llamaremos* ***rango afín*** *de S al número máximo de vectores de S afínmente independientes, y lo denotaremos por $r_a(S)$. Y en el caso de una matriz A, llamaremos rango de A $(rg(A))$ al rango del conjunto de sus vectores columna, que es el mismo que el rango de sus vectores fila.*

Teorema 1.2.2

Sea $S \subseteq \mathbb{R}^n$

1. *Si $0 \in aff(S)$, entonces $r_a(S) = rg(S) + 1$.*
2. *Si $0 \notin aff(S)$, entonces $r_a(S) = rg(S)$*

Definición 1.2.8 (Base)

Si S es un subespacio lineal de $\mathbb{R}^n$, llamaremos ***base*** *de S a cualquier subconjunto finito B de vectores linealmente independientes de S tal que $span(B) = S$. Todas*

las bases de un mismo subespacio lineal tienen el mismo número de vectores.

Definición 1.2.9 (Dimensión)
Si S es un subespacio lineal, llamaremos ***dimensión*** *de S al número de vectores que forman una base de S. Si S es un subespacio afín de $\mathbb{R}^n$, existe un único subespacio lineal S' de $\mathbb{R}^n$ tal que $S' = \{x - x^* : x \in S\}$, para cualquier $x^* \in S$. Llamaremos* ***dimensión*** *de S a la dimensión de S'. Por último, si S es un subconjunto arbitrario de $\mathbb{R}^n$, llamaremos* ***dimensión*** *de S a la dimensión de aff(S) y la denotaremos por $dim(S)$.*

1.3. Teoría de poliedros

Muchos de los problemas de optimización combinatoria, como los problemas de rutas por arcos, se pueden formular como $min\{c^T x : x \in S\}$, donde $x \in \mathbb{R}^{\kappa}$ es el vector de las variables de decisión, $c \in \mathbb{R}^{\kappa}$ es el vector de coeficientes de la función objetivo y S es el conjunto de soluciones posibles. La mayoría de las veces S viene definido por un sistema de ecuaciones y desigualdades lineales con coeficientes enteros, por lo que existe una matriz $A \in \mathbb{Z}^{m \times n}$ y un vector $b \in \mathbb{Z}^m$, de manera que $S = \{x \in \mathbb{Z}^n : Ax \leq b\}$.

Definición 1.3.1 (Hiperplano y semiespacio)
Sean $a \in \mathbb{R}^n$ y $\alpha \in R$.

- *Llamaremos* ***hiperplano*** *de $\mathbb{R}^n$ al conjunto $\{x \in \mathbb{R}^n : a^T x = \alpha\}$.*
- *Llamaremos* ***semiespacio*** *de $\mathbb{R}^n$ al conjunto $\left\{x \in \mathbb{R}^n : a^T x \leq \alpha\right\}$.*

Definición 1.3.2 (Poliedro)
Llamaremos ***poliedro de*** *$\mathbb{R}^n$ a la intersección de un número finito de semiespacios de $\mathbb{R}^n$ o, equivalentemente, al conjunto de soluciones de un sistema de inecuaciones*

de la forma $Ax \leq b$, donde A es una matriz $m \times n$ y $b \in \mathbb{R}^m$. Llamaremos ***polítopo*** *a un poliedro acotado. Diremos que un poliedro P es de* ***dimensión completa*** *si $dim(P) = n$*

Teorema 1.3.1

$P \subseteq \mathbb{R}^n$ es un poliedro si y sólo si existen V y E, subconjuntos finitos de $\mathbb{R}^n$, tales que $P = conv(V) + cone(E)$.

Definición 1.3.3

Sean $a \in \mathbb{R}^n$, $\alpha \in R$.

- *Diremos que $a^T x \leq \alpha$ es una* ***desigualdad válida*** *para el poliedro P si $P \subseteq \{x \in \mathbb{R}^n : a^T x \leq \alpha\}$.*
- *Diremos que $F \subseteq \mathbb{R}^n$ es una* ***cara*** *del poliedro P si existe una desigualdad válida $a^T x \leq \alpha$ para P tal que $F = P \cap \{x \in \mathbb{R}^n : a^T x = \alpha\}$. En este caso diremos que la cara F está inducida por la desigualdad $a^T x \leq \alpha$. Varias desigualdades pueden inducir la misma cara de un poliedro P. En este caso diremos que esas desigualdades son equivalentes respecto a P.*
- *Llamaremos* ***caras propias*** *a las caras no vacías de un poliedro P. Nótese que, con esta definición, los vértices y aristas de un poliedro son caras.*
- *Así mismo, llamaremos* ***faceta*** *del poliedro P a cualquier cara propia no vacía de P que sea maximal respecto a la inclusión de conjuntos. De esta manera, el concepto de faceta se corresponde con la idea coloquial de cara de un poliedro.*

Teorema 1.3.2

Sea $P \subseteq \mathbb{R}^n$ un poliedro y sea F una cara propia no vacía de P inducida por una desigualdad válida $a^T x \leq \alpha$. Entonces, son equivalentes:

1. *F es una faceta de P.*
2. *$dim(F) = dim(P) - 1$.*

3. *Si* $b^T x \leq \beta$ *es válida para* P, $F \subseteq P \cap \{x \in \mathbb{R}^n : b^T x = \beta\}$ *y* $aff(P) = \{x \in \mathbb{R}^n : Ax = d\}$, *donde* A *es una matriz* $m \times n$, $d \in \mathbb{R}^m$, *entonces existen* $\lambda \in \mathbb{R}^n$ *y* $\mu \geq 0$ *tales que* $b^T = \mu a^T + \lambda^T A$.

Definición 1.3.4 (Vértice)

Llamaremos ***vértice*** *del poliedro* P *a cualquier cara propia no vacía de* P *que sea minimal respecto a la inclusión de conjuntos, es decir, a cualquier cara de* P *que esté formada por un único punto* $F = \{v\}$. *Llamaremos* ***poliedro entero*** *a cualquier poliedro cuyos vértices tienen todas sus componentes enteras.*

Teorema 1.3.3

$v \in P$ *es un vértice si y sólo si* v *no puede expresarse como combinación convexa de otros puntos de* P.

Los artículos de Bachem y Grötschel (1982) y Pulleyblank (1983) y los libros de Schrijver (1986) y Nemhausser y Wosley (1988) proporcionan una información abundante sobre los conceptos y resultados de la Teoría de Poliedros y la Combinatoria Poliédrica.

1.4. Teoría de la complejidad algorítmica

En Matemáticas, la palabra problema generalmente hace referencia a una cuestión concreta a resolver o una tarea a realizar. Sin embargo, nosotros llamaremos **problema** a una cuestión general planteada en términos de algunos "parámetros abiertos" para la cual se busca una "solución". Un problema se describe mediante una lista de sus "parámetros abiertos" y una lista de las propiedades que deben satisfacer sus soluciones. Si especificamos todos los "parámetros abiertos" con unos valores concretos, entonces tenemos lo que llamaremos **instancia** del problema. Por ejemplo, el problema "resolver un sistema de ecuaciones lineales" puede ser definido como

"dada una matriz A mxn y un vector $b \in \mathbb{R}^m$, ¿existe algún vector $x \in \mathbb{R}^n$ tal que $Ax = b$?" A y b son los parámetros abiertos. Una instancia de este problema sería, por ejemplo, encontrar x, y, z tales que:

$$\begin{aligned} 3x - y + 2z &= 1 \\ -x + 7y - 4z &= 3 \end{aligned}$$

Llamaremos **algoritmo** a una secuencia de pasos sin ambigüedad que llevan a la solución de un problema. Diremos que un algoritmo resuelve un problema si encuentra una solución para cada instancia del problema. Para determinar la "eficiencia" de un algoritmo debemos considerar todos los "recursos" que emplea dicho algoritmo. Por ejemplo, si utilizamos un ordenador para ejecutar un algoritmo, el uso de los distintos recursos del ordenador (tiempo de ejecución, memoria, etc.) nos proporcionará distintas medidas para evaluar la eficiencia del algoritmo. Sin embargo, normalmente el tiempo de ejecución es el recurso dominante y se acepta que el algoritmo más rápido para resolver un problema es el más eficiente. Por último, para obviar las diferencias que hay entre los distintos ordenadores, podemos definir el **tiempo de ejecución** de un algoritmo como el número de operaciones elementales requeridas para llegar a una solución del problema.

Naturalmente, el tiempo de ejecución depende del "tamaño" de la instancia, luego es razonable expresar el tiempo de ejecución de un algoritmo en función del tamaño de las instancias.

Codificar una instancia significa representarla mediante una secuencia finita de símbolos (que servirá de entrada al ordenador). Llamaremos tamaño o longitud de una instancia I al número de símbolos necesarios para codificarla, y lo denotaremos por $l(I)$. Claramente, $l(I)$ depende no sólo de I sino del esquema de codificación empleado.

Sea Π un problema, A un algoritmo que resuelve Π y sea I una instancia de Π.

Llamaremos tiempo de ejecución de A para resolver I, $t_A(I)$, al número de pasos elementales necesarios para llegar a una solución de I. Llamaremos función de tiempo de ejecución de A a la función $f_A : \mathbb{N} \to \mathbb{N}$ tal que $f_A(n) = max\{t_A(I) : l(I) \leq n\}$, es decir, que asigna a cada $n \in \mathbb{N}$ el máximo tiempo de ejecución necesario para resolver cualquier instancia I de tamaño menor o igual que n. Diremos que A es un **algoritmo polinómico** si existe un polinomio $p(n)$ tal que $f_A(n) = O(p(n))$, es decir, si existe una constante M tal que $|f_A(n)| \leq M\,|p\,(n)|\,,\ \forall n$. En otro caso diremos que A es un **algoritmo exponencial**.

Aceptaremos que los algoritmos polinómicos son eficientes, puesto que su tiempo de ejecución crece de un modo "razonable" a medida que crece el tamaño de la instancia (aunque este crecimiento puede ser considerablemente grande si el grado del polinomio lo es). De este modo, podemos decir que un problema es "fácil" si puede ser resuelto mediante un algoritmo polinómico y que es "difícil" en otro caso. Para obtener una clasificación más rigurosa de problemas según su complejidad algorítmica procederemos como sigue:

Consideremos primero los problemas de decisión, es decir, problemas cuyas instancias tienen solamente dos posibles soluciones: "sí" o "no". Denotaremos por $\boldsymbol{P}$ la clase de todos los problemas de decisión que pueden ser resueltos mediante un algoritmo polinómico. Por ejemplo, el problema Π_1 : "dado un grafo G, ¿contiene un ciclo?" es un problema de decisión, y puede probarse que está en P. Sin embargo, para el problema de decisión Π_2: "dado un grafo G, ¿contiene un ciclo hamiltoniano?", no se conoce ningún algoritmo polinómico que lo resuelva, luego no se sabe si está en P o no. Observemos que una instancia I de Π_2 cuya solución sea "sí" contendrá un ciclo hamiltoniano y, conociendo ese ciclo, podemos comprobar la veracidad de la solución "sí" en un tiempo polinómico en $l(I)$ (basta comprobar que efectivamente es un ciclo hamiltoniano). Diremos que los problemas con esta propiedad están en la clase $\boldsymbol{NP}$.

En general, diremos que un problema de decisión Π está en la clase NP si para cada instancia I de Π con solución "sí" existe una estructura S (el ciclo hamiltoniano en el problema Π_2) tal que, conociendo S, podemos comprobar la veracidad de la solución "sí" en un tiempo polinómico en $l(I)$. Notemos que en esta definición se exige solamente la existencia de S y no necesariamente un procedimiento para encontrarla.

Obviamente $P \subseteq NP$. La cuestión "¿es $P = NP$?" es el principal problema abierto de la Teoría de la Complejidad Algorítmica, aunque una conjetura generalmente aceptada es que la inclusión es estricta.

Sean Π_1 y Π_2 dos problemas de decisión. Llamaremos transformación polinómica de Π_1 en Π_2 a cualquier algoritmo polinómico que para cada instancia I_1 produce una instancia I_2 de Π_2 tal que la solución para I_1 es "sí" si y sólo si la solución para I_2 es "sí". Diremos que un problema de decisión Π^* es NP**-completo** si está en NP y cualquier otro problema de NP es transformable polinómicamente en Π^* . Esta definición implica que si Π^* es NP-completo y $\Pi^* \in P$, entonces $P = NP$. Por lo tanto, si aceptamos la conjetura de que $P \neq NP$, no debe existir ningún algoritmo polinómico que resuelva Π^*. En este sentido, los problemas NP-completos son los más difíciles de la clase NP. De la definición de NP-completo se deduce que un problema Π es NP -completo si está en NP y otro problema NP-completo conocido es transformable polinómicamente en Π. Este hecho ha servido para demostrar que una gran cantidad de problemas son NP-completos. El primer problema NP-completo conocido fue el problema de la Satisfabilidad (Cook, 1971). El problema del Ciclo Hamiltoniano considerado anteriormente es también NP-completo.

Diremos que un problema Π_1 es reducible polinómicamente a otro problema Π_2 si:

1. Existe un algoritmo A_1 que resuelve Π_1 usando como subrutina un algoritmo A_2 para Π_2.

2. A_1 es un algoritmo polinómico si y sólo si A_2 es un algoritmo polinómico.

Los problemas que vamos a tratar en esta tesis son problemas de Optimización Combinatoria, que no son propiamente problemas de decisión y, por lo tanto, no pueden ser NP-completos. Así mismo, hay problemas de decisión en los que son transformables polinómicamente todos los problemas de NP pero no está demostrada su pertenencia a NP y, por lo tanto, tampoco se les puede clasificar como NP-completos. Para todos estos tipos de problemas se reserva el término NP**-difícil** (NP-hard). Un problema es NP-difícil si todos los problemas de NP son reducibles (no necesariamente transformables) polinómicamente a él.

Una gran cantidad de problemas de optimización son NP-difíciles y, por lo tanto, si asumimos que $P \neq NP$ no podremos encontrar algoritmos polinómicos que los resuelvan. Por este motivo, la investigación actual sobre estos problemas se divide básicamente en tres ramas distintas pero complementarias:

- Diseñar algoritmos heurísticos que sean rápidos y que produzcan soluciones "próximas" a la solución óptima.
- Buscar casos especiales de esos problemas que sí puedan ser resueltos en tiempo polinómico.
- Diseñar algoritmos que busquen la solución óptima.

Aunque hemos visto que no podemos esperar encontrar algoritmos polinómicos que resuelvan cualquier instancia, es interesante diseñar algoritmos exponenciales exactos por varios motivos. Primero, para resolver instancias de tamaño no demasiado grande. Si el tiempo de ejecución crece de modo exponencial con el tamaño de la instancia, no podremos resolver instancias de "cualquier" tamaño, pero al menos sí podremos resolver instancias "pequeñas", y cuanto mejor sea el algoritmo, mayores serán las instancias que seremos capaces de resolver. En segundo lugar, la definición de tiempo de ejecución se hace sobre el peor caso, es decir, para resolver una

instancia de tamaño n necesitamos realizar $t_A(n)$ operaciones como máximo. Pero puede ocurrir que instancias concretas de gran tamaño tomadas del mundo real se resuelvan en un tiempo razonable.

Una presentación más extensa y rigurosa de estos conceptos puede encontrarse en Papadimitriou (1976, 1984).

1.5. Programación lineal y entera

Llamaremos problema de **Programación Lineal**, abreviadamente PL, al problema de minimizar (maximizar) una función lineal $f(x) = c^T x$, $c \in \mathbb{R}^n$, que llamaremos función objetivo, sobre un poliedro $P \subseteq \mathbb{R}^n$. Llamaremos solución posible a cualquier $x \in P$ y llamaremos **solución óptima** a cualquier $x^* \in P$ tal que:

$$c^T x^* = min\{c^T x : x \in P\}$$

Teorema 1.5.1

Si P tiene al menos un vértice y $min\{c^T x : x \in P\}$ es finito, entonces existe al menos un vértice en P que es solución óptima.

Teorema 1.5.2

Para cada vértice x^* de P, existe un vector $c \in \mathbb{R}^n$ tal que $c^T x^* < c^T x$ para cada $x \in P \backslash \{x^*\}$.

Corolario 1.5.1

Un poliedro P es entero si y sólo si para cada $c \in \mathbb{R}^n$ tal que $min\{c^T x : x \in P\}$ es finito, existe un vector entero $x^* \in P$ tal que:

$$c^T x^* = min\{c^T x : x \in P\}$$

Dantzig desarrolló en 1945, un algoritmo para la resolución de Problemas de Programación Lineal, llamado **método simplex** (Dantzig, 1963). Este algoritmo no es polinómico y para demostrar este resultado se encontró un conjunto de instancias de PL para las que el simplex necesita hacer un número de operaciones que depende exponencialmente del tamaño de la instancia. Por otro lado, Khachian (1979) propuso un algoritmo polinómico, llamado **método de las elipsoides**, para la resolución de los problemas de PL. La aportación más importante de este algoritmo es la constatación de que PL es un problema perteneciente a la clase P.

Definición 1.5.1 (Problema de Programación Lineal Entera)
Llamaremos ***problema de Programación Lineal Entera*** *o, abreviadamente* PLE*, al problema de minimizar (maximizar) una función lineal sobre los vectores enteros de un poliedro* $P \subset \mathbb{R}^n$.

Definición 1.5.2 (Relajación lineal)
Llamamos ***relajación lineal*** *de un* PLE *al* PL *resultante de relajar la restricción de integridad en el PLE.*

El PLE es un problema NP-difícil, pues una gran cantidad de problemas NP-difíciles conocidos se pueden formular como un PLE. Los métodos más utilizados para resolver el PLE son los llamados métodos de **Branch and Bound**, y el modo de hacer más eficientes estos algoritmos consiste en encontrar cotas superiores e inferiores al valor óptimo de la función objetivo lo más ajustadas posible. Otros métodos más recientes, como el de **Branch and Cut** y **Branch and Price**, han demostrado su gran potencial al resolver problemas específicos de PLE.

Las siguientes referencias, entre otras, son excelentes textos de Programación Lineal y Programación Lineal Entera: Dantzig (1963), Garfinkel y Nemhausser (1972), Bazaraa y Jarvis (1977), Chvátal (1983), Schrijver (1986) y Nemhausser y Wolsey (1988).

1.6. Combinatoria poliédrica

En esta sección resumiremos la idea fundamental de la aproximación poliédrica a la resolución de los problemas de Optimización Combinatoria, especialmente de los NP-difíciles.

Dado un conjunto E finito (llamado conjunto base) con una función de coste y una familia $\mathcal{F}$ (finita o infinita numerable) de subconjuntos (o listas) de E llamados soluciones posibles, el **problema de optimización combinatoria** (POC) con función objetivo lineal consiste en encontrar un $F^* \in \mathcal{F}$ tal que $c(F^*) = \sum_{e \in F^*} c_e x_e$ sea mínimo (o máximo), donde x_e denota el número de veces que $e \in E$ aparece en F^*.

La aproximación poliédrica a los POC comienza con la creación de un poliedro $\mathrm{P}_{\mathcal{F}}$, cuyos vértices, posiblemente junto a otros puntos de $P_{\mathcal{F}}$, siguen una correspondencia uno-a-uno con las soluciones posibles de $\mathcal{F}$. Se define el vector de incidencia x^F de una solución posible F en $\mathcal{F}$ como $x^F = \left(x_e^F\right)_{e \in E} \in Z^{|E|}$, donde x_e^F denota el número de veces que e aparece en F, y entonces definimos un conjunto convexo $P_{\mathcal{F}}$ como: $P_{\mathcal{F}} = conv\{x^F : F \in \mathcal{F}\}$.

Aunque el conjunto $P_{\mathcal{F}}$ no es un poliedro en general, para la mayor parte de los POC se puede demostrar que sí lo es. Supondremos pues que $P_{\mathcal{F}}$ es un poliedro. Claramente, cada solución posible F en $\mathcal{F}$ corresponde a un punto de $P_{\mathcal{F}}$ y cada vértice de $P_{\mathcal{F}}$ corresponde a una solución posible en $\mathcal{F}$. Por lo tanto, para cualquier función de costes c para la que el POC tiene un valor óptimo finito, podemos encontrar una solución óptima del POC por medio de un vértice óptimo del PL:

$$\min_{x \in P_{\mathcal{F}}} \quad cx \tag{1.1}$$

Lo que hemos hecho ha sido transformar de un modo natural el POC en el PL

(1.1). Sin embargo, sin tener conocimiento alguno del sistema lineal que describe $P_{\mathcal{F}}$, esta transformación es inútil desde el punto de vista algorítmico. Para abordar el problema (1.1) con las técnicas de la Programación Lineal, es necesario conocer todo (o una parte suficientemente grande de) el sistema lineal que describe $P_{\mathcal{F}}$.

Sin embargo, aunque conociéramos una descripción lineal completa del poliedro $P_{\mathcal{F}}$, no podríamos generar y almacenar todos los datos del PL asociado, ya que, aún siendo el sistema lineal finito, cabe esperar que sea extremadamente grande (en muchos problemas, el número de filas crece exponencialmente con el número de variables del problema). Una alternativa a la enumeración de todas las filas conocidas del sistema lineal es resolver periódicamente el siguiente problema:

Problema de Identificación de Facetas (Problema de Separación)

Dado un punto $\overline{x} \in \mathbb{R}^{|E|}$ y un poliedro P_F, encontrar una desigualdad lineal $f(x) \leq f_0$ que defina una faceta de $P_{\mathcal{F}}$ que sea violada por $\overline{x}$ (es decir, $f(\overline{x}) > f_0$), o demostrar que tal desigualdad no existe (en cuyo caso $\overline{x} \in P_{\mathcal{F}}$).

Utilizando el problema de Identificación de Facetas como subrutina en un código de Programación Lineal, podríamos generar desigualdades (que definen facetas de $P_{\mathcal{F}}$) violadas a medida que las necesitamos para "cortar" los óptimos fraccionarios de los problemas lineales.

El procedimiento iterativo para la resolución de los POC, introducido por Grötschel y Padberg (1985), es:

Método de Relajación para Problemas de Optimización Combinatoria:

1. **Inicialización**. Sea (PL_0) una relajación lineal del POC que consideramos. Hacer $k = 0$ e ir a (2).

2. **Resolución del** PL. Resolver (PL_k). Sea x^k una solución óptima de (PL_k). Ir a (3).

3. **Identificación de facetas**. Resolver el problema de identificación de facetas para x^k y $P_{\mathcal{F}}$.

   3.1 Si se encuentran una o más desigualdades (que definan facetas de $P_{\mathcal{F}}$) violadas por x^k , definir (PL_{k+1}) como $(PL_k$) junto a las desigualdades encontradas. Hacer $k = k + 1$ e ir a (2).

   3.2 Si no existe una desigualdad violada, parar.

Si en el paso (2) utilizamos el método simplex, el procedimiento anterior será finito y convergente, y la terminación en (3.2) ocurrirá en una solución óptima del P.O.C, siempre y cuando conozcamos la descripción lineal completa de $P_{\mathcal{F}}$.

Desgraciadamente, la descripción completa del sistema 1ineal asociado al poliedro $P_{\mathcal{F}}$ se conoce únicamente para unos pocos problemas de Optimización Combinatoria. Además, éste no es el caso de ningún problema NP-difícil. De hecho, resultados de la Teoría de la Complejidad Algorítmica conducen a pensar que es altamente improbable llegar a un conocimiento completo del sistema lineal que represente el poliedro $P_{\mathcal{F}}$ asociado a un problema NP-difícil (Papadimitrou, 1977).

Sin embargo, los descubrimientos en Combinatoria Poliédrica de los últimos años han demostrado que incluso un conocimiento parcial del sistema lineal que describe $P_{\mathcal{F}}$ es, muy frecuentemente, fuente de nuevos resultados significativos tanto en el aspecto teórico como en el computacional. Se puede utilizar la descripción del poliedro $P_{\mathcal{F}}$ para mejorar la relajación lineal inicial, añadiendo una lista parcial de desigualdades (que definen facetas) en el Método de Relajación.

Si durante la aplicación del Método de Relajación ocurre que no encontramos una

desigualdad violada, puede ser porque:

i) no existe.

ii) no conocemos la descripción completa de $P_{\mathcal{F}}$.

iii) no sabemos como generar algorítmicamente una desigualdad conocida.

Entonces, cuando lleguemos a (3.2) en el Método de Relajación, podemos acudir a un esquema del tipo de Branch and Bound o de Branch and Cut, pero con mayor probabilidad de alcanzar el óptimo, ya que tenemos una relajación lineal mucho más ajustada.

Así pues, uno de los retos fundamentales en Combinatoria Poliédrica es encontrar un subsistema suficientemente grande del sistema lineal completo que describe el poliedro $P_{\mathcal{F}}$ para diseñar estrategias eficientes de resolución del problema.

Algunos de los primeros ejemplos en los que esta aproximación ha llevado a resultados importantes son el problema del Acoplamiento (Grötschel y Holland, 1985), el problema del Viajante (Grötschel y Padberg, 1985 y Padberg y Rinaldi, 1990), el problema de la Ordenación Lineal (Grötschel, Jünger y Reinelt, 1984) y el VRP o problema de Rutas de Vehículos (Augerat et al., 1995). Esta aproximación a los POC es, de hecho, el centro de atención de la Combinatoria Poliédrica. Excelentes artículos sobre el particular son los de Pulleyblank (1983), Grötschel (1984), Hoffman y Padberg (1985) y el libro de Nemhausser y Wolsey (1988).

Capítulo 2

Problemas de Rutas

En este capítulo presentamos un resumen más exhaustivo de los problemas de rutas más conocidos. Los problemas de rutas son problemas de Optimización Combinatoria, que consisten en encontrar la mejor solución (**óptima**) entre un número finito (pero enorme) o infinito numerable de soluciones. El número de soluciones generalmente depende de $n!$ o de k^n (donde n podría representar el tamaño del problema y k es un entero mayor o igual que 2). Por ello para resolver este tipo de problemas es necesario estudiar sus propiedades particulares que nos permitan diseñar un método de resolución específico para ese problema.

Los problemas de rutas se modelizan en un grafo $G = (V, E)$, donde existe un coste no negativo asociado a los arcos o/y aristas del grafo. Estos problemas se dividen en dos clases, los problemas de rutas por vértices y los problemas de rutas por arcos.

Los primeros se caracterizan porque el servicio a realizar se hace en los vértices del grafo, mientras que en los segundos el servicio se realiza en los arcos o aristas del grafo. En ambos casos el objetivo es encontrar un tour de mínimo coste que pase por los vértices, arcos y aristas del grafo en los que hay que realizar un servicio.

La importancia del estudio de estos problemas se fundamenta en la gran cantidad de problemas del mundo real que pueden ser formulados a través de ellos. Por ejemplo, la mayor parte de organismos públicos o empresas privadas han de plantearse problemas relacionados con el reparto del correo, recogida de basuras, limpieza, inspección o mantenimiento de calles, carreteras o redes eléctricas, distribución de todo tipo de productos, mercancías o mensajes, visitas a clientes, transporte de personal, etc. Esta es la razón por la que en los últimos 35 años se haya hecho un estudio importante en este campo, conseguiéndose un avance significativo tanto en la formulación de los problemas como en el diseño, análisis e implementación de algoritmos para su resolución.

2.1. Problemas de rutas por Vértices

Como hemos comentado anteriormente la característica principal de estos problemas es que los clientes que requieren servicio están representados como vértices del grafo. Básicamente consisten en la obtención de uno o varios ciclos que pasen por todos (o algunos de) los vértices del grafo y de forma que la distancia total recorrida sea mínima. Dentro de este tipo de problemas podemos destacar:

- El problema del Viajante (Traveling Salesman Problem, TSP), donde un único vehículo ha de atravesar todos los vertices del grafo una única vez de manera que la distancia recorrida sea mínima. El planteamiento clásico del TSP se realiza sobre un grafo completo. El TSP es uno de los problemas más estudiados en la literatura de los problemas de rutas y fue Karp (1972) quien demostró que el TSP es un problema NP-difícil.

- El problema Gráfico del Viajante (Graphical Traveling Salesman Problem, GTSP), que es una variante del TSP introducida por Fleischmann (1985, 1988) y Cornuèjols, Fonlupt y Naddef (1985) en la que el grafo no tiene por qué ser

completo y el vehículo puede visitar más de una vez cada vértice del grafo.

- El problema de rutas de vehículos (Vehicle Routing Problem, VRP), que es una generalización del TSP en la que la demanda de los clientes requiere el uso de más de un vehículo. Este problema es más difícil que el TSP puesto que se hace necesario particionar el conjunto de clientes para que éstos puedan ser atendidos por los vehículos y después determinar el orden de servicio de cada cliente.

Casos particulares del VRP son:

- El problema de rutas de vehículos con restricciones de distancia (Distance Constrained VRP, DCVRP). Cada vehículo, además de tener una capacidad o carga máxima, también está limitado por un tiempo maximo que puede emplear o una distancia máxima que puede recorrer.

- El problema de rutas de vehículos con ventanas de tiempo (VRP with Time Windows, VRPTW). En este caso el servicio a cada cliente debe realizarse dentro de un horario o ventana de tiempo. Un cliente puede tener más de una ventana de tiempo. El vehículo no puede atender la demanda de un cliente fuera de su ventana horaria.

- El problema de rutas de vehículos con demanda compartida (Split Delivery VRP, SDVRP). En este problema es posible atender la demanda de un cliente por más de un vehículo. Esta relajación del VRP puede significar la reducción del número de vehículos necesarios para atender la demanda total.

- El problema de rutas de vehículos con restricciones de precedencia (Precedence Constrained VRP, PCVRP). En ocasiones la colocación de la mercancía en el vehículo obliga a que unos clientes sean servidos antes que otros. Esto supone una secuenciación en el orden de visita de los clientes.

- El problema del Dial-A-Ride (DRP). Este problema está definido por el hecho

de que, además de que las rutas deben satisfacer unas ventanas de tiempo de recogida y entrega de los clientes, éstos no deben permanecer en el vehículo más de un tiempo límite (ride time).

- El problema de rutas de vehículos abierto (OVRP). Aquí se considera la posibilidad de que las rutas sean abiertas, es decir, caminos que visitan clientes sin necesidad de retornar al depósito.
- El Problema de rutas de vehículos con múltiples depósitos (MDVRP). En este caso, existen diversos depósitos que sirven de punto de partida y retorno a los vehículos.

2.2. Problemas de rutas por arcos

Los problemas que tratamos en esta tesis se engloban dentro de los problemas de rutas por arcos. Como hemos comentado anteriormente, en este tipo de problemas el vehículo ha de recorrer un conjunto de arcos y/o aristas del grafo, donde se encuentra el servicio a realizar. Según el tipo de grafo en el que se definen los problemas de rutas por arcos, éstos se clasifican en:

- **No dirigidos:** el grafo está formado únicamente por aristas. Éstas suelen utilizarse para modelizar, por ejemplo, calles de doble sentido. El coste de atravesar una arista es el mismo en los dos sentidos.
- **Dirigidos:** el grafo está formado por arcos únicamente. Suelen utilizarse para modelizar calles de sentido único.
- **Mixtos:** el grafo contiene arcos y aristas.
- **Windy:** el grafo está formado por aristas, pero el coste de recorrerlas depende del sentido en que se haga.

En lo que sigue describiremos con mayor detalle algunos de los problemas más importantes de rutas por arcos. Para cada uno de ellos, describiremos su complejidad algorítmica y repasaremos las estrategias más importantes que se conocen para solucionarlos.

Los trabajos de Bodin, Golden, Assad y Ball (1983), Benavent, Campos, Corberán y Mota (1983) fueron de los primeros en proporcionar una panorámica general sobre los problemas de rutas por arcos. Bodin y Golden (1981) ofrecen una clasificación detallada de estos problemas y Lenstra y Rinnooy-Kan (1981) discuten su complejidad. Más recientes y completos, los trabajos de Eiselt, Gendreau y Laporte (1995a, 1995b) , Assad y Golden (2000), y el libro editado por Dror (2000) actualizan gran parte del trabajo realizado sobre los problemas de rutas por arcos. El artículo de Corberán y Prins (2010) comenta los últimos trabajos publicados en este campo.

En las siguientes secciones, describiremos los problemas de rutas por arcos más importantes con un único vehículo: aquellos cuyas definiciones aparecen de un modo natural según la demanda esté en todos los arcos o aristas de un grafo, en algunos de ellos o, incluso, simultáneamente en algunos vértices y en algunos arcos o aristas.

2.2.1. El problema del Cartero Chino

El problema del Cartero Chino fue introducido en 1960 (en 1962 en su traducción al inglés) por el matemático chino Meigu Guan al diseñar el recorrido que debía realizar un cartero para repartir la correspondencia. El problema consiste en encontrar una ruta cerrada de coste mínimo que recorra cada arco o arista del grafo al menos una vez. Fue Edmonds (1965) quien demostró que la versión no dirigida del problema es resoluble polinómicamente. Posteriormente se ha demostrado que también la versión dirigida puede ser resuelta en tiempo polinómico, por lo que ambas versiones se consideran problemas resueltos.

El problema del Cartero Chino en un grafo no dirigido (CPP)

Consideremos un grafo no dirigido $G = (V, E)$ con un coste c_e asociado a cada arista $e \in E$. El CPP consiste en encontrar un tour de coste mínimo en G que recorra cada arista de E al menos una vez.

Es fácil ver que, para que exista una solución óptima finita, hemos de suponer que el grafo G es conexo y que los costes c_e son no negativos. Un tratamiento exhaustivo del CPP desde el punto de vista de la Teoría de Grafos puede encontrarse en Fleischner (1990, 1991). Nosotros presentaremos aquí los resultados más importantes sobre el problema desde el punto de vista de la optimización.

El CPP está muy relacionado con el problema de encontrar un acoplamiento perfecto de mínimo coste, puesto que a partir de G, añadiendo aristas que unan los vértices impares dos a dos, podemos obtener un grafo Euleriano (Edmonds y Johnson, 1973; Christofides, 1973). Edmonds y Johnson (1973) resolvieron el CPP utilizando una adaptación del algoritmo *blossom* para el problema de acoplamiento perfecto (Edmonds, 1965). Demostraron, pues, que el CPP es resoluble polinómicamente y dieron una descripción completa del poliedro asociado.

En el contexto del problema del acoplamiento perfecto, Grötschel y Holland (1985) implementaron un algoritmo de planos de corte que demostró ser tan eficiente como los algoritmos existentes de tipo combinatorio. Así pues, su procedimiento puede ser también considerado como un método exacto para resolver el CPP .

El problema del Cartero Chino en un grafo dirigido (DCPP)

Consideremos ahora un grafo dirigido $G = (V, A)$, con un coste c_a asociado a cada arco $a \in A$. El DCPP se define como el problema de encontrar un tour en G que

recorra cada arco de A al menos una vez, con coste total mínimo.

Es fácil ver que, para que exista una solución óptima finita, hemos de suponer que el grafo G es fuertemente conexo y que no existe ningún ciclo de coste negativo. Este último supuesto lo sustituiremos por el de que los costes c_a sean no negativos.

Bajo las condiciones anteriores, un grafo Euleriano de mínimo coste puede obtenerse resolviendo un problema de transporte: para cada vértice i, sea d_i el número de arcos entrando en i menos el número de arcos que salen de i. Sea S el conjunto de vértices i con $d_i > 0$ y T el conjunto de vértices i con $d_i < 0$. El DCPP puede plantearse como el problema de encontrar el plan de transporte de coste mínimo entre las plantas S y los destinos T (Liebling, 1970; Edmonds y Johnson, 1973).

Así pues, el DCPP es también resoluble polinómicamente y se conoce una descripción completa de su poliedro asociado.

El problema del Cartero Chino en un grafo mixto (MCPP)

Consideramos ahora el caso en el que tenemos un grafo con aristas, que pueden ser recorridas en uno u otro sentido con el mismo coste, y arcos, que pueden ser recorridos solamente en un sentido. El MCPP puede definirse del modo siguiente:

Sea $G = (V, E, A)$ un grafo mixto con costes c_e asociados a los arcos y aristas $e \in E \cup A$. Encontrar un tour de coste mínimo en G que recorra cada enlace de $E \cup A$ al menos una vez.

De nuevo es fácil ver que, para que exista una solución óptima finita, hemos de suponer que el grafo G es fuertemente conexo y que no existe ningún ciclo de coste negativo. Como en el caso dirigido, este último supuesto lo sustituiremos por el de que los costes c_e sean no negativos.

Papadimitriou (1976) demostró que, a diferencia de las versiones dirigida y no dirigida del CPP, el MCPP es un problema NP-difícil. Christofides et al. (1984) propusieron una formulación con una variable por cada arco, dos variables por cada arista (representando el número de veces que es atravesada en cada dirección) y una variable por cada vértice. Desarrollaron un algoritmo de branch and bound en el que dos cotas inferiores diferentes, obtenidas de relajar lagrangianamente dos tipos de restricciones, eran calculadas en cada nodo del árbol de ramificación. Usando este algoritmo resolvieron un conjunto de 34 instancias generadas aleatoriamente con $7 \leq |V| \leq 50$, $3 \leq |A| \leq 85$ y $4 \leq |E| \leq 39$. Nobert y Picard (1996) propusieron una formulación para el MCPP en la que hay una única variable asociada a cada arista de E. La solución entera, pues, no especifica la dirección en que las aristas son recorridas. Basándose en la condición necesaria y suficiente para que un grafo mixto sea Euleriano (Ford and Fulkerson, 1962), los autores formulan restricciones que aseguran que el grafo solución será par y equilibrado. Su principal contribución es la demostración de que el problema de separación para las restricciones de equilibrio puede ser resuelto polinómicamente. Basándose en ello, diseñan un algoritmo de planos de corte con el que resuelven 148 de las 180 instancias probadas. Los tamaños de estas instancias están en los rangos $10 \leq |V| \leq 169$, $2 \leq |A| \leq 2876$ y $15 \leq |E| \leq 1849$. Las dos formulaciones propuestas para el MCPP , con una y dos variables por arista descritas anteriormente, han sido comparadas teórica y computacionalmente en Corberán, Mota y Sanchis (2006).

Hasta donde sabemos, el mejor algoritmo exacto para el MCPP es el propuesto por Corberán et al. (2012) para el WPP. Es capaz de resolver óptimamente instancias del MCPP con hasta 3000 vértices y 9000 arcos y aristas. Únicamente 12 de las 120 instancias probadas no pudieron ser resueltas en dos horas de computación, y en ellas el gap final obtenido fue menor del 1 %.

Es importante notar que, si G es par, el MCPP es resoluble en tiempo polinómico (Edmonds y Johnson, 1973). El algoritmo exacto para el caso en que G sea par es la

base de dos heurísticos para el caso general sugeridos por Edmonds y Johnson (1973) y desarrollados y mejorados por Frederickson (1979). El primer algoritmo equivaldría a transformar primero G en un grafo par y aplicar luego el método antes citado. El segundo algoritmo puede considerarse como la versión inversa: primero obtiene un grafo simétrico y luego lo mantiene simétrico mientras lo hace par. Frederickson (1979) demostró que ambos algoritmos tienen radio del peor caso 2, mientras que si aplicamos ambos algoritmos y seleccionamos la mejor solución, el radio del peor caso es, ahora, 5/3. Raghavachari y Veerasamy (1999) han propuesto una modificación al algoritmo conjunto anterior cuyo radio del peor caso es 3/2. Corberán, Martí y Sanchis (2002) han desarrollado un algoritmo tipo GRASP que ha proporcionado soluciones posibles distantes, en media, un 1.12 % de cotas inferiores en 225 instancias con $50 \leq |V| \leq 200$ y $120 \leq |E \cup A| \leq 600$.

El poliedro del MCPP como caso particular del MRPP y del MGRP, ha sido estudiado en Romero (1997), Corberán, Romero y Sanchis (2003) y Corberán, Mejía y Sanchis (2003).

El problema del Cartero Chino con viento (WPP)

Supondremos ahora que las aristas del grafo pueden recorrerse en uno u otro sentido pero con costes diferentes según el sentido del recorrido. El problema del Cartero Chino con viento (Windy Postman Problem, WPP) puede definirse como sigue.

Sea $G = (V, E)$ un grafo no dirigido con dos costes asociados a cada arista $(i, j) \in E$: c_{ij}, que es el coste de recorrer dicha arista desde i hasta j y c_{ji}, que es el coste de recorrer la arista de j a i. El WPP consiste en hallar un tour en G de coste mínimo que recorra cada arista de E al menos una vez.

Una condición suficiente para que exista una solución óptima finita es que el grafo

G sea conexo y que todos los costes sean no negativos.

Este problema, propuesto por Minieka (1979), contiene como casos particulares a las tres versiones anteriores del CPP y, puesto que el MCPP es NP-difícil, el WPP también lo es.

Un buen estudio sobre el WPP puede encontrarse en la tesis de Win (1989). Allí se demuestra que el WPP se puede resolver en tiempo polinómico en algunos supuestos, entre ellos si G es par. Este resultado es la base de un algoritmo heurístico, con radio del peor caso igual a 2, que primero convierte el grafo en un grafo par y luego aplica el algoritmo exacto para grafos pares. Raghavachari and Veerasamy (1999) proponen una modificación de este algoritmo y demuestran que tiene una cota del peor caso de 3/2. Win (1989) también propone otro heurístico basado en redondear la solución fraccionaria proporcionada por la relajación lineal del WPP. Este heurístico tiene cota del peor caso 2.

Grötschel y Win (1992) describen un algoritmo de planos de corte para el WPP basado fundamentalmente en la identificación de restricciones de corte impar violadas por la solución lineal. Este problema de separación es resoluble en tiempo polinómico (Padberg y Rao, 1982), lo que les permite resolver instancias del WPP de entre 52 y 264 vértices y de entre 78 y 489 aristas. La solución lineal así obtenida proporciona la solución óptima del WPP en 31 de las 36 instancias probadas. Corberán et al. (2012) realizan un estudio del poliedro del WPP y, basándose en el mismo, presentan un procedimiento de branch & cut capaz de resolver óptimamente instancias de gran tamaño (con hasta 3000 vértices y 9000 aristas). Está basado en un algoritmo de planos de corte que identifica restricciones violadas de corte impar, desigualdades zigzag y otros dos tipos de desigualdades.

2.2.2. El problema del Cartero Rural

Supongamos ahora que el cartero que mencionábamos antes debe repartir correo no en todas, sino sólo en algunas calles de la ciudad. Tenemos así el llamado problema del Cartero Rural. De nuevo presentaremos los resultados conocidos sobre este problema según si el grafo sobre el que está definido es no dirigido, dirigido, mixto o windy.

El problema del Cartero Rural en un grafo no dirigido (RPP)

Sea $G = (V, E)$ un grafo no dirigido, con costes $c_e \geq 0$ asociados a las aristas $e \in E$, y sea $E_R \subseteq E$ un subconjunto no vacío de aristas de G, a las que llamaremos *requeridas*. El problema del Cartero Rural consiste en encontrar un tour en G de coste total mínimo que recorra cada arista de E_R al menos una vez.

Este problema fue definido por primera vez por Orloff (1974). Lenstra y Rinnooy-Kan (1976) demostraron que el RPP es un problema NP-difícil. Orloff señaló que la complejidad del RPP parece aumentar con el número de componentes conexas del subgrafo inducido por las aristas requeridas. Este hecho coincide con la observación de Frederickson (1979) de que existe un algoritmo recursivo exacto para el RPP que es exponencial en el número de tales componentes.

Frederickson (1979) y Christofides et al. (1981) propusieron un algoritmo aproximado de resolución para el RPP basado en el cálculo de un árbol generador de mínimo coste que conecta las componentes conexas inducidas por las aristas requeridas y en la resolución, posteriormente, de un problema de acoplamiento mínimo sobre los vértices impares (respecto de las aristas requeridas y las aristas del árbol calculado anteriormente). Este algoritmo heurístico, similar al propuesto por Christofides

(1976) para el TSP, tiene una cota del peor caso de 1.5 (Benavent et al., 1985). Hertz, Laporte y Nanchen (1999) han desarrollado varios procedimientos de mejora de soluciones posibles generadas con algoritmos heurísticos constructivos. Estos procedimientos, que pueden extenderse fácilmente a grafos dirigidos y mixtos, producen buenos resultados computacionales.

Christofides et al. (1981) también describieron un algoritmo de branch & bound para la resolución exacta del RPP. Las cotas inferiores son calculadas a partir del problema del árbol generador de mínimo coste que surge como una relajación lagrangiana. La ramificación en el árbol de búsqueda se realiza sobre aristas que unen componentes conexas inducidas por las aristas requeridas. Con este algoritmo resuelven problemas de hasta 84 vértices, 180 aristas y 74 aristas requeridas.

Sanchis (1990), Corberán y Sanchis (1994, 1998) y Letchford (1997, 1999) estudiaron el poliedro del RPP y de su generalización, el problema General de Rutas (General Routing Problem, GRP), que será introducido más adelante. Basándose en el estudio poliédrico anterior, Corberán, Letchford y Sanchis (2001) diseñaron un algoritmo de planos de corte capaz de resolver las 26 instancias de Christofides et al. (1981) y Corberán y Sanchis (1994), así como las 92 instancias del RPP generadas por Hertz, Laporte y Nanchen (1999).

Otra aproximación fue propuesta por Ghiani y Laporte (2000). Estos autores formularon el RPP usando únicamente variables 0/1. De esta forma implementaron un algoritmo de Branch & Cut con el que consiguieron muy buenos resultados en instancias con hasta 350 vértices. Esta aproximación ha sido explotada después con éxito por Theis (2005) y por Reinelt y Theis (2005, 2008). Basándose en otra formulación diferente, Fernández et al. (2003) propusieron un algoritmo de Branch & Cut con el que obtuvieron buenos resultados computacionales. Sin embargo, hasta donde sabemos, los mejores resultados para el RPP han sido obtenidos por el Branch & Cut propuesto por Corberán, Plana y Sanchis (2007) para la resolución del RPP en

un grafo windy (del que el RPP es un caso particular).

El problema del Cartero Rural en un grafo dirigido (DRPP)

El problema del Cartero Rural Dirigido se define como sigue. Consideremos un grafo dirigido $G = (V, A)$, con un coste $c_a \geq 0$ asociado a cada arco $a \in A$, y sea $A_R \subseteq A$ un subconjunto no vacío de arcos de G que llamaremos *requeridos*. Buscamos un tour de coste mínimo en G que recorra al menos una vez cada arco de A_R.

Como en el caso no dirigido, el DRPP es NP-difícil. Basándose en la formulación del problema dada por Christofides et al. (1986), Savall (1990) define el poliedro de soluciones del DRPP y estudia algunas de sus propiedades. También, de forma independiente, ha sido estudiado por Gun (1993).

Campos y Savall (1995) realizan un estudio comparativo de tres heurísticos existentes para el DRPP: uno propuesto por ellos, otro por Ball y Magazine (1988) y un tercero por Christofides et al. (1986). La idea general de los tres heurísticos es similar. Básicamente, o se cambia el orden de los pasos efectuados o la forma de conectar las componentes conexas inducidas por los arcos requeridos. El estudio del peor caso del algoritmo heurístico de Christofides et al. (1986) es presentado en Benavent et al. (1983).

Respecto de la resolución exacta del DRPP, Christofides et al. (1986) describen un algoritmo de branch & bound cuya cota superior se obtiene con el heurístico antes mencionado; la cota inferior es obtenida relajando lagrangianamente las condiciones de simetría y calculando una arborescencia de mínimo coste.

Algunos de los resultados en Romero (1997) y Corberán, Romero y Sanchis (2003) para el RPP en un grafo mixto se pueden aplicar directamente al DRPP. También algunos de los resultados obtenidos por Plana (2005) y Corberán, Plana y Sanchis

(2008) para el RPP en un grafo windy son de aplicación aquí. Los resultados obtenidos con el Branch & Cut de Corberán, Plana y Sanchis (2008) son los mejores conocidos hasta el momento para el DRPP. Con dicho algoritmo, los autores fueron capaces de resolver instancias con 1000 vértices, 3100 arcos y 213 componentes conexas inducidas por los arcos requeridos en menos de 1 minuto de tiempo de computación (en promedio), la mayor parte de ellas en el nodo raíz del árbol.

El problema del Cartero Rural en un grafo mixto (MRPP)

Consideremos ahora un grafo mixto $G = (V, E, A)$, con costes c_e asociados a los arcos y aristas $e \in E \cup A$, y sea $E_R \subseteq E$ un subconjunto no vacío de aristas de G que llamaremos requeridas y $A_R \subseteq A$ un subconjunto no vacío de arcos de G que llamaremos requeridos. El problema del Cartero Rural en un grafo mixto consiste en encontrar un tour en G con coste total mínimo que recorra cada arco y arista de $E_R \cup A_R$ al menos una vez .

Una condición suficiente para que el MRPP tenga solución óptima finita es que el grafo sea fuertemente conexo y no existan aristas con coste negativo ni ciclos con coste total negativo. Nosotros supondremos que todos los costes asociados a enlaces del grafo son no negativos.

Nótese que, si $A = \emptyset$ ($E = \emptyset$), el MRPP se reduce al RPP (DRPP) y que si $E_R = E$ y $A_R = A$, obtenemos el MCPP (que a su vez generaliza al CPP y al DCPP). Es evidente, pues, que el MRPP es, en general, un problema NP-difícil.

Algunos trabajos sobre el MRPP son los de Romero (1997) y Corberán, Romero y Sanchis (2003). En ellos, dada una instancia del MRPP, ésta es transformada en una nueva en la que cada vértice $v \in V$ es incidente con, al menos, un arco o arista requerida y donde cada arista $e = (i, j)$ no requerida es reemplazada por dos arcos

no requeridos (i, j) y (j, i) y, así, $E = E_R$.

En su artículo sobre el poliedro del MGRP, del cual el poliedro del MRPP es un caso particular, Corberán, Romero y Sanchis (2003) presentan algunos resultados computacionales con un algoritmo de planos de corte sobre 100 instancias generadas aleatoriamente con $20 \leq |V| \leq 100$, $15 \leq |E| \leq 200$, $55 \leq |A| \leq 350$ y hasta 15 componentes R-conexas. Este algoritmo produce la solución óptima en 28 de ellas y, en promedio, la cota obtenida dista menos de 0.5 % del coste de una solución posible obtenida por los procedimientos heurísticos descritos en Corberán, Martí y Romero (2000).

En un trabajo posterior, Corberán, Mejía y Sanchis (2005) amplían el estudio del poliedro del MRPP y mejoran el algoritmo de planos de corte anterior. Finalmente, y como en los casos anteriores, los mejores resultados computacionales para el MRPP se obtienen con el Branch & Cut de Corberán, Plana y Sanchis (2007).

El problema del Cartero Rural con viento (WRPP)

Dado un grafo no dirigido $G = (V, E)$, con dos costes c_{ij}, c_{ji} asociados a cada arista $(i, j) \in E$ (representando el coste de recorrer dicha arista en cada sentido), y dado un subconjunto de aristas requeridas $E_R \subseteq E$, el problema del Cartero Rural con viento (Windy RPP, WRPP) consiste en hallar un tour en G que recorra al menos una vez todas las aristas de E_R (sin importar en qué sentido) que tenga coste total mínimo.

Una condición suficiente para que el WRPP tenga solución óptima finita es que el grafo sea conexo y no existan ciclos de coste total negativo. De nuevo, supondremos que los costes de las aristas son todos no negativos.

En el caso particular en que $E_R = E$, tenemos el WPP y, de la misma forma en

la que el WPP generaliza al CPP, DCPP y MCPP, el RPP, DRPP y MRPP son casos particulares del WRPP. Así pues, el WRPP es un problema NP-difícil que generaliza a la mayor parte de los problemas de rutas por arcos con un único vehículo.

Los primeros trabajos realizados sobre el WRPP son los de Benavent et al. (2007) y Benavent et al. (2005). En el primero de ellos, se propone una formulación entera del problema, así como una cota inferior (obtenida mediante un algoritmo de planos de corte) y varios algoritmos heurísticos constructivos. Los resultados computacionales aportados demuestran que tanto las cotas inferiores como las superiores son buenas en instancias de tamaño medio. El segundo trabajo propone distintos algoritmos heurísticos para la resolución aproximada del WRPP. Entre ellos, dos heurísticos constructivos que son modificaciones de los algoritmos propuestos por Benavent et al. (2003). La aleatorización de algunos de los algoritmos constructivos iniciales y de sus modificaciones dan lugar a algoritmos metaheurísticos de tipo Multiarranque (Multi-Start) que, como se demuestra, proporcionan muy buenas soluciones posibles en tiempos razonables de computación. Finalmente, en Benavent et al. (2005) se propone un sofisticado algoritmo de Búsqueda Dispersa (Scatter Search) que proporciona excelentes soluciones, aunque con tiempos mayores de computación.

El estudio del poliedro del WRPP ha sido realizado por Corberán, Plana and Sanchis (2007) en el contexto del problema general de rutas en un grafo windy. En ese artículo se demuestra que todas las desigualdades válidas propuestas en Benavent et al. (1983) definen facetas del poliedro del WRPP. También, nuevas rutinas de separación han llevado al diseño e implementación del ya mencionado algoritmo de Branch & Cut de Corberán, Plana and Sanchis (2007), capaz de resolver instancias del WRPP de gran tamaño. Por ejemplo, 54 instancias generadas aleatoriamente con hasta 1000 vértices, 4000 aristas y 150 componentes R-conexas fueron resueltas óptimamente, así como 65 de 72 instancias asociadas a grafos generados intentando simular redes viarias con hasta 1000 vértices, 3070 aristas y 202 componentes R-

conexas.

2.3. El problema general de rutas

Es el caso más general de entre los problemas de rutas con un único vehículo: aquél en el que la demanda de servicio se puede encontrar tanto en los arcos y aristas como en los vértices del grafo.

2.3.1. El problema general de rutas en un grafo no dirigido (GRP)

El problema general de rutas en un grafo no dirigido se define como sigue:

Sea un grafo no dirigido $G = (V, E)$, con costes $c_e \geq 0$ asociados a las aristas $e \in E$, y consideremos un subconjunto de aristas, que llamaremos requeridas, $E_R \subseteq E$ y un subconjunto de vértices, que llamaremos requeridos, $V_R \subseteq V$. El objetivo es ncontrar un tour en G de coste mínimo que recorra cada arista requerida y pase por cada vértice requerido al menos una vez.

Este problema fue introducido por Orloff (1974) y es NP-difícil (Lenstra y Rinnooy-Kan, 1976). Incluye como casos particulares la mayoría de los problemas sobre grafos no dirigidos presentados anteriormente. En concreto, si $E_R = E$ tenemos el CPP, si $V_R = \emptyset$ resulta el RPP y si $E_R = \emptyset$ y $V_R = V$ se reduce al problema del Viajante Gráfico (GTSP).

La formulación como PLE del GRP es exactamente la misma que para el RPP. Así pues, todas las desigualdades válidas para el RPP lo son para el GRP y viceversa.

El problema y su poliedro han sido extensamente estudiados en Corberán y Sanchis (1994, 1998) y Letchford (1997, 1999). Corberán, Letchford y Sanchis (2001) han desarrollado un algoritmo de planos de corte que ha producido excelentes resultados computacionales en instancias del RPP, GTSP y GRP. El trabajo sobre el RPP de Ghiani y Laporte (2000), mencionado en la sección 2.2.2 puede considerarse también un estudio sobre el GRP y su resolución. Así mismo los trabajos de Theis (2005) y, Reinelt y Theis (2005, 2008), mencionados en la sección del RPP no dirigido, muestran un estudio profundo del poliedro del GRP que podría utilizarse en el diseño de algoritmos eficientes para la resolución óptima del GRP.

2.3.2. El problema general de rutas en un grafo mixto (MGRP)

Sea ahora un grafo mixto $G = (V, E, A)$ con costes no negativos asociados a sus arcos y aristas. Consideremos un subconjunto de arcos $A_R \subseteq A$ y un subconjunto de aristas $E_R \subseteq E$, que llamaremos requeridos, y un subconjunto $V_R \subseteq V$ de vértices requeridos. El MGRP es el problema de encontrar un tour en G que pase por cada arco, arista y vértice requerido al menos una vez y tenga coste total mínimo.

Éste problema es NP-difícil ya que contiene a todos los problemas anteriores (excepto al WPP) como caso particular.

La descripción del poliedro del MGRP hecha en los trabajos de Corberán, Romero y Sanchis (2003) y Corberán, Mejía y Sanchis (2005) es bastante completa, presentan además un algoritmo de planos de corte que proporciona buenos resultados computacionales. Como el RPP, las versiones dirigida y mixta del GRP pueden transformarse fácilmente en problemas equivalentes de rutas por vértices. Blais and Laporte (2003) usan esas transformaciones para resolver el GRP en un grafo dirigido (DGRP) y el MGRP como un problema del Viajante Asímétrico (ATSP) y un TSP Generalizado Asímétrico, respectivamente. Esta aproximación funciona mejor

en el caso del DGRP. El Branch & Cut de Corberán, Plana y Sanchis (2007) es, hasta el momento, el mejor método exacto para el MGRP: 14 de 18 instancias con 1000 vértices y, en promedio, 1240 aristas y 1210 arcos, de los que 1200 son aristas y arcos requeridos definiendo 170 componentes R-conexas, y han sido resueltas óptimamente.

2.3.3. El problema general de rutas con Viento (WGRP)

Consideremos un grafo no dirigido $G = (V, E)$ con dos costes no negativos c_{ij}, c_{ji} asociados a cada arista $(i, j) \in E$, representando el coste de recorrer dicha arista en cada sentido. Dado un subconjunto de aristas, $E_R \subseteq E$, que llamaremos requeridas y un subconjunto $V_R \subseteq V$ de vértices requeridos, el WGRP consiste en encontrar un tour en G que pase por cada arista y vértice requerido al menos una vez y que tenga coste total mínimo.

Este problema es obviamente NP-difícil porque tiene como caso particular al WRPP (cuando $V_R = \emptyset$). Nótese que si, además, todas las aristas son requeridas ($E = E_R$) obtenemos el WPP. Por otra parte si $c_{ij} = c_{ji}\ \forall (i, j) \in E$, obtenemos el GRP, por lo que muchos de los problemas anteriores se pueden ver como un caso particular de éste.

Una buena descripción del poliedro del WGRP es la hecha por Plana (2005) y Corberán, Plana, Sanchis (2006). En esos trabajos, se presentan unas nuevas desigualdades, las desigualdades ZigZag, que son también útiles en otros problemas de rutas por arcos como el WPP. Por otra parte, en Corberán, Plana, Sanchis (2007) se presenta un algoritmo de Branch & Cut que, como se ha comentado, es, hasta la fecha, el mejor procedimiento exacto para la resolución de los problemas descritos en estas secciones. En lo que respecta a su comportamiento en instancias del WGRP, el algoritmo resuelve 46 de 51 instances con 500 vertices y hasta 1550 aristas y 280

componentes R-conexas.

Capítulo 3

El problema de la Grúa

En este capítulo, presentamos con detalle el Problema de la Grúa, o Stacker Crane Problem (SCP). Inicialmente, expondremos las aplicaciones tanto teóricas como prácticas del SCP. A continuación propondremos un heurístico basado en una búsqueda local iterada y finalizaremos presentando una formulación lineal entera.

3.1. Descripción del problema

Dado un grafo mixto, el Problema de la Grúa o Stacker Crane Problem consiste en encontrar un tour de coste mínimo que recorra al menos una vez todos sus arcos, comenzando y finalizando en un vértice llamado depósito.

El SCP fue originalmente propuesto por Frederickson, Hecht y Kim (1978), quienes lo definieron como un problema de rutas por arcos. Frederickson, Hecht y Kim (1978) demuestran la complejidad del problema probando que cualquier instancia del problema del Viajante (TSP), que es NP-difícil, puede transformarse en otra del

SCP. Proponen, además, un heurístico con cota del peor caso de 9/5, cota que no se ha mejorado por el momento, y complejidad algorítmica de $O(n^3)$, donde n es el número de arcos a recorrer. El heurístico propuesto escoge la mejor solución dada por dos heurísticos, "LARGEARCS", que funciona mejor cuando el coste de los arcos es menor que el de las aristas usadas en la solución óptima, y "LARGEEDGE", que funciona mejor cuando el coste de los arcos es parecido al de las aristas. El primero se basa en resolver un problema de acoplamiento de mínimo coste, seguido del cálculo de un árbol de mínimo coste y el segundo algoritmo trata de realizar una transformación del SCP en un TSP con el objetivo de poder aplicarle el conocido algoritmo de Christofides (1976) para el TSP. Tanto "LARGEARCS" como "LARGEEDGE" necesitan que la instancia del SCP satisfaga la desigualdad triangular.

Dentro de la literatura sobre el problema podemos encontrar artículos como el de Zhang (1992), quien propone una simplificación de los heurísticos propuestos por Frederickson, Hecht and Kim (1978), reduciendo así la complejidad algorítmica de $O(n^3)$ a $O(n^2)$, aunque con una cota del peor caso de 2. Poco tiempo después, Zhang y Zheng (1995) publican otro artículo donde se muestra una posible transformación del SCP en un ATSP. Ellos proponen esta transformación con el objetivo de poder aplicarle algoritmos genéticos previamente propuestos para el ATSP. Hassin y Khuller (2001) proponen una $\frac{1}{2}$ z-aproximación para el TSP que se puede aplicar al SCP. Berbeglia et al. (2007) mencionan el SCP, definiéndolo como un problema de recogida y entrega. Recientemente, Srour y van de Velde (2011) realizan un estudio estadístico para comparar la dificultad del SCP con la del ATSP. En este último artículo se pretende demostrar que, aunque en la literatura el SCP ha sido abordado casi siempre como un problema reducible al ATSP y fácil de resolver (tal y como comenta Laporte, 1997), pueden haber instancias del SCP tan difíciles de resolver como otras del ATSP. Otros artículos donde se menciona al SCP son el survey de Eiselt, Gendreau y Laporte (1995a) y el artículo de Cirasella et al. (2007), donde se realiza una comparación de heurísticos para el ATSP sobre once clases de instancias aleatorias, una de las cuales corresponde a instancias del SCP.

3.1.1. Definición del problema

Sea $G = (V, E, A)$ un grafo mixto, donde V es un conjunto de vértices, E es un conjunto de aristas y A es un conjunto de arcos. Consideremos un coste no negativo asociado al recorrido de cada arco y arista del grafo y un vértice inicial o *depósito*. El objetivo del SCP es encontrar un tour de coste mínimo que comience y acabe en el depósito y que pase por todos los arcos de A al menos una vez.

Supondremos que el grafo G cumple las siguientes propiedades:

- Todo vértice es el final o el origen de al menos un arco, es decir, $V = V(A)$.
- El grafo $G(V, E)$, que contiene únicamente las aristas del grafo, es conexo.
- Dado cualquier arco $a_i = (t_i, h_i) \in A$, existe un camino en $G(V, E)$ desde su vértice inicial t_i al final h_i, con coste menor o igual al del arco.

Si el grafo no cumpliera que $V = V(A)$, siempre se podría transformar en otro que sí lo hiciera. Nótese que esta condición implica que cualquier tour que recorra todos los arcos del grafo, visitará el depósito necesariamente. Exigir que el grafo $G(V, E)$ sea conexo es una condición necesaria para asegurar que siempre existe un tour que pasa por todos los arcos al menos una vez. La última propiedad exigida asegura que siempre existe una solución óptima en la que se usará cada arco una única vez.

3.1.2. Aplicaciones

En la actualidad, los sistemas de almacenamiento y manejo de materiales se estima que pueden absorber hasta el 20 % del coste de distribución física de una empresa. Por ello es necesario una buena optimización del espacio utilizado para almacenar los productos, así como el tiempo y coste en el acceso y guardado de los mismos. Muchas

empresas que han de almacenar un surtido muy variado de productos, como pueden ser empresas de alimentación, suelen utilizar grúas para transportar los palets o cajas de un lugar a otro (Figuras 3.1).

Ésta es una de las aplicaciones más comunes del SCP, ya que nos permite resolver el problema de organizar de forma óptima el movimiento de una grúa que debe recoger los palets o cajas de unas posiciones determinadas y colocarlos en otras posiciones también especificadas. Los movimientos que debe realizar la grúa para transportar los objetos se representan mediante arcos en el grafo, mientras que las aristas del mismo corresponden a movimientos de la grúa sin carga para ir a recoger otro objeto.

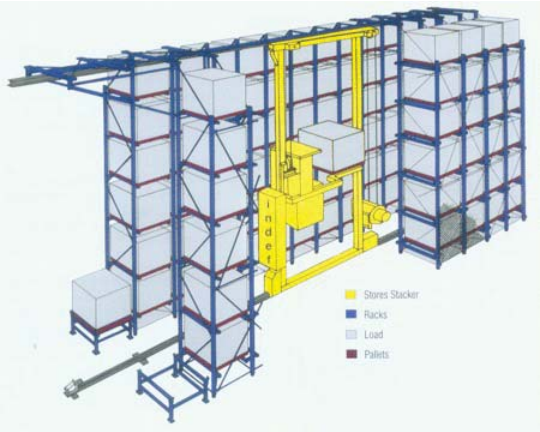

Figura 3.1: *Grúa*

Berbeglia et al. (2007) presentan otras posibles aplicaciones del SCP, proponiendo éste como un problema estático de recogidas y entregas. Esta importante clase de problemas de rutas modeliza la situación en la que unos objetos o personas han de ser transportados desde unos orígenes a unos destinos, y surgen en el reparto de correo, servicios ambulatorios, robótica, etc. Berbeglia et al. (2007) definen el SCP como un problema con las siguientes características: cada producto a transportar tiene un origen y un destino determinados, existe un único vehículo con una unidad de capacidad y cada vértice del grafo es un nodo de entrega o recogida.

En el sector del transporte marítimo y logística, el SCP tiene aplicación en el transporte de mercancías a corta distancia, a menudo como parte de un movimiento global más largo. Este tipo de problemas son los que en inglés se conocen como "Drayage", y se basan en trasportar contenedores desde puntos de entrada e interiores de las terminales marítimas (Figura 3.2).

Figura 3.2: *Movimiento de mercancías en terminales marítimas.*

3.2. Un metaheurístico para el SCP

En esta sección se presenta un metaheurístico para el SCP cuyas componentes son un algoritmo multi-arranque (Multi-Start, MS), una búsqueda por entornos variables descendente (Variable Neighborhood Descent, VND), y una búsqueda local iterada (Iterated Local Search, ILS). Este algoritmo está basado en los algoritmos propuestos por Belenguer et al. (2010) para el Split Delivery Capacitated Arc Routing Problem y por Benavent, Corberán y Sanchis (2010) para el Min-Max k-Vehicles Windy Rural Postman Problem.

Su estructura básica es la siguiente: el MS comienza generando una solución factible para el SCP, la cual es mejorada por el VND. Si la solución mejorada es diferente de la solución previa, será considerada como la solución inicial para el algoritmo ILS.

Este último algoritmo perturba la solución, destruyéndola para luego reconstruirla de manera que obtengamos una solución diferente de la que partíamos, la cual volverá a ser mejorada por el VND. La mejor solución encontrada es usada como solución de partida en la siguiente iteración del ILS. Finalmente, la salida del algoritmo es la mejor solución encontrada.

A continuación describimos los diferentes elementos que componen el algoritmo metaheurístico.

3.2.1. Algoritmo multi-arranque

El algoritmo multi-arranque consiste en la ejecución de un número fijo de iteraciones hasta que se cumple un criterio de parada. En cada iteración, se genera una solución diferente con un algoritmo constructivo aleatorizado, que luego es mejorada a través de un método de búsqueda local. La salida del algoritmo es la mejor solución encontrada.

En nuestra implementación del algoritmo multi-arranque, en la fase de búsqueda local utilizamos un algoritmo de búsqueda por entornos variables descendente seguido de un algoritmo ILS.

3.2.2. Algoritmo constructivo

Cada iteración del algoritmo multiarranque comienza con la generación de una solución diferente mediante un algoritmo constructivo aleatorizado, que luego es mejorada a través de un método de búsqueda local.

Para aplicar el algoritmo constructivo, dados dos arcos a_i y a_j, definimos la distancia

de a_i a a_j, $d(a_i, a_j)$, como el coste del camino más corto entre el vértice final de a_i, h_i, y el vértice inicial de a_j, t_j. Obsérvese que podemos asegurar que existe una solución óptima que para ir desde el final de un arco cualquiera al inicio de otro, pasará por el camino más corto entre dichos vértices usando únicamente aristas. Por tanto, para obtener una solución del SCP es suficiente con hallar la secuencia en la que se recorren los arcos, ya que asumiremos que de un arco al siguiente vamos siempre por el camino más corto usando aristas. El algoritmo constructivo genera una solución

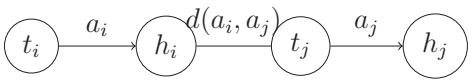

Figura 3.3: *Distancia entre dos arcos.*

añadiendo iterativamente arcos de A hasta completar un tour de la siguiente manera: el primer arco a insertar se elige de manera aleatoria. En los subsiguientes pasos, sea a_j el último elemento añadido a nuestro tour parcial. De entre los arcos de A aún no insertados, preseleccionamos un conjunto $S(a_j)$ con los *nclose* arcos más cercanos a a_j según la distancia d. De este conjunto de arcos candidatos, elegimos uno aleatoriamente y lo insertamos al final del tour parcial. El algoritmo finaliza cuando todos los elementos de A han sido añadidos al tour. Después de algunas pruebas computacionales, hemos fijado el número de arcos preseleccionados en cada paso a $nclose = 5$.

Los elementos aleatorios del algoritmo nos permiten la construcción de diferentes soluciones posibles que serán usadas como puntos de partida para el algoritmo multi-arranque.

Algoritmo 1 Constructivo

Entrada: Grafo dirigido $G = (V, A)$.

Salida: Un tour T para el SCP.

1: Elegimos un elemento $a_0 \in A$ de manera aleatoria.
2: $T := \{a_0\}$ y $A := A \setminus \{a_0\}$.
3: **mientras** $A \neq \emptyset$ **hacer**
4: Calcular $S(a_0)$ con los *nclose* arcos de A más cercanos a a_0.
5: Elegir un elemento $a_j \in S(a_0)$ de manera aleatoria.
6: $T := T \cup \{a_j\}$ y $A := A \setminus \{a_j\}$.
7: $a_0 := a_j$.
8: **fin mientras**
9: Completamos T con los caminos más cortos entre los arcos.
10: **devolver** T

3.2.3. Búsqueda por entornos variables descendientes

La búsqueda por entornos variables descendente (VND), introducida por Maldenovic y Hansen (1997), es una búsqueda local mejorada que se basa en una secuencia $(N_1, ..., N_K)$ de K entornos con cardinal creciente. Comenzando con $k = 1$, cada iteración en el VND busca en el entorno N_k. Si una mejora es detectada, ésta se ejecuta y k es reiniciado a 1. En otro caso, k aumenta en una unidad. El algoritmo finaliza cuando k alcanza K sin realizar ninguna mejora.

En el algoritmo presentado se trabaja con $K = 3$. Estos tres entornos están asociados con los procedimientos que se describen a continuación.

Sea T una secuencia de arcos correspondientes a una solución posible. Los movimientos definidos son los siguientes:

1-opt: Para cada arco $a \in T$, se intenta insertar a en una posición diferente de T.

En la primera posición tal que el coste de la nueva secuencia sea mejor, se realiza el cambio. Este procedimiento se aplica hasta que, probado el cambio de posición de todos los arcos, no se encuentra ninguna mejora.

2-opt: Dada una pareja de arcos consecutivos en T, se prueba a insertarla en diferente posición, y se realiza el cambio si el coste es inferior. Al igual que antes, el procedimiento se aplica hasta que no se encuentra ninguna mejora.

3-opt: El procedimiento es el mismo que el anterior pero ahora con grupos de tres arcos consecutivos de T.

3.2.4. Búsqueda local iterada

La búsqueda local iterada (ILS), introducida por Lourenço et al. (2002), es un metaheurístico que parte de una solución inicial y tiene como principales componentes una búsqueda local y una perturbación. Su estructura es la siguiente. Dada una solución inicial, ésta es mejorada mediante la búsqueda local. A continuación, el algoritmo realiza $niterILS$ iteraciones. En cada iteración, se modifica aleatoriamente una copia de la mejor solución encontrada hasta el momento usando un algoritmo de perturbación. La solución resultante se mejora también mediante la búsqueda local y se actualiza la mejor solución, si corresponde. La búsqueda local usada es el mismo VND descrito en la sección anterior. El procedimiento de perturbación es el siguiente.

Dada una solución T del SCP en forma de secuencia de arcos, *ndel* arcos son eliminados de manera aleatoria de T. A continuación, cada uno de estos arcos es reinsertado en T en la posición que resulte en un menor incremento del coste, obteniendo así una solución que puede ser diferente a la que teníamos originalmente.

Algoritmo 2 Perturbación

Entrada: Una solución en forma de secuencia de arcos $T = \{a_{i_1}, ..., a_{i_n}\}$.

Salida: Una solución T' diferente.

1: Sea $S \subset T$ un succonjunto de $ndel$ arcos aleatorios.
2: $T' := T \setminus S$.
3: **mientras** $S \neq \emptyset$ **hacer**
4: Elige $a_k \in S$ de manera aleatoria.
5: Elige $a_{i_j} = \underset{a_{i_j} \in T'}{argmin}\{d(a_{i_j}, a_k) + d(a_k, a_{i_{j+1}})\}$, tal que a_k no estuviera entre a_{i_j} y $a_{i_{j+1}}$ en T.
6: Insertamos a_k en T' entre a_{i_j} y $a_{i_{j+1}}$. $S := S \setminus \{a_k\}$.
7: **fin mientras**
8: **devolver** T'

Algoritmo 3 Metaheurístico

Entrada: $G = (V, E, A)$ un grafo mixto donde A es el conjunto de arcos requeridos.

Salida: Un tour T para el SCP.

1: **Para** $iterMS = 1$ **hasta** $nsol$ **hacer**
2: $SCPsol :=$ **Constructivo**(G)
3: $BestSol :=$ **VND**$(SCPsol)$
4: **Para** $iterILS = 1$ **hasta** $niterILS$ **hacer**
5: $ILSsol :=$ **Perturbación**$(BestSol)$
6: $ILSol :=$ **VND**$(ILSsol)$
7: **Actualizamos** $BestSol$
8: **fin Para**
9: **Actualizamos** $GlobalBestSol$
10: **fin Para**
11: **devolver** $GlobalBestSol$

3.3. Resultados computacionales

En esta sección presentamos los resultados computacionales obtenidos con el algoritmo metaheurístico propuesto sobre diferentes conjuntos de instancias.

Para probar la eficiencia del metaheurístico propuesto para el SCP se han utilizado 92 instancias generadas a partir de instancias del Problema del Cartero Rural (Rural Postman Problem, RPP) propuestas por Hertz, Laporte y Nanchen (1999).

En las instancias originales del RPP, los vértices son puntos del plano euclídeo y los costes de las aristas son proporcionales a las distancias euclídeas entre los vértices. Las instancias del RPP están agrupadas en tres tipos, en función de las características del grafo correspondiente:

- Conjunto **Hertzd:** Se compone de 36 instancias en las que el conjunto de aristas se ha seleccionado de manera que todos los vértices tengan grado 4.
- Conjunto **Hertzg:** Se compone de 36 instancias, cuyo grafo tiene estructura de rejilla o *grid* y las aristas tiene coste unitario.
- Conjunto **Hertzr:** Se compone de 20 instancias, en las que el conjunto de aristas está generado aleatoriamente y su coste es la distancia más corta.

Para la generación de instancias del SCP a partir de las del RPP, simplemente se ha asignado una dirección aleatoria a las aristas requeridas para obtener así los arcos requeridos. Finalmente, los grafos han sido simplificados para que todos los vértices sean incidentes con al menos un arco.

Las instancias del SCP generadas corresponden a grafos con $5 \leqslant |V| \leqslant 100$, $3 \leqslant |A| \leqslant 116$, $5 \leqslant |E| \leqslant 423$. El algoritmo ha sido programado en $C++$ y ejecutado en un procesador Intel Core 2 Duo a 2,4 GHZ.

Para todos los tests se ha fijado **nclose**= 5, y se han hecho varias pruebas para diferentes valores de los parámetros **nsol** y **niterILS**. Los valores que se han utilizado son **nsol** $= 100, 50, 25, 10$ y **niterILS** $= 0, 15, 25, 70$. Los resultados para cada conjunto de instancias se muestran en la Tabla *3.1*. En esta tabla se muestra el número de instancias para las que el algoritmo ha alcanzado la mejor solución conocida (**nº Mejor sol**), la desviación media respecto a la mejor solución encontrada (**Gap(MS)**), la desviación respecto a la cota inferior (**Gap(LB)**), la cual hemos obtenido con un algoritmo preliminar de planos de corte basado en la formulación propuesta en la sección siguiente, y el tiempo medio empleado en segundos (**CPU**). Para que las medias puedan ser más representativas se han agrupado las instancias en tres grupos según el número de arcos.

Se puede comprobar que los mejores resultados se obtienen cuando el número de soluciones del multi-arranque *nsol* es 25 o 50 y el número de veces que se ejecuta el ILS, $niterILS$, es 15. Sin embargo el tiempo de ejecución del algoritmo es mayor que en los demás casos. El algoritmo es más rápido cuando ejecutamos únicamente el multi-arranque es decir cuando $niterILS = 0$, pero excepto en el caso de Hertzd, las soluciones son con diferencia peores que para los demás parámetros. Finalmente, en el caso para el que se generan 10 soluciones y se aplica 70 veces el ILS, podemos comprobar que no existe una mejora importante ni en el tiempo de ejecución ni tampoco en la resolución de las instancias, con lo que podemos concluir que es mejor la generación de posibles soluciones que estar aplicando el ILS un número elevado de veces. Concluímos, por tanto, que los mejores resultados se han obtenido cuando se ha tenido un equilibrio entre los dos parámetros. En el caso de las instancias Hertzr, cualquier valor de los parámetros produce buenas soluciones. Esto es debido a que el número de arcos de estas instancias es pequeño.

Además de las instancias anteriores y con idea de comparar con métodos previamente publicados, hemos probado el algoritmo con dos conjuntos de instancias presentados en Srour and van de Velde (2011). El primer conjunto de instancias, llamadas PK,

		Hertd			Hertzg			Hertzr		
		$\|A\| < 20$	$20 < \|A\| < 40$	$\|A\| > 40$	$\|A\| < 20$	$20 < \|A\| < 50$	$\|A\| > 50$	$\|A\| < 10$	$\|A\| > 10$
100-0	nºMejor sol	13	7	7	12	9	2	9	10
	Gap(MS)	0,00	0,01	0,44	0,00	0,00	2,39	0,00	0,00
	Gap(LB)	8,73	2,48	4,92	2,19	1,29	5,36	7,92	14,26
	CPU	0,26	2,94	76,32	0,08	1,55	37,38	0,04	0,47
50-15	nºMejor sol	13	8	10	12	9	10	9	10
	Gap(MS)	0,00	0,00	0,25	0,00	0,00	0,37	0,00	0,00
	Gap(LB)	8,73	2,47	4,72	2,19	1,29	3,29	7,92	14,26
	CPU	0,78	8,68	110,61	0,32	5,80	71,75	0,06	1,54
25-25	nºMejor sol	13	8	6	12	9	12	9	10
	Gap(Best)	0,00	0,00	0,63	0,00	0,00	0,28	0,00	0,00
	Gap(LB)	8,73	2,47	5,12	2,19	1,29	3,18	7,92	14,26
	CPU	0,56	6,53	78,24	0,23	4,46	53,00	0,04	1,25
10–70	nºMejor sol	13	8	1	12	9	7	9	10
	Gap(Best)	0,00	0,00	0,90	0,00	0,00	1,12	0,00	0,00
	Gap(LB)	8,73	2,47	5,39	2,19	1,29	4,05	7,92	14,26
	CPU	0,59	6,74	73,27	0,26	4,75	52,76	0,05	1,42

Tabla 3.1: *Resultados en las instancias del SCP*

provienen de problemas reales de "drayage" del puerto de Rotterdam, Holanda. Estas instancias están transformadas en instancias para el ATSP y resueltas de manera óptima mediante algoritmos exactos específicos para dicho problema. Aunque todas estas instancias tienen 131 arcos, no conocemos sus características exactas, por lo que hemos usado directamente las instancias trasformadas. El segundo conjunto de instancias, llamadas Crane, fueron usadas en Cirasella et al. (2007) y Srour and van de Velde (2011). Las instancias Crane fueron generadas de manera aleatoria usando un único parámetro $u \geq 1$ para construir cada par origen-destino en la forma siguiente: el origen es elegido de manera aleatoria en el cuadrado $10^6 \times 10^6$; a continuación dos enteros x e y son elegidos aleatoriamente en el intervalo $[-10^6/u, 10^6/u]$ y el destino es determinado añadiendo el vector (x, y) al origen. Diez de estas instancias tienen $|A| = 100$ y otras diez tienen $|A| = 316$. Estas últimas han sido generadas con el valor $u = 10$.

En las Tablas 3.2 y 3.3 se muestran los resultados obtenidos con nuestro metaheurístico sobre las instancias PK y Crane, respectivamente, usando la combinación de parámetros $\{50, 15\}$ y un tiempo limite de 900 segundos. La columna *Heurístico* muestra el coste de la solución dada por nuestro algoritmo, mientras que *Óptimo* da el coste óptimo. El tiempo de computación, en segundos, del algoritmo se muestra en *Tiempo* y la columna *Gap* corresponde al porcentaje de desviación de la solución obtenida con respecto a la solución óptima. Podemos observar que el comportamiento de nuestro algoritmo es bueno sobre las instancias PK, encontrando un total de 33 soluciones óptimas. El Gap medio para estas instancias es de 1.24 %, con un tiempo medio de 240.2 segundos. Por otro lado, las instancias Crane resultan considerablemente más difíciles que las PK. Sin embargo, existe una clara diferencia entre las instancias con 100 arcos y las de 316 en lo que se refiere al comportamiento de nuestro algoritmo. Mientras que en el primer caso el Gap es del orden del 4.19 %, en el segundo caso el gap obtenido en el tiempo límite fijado es del 12.93 %, lo que significa que el algoritmo necesitaría bastante más tiempo de computación para encontrar mejores soluciones en las instancias grandes.

Nombre	Heurístico	Tiempo	Óptimo	Gap	Nombre	Heurístico	Tiempo	Óptimo	Gap
PK1	65005	239.30	63484	2.40	PK18	72336	234.76	69367	4.28
PK2	64786	229.13	63843	1.48	PK19	48981	264.48	48981	0.00
PK3	52015	233.78	50139	3.74	PK20	51145	245.79	51145	0.00
PK4	51712	236.84	51712	0.00	PK21	51477	243.64	50474	1.99
PK5	51092	217.43	50967	0.25	PK22	70582	245.93	68288	3.36
PK6	60349	237.90	57822	4.37	PK23	50282	266.71	49345	1.90
PK7	63222	256.30	62497	1.16	PK24	68105	248.22	67556	0,81
PK8	48602	214.23	48601	0.00	PK25	56265	259.36	56153	0.20
PK9	60329	229.47	59742	0.98	PK26	52480	237.82	52062	0.80
PK10	58983	249.42	58046	1.61	PK27	68756	239.32	68260	0.73
PK11	63007	227.57	62010	1.61	PK28	80588	251.09	80499	0.11
PK12	67247	232.86	67247	0.00	PK29	54009	244.56	52175	3.52
PK13	51027	230.19	51027	0.00	PK30	48807	215.40	48807	0.00
PK14	67180	250.64	66937	0.36	PK31	38164	263.73	37893	0.72
PK15	52720	253.11	51943	1.50	PK32	72517	244.03	72517	0.00
PK16	52798	222.19	51729	2.07	PK33	47331	223.82	47331	0.00
PK17	64580	237.43	64051	0.83					

Tabla 3.2: *Resultados computacionales instancias PK*

Nombre	Heurístico	Tiempo	Óptimo	Gap	Nombre	Heurístico	Tiempo	Óptimo	Gap
*crane*0	8110760	234.39	7777997	4.28	*crane*10	15716000	900	13132907	19.67
*crane*1	7893750	242.15	7615069	3.66	*crane*11	14931500	900	13194492	13.16
*crane*2	8288450	237.46	8062054	2.81	*crane*12	14714300	900	12968098	13.47
*crane*3	7402210	242.53	7018782	5.46	*crane*13	14520000	900	12350138	17.57
*crane*4	8074990	242.47	7786309	3.71	*crane*14	14575400	900	12744226	14.37
*crane*5	8233300	235.59	7933180	3.78	*crane*15	15606200	900	13120938	18.94
*crane*6	7718400	241.78	7208062	7.08	*crane*16	15464800	900	12900450	19.88
*crane*7	7658470	239.29	7474720	2.46	*crane*17	15373200	900	13145193	16.95
*crane*8	8021430	232.68	7676198	4.50	*crane*18	15063100	900	13289851	13.34
*crane*9	7974160	237.83	7561872	5.45	*crane*19	15372400	900	13164800	16.77

Tabla 3.3: *Resultados computacionales instancias Crane*

3.4. Formulación

En esta sección presentamos una formulación lineal entera para el SCP.

Llamaremos **tour** para el SCP a cualquier camino cerrado que, partiendo y volviendo al depósito, pase exactamente una vez por cada arco de A, y **semitour** a cualquier subconjunto de aristas de E tales que, al añadirle los arcos de A, puedan ser orientadas de manera que obtengamos un tour.

Dado un vértice $i \in V$, definimos $d^+(i)$ y $d^-(i)$ como el número de arcos entrantes en i y salientes de i, respectivamente. Con cada arista $e = (i, j)$, asociamos dos variables, x_{ij} y x_{ji}, representando el número de veces que e es atravesada de i a j y de j a i, respectivamente. Dado un subconjunto de vértices $S \subset V$, definimos $x^+(S) = \sum_{i \in S, j \in V \setminus S} x_{ij}$ y $x^-(S) = \sum_{i \in S, j \in V \setminus S} x_{ji}$.

Notar que un tour para el SCP define un grafo Euleriano. Ahora bien, como dado cualquier arco $a_j = (t_j, h_j)$ existe siempre un camino de igual o menor coste desde t_j hasta h_j usando únicamente aristas, podemos asegurar que existe una solución óptima atravesando únicamente una vez cada arco. Esta propiedad nos permite trabajar con una formulación para semitours sin tener que asociar variables a los arcos, ya que podemos fijar esas variables a uno.

Denotamos por $V_1, V_2, \ldots, V_p$ a los conjuntos de vértices que forman las p componentes débilmente conexas inducidas por los arcos requeridos. Les llamaremos R-sets y cumplen que $V_1 \cup \cdots \cup V_p = V_R$.

El SCP se puede formular como sigue:

$$\text{Minimizar} \qquad \sum_{(i,j) \in E} c_{ij}(x_{ij} + x_{ji})$$

s.a:

$$x^+(S) \geq 1 \quad \forall S = \bigcup_{i \in Q} V_i, \quad Q \subset \{1, 2, ..., p\} \tag{3.1}$$

$$x^+(i) + d^+(i) = x^-(i) + d^-(i) \quad \forall i \in V \tag{3.2}$$

$$x_{ij}, x_{ji} \geq 0 \quad \forall (i,j) \in E \tag{3.3}$$

$$x_{ij}, x_{ji} \quad enteras \quad \forall (i,j) \in E \tag{3.4}$$

Dado que los R-sets son subgrafos débilmente conexos, las restricciones (3.1) garantizan la conectividad entre ellos y fuerzan a que la solución sea conexa. Las restricciones (3.1) son las restricciones de simetría sobre los vértices y, junto con (3.1), garantizan que el grafo asociado con una solución posible sea fuertemente conexo. Los vectores $x \in \mathbb{R}^{2|E|}$ satisfaciendo (3.1) a (3.4) forman el conjunto de semitours del SCP.

Capítulo 4

El problema general de rutas en un grafo dirigido

4.1. Descripción del problema

En los problemas de rutas por arcos la demanda del servicio está situada en los arcos del grafo. Sin embargo, hay situaciones reales que tienen que ser formuladas considerando que la demanda de servicio está, además, en los vértices del grafo. El Problema General de Rutas en un grafo dirigido (Directed General Routing Problem, DGRP) contempla este caso. En el DGRP, la demanda de servicio puede encontrarse en algunos arcos y en algunos vértices, que llamamos requeridos, de un grafo dirigido. El objetivo es encontrar una ruta de longitud total mínima que pase, al menos una vez, por todos los arcos y todos los vértices requeridos. Blais y Laporte (2003) transforman el DGRP en un problema del viajante asimétrico (Asymmetric Traveling Salesman Problem, ATSP) equivalente y lo resuelven usando un algoritmo exacto propuesto por Carpaneto, dell'Amico, and Toth (1995), consiguiendo resolver óptimamante instancias de gran tamaño. Aunque no hay casi trabajos que traten

directamente al DGRP como un problema de rutas por arcos, sí hay muchos artículos dedicados al estudio y resolución del Problema General de Rutas en un grafo mixto (Mixed General Routing Problem, MGRP), el cual puede considerarse como un caso general del DGRP. En concreto, en Corberán, Romero y Sanchis (2003) y en Corberán, Mejía y Sanchis (2005) se propone una formulación y se da una descripción parcial del poliedro de soluciones, que utilizan para desarrollar un algoritmo de planos de corte que produce buenos resultados computacionales. Muchos de los resultados presentados en estos trabajos para el MGRP pueden aplicarse al DGRP considerando $E = \emptyset$. Otros dos trabajos cercanos son los de Savall (1990) y Savall y Campos (1995) sobre el DRPP, que puede considerarse un caso particular del DGRP cuando $V_R = \emptyset$. En el primero de estos trabajos se hace un estudio del poliedro de soluciones para el DRPP y se proponen algunos algoritmos heurísticos para su resolución, que son mejorados y publicados más tarde en Savall y Campos(1995).

Uno de los motivos de estudio del DGRP se halla en la cantidad de problemas que tiene como casos particulares. Por ejemplo, si no hay vértices requeridos se tiene un DRPP. Además, en el caso en que todos los arcos sean requeridos tenemos un DCPP, y si no hay arcos requeridos y todos los vértices son requeridos, tenemos un problema Gráfico del Viajante Asimétrico (Graphical Asymmetric Traveling Salesman Problem, GATSP).

4.1.1. Definición del problema

Sea $G = (V, A)$ un grafo dirigido con un coste o longitud c_{ij} asociado a cada arco $(i, j) \in A$. Sean $A_R \subseteq A$, $V_R \subseteq V$ sendos subconjuntos de arcos y de vértices que llamamos requeridos. El Problema General de Rutas en un grafo dirigido (DGRP) consiste en encontrar un tour (camino cerrado) de longitud mínima que pase, al menos una vez, por cada arco y por cada vértice requeridos.

Notar que si un arco (i, j) es requerido, podemos considerar que sus vértices inicial y final son requeridos, es decir, que si $(i, j) \in A_R$ entonces $i, j \in V_R$. Además, como es usual en el estudio de problemas de rutas por arcos, podríamos suponer que $V = V_R$ (todos los vértices del grafo son requeridos), ya que una instancia para el DGRP con vértices no requeridos, puede ser transformada en una instancia equivalente que sólo tiene los vértices requeridos.

Esta transformación es similar a la de Christofides et al. (1981) para el RPP no dirigido (ver también Eiselt, Gendreau and Laporte, 1995b, o Christofides et al. 1986):

Paso 1: Añadir al grafo $G_R = (V_R, A_R)$ un arco no requerido (i, j) entre cada par de vértices i, j de V_R y asignarle un coste c_{ij} igual al coste del camino más corto en G desde i hasta j, si dicho camino no usa ningún arco requerido.

Paso 2: Eliminar: a) cada arco no requerido que sea paralelo a uno requerido con el mismo coste, y b) todos los arcos (i, j) no requeridos para los que existe un vértice k tal que $c_{ij} = c_{ik} + c_{kj}$.

Este tipo de transformaciones son utilizadas en el estudio de los problemas de rutas por arcos porque simplifican la estructura del problema y su formulación, y pueden producir instancias más sencillas y fáciles de resolver.

Sin embargo, en algunas ocasiones puede ocurrir que la instancia transformada sea mucho más grande y difícil de resolver que la instancia original. Este es el caso de las instancias propuestas en Blais y Laporte (2003) para el DGRP. Estas instancias están generadas a partir de un grafo original con 5000 vértices y 50000 arcos. Supongamos que, sobre este grafo, generamos una instancia del DGRP seleccionando aleatoriamente 1000 vértices requeridos y 1500 arcos requeridos. La instancia sin transformar puede tener 4000 vértices requeridos y 1000 no requeridos, 1500 arcos requeridos y 48500 arcos no requeridos. Una transformación como la anterior para

eliminar los vértices no requeridos puede producir un grafo con sólo 4000 vértices (los requeridos) pero con varios millones de arcos no requeridos (los arcos de un grafo completo de 4000 vértices excepto los que podamos eliminar en el Paso 2 de la transformación). En este caso, es más costoso computacionalmente resolver la instancia transformada que la instancia original. Sin embargo, si en el mismo grafo original con 5000 vértices y 50000 arcos seleccionamos 100 vértices requeridos y 150 arcos requeridos, sí resultará computacionalmente interesante trabajar con la instancia transformada.

Por lo tanto, unas veces interesa trabajar con la instancia transformada, que cumple que $V = V_R$, y otras veces interesa trabajar con la instancia original, que tiene vértices no requeridos. Para hacer este trabajo más general y más original, sobre todo comparado con otros trabajos sobre problemas de rutas por arcos conocidos, vamos a trabajar con un grafo dirigido

$$G = (V, A) = \big(V_R \cup V_{NR}\,,\; A_R \cup A_{NR}\big),$$

donde A_{NR} es el conjunto de arcos no requeridos y V_{NR} es el conjunto de vértices no requeridos (ni incidentes con arcos requeridos). Obviamente, los resultados que obtengamos serán también válidos para el caso particular en el que $V_{NR} = \emptyset$, es decir, para instancias transformadas.

4.2. Formulación y propiedades básicas

En esta sección presentaremos una formulación como problema de programación lineal entera del DGRP, definiremos el poliedro asociado a sus soluciones posibles y discutiremos algunas propiedades básicas del mismo.

4.2.1. Formulación

La notación que usamos para este problema es muy parecida a la que hemos usado para el SCP. Denotamos por $V_1, V_2, ..., V_p$ (R-sets) a los conjuntos de vértices que forman las p componentes débilmente conexas inducidas por los vértices y arcos requeridos, cumpliéndose que $V_1 \cup \cdots \cup V_p = V_R$. Dados $S, S_1, S_2 \subset V$, denotamos por $\delta^+(S)$ y $\delta^-(S)$ al conjunto de los arcos salientes y entrantes de S, $A(S)$ es el conjunto de arcos dentro de S y $(S_1 : S_2)$ los arcos con el vértice inicial en S_1 y el vértice final en S_2. Los respectivos conjuntos referidos sólo a los arcos requeridos se representan por $\delta_R^+(S)$, $\delta_R^-(S)$, $(S_1 : S_2)_R$, etc., y los referidos sólo a arcos no requeridos por $\delta_{NR}^+(S)$, $\delta_{NR}^-(S)$, $(S_1 : S_2)_{NR}$, etc.

En principio, los tours que buscamos como soluciones posibles para el DGRP en G son los caminos cerrados (closed walks) que pasan, al menos una vez, por todos los arcos y vértices requeridos. Sin embargo, para que la envoltura convexa de estas soluciones posibles sea un poliedro, y poder así resolver el DGRP con las técnicas de la combinatoria poliédrica, necesitamos que, dada una solución posible para el DGRP en G y dado cualquier camino cerrado formado por arcos de G, la suma de ambos sea también una solución posible (ver Teorema 4.2.1). Puesto que el grafo G tiene vértices y arcos no requeridos, esto nos lleva a tener que considerar como soluciones posibles para el DGRP en G a aquellas formadas por un camino cerrado que pasa por todos los arcos y vértices requeridos más otros caminos cerrados desconectados del primero formados por arcos y vértices no requeridos. Así, definimos:

Definición 4.2.1 (Tour para el DGRP)
Llamaremos tour para el DGRP *en G a cualquier conjunto de caminos cerrados (closed walks) en G tal que uno de ellos pase, al menos una vez, por todos los arcos y vértices requeridos.*

A cada tour para el DGRP en G le asociamos un vector de incidencia $x \in \mathbb{R}^{|A|}$ con una componente o variable x_{ij} por cada arco $(i,j) \in A$ representando el número de veces que el tour x recorre el arco (i,j). Haciendo abuso de notación llamaremos tour para el DGRP en G tanto al conjunto de arcos como a su vector de incidencia. El DGRP se puede formular como sigue:

$$\text{Minimize} \qquad \sum_{(i,j)\in A} c_{ij}x_{ij}$$

s.t.:

$$x(\delta^+(i)) = x(\delta^-(i)), \quad \forall i \in V \tag{4.1}$$

$$x(\delta^+(S)) \geq 1, \ \forall S = \left(\bigcup_{i\in Q} V_i\right) \cup W, \ \emptyset \neq Q \subsetneq \{1,\ldots,p\}, \ W \subseteq V_{NR} \tag{4.2}$$

$$x_{ij} \geq 1, \quad \forall (i,j) \in A_R \tag{4.3}$$

$$x_{ij} \geq 0, \quad \forall (i,j) \in A_{NR} \tag{4.4}$$

$$x_{ij} \text{ entero}, \qquad \forall (i,j) \in A \tag{4.5}$$

Las igualdades (4.1) son las llamadas de *ecuaciones de simetría*, que aseguran que en cada tour hay el mismo número de arcos entrantes y salientes en cada uno de los vértices del grafo (y, por extensión, hay el mismo número de arcos en ambos sentidos de cada cortadura del grafo). Las desigualdades (4.2) son las desigualdades de *conectividad*, que obligan a la solución a conectar todas las componentes inducidas por los arcos y vértices requeridos (todos los R-sets). Las desigualdades (4.3) son las de *obligatoriedad*, que aseguran que los arcos requeridos serán recorridos al menos una vez, y las desigualdades (4.4) son las *triviales*. Y por último se tienen las condiciones de *integridad* (4.5), que obligan a que las variables sean enteras. No-

tar que esta formulación permite soluciones posibles con algún camino cerrado de arcos y vértices no requeridos desconectado del camino cerrado que recorre todo lo requerido (ver Definición 4.2.1), pero ninguna de éstas será solución óptima.

4.2.2. El poliedro de soluciones

Definimos el poliedro de soluciones del DGRP en el grafo G, al que denotamos por DGRP(G), como la envoltura convexa de todos los tours para el DGRP en G, es decir:

$$\text{DGRP}(G) = conv\big\{x \in \mathbb{R}^{|A|} : \quad x \text{ es un tour para el DGRP en } G\big\}.$$

A continuación demostraremos que DGRP(G) es un poliedro (no es obvio pues es la envoltura convexa de un número infinito de puntos) y estudiaremos su dimensión. Para demostrar que se trata de un poliedro, hemos de encontrar dos conjuntos finitos de vectores en $\mathbb{R}^{|A|}$, W_1 y W_2 cumpliendo,

$$DGRP(G) = conv(W_1) + cone(W_2)$$

Definición 4.2.2 (Tour elemental)
Llamaremos ***tour elemental*** *para el* DGRP *a cualquier tour que no pueda ser obtenido a partir de otro tour eliminando algunos ciclos de éste. Denotaremos por T al conjunto de tours elementales, que obviamente es finito.*

Definición 4.2.3
Llamaremos a $\mathbb{Z}$ al conjunto de los vectores de incidencia de todos los ciclos en G. $\mathbb{Z}$ es también un conjunto finito.

Proposición 4.2.1
Sea X el conjunto de todos los vectores que son tours para el DGRP. Para cualesquiera $x \in X$ y $z \in \mathbb{Z}$ se cumple que $x' = x + nz \in X \ \forall\, n \geqslant 0$ entero.

Demostración: Es fácil ver que como $x \in X$ es un tour para el DGRP en G, al añadirle n copias del ciclo z obtenemos un vector $x' = x + nz$ que es también un tour para el DGRP y, por lo tanto, $x' \in X$. ♦

Proposición 4.2.2

Si $x \in X$, $z_1, ..., z_p \in T$, entonces $x + n_1 z_1 + ... + n_p z_p \in X$, $\forall n_i \geqslant 0$ entero, $i = 1, ..., p$.

Demostración: Obvio aplicando p veces la preposición anterior. ♦

Proposición 4.2.3

Sea $x \in DGRP(G) = conv(X)$ y $z_1, ..., z_p \in T$, entonces $x + n_1 z_1 + ... + n_p z_p \in DGRP(G)$ $\forall n_i \geqslant 0$ entero $i = 1, ..., p$.

Demostración: Si $x \in DGRP(G) = conv(X)$ se cumple que $x = \lambda_1 x^1 + ... + \lambda_k x^k$ con $x^i \in X$ y $\sum \lambda_i = 1, \lambda_i \geqslant 0, i = 1, ..., k$. Así, $x + n_1 z_1 + ... + n_p z_p = \lambda_1 x^1 + ... + \lambda_k x^k + n_1 z_1 + ... + n_p z_p = \lambda_1 x^1 + ... + \lambda_k x^k + \sum n_i z_i = \lambda_1 (x^1 + \sum n_i z_i) + ... + \lambda_k (x^k + \sum n_i z_i)$, que pertenece a $DGRP(G)$ ya que cada $x^j + \sum n_i z_i$ es un tour para el DGRP en G. ♦

Proposición 4.2.4

Dado cualquier tour $x \in X$, existe $\overline{x} \in T$ y existen enteros no negativos n_i y ciclos $z_i \in \mathbb{Z}$ tal que $x = \overline{x} + \sum n_i z_i$.

Demostración: Si x contiene un ciclo z_1 tal que $x' = x - z_1$ es un tour para el DGRP, se obtiene que $x = x' + z_1$. En otro caso, $x \in T$. Repitiendo el razonamiento, con x', se obtiene que $x = x'' + z_1 + z_2$ o bien $x' \in T$. Así sucesivamente, se llega a

$$x = \overline{x} + \sum n_i z_i$$

con $n_i \geqslant 0$ enteros y $\overline{x} \in X$. ♦

Teorema 4.2.1

Si $G = (V, A) = (V_R \cup V_{NR}, A_R \cup A_{NR})$ es un grafo dirigido fuertemente conexo,

entonces DGRP(G) es un poliedro no acotado.

Demostración: Basta ver que DGRP(G)= $conv(T) + cone(\mathbb{Z})$:

Veamos primero que $DGRP(G) \subseteq conv(T) + cone(\mathbb{Z})$. Sea $x \in DGRP(G) = conv(X)$ por lo tanto $x = \lambda_1 x^1 + ... + \lambda_k x^k$ con $x^i \in X$, $\sum \lambda_j = 1$, $\lambda_j \geq 0$ $i = 1, ..., k$. Hemos demostrado anteriormente que cada tour x^j puede expresarse como $x^j = \overline{x}^j + \sum n_i^j z_i$ con $\overline{x}^j \in T$, $n_i^j \geq 0$ y enteros, y $z_i \in \mathbb{Z}$. Por lo tanto se tiene que $x = \lambda\overline{x}^1 + ... + \lambda_k \overline{x}^k + \sum_j \sum_i \lambda_j n_i^j z_i \in conv(T) + cone(\mathbb{Z})$.

Veamos ahora que, $conv(T) + cone(\mathbb{Z}) \subseteq DGRP(G)$. Sea $\overline{x} \in T$ y $z \in \mathbb{Z}$, obviamente $\overline{x} + kz \in DGRP(G)$ $\forall k = 0, 1, 2, ...$ Veamos también que $\forall \lambda \geq 0, \overline{x} + \lambda z \in DGRP(G)$. Si λ no fuera entero tenemos: $\overline{x} + \lfloor \lambda \rfloor z \in DGRP(G)$ y $\overline{x} + \lceil \lambda \rceil z \in DGRP(G)$. Sea $\beta = \lceil \lambda \rceil - \lambda$, $0 < \beta < 1$, se cumple que $\overline{x} + \lambda z = \beta(\overline{x} + \lfloor \lambda \rfloor z) + (1 - \beta)(\overline{x} + \lceil \lambda \rceil z) \in DGRP(G)$, por ser convexo. Luego $\overline{x} + \lambda z \in DGRP(G)$, $\forall \lambda \geq 0$. Por lo tanto, $T + cone(\mathbb{Z}) \subset DGRP(G)$, y $conv(T + cone(\mathbb{Z})) \subseteq conv(DGRP(G)) = DGRP(G)$, por ser convexo $DGRP(G)$. Es decir, $conv(T) + conv(cone(\mathbb{Z})) \subseteq DGRP(G)$. Como, por otro lado, se tiene que $conv(cone(\mathbb{Z})) = cone(\mathbb{Z})$, obtenemos que $conv(T) + cone(\mathbb{Z}) \subseteq DGRP(G)$. ♦

Antes de introducir el teorema siguiente necesitamos un resultado sobre la dimensión del poliedro del Problema del Viajante Asimétrico Gráfico (Graphical Asymmetric Traveling Salesman Problem, GATSP), obtenido por Chopra y Rinaldi (1996). El GATSP se define en un grafo dirigido $G = (V, A)$ fuertemente conexo y consiste en encontrar un tour de coste mínimo que pase al menos una vez por todos los vértices del grafo. Su poliedro asociado, GATSP(G), es la envoltura convexa de todos los tours para el GATSP en G. Chopra y Rinaldi (1996) proporcionan el siguiente resultado:

Teorema 4.2.2

Si $G = (V, A)$ un grafo dirigido fuertemente conexo, entonces

$$dim(GATSP(G)) = |A| - |V| + 1.$$

Con este resultado, podemos demostrar la dimensión de nuestro poliedro DGRP(G).

Teorema 4.2.3 (Dimensión del poliedro)

Si $G = (V, A) = (V_R \cup V_{NR}, A_R \cup A_{NR})$ es un grafo dirigido fuertemente conexo, entonces

$$dim(DGRP(G)) = |A| - |V| + 1.$$

Demostración: Como la matriz de coeficientes de las ecuaciones (4.1) tiene rango $|V| - 1$, tenemos que $dim(DGRP) \leq |A| - (|V| - 1)$. Para obtener la otra desigualdad, consideremos el GATSP en el grafo G. Como G es fuertemente conexo, por el Teorema 4.2.2, la dimensión del poliedro GATSP(G), es $m = |A| - |V| + 1$, por lo que podemos encontrar $x^1, ..., x^{m+1}$ vectores afínmente independientes que son tours para el GATSP en G, es decir, todos estos tours pasan, al menos una vez, por todos los vértices de G, requeridos y no requeridos. Sin embargo, no tienen por qué pasar por todos los arcos requeridos. Para cada arco requerido $(i, j) \in A_R$ podemos encontrar un camino en G de j a i, digamos P_{ji} (pues G es fuertemente conexo). Entonces, $P_{ji} \cup \{(i, j)\}$ es un ciclo en G que contiene al arco requerido (i, j). Sea $\overline{x}$ el vector de incidencia del conjunto de todos estos ciclos. Sumando éste a cualquier tour para el GATSP x^j anterior, obtenemos un tour para el DGRP en G, $x^j + \overline{x}$. Así, tenemos $m + 1$ tours para el DGRP en G, $x^1 + \overline{x}, ..., x^{m+1} + \overline{x}$, afínmente independientes, por lo que dim(DGRP(G))$\geq |A| - |V| + 1$ y tenemos la igualdad. ♦

Nota 4.2.1

Como no es de dimensión completa, varias desigualdades pueden definir la misma faceta del poliedro DGRP(G). Diremos que estas desigualdades son equivalentes.

Teorema 4.2.4 (Facetas triviales)

Sea $(i,j) \in A_{NR}$*. Si* (i,j) *no es un arco de corte de* G*, la desigualdad trivial* (4.4) $x_{ij} \geq 0$ *define faceta de DGRP(G).*

Demostración: Tenemos que encontrar $m = |A| - |V| + 1$ tours para el DGRP en G afínmente independientes cumpliendo la desigualdad con igualdad. Si (i,j) no es un arco de corte de G, entonces $G\backslash\{(i,j)\}$ sigue siendo fuertemente conexo y, por el Teorema 4.2.3, la dimensión de DGRP($G\backslash\{(i,j)\}$) es $(|A|-1)-|V|+1 = |A|-|V| = m-1$. Así, podemos encontrar $x^1, ..., x^m$ tours para el DGRP en $G\backslash\{(i,j)\}$ afínmente independientes. Estos m vectores, completados con la componente $x_{ij} = 0$, son también tours para el DGRP en G y todos satisfacen $x_{ij} = 0$, luego $x_{ij} \geq 0$ define faceta de DGRP(G). ♦

Teorema 4.2.5 (Facetas de obligatoriedad)

Sea $(i,j) \in A_R$*. Si* (i,j) *no es un arco de corte de* G*, la desigualdad de obligatoriedad* (4.3) $x_{ij} \geq 1$ *define faceta de DGRP(G).*

Demostración: Si (i,j) no es un arco de corte, entonces $G\backslash\{(i,j)\}$ es fuertemente conexo, la dimensión de DGRP($G\backslash\{(i,j)\}$) es $|A|-|V| = m-1$ y podemos encontrar $x^1, ..., x^m$ tours para el DGRP en $G\backslash\{(i,j)\}$ afínmente independientes. Notar que éstos no son tours para el DGRP en G ya que no pasan por el arco requerido (i,j). Sea ahora P_{ji} un camino de j a i en $G\backslash\{(i,j)\}$, que existe por ser $G\backslash\{(i,j)\}$ fuertemente conexo. Si sumamos el vector de incidencia del ciclo $P_{ji} \cup \{(i,j)\}$ a cada uno de los vectores $x^1, ..., x^m$ obtenemos m tours para el DGRP en G afínmente independientes, y todos cumplen que $x_{ij} = 1$. Por lo tanto, $x_{ij} \geq 1$ define faceta de $DGRP(G)$. ♦

4.3. Poliedro I: Desigualdades de configuración, conectividad y R-imparidad

Las desigualdades de configuración fueron propuestas por Nadeff y Rinaldi en 1991 para el GTSP. Están asociadas a un grafo "reducido" que resulta de comprimir el grafo original G y al que se le llama grafo de configuración. Nadeff y Rinaldi (1991) demostraron que todas las desigualdades que inducen faceta del GTSP, excepto las desigualdades triviales, son desigualdades de configuración. Además, demostraron un teorema de "lifting", que establece condiciones para que, si una desigualdad induce faceta del poliedro asociado al grafo de configuración, entonces también induzca faceta del poliedro asociado al grafo original G. Posteriormente, Corberán, Mejía y Sanchis (2005) aplicaron estos conceptos al MGRP en grafos transformados, es decir, cumpliendo que $V_R = V$. Aquí vamos a estudiar estos conceptos para el DGRP en grafos no transformados $G = (V, A) = (V_R \cup V_{NR}, A_R \cup A_{NR})$.

4.3.1. Desigualdades de configuración

Sea $G = (V, A) = (V_R \cup V_{NR}, A_R \cup A_{NR})$ un grafo dirigido fuertemente conexo. Llamamos *configuración* a un par $\mathcal{C} = (\mathbf{B}, c)$, donde $\mathbf{B} = \{B_1, ..., B_k\}$ es una partición de V y c es una función de costes definida en $\mathbf{B} \times \mathbf{B}$, satisfaciendo:

1. Cada subgrafo inducido $G(B_r)$ es fuertemente conexo, $r = 1, \ldots, k$.
2. No existe ningún camino cerrado $B_r, B_q, \ldots, B_p, B_r$ con c-coste total, $c(B_r, B_q) + \cdots + c(B_p, B_r)$, negativo.

Asociado a una configuración $\mathcal{C}$ en G, tenemos un *grafo de configuración*, $G_{\mathcal{C}}$, que tiene como conjunto de vértices a $\mathbf{B}$, de los cuales son vértices requeridos aquellos

B_r tales que $B_r \cap V_R \neq \emptyset$ y son vértices no requeridos los demás, tiene un arco requerido (B_r, B_q) con un coste igual a $c(B_r, B_q)$ por cada arco requerido en el grafo original que va desde un vértice de B_r a otro de B_q y tiene un arco no requerido (B_r, B_q) un coste igual a $c(B_r, B_q)$ por cada conjunto $(B_r : B_q)_{NR} \neq \emptyset$ en el grafo original. En otras palabras, $G_{\mathcal{C}}$ es el grafo que resulta de comprimir en el grafo original cada conjunto B_r en un único vértice y, después, reemplazar cada conjunto de arcos no requeridos en paralelo por un único arco no requerido, manteniendo todos los requeridos. Por ejemplo, consideremos el grafo de la Figura 4.1(a), donde los arcos requeridos están dibujados con trazo continuo y los arcos no requeridos con trazo discontinuo. Si tomamos la partición del conjunto de vértices formada por $B_1 = \{1, 2, 4\}$, $B_2 = \{3, 6, 7, 8\}$, $B_3 = \{5\}$, obtenemos el grafo de configuración de la Figura 4.1(b). Notar que se mantienen los arcos requeridos $\{(1, 3), (2, 3), (4, 5)\}$ pero que sólo tenemos un arco no requerido por cada par $(B_p : B_q) \neq \emptyset$ en el grafo original.

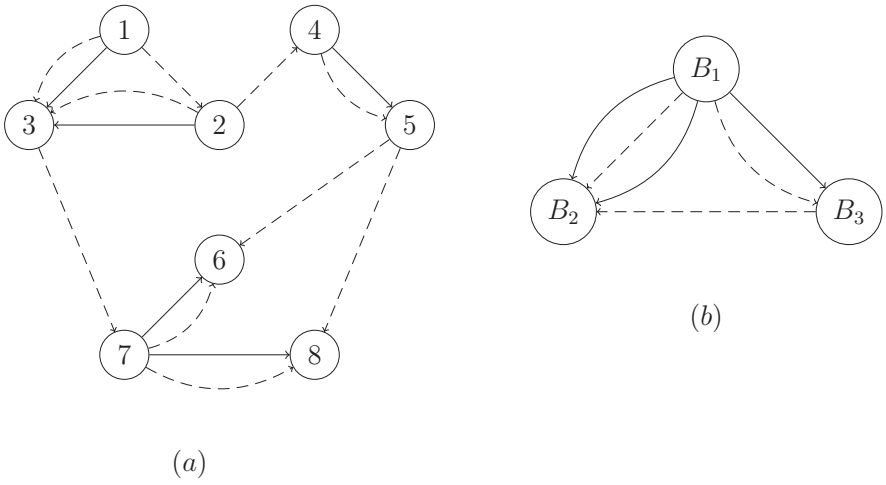

Figura 4.1: *Un ejemplo de grafo de configuración $G_{\mathcal{C}}$.*

Una configuración $\mathcal{C}$ define la correspondiente *desigualdad de configuración*,

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \geq c_0, \tag{4.6}$$

donde

1. $c_{ij} = 0, \quad \forall (i,j) \in A(B_r)$, r=1,...,k,
2. $c_{ij} = c(B_r, B_q), \quad \forall (i,j) \in (B_r : B_q)$, y
3. c_0 es el c-coste del tour para el $DGRP$ en $G_{\mathcal{C}}$ de mínimo c-coste.

Notar que el vector c de la configuración juega un doble papel: por un lado es el vector de los costes del grafo de configuración y también es el vector de los coeficientes de las variables en la desigualdad de configuración correspondiente. Para demostrar que todas las desigualdades que inducen faceta de DGRP(G), excepto las equivalentes a las triviales o a las de obligatoriedad, son desigualdades de configuración, necesitamos un lema previo.

Lema 4.3.1

Sea $F(x) \geq c_0$ una desigualdad que induce faceta de DGRP(G) que no es equivalente a una trivial (4.4) *ni a una de obligatoriedad* (4.3)*. Para cada arco $(i,j) \in A_{NR}$ existe, al menos, un tour para el* DGRP *en G que cumple $F(x) = c_0$ y $x_{ij} \geq 1$. Para cada arco $(i,j) \in A_R$ existe, al menos, un tour para el* DGRP *en G que cumple $F(x) = c_0$ y $x_{ij} \geq 2$.*

Demostración: Dado un arco $(i,j) \in A_{NR}$, si todos los tours para el DGRP en G que cumplen $F(x) = c_0$ cumpliesen también que $x_{ij} = 0$, entonces

$$\big\{x \in DGRP(G) : F(x) = c_0\big\} \subset \big\{x \in DGRP(G) : x_{ij} = 0\big\},$$

y por lo tanto, o $F(x) \geq c_0$ es equivalente a la desigualdad trivial $x_{ij} \geq 0$ o está dominada por ésta y no induce faceta. Del mismo modo, dado un arco $(i,j) \in A_R$, si todos los tours para el DGRP en G que cumplen $F(x) = c_0$ cumpliesen también que $x_{ij} = 1$, entonces

$$\big\{x \in DGRP(G) : F(x) = c_0\big\} \subset \big\{x \in DGRP(G) : x_{ij} = 1\big\},$$

y por lo tanto, o $F(x) \geq c_0$ es equivalente a la desigualdad de obligatoriedad $x_{ij} \geq 1$ o está dominada por ésta y no induce faceta. ♦

Nota 4.3.1

En ambos casos, es decir, $(i,j) \in A_{NR}$ y el tour x cumple $x_{ij} \geq 1$, o bien $(i,j) \in A_R$ y el tour x cumple $x_{ij} \geq 2$, diremos que x recorre el arco (i,j) *de modo extra.*

Teorema 4.3.1

Todas las desigualdades que inducen facetas de DGRP(G), excepto las equivalentes a las triviales (4.4) *o a las de obligatoriedad* (4.3)*, son desigualdades de configuración.*

Demostración: Sea $F(x) = \sum_{(i,j)\in A} c_{ij}x_{ij} \geq c_0$ una desigualdad que induce faceta de DGRP(G) y que no es equivalente a una trivial (4.4) ni a una de obligatoriedad (4.3). Sea $A^0 = \{(i,j) \in A : c_{ij} = 0\}$. Sea S el conjunto de vértices de una de las componentes fuertemente conexas del grafo $G^0 = (V, A^0)$. Supongamos que existe un arco $(i,j) \in A(S) \setminus A^0$, es decir, un arco en $A(S)$ tal que $c_{ij} > 0$ ó $c_{ij} < 0$. En el primer caso, sea x un tour para el DGRP en G que recorre (i,j) de modo extra y que cumple $F(x) = c_0$ (existe por el Lema 4.3.1). Como el grafo $G(S) = (S, A^0(S))$ es fuertemente conexo, existe un camino de i a j usando arcos de A^0 y, si sustituimos el arco (i,j) en x por los arcos de este camino, obtendríamos un nuevo tour x^* tal que $F(x^*) = F(x) - c_{ij} < c_0$, lo que contradice que $F(x) \geq c_0$ sea una desigualdad válida. En el caso en que $c_{ij} < 0$, podemos construir un ciclo de coste negativo uniendo (i,j) y el camino en $G(S)$ de j a i, y sumando un número suficiente de veces este ciclo a x obtendríamos un tour que no cumple la desigualdad, lo que vuelve a ser una contradicción. Así pues, $c_{ij} = 0 \ \forall\, (i,j) \in A(S)$ y consideramos como conjuntos B_r de la configuración a las componentes fuertemente conexas de $G^0 = (V, A^0)$.

Sean ahora B_r y B_q dos conjuntos tales que $(B_r : B_q) \neq \emptyset$. Supongamos que existen dos arcos (i,j) y (u,v) en $(B_r : B_q)$ con $c_{ij} > c_{uv}$. Sea x un tour que pasa por (i,j) de modo extra y que cumple $F(x) = c_0$. Podemos sustituir el arco (i,j) por el camino entre i y u en B_r, el arco (u,v) y el camino entre v y j en B_q, obteniendo un nuevo tour x^* que cumple $F(x^*) = F(x) - c_{ij} + c_{uv} < c_0$, luego la desigualdad no sería válida.

Finalmente supongamos que existe un ciclo en $G_{\mathcal{C}}$ con coste negativo. Dado cualquier

tour x con $F(x) = c_0$, podríamos añadirle ese ciclo obteniendo un nuevo tour x^* que no cumpliría la desigualdad, lo cual es una contradicción. ♦

Una configuración $\mathcal{C}$ sobre G puede también considerarse una configuración sobre el grafo comprimido $G_{\mathcal{C}} = (\mathbf{B}, A_{\mathcal{C}})$ y, por lo tanto también define una desigualdad sobre este grafo:

$$\sum_{(B_r,B_q)\in A_{\mathcal{C}}} c(B_r, B_q)x_{B_rB_q} \geq c_0$$

donde $x_{B_rB_q}$ representa el número de veces que el arco (B_r, B_q) es recorrido. Se tiene además que cualquier tour x para el DGRP en G puede ser comprimido para convertirlo en un tour $x^{\mathcal{C}}$ para el DGRP en $G_{\mathcal{C}}$ y ambos tendrían el mismo c-coste. Por otra parte, un tour $x^{\mathcal{C}}$ para el DGRP en $G_{\mathcal{C}}$ puede convertirse en un tour x para el DGRP en G, puesto que los subgrafos $G(B_r)$ son fuertemente conexos, y con el mismo c-coste, puesto que los coeficientes de las variables asociadas a los arcos de $A(B_r)$ son todos cero. Por lo tanto, si la desigualdad de configuración es válida para el poliedro DGRP($G_{\mathcal{C}}$), también lo es para DGRP(G).

Teorema 4.3.2 (de "lifting")
Sea $\mathcal{C}$ una configuración en G. Si la desigualdad de configuración asociada a $\mathcal{C}$ como configuración en el grafo $G_{\mathcal{C}}$ define una faceta de DGRP($G_{\mathcal{C}}$), entonces la desigualdad de configuración asociada a $\mathcal{C}$ como configuración en el grafo G define una faceta de DGRP(G).

Demostración: Por inducción, basta demostrar el resultado para el grafo G resultante de sustituir en $G_{\mathcal{C}}$ el nodo B_1 por un grafo fuertemente conexo $G(B_1)$. Por simplicidad, denotamos por $F(x) \geq c_0$ tanto a la desigualdad en G como a la desigualdad en $G_{\mathcal{C}}$.

Sea $K = dim(DGRP(G_{\mathcal{C}})) = |A_{G_{\mathcal{C}}}| - |V_{G_{\mathcal{C}}}| + 1$ y supongamos que $c_0 \neq 0$. Existen K tours para el DGRP en $G_{\mathcal{C}}$ $x^1, ..., x^K$ linealmente independientes cumpliendo

$F(x^i) = c_0$ (si $c_0 = 0$, serían $K-1$ en lugar de K, pero la demostración es similar). Al sustituir B_1 por $G(B_1)$ aparecen arcos no requeridos entre vértices de $G(B_1)$ y nodos B_r, $r \neq 1$. Llamemos $G'_{\mathcal{C}}$ al grafo obtenido de $G_{\mathcal{C}}$ añadiéndole Q arcos no requeridos $a'_1, \ldots, a'_Q$, cada uno de ellos paralelo a un arco $a_1 \ldots, a_Q$ de $G_{\mathcal{C}}$ entre B_1 y otros B_r. Para cada arco a'_i sea x^j el tour para el DGRP en $G_{\mathcal{C}}$ que recorre a_i cumpliendo $F(x) = c_0$ y reemplazamos el arco a_i por el arco a'_i. Así, tenemos $K+Q$ tours para el DGRP en $G'_{\mathcal{C}}$ cumpliendo $F(x) = c_0$. Como el grafo $G(B_1)$ es fuertemente conexo, cada uno de estos tours puede ser completado para obtener $K+Q$ tours para el DGRP en G cumpliendo $F(x) = c_0$, que llamamos $x^1, ..., x^{K+Q}$.

Consideremos el DGRP en el grafo $G(B_1)$ y sea $M = dim(DGRP(G(B_1))) = |A_{G(B_1)}| - |V_{G(B_1)}| + 1$. Existen M tours para el DGRP en $G(B_1)$ linealmente independientes, $y^1, ..., y^M$. Sumandoles x^1, obtenemos $x^1 + y^1, ..., x^1 + y^M$ que son tours para el DGRP en G cumpliendo $F(x) = c_0$. Expresando los $K+Q+M$ tours para el DGRP en G que hemos construido como filas y restando x^1 a las M últimas, obtenemos la matriz de la figura 4.2, que es de rango completo. Sólo nos falta ver que $K+Q+M$ es la dimensión del poliedro:

$$dim(DGRP(G)) = |A| - |V| + 1 = \Big(|A_{G_{\mathcal{C}}}| + Q + |A_{G(B_1)}|\Big) - \Big(|V_{G_{\mathcal{C}}}| + |V_{G(B_1)}| - 1\Big) + 1 =$$

$$= \Big(|A_{G_{\mathcal{C}}}| - |V_{G_{\mathcal{C}}}| + 1\Big) + Q + \Big(|A_{G(B_1)}| - |V_{G(B_1)}| + 1\Big) = K + Q + M. \qquad \blacklozenge$$

$$\left(\begin{array}{c|c} x_1 & \\ \vdots & * \\ x^{K+Q} & \\ \hline & y^1 \\ 0 & \vdots \\ & y^M \end{array}\right)$$

Figura 4.2: *Matriz del teorema 4.3.2*

4.3.2. Desigualdades de conectividad

Como ejemplo de aplicación del concepto de desigualdad de configuración y del Teorema 4.3.2 de "lifting", vamos a demostrar que las desigualdades de conectividad (4.2) de la formulación inducen faceta del poliedro de soluciones.

Teorema 4.3.3
Las desigualdades de conectividad (4.2), $x(\delta^+(S)) \geq 1$, *con* $S = \left(\underset{i\in Q}{\cup} V_i\right) \cup W$, $\emptyset \neq Q \subsetneq \{1,\ldots,p\}$, $W \subseteq V_{NR}$, *inducen faceta de* $DGRP(G)$ *si los subgrafos inducidos* $G(S)$ *y* $G(V\backslash S)$ *son fuertemente conexos.*

Demostración: Si $G(S)$ y $G(V\backslash S)$ son fuertemente conexos, la desigualdad de conectividad es una desigualdad de configuración cuyo grafo de configuración, $G_{\mathcal{C}}$, tiene dos vértices, $B_1 = S$, requerido pues $Q \neq \emptyset$, y $B_2 = V\backslash S$, requerido pues $Q \subsetneq \{1,\ldots,p\}$, y tiene dos arcos no requeridos con sentidos opuestos, pues G es fuertemente conexo. Por lo tanto, $dim(DGRP(G_{\mathcal{C}})) = 2 - 2 + 1$. Por otro lado, el tour $x_{B_1B_2} = x_{B_2B_1} = 1$ cumple que $x(\delta^+(B_1)) = 1$, por lo que la desigualdad induce faceta de DGRP($G_{\mathcal{C}}$) y, por el Teorema 4.3.2 de 'lifting', la desigualdad (4.2) induce faceta de DGRP(G). ♦

4.3.3. Desigualdades de R-imparidad

Decimos que un subconjunto de vertices $S \subseteq V$ es R-impar si $|\delta_R(S)|$ es impar, es decir, si en la cortadura $\delta(S)$ hay un número impar de arcos requeridos (ver Figura 4.3). Puesto que todos los tours para el DGRP tienen que cruzar cualquier cortadura un número par de veces, todos los tours x pasan por todos los arcos requeridos una vez y, al menos una vez más por algún arco, requerido o no requerido, de $\delta(S)$. Así,

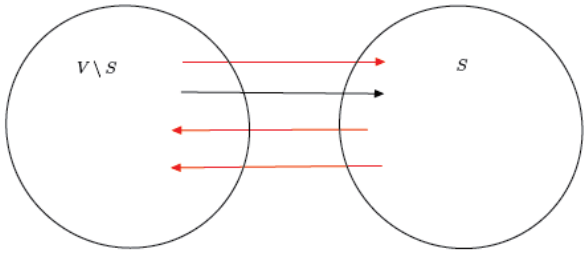

Figura 4.3: *Cortadura R-impar*

la siguiente desigualdad es válida para el DGRP en G:

$$x(\delta(S)) \geq |\delta_R(S)| + 1, \quad \forall\, S \subset V : \ |\delta_R(S)| \ \text{ es impar.} \tag{4.7}$$

Estas desigualdades se conocen como desigualdades de R-imparidad en otros problemas de rutas por arcos e inducen faceta de sus correspondientes poliedros. Sin embargo, en el DGRP están implicadas por las ecuaciones de simetría (4.1). Es fácil ver que sumando las desigualdades (4.1) para cada vértice $i \in S$ obtenemos la ecuación $x(\delta^+(S)) = x(\delta^-(S))$. Por otro lado, si $|\delta_R(S)|$ es impar, podemos suponer, por ejemplo, que $|\delta_R^+(S)| \geq |\delta_R^-(S)| + 1$, y se tiene que $x(\delta(S)) = x(\delta^+(S)) + x(\delta^-(S)) = x(\delta^+(S)) + x(\delta^+(S)) \geq |\delta_R^+(S)| + |\delta_R^+(S)| \geq |\delta_R^+(S)| + |\delta_R^-(S)| + 1 = |\delta_R(S)| + 1$. Por lo tanto estas desigualdades de R-imparidad, aunque son válidas para el DGRP(G) no definen faceta y no serán utilizadas.

4.4. Poliedro II: Desigualdades K-C, HC y PB

En esta sección, estudiaremos otras familias de desigualdades válidas para el DGRP(G) que definen faceta bajo ciertas condiciones. Se trata de algunas de las desigualdades más usadas en los problemas de rutas por arcos como son las K-C (1994), Path-Bridge (1997) o Honeycomb (1998).

4.4.1. Desigualdades K-C

Las desigualdades K-C fueron introducidas por Corberán y Sanchis (1994) para el Problema del Cartero Rural en un grafo no dirigido, RPP. Fueron adaptadas al MGRP en Corberán, Romero y Sanchis (2003) y al WGRP en Corberán, Plana y Sanchis (2007). En esta sección presentamos las desigualdades K-C para el DGRP.

Una configuración K-C está definida por una partición del conjunto de vértices $V = V_R \cup V_{NR}$ en $K+1$ subconjuntos, $\mathbf{B} = \{M_0, M_1, ..., M_{K-1}, M_K\}$, con $K \geq 3$, cumpliendo las siguientes condiciones:

- Cada R-set V_i está contenido en $M_0 \cup M_K$ o en alguno de los conjuntos M_j, $j = 1, ..., K-1$ y, recíprocamente, cada uno de estos conjuntos contiene a algún R-set V_i (así, todos los nodos del grafo de configuración son requeridos).
- Los subgrafos $G(M_i)$, $i = 0, ..., K$ son fuertemente conexos.
- Existe un número par de arcos requeridos entre M_0 y M_K, de manera que $\left|(M_0 : M_K)_R\right| = \left|(M_K : M_0)_R\right|$.
- Los conjuntos $(M_i : M_{i+1})$ y $(M_{i+1} : M_i)$, $i = 0, .., K-1$ son no vacíos.

Definimos el vector c de la configuración K-C del modo siguiente:

- $c(M_0, M_K) = c(M_K, M_0) = K - 2$.
- $c(M_i, M_j) = |i - j|, \quad \forall i, j : \{i, j\} \neq \{0, K\}$.

La partición $\mathbf{B} = \{M_0, M_1, ..., M_{K-1}, M_K\}$ y el vector c definen el grafo de configuración $G_{\mathcal{C}}$ cuyo esqueleto se muestra en la Figura 4.4. En esta figura, los arcos requeridos (entre M_0 y M_K) se muestran con trazo más grueso. El número que aparece al lado de cada arco es el correspondiente valor de c, es decir, es el coste del correspondiente arco el grafo configuración y es también el coeficiente en la de-

sigualdad K-C de las variables asociadas a los arcos en el grafo original. Todos los nodos de $G_{\mathcal{C}}$ son requeridos, pues M_0 y M_K tienen arcos requeridos incidentes y los demás conjuntos $M_1, \ldots, M_{K-1}$ contienen a algún R-set. Finalmente, definimos $c_0 = (K-2)\Big(|(M_0 : M_K)_R| + |(M_K : M_0)_R|\Big) + 2(K-1)$, que es el c-coste del tour para el DGRP de mínimo c-coste en $G_{\mathcal{C}}$ (ver la demostración del Teorema 4.4.1). Así, la desigualdad K-C correspondiente a esta configuración es:

$$(K-2)\Big(x(M_0 : M_K) + x(M_K : M_0)\Big) + \sum_{\substack{0 \le i,j \le K \\ (i,j) \neq (0,K)}} |i-j| x(M_i : M_j) \geq c_0 \qquad (4.8)$$

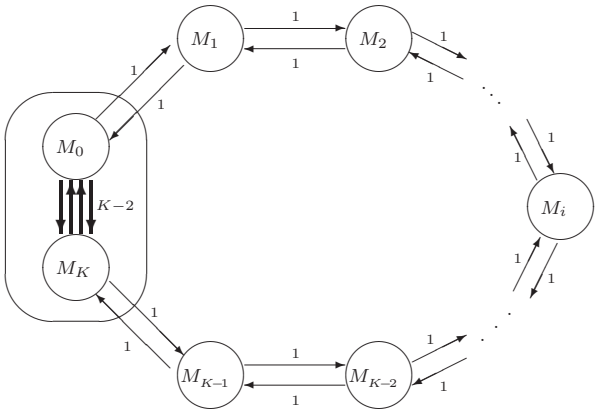

Figura 4.4: *Configuración K-C.*

En lo que sigue llamaremos:

- Arcos externos a los de $\mathcal{E}x = \bigcup_{i=0}^{K-1}\Big((M_i : M_{i+1}) \cup (M_{i+1} : M_i)\Big)$.
- Arcos internos a los de $\mathcal{I}n = \Big\{ \bigcup (M_i : M_j) : |i-j| > 1, (i,j) \neq (0,K) \Big\}$.
- Arcos de la cortadura a los de $\mathcal{C} = (M_0 : M_K) \cup (M_K : M_0)$.

Teorema 4.4.1

Las desigualdades K-C (4.8) *son válidas para el DGRP(G).*

Demostración: Sea x un tour para el DGRP en $G_{\mathcal{C}}$ de c-coste mínimo. Veamos que su c-coste es exactamente c_0. Si x usa un arco interno, por ejemplo de M_i a M_j

(con c-coste $|i-j|$), podemos sustituir este arco por los $|i-j|$ arcos del camino que va de M_i a M_j formado con arcos externos (con c-coste 1) y obtenemos otro tour con el mismo c-coste. Por lo tanto, podemos suponer que x no usa arcos internos.

Como $\big|(M_0 : M_K)_R\big| = \big|(M_K : M_0)_R\big|$ y todos los tours para el DGRP en $G_{\mathcal{C}}$ han de pasar por todos los vértices de $G_{\mathcal{C}}$, tenemos dos tipos posibles de tours de c-coste mínimo que no usan arcos internos:

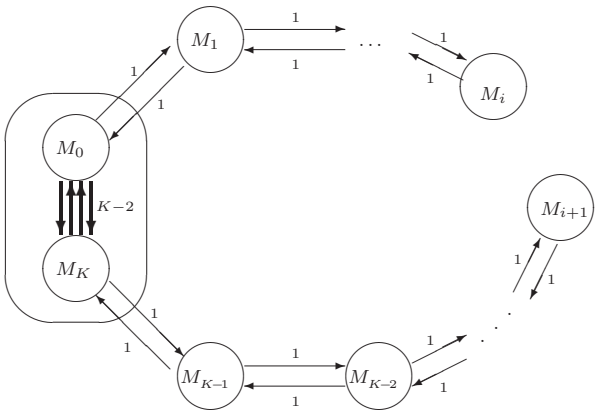

Figura 4.5: *Tours para el* DGRP *en $G_{\mathcal{C}}$ que cumplen la desigualdad K-C con igualdad.*

- Tours que recorren los arcos requeridos entre M_0 y M_k exactamente una vez y recorren una vez los dos arcos opuestos externos no requeridos entre cada M_j y M_{j+1}, para todo $j \in \{0, .., K-1\}$, excepto uno de ellos (ver Figura 4.5). Estos tours tienen un c-coste de $(K-2)\Big(|(M_0 : M_K)_R| + |(M_K : M_0)_R|\Big) + 2(K-1) = c_0$.

- Tours que recorren el camino $M_0, M_1, \ldots, M_K$, recorren todos los arcos requeridos entre M_0 y M_k y uno más (bien un arco no requerido o uno requerido de modo extra, ver Figura 4.6). Estos tours tienen un c-coste de $(K-2)\Big(|(M_0 : M_K)_R| + |(M_K : M_0)_R| + 1\Big) + K = c_0$. ♦

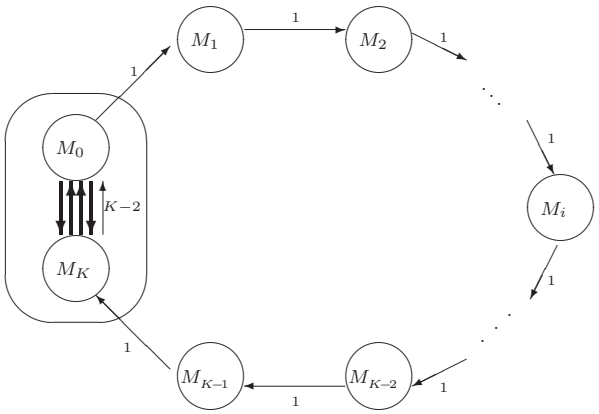

Figura 4.6: *Tours para el* DGRP *en* $G_{\mathcal{C}}$ *que cumplen la desigualdad K-C con igualdad.*

Teorema 4.4.2

Las desigualdades K-C (4.8) *inducen faceta de* $DGRP(G)$.

Demostración: Veamos primero que definen faceta de DGRP($G_{\mathcal{C}}$). Denotemos por $F(x) \geq c_0$ a la desigualdad K-C. Vamos a construir $|A_{\mathcal{C}}| - |V_{\mathcal{C}}| + 1$ tours afínmente independientes para el DGRP en $G_{\mathcal{C}}$ cumpliendo $F(x) = c_0$.

- Para cada arco interno (M_i, M_j), por ejemplo con $i < j - 1$, podemos generar el tour que utiliza el ciclo $M_i, M_j, M_{j-1}, \ldots, M_i$ y lo completamos con pares de arcos opuestos externos entre conjuntos M_r consecutivos, excepto uno de esos pares, y con los arcos requeridos de $\mathcal{C}$ (ver Figura 4.7).

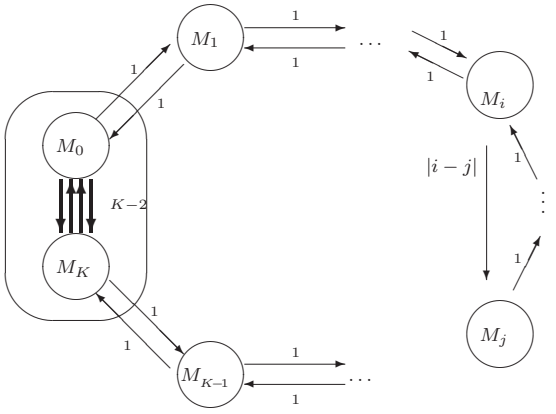

Figura 4.7: *Tours para el* DGRP *en* $G_{\mathcal{C}}$ *que cumplen la desigualdad K-C con igualdad.*

- Por cada arco entre M_0 y M_K, requerido ($\mathcal{C}_R$) o no requerido ($\mathcal{C}_{NR}$) , podemos generar un tour como el de la Figura 4.6 que lo utiliza de modo extra, además de arcos externos y los arcos requeridos de $\mathcal{C}$.

- Para cada par de arcos opuestos externos entre M_i y M_{i+1}, $i \in \{0, .., K-1\}$, generamos un tour como el de la Figura 4.5. Estos tours utilizan cada arco requerido de $\mathcal{C}$ una vez y todos los arcos externos excepto los dos arcos opuestos de (M_i, M_{i+1}).

En total tenemos $(|\mathcal{E}x| - K) + |\mathcal{I}n| + |\mathcal{C}| = |A_{\mathcal{C}}| - |V_{\mathcal{C}}| + 1$ tours, todos satisfaciendo $F(x) = c_0$. Si expresamos estos tours como filas y los arcos como columnas, obtenemos la matriz de bloques siguiente:

$\mathcal{I}n$	$\mathcal{C}_{NR}$	$\mathcal{C}_R$	$\mathcal{E}x$
I	0	1	$*$
0	I	1	$*$
0	0	1+I	B

donde $B = \begin{pmatrix} 0 & 0 & 1 & 1 & .. & 1 & 1 \\ 1 & 1 & 0 & 0 & .. & 1 & 1 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots \\ 1 & 1 & 1 & 1 & .. & 0 & 0 \end{pmatrix}$,

I representa la matriz identidad, 0 y 1 representan, respectivamente, bloques de ceros y de unos y $*$ representa un bloque con entradas cualesquiera. Es fácil ver que esta matriz es de rango completo, por lo que la desigualdad K-C induce faceta de DGRP($G_{\mathcal{C}}$) y, aplicando el Teorema 4.3.2 de "lifting", también induce faceta de DGRP(G). ♦

4.4.2. Desigualdades Honeycomb

Las desigualdades Honeycomb fueron propuestas originalmente para el GRP (no dirigido) por Corberán y Sanchis (1998), y fueron adaptadas al MGRP en Corberán, Mejía y Sanchis(2005) y al WGRP en Corberán, Plana y Sanchis (2007). En esta sección, presentamos las desigualdades Honeycomb para el DGRP.

Las desigualdades Honeycomb pueden verse como una generalización de las K-C en el sentido siguiente. En una K-C, un R-set (o una agrupación de R-sets) se divide en dos conjuntos, M_0 y M_K. En las Honeycomb que presentamos aquí, un R-set (o una agrupación de R-sets) se divide en un número $L \geq 2$ de conjuntos (en el caso $L = 2$, tenemos una desigualdad K-C).

Una configuración Honeycomb está definida por una partición del conjunto de vértices $V = V_R \cup V_{NR}$, $\mathbf{B} = \{B_1, ..., B_L, B_{L+1}, ..., B_K\}$, con $3 \leq K-L+1 \leq p$ (recordemos que p es el número de R-sets de G), $2 \leq L \leq K$ y tal que los subgrafos inducidos $G(B_i)$ son fuertemente conexos. Suponemos que cada R-set V_i está contenido en $B_1 \cup \cdots \cup B_L$ o en alguno de los conjuntos B_r, $r = L+1, ..., K$ y, recíprocamente, que cada uno de estos conjuntos contiene a algún R-set V_i (así, todos los nodos del grafo de configuración son requeridos). Por lo tanto, hemos dividido un R-set (o un grupo de R-sets) en L trozos $B_1, \ldots, B_L$. Suponemos que el grafo que tiene como vértices a $B_1, \ldots, B_L$ y tiene un arco (B_r, B_q) por cada arco requerido de $(B_r : B_q)_R$, es un grafo par y fuertemente conexo (ver Figura 4.8). Esta condición es la correspondiente a $\left|(M_0 : M_K)_R\right| = \left|(M_K : M_0)_R\right|$ para las K-C.

Sea T un conjunto de pares de arcos opuestos no requeridos tal que el grafo no dirigido con conjunto de nodos $\mathbf{B}$ y que tiene una arista (B_r, B_q) por cada par de arcos opuestos de T, es un árbol con grado de incidencia 1 para los nodos $B_1, \ldots, B_L$ y grado de incidencia mayor que 1 para los nodos $B_{L+1}, \ldots, B_K$. Para cada par de nodos B_r, B_q, denotemos por $d(B_r, B_q)$ al número de arcos del único camino en el grafo $(\mathbf{B}, T)$ uniendo B_r con B_q. Supondremos que $d(B_r, B_q) \geq 3$ para todo $r, q \in \{1, ..., L\}$.

Definimos el vector c de la configuración Honeycomb del modo siguiente:

- $c(B_r, B_q) = d(B_r, B_q) - 2$, para todo B_r, B_q con $r, q \leq L$,
- $c(B_r, B_q) = d(B_r, B_q)$, en otro caso.

La partición $\mathbf{B} = \{B_1, ..., B_L, B_{L+1}, ..., B_K\}$ y el vector c definen el grafo de configuración $G_{\mathcal{C}}$ cuyo esqueleto se muestra en la Figura 4.8 con valores $L = 4$ y $K = 9$. En esta figura, los arcos de T están representados con líneas finas y los arcos requeridos (entre conjuntos B_r y B_q con $r, q \leq L$) con línea más gruesa. Una configuración Honeycomb parece un panal de abejas (Honeycomb) en el que cada celda es un configuración K-C definida por un par de nodos B_r, B_q, con $r, q \leq L$ y por el único camino que une B_r con B_q usando arcos de T. El número que aparece al lado de cada arco es el correspondiente valor de c, es decir, es el coste del arco en el grafo de configuración y también el coeficiente en la desigualdad de las variables asociadas a los arcos en el grafo original.

Llamemos $\mathcal{A}_R$ al conjunto de los arcos requeridos en el grafo de configuración. Definimos $c_0 = 2(K - L) + \sum_{a \in \mathcal{A}_R} c_a$, que es el c-coste del tour para el DGRP de mínimo c-coste en $G_{\mathcal{C}}$. El término $2(K-L)$ es el c-coste mínimo de conectar $K-L+1$ nodos de la configuración asociados a R-sets diferentes, mientras que el segundo término es la suma del c-coste de los arcos requeridos en el grafo de configuración (ver la demostración del Teorema 4.4.3). Por ejemplo, en la configuración de la Figura 4.8, tenemos que $c_0 = 2(9 - 4) + 4 + 2 + 1 + 1 = 18$. Así, la desigualdad Honeycomb correspondiente a esta configuración es:

$$\sum_{(i,j) \in A} c_{ij} x_{ij} \geq 2(K - L) + \sum_{a \in \mathcal{A}_R} c_a, \tag{4.9}$$

donde $c_{ij} = c(B_r, B_q)$ para todo $(i, j) \in (B_r : B_q)$, $c_{ij} = 0$ para todo $(i, j) \in A(B_r)$.

Teorema 4.4.3

Las desigualdades Honeycomb (4.9) *son válidas para el DGRP(G).*

Demostración: Denotemos por $F(x) \geq c_0$ a la desigualdad Honeycomb (4.9) sobre el grafo configuración $G_{\mathcal{C}}$. Consideremos, sin pérdida de generalidad, que los arcos

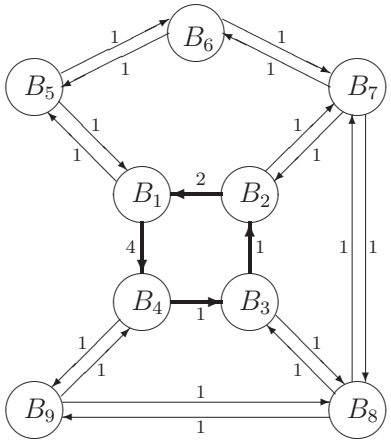

Figura 4.8: *Una configuración Honeycomb con $L = 4$ y $K = 9$.*

no requeridos de $G_{\mathcal{C}}$ forman un grafo completo. Sea x un tour para el DGRP en $G_{\mathcal{C}}$. Tenemos que demostrar que $F(x) \geq 2(K-L) + \sum_{a \in \mathcal{A}_R} c_a$. Notar que x recorre todos los arcos requeridos de $G_{\mathcal{C}}$ al menos una vez, lo que supone un c-coste de $\sum_{a \in \mathcal{A}_R} c_a$. Así, tenemos que ver que el c-coste de los arcos no requeridos más el c-coste de los arcos requeridos recorridos de modo extra por x es, al menos, $2(K-L)$.

Si x recorre un arco (B_r, B_q) que no está en T ni se cumple que $r, q \leq L$, podemos reemplazarlo por los arcos del único camino en el grafo $(\mathbf{B}, T)$ que va de B_r a B_q, y obtendríamos otro tour para el DGRP en $G_{\mathcal{C}}$ con el mismo c-coste, pues $c(B_r, B_q) = d(B_r, B_q)$. Si x recorre de modo extra los arcos de un ciclo cualquiera entre los nodos $B_1, \ldots, B_L$, podemos eliminar esos arcos y obtendríamos otro tour para el DGRP en $G_{\mathcal{C}}$ con c-coste menor. Si x recorre de modo extra dos arcos adyacentes entre los nodos $B_1, \ldots, B_L$, digamos (B_r, B_q) y (B_q, B_s), reemplazándolos por el arco (B_r, B_s) obtendríamos otro tour para el DGRP en $G_{\mathcal{C}}$ con c-coste menor o igual.

Por lo tanto, podemos suponer que x recorre, además de los arcos requeridos, sólo arcos de T y algunos arcos aislados entre los nodos $B_1, \ldots, B_L$ de modo extra. Si x recorre de modo extra un arco, por ejemplo, entre conjuntos B_r y B_q con

$r, q \leq L$, entonces x también recorre los arcos del camino de T que une B_q con B_r. Reemplazando este arco por los arcos en el único camino de T que une B_r con B_q obtenemos un tour para el DGRP en $G_{\mathcal{C}}$ con un c-coste 2 unidades mayor. Pero podemos eliminar los dos arcos opuestos incidentes con B_r (por ejemplo) y obtenemos otro tour para el DGRP en $G_{\mathcal{C}}$ con el mismo c-coste que x.

Así, podemos finalmente considerar que x recorre, además de una vez cada arco requerido, sólo arcos de T. El tour x tiene que conectar al menos uno de los nodos $B_1, ..., B_L$ (correspondientes a un mismo R-set) y todos los nodos $B_{L+1}, ..., B_K$ (correspondientes a distintos R-sets). Por la estructura de árbol de T, para conectar nodos correspondientes a $K-L+1$ R-sets hacen falta, al menos, $K-L$ pares de arcos opuestos de T, lo que tiene un c-coste de $2(K-L)$. Por lo tanto, tenemos que $F(x) \geq 2(K-L) + \sum_{a \in \mathcal{A}_R} c_a$ y la desigualdad Honeycomb (4.9) es válida para DGRP($G_{\mathcal{C}}$) y, por lo tanto, también lo es para DGRP(G). ♦

Teorema 4.4.4

Las desigualdades Honeycomb (4.9) *definen faceta de DGRP(G).*

Demostración: Veamos primero que definen faceta de DGRP($G_{\mathcal{C}}$). Denotemos por $F(x) \geq c_0$ a la desigualdad Honeycomb. Vamos a construir $l = |A_{\mathcal{C}}| - |V_{\mathcal{C}}| + 1 = |A_{\mathcal{C}}| - K + 1$ tours afínmente independientes para el DGRP en $G_{\mathcal{C}}$ cumpliendo $F(x) = c_0$.

Denotemos por $V_L = \{B_1, \ldots, B_L\}$ y por $V_K = \{B_{L+1}, \ldots, B_K\}$. Como el grafo que tiene como conjunto de vértices a V_L y tiene un arco (B_r, B_q) por cada arco requerido de $(B_r : B_q)$ es un grafo euleriano, podemos encontrar un ciclo que visite cada nodo de V_L y cada arco requerido, con c-coste $\sum_{a \in \mathcal{A}_R} c_a$. Sea $x^R \in R^{|A_{\mathcal{C}}|}$ el vector de incidencia de ese ciclo. A partir de x^R vamos a construir cinco tipos de tours para el DGRP en $G_{\mathcal{C}}$ asociados a los distintos arcos del grafo de configuración. Denotemos por T_1 al conjunto de arcos resultante de eliminar de T los arcos incidentes con nodos

de V_L y por $G(T_1)$ al subgrafo inducido por T_1.

- **Tipo 1:** Para cada par de arcos opuestos $(B_r, B_q), (B_q, B_r) \in T \backslash T_1$, construimos el tour que recorre estos dos arcos y todos los arcos de T_1, al que le sumamos x^R. Así obtenemos L tours con c-coste c_0, similares al mostrado en la Figura 4.9a para $(B_r, B_q) = (B_1, B_5)$.

- **Tipo 2:** Fijemos un par de arcos opuestos $(B_r, B_q), (B_q, B_r) \in T \backslash T_1$. Para cada arco $(B_i, B_j) \notin T$, con $B_i, B_j \in V_K$, construimos un tour que recorra el arco (B_i, B_j), los arcos del camino de T_1 de B_j a B_i, el resto de los arcos de T_1 y los arcos $(B_r, B_q), (B_q, B_r)$. Un tour similar se muestra en la Figura 4.9b, con $(B_r, B_q) = (B_1, B_5)$ y $(B_i, B_j) = (B_7, B_5)$. Podemos construir así $|A_k| - |T_1| = |A_k| - (2(K-1) - 2L)$ tours, todos con c-coste c_0.

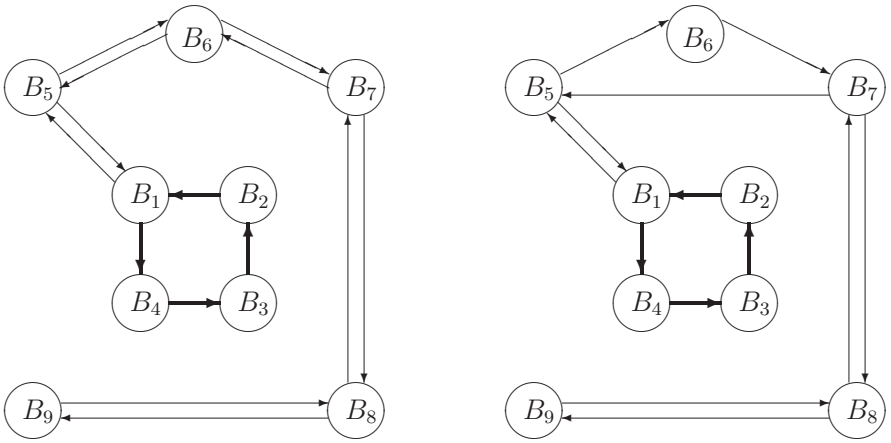

Figura 4.9: *Tours de tipo 1 y tipo 2 que aparecen en la demostración del Teorema 4.4.4.*

- **Tipo 3:** Para cada arco $(B_r, B_q) \in \Big((V_L : V_K) \cup (V_K : V_L)\Big) \setminus T$ construimos un tour recorriendolo y recorriendo el camino de B_q a B_r en T, similar al que se muestra en la Figura 4.10a para $(B_r, B_q) = (B_6, B_1)$. Podemos contruir así $|(V_L : V_K)| \cup |(V_K : V_L)| - 2L$ tours, todos con c-coste c_0.

- **Tipo 4:** Para cada arco $(B_r, B_q) \in G(V_L)$, construimos un tour recorriendo

este arco, los arcos del camino en T de B_q a B_r y todos los arcos restantes de T_1. En la Figura 4.10b se muestra uno de estos $|A_{V_L}|$ tours con c-coste c_0 para $(B_r, B_q) = (B_2, B_1)$.

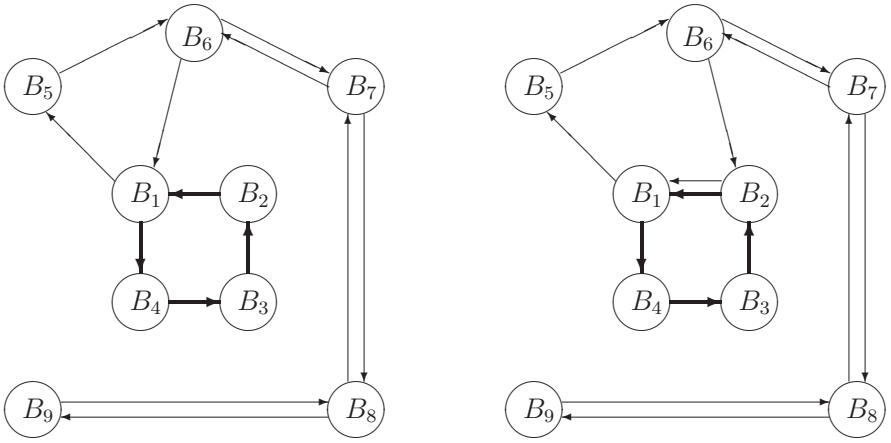

Figura 4.10: *Tours de tipo 3 y tipo 4 que aparecen en la demostración del Teorema 4.4.4.*

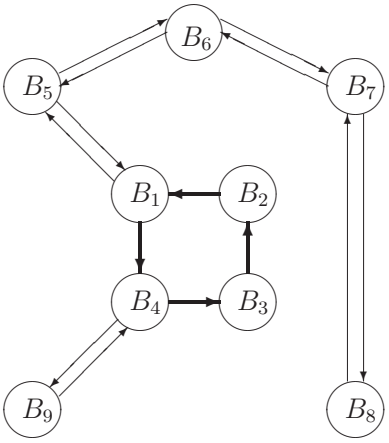

Figura 4.11: *Tours de tipo 5 que aparecen en la demostración del Teorema 4.4.4.*

- **Tipo 5:** Para cada par de arcos opuestos $(B_r, B_q), (B_q, B_r) \in T_1$, sean G^1 y G^2 los dos subgrafos en que queda dividido $G(T_1)$ al eliminar estos dos arcos. Escogemos dos pares de arcos (B_{i_0}, B_{j_0}),(B_{j_0}, B_{i_0}) con $B_{i_0} \in G_1$ y $B_{j_0} \in V_L$ y (B_{i_1}, B_{j_1}),(B_{j_1}, B_{i_1}) con $B_{i_1} \in G_2$ y $B_{j_1} \in V_L$. Ahora el tour estará formado por todos los arcos de $T_1 \setminus \{(B_r, B_q), (B_q, B_r)\}$ más los ar-

cos $\{(B_{i_0}, B_{j_0}), (B_{j_0}, B_{i_0}), (B_{i_1}, B_{j_1}), (B_{j_1}, B_{i_1})\}$. Un tour similar se muestra en la figura 4.11, con $(B_r, B_q) = (B_8, B_9)$, $(B_{i_0}, B_{j_0}) = (B_1, B_5)$ y $(B_{i_1}, B_{j_1}) = (B_4, B_9)$. Obtenemos así $|T_1| = (k-1) - L$ tours, todos con c-coste c_0.

Tenemos un total de $L + |A_{V_k}| - 2(K-1) + 2L + |V_K : V_L| + |V_K : V_L| - 2L + |A_{V_L}| + (K-1) - L = |A_{\mathcal{C}}| - K + 1$ tours para el DGRP en $G_{\mathcal{C}}$ cumpliendo la desigualdad con igualdad. Si expresamos estos tours como filas de una matriz y les restamos el vector x^R obtenemos la matriz de la Figura 4.12.

$$
\begin{array}{c}
 \\
\text{Tipo 5} \\
\text{Tipo 1} \\
\text{Tipo2} \\
\text{Tipo 3} \\
\text{Tipo 4}
\end{array}
\begin{array}{c}
\begin{array}{ccccc}
T \setminus T_1 & T_1 & A(V_K) \setminus T & (V_L : V_K) \setminus T & A(V_L)
\end{array} \\
\left(\begin{array}{c|c|c|c|c}
* & A & 0 & 0 & 0 \\ \hline
B & 1 & 0 & 0 & 0 \\ \hline
* & * & I & 0 & 0 \\ \hline
* & * & 0 & I & 0 \\ \hline
* & * & 0 & 0 & I
\end{array}\right)
\end{array}
$$

$$
A = \begin{pmatrix} 0\,0\,1\,1\,\ldots\,1\,1 \\ 1\,1\,0\,0\,\ldots\,1\,1 \\ \vdots \quad\quad \vdots \\ 1\,1\,1\,1\,\ldots\,0\,0 \end{pmatrix} \text{ y } \quad B = \begin{pmatrix} 1\,1\,0\,0\,\ldots\,0\,0 \\ 0\,0\,1\,1\,\ldots\,0\,0 \\ \vdots \quad\quad \vdots \\ 0\,0\,0\,0\,\ldots\,1\,1 \end{pmatrix}.
$$

Figura 4.12: *Matrices usadas en el teorema 4.4.4*

Para ver que son afínmente independientes solo queda comprobar que la matriz de la Figura 4.13 tiene rango completo. Supongamos que existe una combinación lineal de sus filas, con coeficientes $\alpha_1, .., \alpha_s, \beta_1, ..., \beta_{\frac{r}{2}}$, igual al vector nulo. Tendríamos las siguientes ecuaciones:

$$\begin{pmatrix} * & A \\ B & 1 \end{pmatrix} = \left(\begin{array}{c|c} \begin{matrix} a_{11}\ a_{12}\ ...\ a_{1r} \\ a_{21}\ a_{22}\ ...\ a_{2r} \\ \vdots \qquad \vdots \\ a_{s1}\ a_{s2}\ ...\ a_{sr} \end{matrix} & \begin{matrix} 00\ 11\ ...\ 11 \\ 11\ 00\ ...\ 11 \\ \vdots \quad \vdots \\ 11\ 11\ ...\ 00 \end{matrix} \\ \hline \begin{matrix} 11\ 00\ ...\ 00 \\ 00\ 11\ ...\ 00 \\ \vdots \quad \vdots \\ 00\ 00\ ...\ 11 \end{matrix} & 1 \end{array}\right)$$

Figura 4.13: *Matriz usada en el teorema 4.4.4*

$$\sum_{s}^{i=1} \alpha_i a_{i1} + \beta_1 = 0 \qquad (1)$$

$$\sum_{s}^{i=1} \alpha_i a_{i2} + \beta_1 = 0 \qquad (2)$$

$$\vdots$$

$$\sum_{i=1}^{s} \alpha_i a_{i1} + \beta_{\frac{r}{2}} = 0 \qquad (r)$$

$$\sum_{i=2}^{s} \alpha_i + \sum_{i=1}^{r/2} \beta_{\frac{r}{2}} = 0 \qquad (r+1)$$

$$\sum_{i=1, i\neq 2}^{s} \alpha_i \sum_{i=1}^{r/2} \beta_{\frac{r}{2}} = 0 \qquad (r+2)$$

$$\vdots$$

$$\sum_{i=2}^{s} \alpha_i + \sum_{i=1}^{r/2} \beta_i = 0 \qquad (r+s)$$

Restando las ecuaciones $(r+1), (r+2)$ obtenemos que $\alpha_1 = \alpha_2$ y, en general, si restamos $(r+i)$ y $(r+j)$ se obtiene $\alpha_i = \alpha_j$. Por lo tanto, tenemos que $\alpha_1 = \alpha_2 =$

$... = \alpha_s = -\frac{1}{s-1}\sum_{i=1}^{r/2}\beta_i$, y podemos reescribir las ecuaciones desde (1) a la (r) como

$$-\frac{1}{s-1}\sum_{i=1}^{r/2}\beta_i\sum_{i=1}^{s}a_{ij} + \beta_{\lfloor \frac{j+1}{2} \rfloor} = 0$$

Sumando las ecuaciones de la (1) a (r) se obtiene:

$$-\frac{1}{s-1}\sum_{i=1}^{r/2}\beta_i\sum_{i=1}^{s}\sum_{j=1}^{r}a_{ij} + 2\sum_{i=1}^{r/2}\beta_i = \sum_{i=1}^{r/2}\beta_i(2 - \frac{1}{s-1}\sum_{i=1}^{s}\sum_{j=1}^{r}a_{ij}) = 0$$

Si $2 - \frac{1}{s-1}\sum_{i=1}^{s}\sum_{j=1}^{r}a_{ij} = 0$ entonces $\sum_{i=1}^{s}\sum_{j=1}^{r}a_{ij} = 2(s-1)$. Pero por la estructura de los tours de tipo 5 tenemos que $\sum_{j=1}^{r}a_{ij} = 4$, teniendo así $4s = 2(s-1)$ y $s = -1$ lo que es una contradicción, por lo que $\sum_{i=1}^{r/2}\beta_i$ tiene que ser igual a cero. Así pues, $\alpha_1 = \alpha_2 = ... = \alpha_s = 0$ y sustituyendo desde la ecuación (1) a la (r) tenemos que $\beta_1 = ... = \beta_{\frac{r}{2}} = 0$. Por lo tanto, los tours son afínmente independientes y las desigualdades Honeycomb (4.9) inducen faceta de DGRP($G_{\mathcal{C}}$). Finalmente, por el Teorema 4.3.2 de lifting, también inducen faceta de DGRP(G). ♦

Hemos estudiado el caso donde un único R-set (o un grupo de R-sets) es dividido en varios nodos, pero podemos definir las Honeycomb de manera más general donde varios R-set son divididos. Consideremos una partición de V en K conjuntos de vertices $\{A_1, A_2, \ldots, A_L, A_{L+1}, \ldots, A_K\}$, $3 \leq K \leq p$, $1 \leq L \leq K$, de manera que cada V_j, $1 \leq j \leq p$, es contenido exactamente en un A_i y el grafo inducido por $G(A_i)$, $i = 1, 2, \ldots, K$, sea fuertemente conexo. Supongamos una nueva partición de cada conjunto A_i, $i = 1, 2, \ldots, L$, en $\gamma_i \geq 2$ subconjuntos, $A_i = B_i^1 \cup B_i^2 \cup \ldots \cup B_i^{\gamma_i}$, satisfaciendo las siguientes condiciones:

- Cada B_i^j es incidente con un número par de arcos requeridos de manera que el grafo cuyos nodos son los B_i^j y cuyos arcos son los arcos requeridos entre B_j^l y B_j^r, sea fuertemente conexo y simétrico.
- Los grafos inducidos por $G(B_i^j), j = 1, 2, \ldots, \gamma_i$, son fuertemente conexos.

Denotamos por $B_i^0 = A_i$, $i = L+1, \ldots, K$. Con ello tenemos la siguiente partición de V:

$$\mathbf{B} = \{B_1^1, B_1^2, \ldots, B_1^{\gamma_1}, B_2^1, B_2^2, \ldots, B_2^{\gamma_2}, \ \ldots \ , B_L^1, B_L^2, \ldots, B_L^{\gamma_L}, \ B_{L+1}^0, \ldots, B_K^0\}$$

Esta partición $\mathbf{B}$ define un grafo configuración $G_{\mathcal{C}} = (\mathbf{B}, \mathbf{A})$ con conjunto de nodos $\mathbf{B}$ y un conjunto de arcos $\mathbf{A}$ formados por los arcos requeridos (B_r^i, B_q^j) por cada arco requerido $a \in (B_r^i : B_q^j)_R$ y un arco no requeridos (B_r^i, B_q^j) entre cada par de nodos $B_r^i, \ B_q^j$ de manera que $A_{NR}(B_r^i : B_q^j) \neq \emptyset$.

Como en el caso para un R-set, supongamos que hay un conjunto T de pares de arcos no requeridos en $G_{\mathcal{C}}$ uniendo nodos correspondientes a diferentes A_j, $j = 1, 2, \ldots K$, de manera que el grafo no dirigido con nodos el conjunto $\mathbf{B}$ y teniendo una arista (B_i^j, B_p^q) por cada par de arcos opuestos (B_i^j, B_p^q), (B_p^q, B_i^j) en T, es un árbol generador. Entonces, por cada par de nodos B_i^j, B_p^q en $\mathbf{B}$, $d(B_i^j, B_p^q)$ denota el número de arcos en el camino $(\mathbf{B}, T)$ uniendo B_i^j y B_p^q. Asumiremos además que se cumplen la condición de $d(B_i^j, B_i^q) \geq 3 \quad \forall\ i = 1, \ldots, L$ y $\forall\ j \neq q$.

El grafo $(\mathbf{B}, T)$ que define el esqueleto de la configuración se puede ver en la Figura 4.14, donde los arcos en T están representados por líneas más finas y los requeridos por líneas más gruesas. Supondremos, además, que para todo arco (B_r^i, B_q^j) no hay más de un nodo asociado con el mismo A_s en el camino en $(\mathbf{B}, T)$ de B_r^i a B_q^j. Definimos los costes de los arcos en la configuración de la siguiente manera:

Para los arcos (B_q^i, B_q^j): $c(B_q^i, B_q^j) = d(B_q^i, B_q^j) - 2$.

Para los restantes arcos (B_r^i, B_q^j), $r \neq q$: $c(B_q^i, B_q^j) = d(B_q^i, B_q^j)$.

Llamaremos desigualdad honeycomb a:

$$\sum_{a \cup A} c_a x_a \geq 2(K-1) + \sum_{a \in A_R} c_a \tag{4.10}$$

donde $c_a = c(B_r^i, B_q^j)\ \forall a \in (B_r^i : B_q^j), \quad c_a = 0\ \forall a \in A(B_r^i)$.

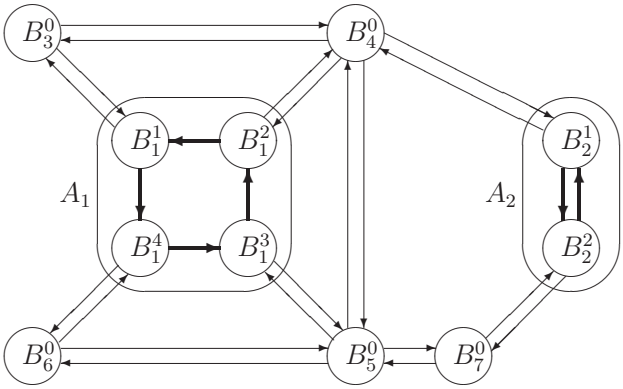

Figura 4.14: *Configuración Honeycomb*

4.4.3. Desigualdades Path-Bridge

Como las desigualdades Honeycomb, las desigualdades Path-Bridge son una generalización de las desigualdades K-C. Sin embargo, la generalización es en una dirección diferente y ninguna de las dos familias contiene a la otra. Estas desigualdades fueron introducidas por Letchford (1997) para el problema General de Rutas por Arcos, GRP, no dirigido. Están inspiradas en las desigualdades "Path" propuestas por Cornuejols, Fonlupt y Naddef (1985) para el Graphical Traveling Salesman Problem, GTSP. Las desigualdades Path-Bridge fueron adaptadas al MGRP en Corberán, Romero y Sanchis (2003) y al WGRP en Corberán, Plana y Sanchis (2007). En esta sección, presentamos las desigualdades Path-Bridge para el DGRP.

Una configuración **Path-Bridge** viene definida por dos enteros P (el número de "paths", caminos en inglés) y B (el número de arcos en el "bridge", puente en inglés) con $P \geqslant 1$, $B \geqslant 0$, $P + B \geqslant 3$ e impar, por P enteros $n_i \geqslant 2$, $i = 1, \ldots, P$ y por una partición de $V = V_R \cup V_{NR}$ en subconjuntos

$$\left\{M_0, M_Z, M_j^i : \ i = 1, \ldots, P, \ j = 1, \ldots, n_i\right\},$$

de modo que cada R-set V_i está contenido en uno de los subconjuntos $M_0 \cup M_Z$ o M_j^i, $i = 1, \ldots, P, \ j = 1, \ldots, n_i$ y, recíprocamente, cada uno de estos subconjuntos

contiene, al menos, un R-set (así, todos los nodos del grafo de configuración son requeridos). Por simplicidad, identificamos M_0^i con M_0 y $M_{n_{i+1}}^i$ con M_Z para todo i. Suponemos que los subgrafos inducidos $G(M_j^i),\ i = 1, \ldots, P,\ j = 0, 1, \ldots, n_i{+}1$, son fuertemente conexos y que los conjuntos $(M_j^i : M_{j+1}^i)$ y $(M_{j+1}^i : M_j^i),\ i = 1, \ldots, P$, $j = 0, \ldots, n_i$ (que forman los P caminos entre M_0 y M_Z), son no vacíos. Además, asumimos que los arcos requeridos están, bien en los $A(M_j^i)$ o bien entre M_0 y M_Z, y que se cumple que $\left|(M_0 : M_Z)_R\right| + \left|(M_Z : M_0)_R\right| = B$. Estos B arcos requeridos entre M_0 y M_Z constituyen el "bridge" de la configuración. Definimos la función c de costes de esta configuración como:

- $c(M_0 : M_Z) = c(M_Z : M_0) = 1$.
- $c(M_j^i, M_q^i) = \dfrac{|j-q|}{n_i - 1},\ \forall\, j, q \in \{0, 1, \ldots, n_i + 1\},\ \ 0 < |j - q| < n_i + 1$.
- $c(M_j^i, M_q^r) = \dfrac{1}{n_i - 1} + \dfrac{1}{n_r - 1} + \left|\dfrac{j-1}{n_i - 1} - \dfrac{q-1}{n_r - 1}\right|,\ \forall\, i, r \in \{1, \ldots, P\},\ i \neq r,$ $j \in \{1, \ldots, n_i\},\ q \in \{1, \ldots, , n_r\}$.

La partición $\mathbf{B} = \left\{M_0, M_Z, M_j^i,\ \ i = 1, \ldots, P,\ \ j = 1, \ldots, n_i\right\}$ y la función c definen un grafo de configuración $G_{\mathcal{C}}$ cuyo esqueleto se muestra en la Figura 4.15. En este grafo hay P caminos de M_0 a M_Z, cada uno de ellos formado por $n_i + 2$ nodos y $n_i + 1$ pares de arcos opuestos. La desigualdad Path-Bridge correspondiente a esta configuración es:

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \geq B + 1 + \sum_{i=1}^{P} \frac{n_i + 1}{n_i - 1} \tag{4.11}$$

donde $c_{ij} = c(M_q^r, M_s^t)$ para todo $(i,j) \in (M_q^r, M_s^t)$, $c_{ij} = 0$ para todo $(i,j) \in A(M_q^r)$.

Llamamos desigualdad Path-Bridge n-regular al caso particular en el que $n_i = n$ para todo i. En este caso, podemos eliminar denominadores multiplicando la desigualdad (4.11) por $n - 1$, y la desigualdad Path-Bridge n-regular puede escribirse como:

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \geq (n-1)(B+1) + Pn + P \tag{4.12}$$

donde:

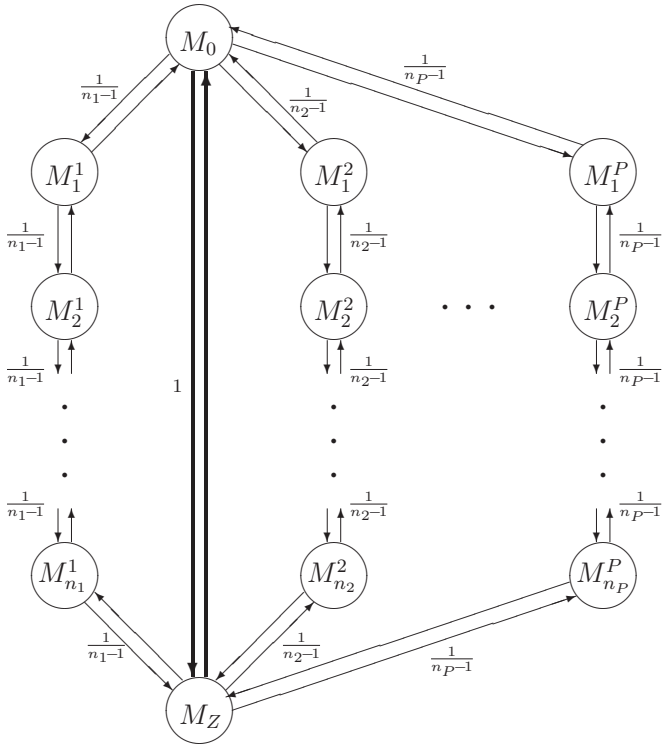

Figura 4.15: *Grafo de configuración de una desigualdad Path-Bridge.*

- $c(M_0 : M_Z) = c(M_Z : M_0) = n - 1.$
- $c(M_j^i : M_q^i) = |j-q|, \ \ \forall j, q \in \{0, 1, \ldots, n+1\}, \ 0 < |j-q| < n+1, \ i = 1, \ldots, P.$
- $c(M_j^i : M_q^r) = |j - q| + 2, \ \forall j, q \in \{0, 1, \ldots, n+1\}, \ i \neq r, \ i, r \in \{1, \ldots, P\}.$

En el caso $P = 1$, la desigualdad Path-Bridge es una desigualdad K-C (con $K = n + 1$). En el caso $B = 0$, el puente está vacío y la desigualdad Path-Bridge es una desigualdad "Path" para el GATSP (Chopra y Rinaldi, 1996), que es la versión dirigida de una desigualdad "Path" para el GTSP (Cornuejols, Fonlupt, y Naddef, 1985).

Teorema 4.4.5

Las desigualdades Path-Bridge (4.11) *son válidas para DGRP(G).*

Demostración: Obvia, puesto que estas desigualdades Path-Bridge son válidas para el MGRP, del que el DGRP es un caso particular. ♦

Aunque son válidas en general, sólo hemos podido demostrar que las desigualdades Path-Bridge inducen faceta del DGRP en el caso P impar:

Teorema 4.4.6

Las desigualdades Path-Bridge (PB) (4.11) *con P impar inducen faceta de DGRP(G) si* $\left|(M_0 : M_Z)_R\right| = \left|(M_Z : M_0)_R\right|$.

Demostración: Si $P = 1$ la desigualdad Path-Bridge es una K-C, que induce faceta de DGRP(G) (Teorema 4.4.2). Así, podemos suponer que $P \geq 3$ (e impar). Veamos primero que definen faceta de DGRP($G_{\mathcal{C}}$). Denotemos por $F(x) \geq c_0$ a la desigualdad Path-Bridge. Vamos a construir $|A_{\mathcal{C}}| - |V_{\mathcal{C}}| + 1$ tours para el DGRP en $G_{\mathcal{C}}$ afínmente independientes cumpliendo $F(x) = c_0$. Consideremos el GATSP en el grafo $G_{\mathcal{C}}$, es decir, consideremos que sólo hemos de pasar por los vértices del grafo $G_{\mathcal{C}}$. Chopra y Rinaldi (1996) demostraron que las desigualdades "Path" (P impar) inducen faceta del GATSP, por lo que podemos encontrar $\dim(\text{GATSP}(G_{\mathcal{C}})) = |A_{\mathcal{C}}| - |V_{\mathcal{C}}| + 1$ tours para el GATSP en $G_{\mathcal{C}}$ afínmente independientes cumpliendo $F(x) = c_0 - B$ (ya que en el GATSP no tenemos arcos requeridos). Sean $x^1, \ldots, x^k$ estos vectores. Como $\left|(M_0 : M_Z)_R\right| = \left|(M_Z : M_0)_R\right|$, si x^R es el vector de incidencia de los B arcos requeridos, sumando x^R a cada uno de los x^i anteriores, $i = 1, \ldots, k$, se obtienen k tours para el DGRP($G_{\mathcal{C}}$) afínmente independientes que cumplen $F(x) = c_0$, con $k = \dim(DGRP(G_{\mathcal{C}}))$. Por lo tanto, la desigualdad induce faceta de DGRP($G_{\mathcal{C}}$) y, por el Teorema 4.3.2 de 'lifting", también induce faceta de DGRP(G). ♦

Para el MGRP y el WGRP hay otras familias de desigualdades, K-C_{02}, Path-Bridge_{02} y Honeycomb_{02}, que inducen faceta del correspondiente poliedro de soluciones. Sin embargo, para el DGRP estas desigualdades no inducen faceta porque están dominadas por las desigualdades descritas arriba, K-C, Path-Bridge, y Honeycomb, respectivamente.

4.5. Poliedro III: 2-Path-Bridge Asimétricas

Las desigualdades vistas en la sección anterior son simétricas en sus coeficientes, es decir, el coeficiente de las variables asociadas a los arcos en, digamos, $(M_j^i : M_q^r)$ es el mismo que el de las variables asociadas a los arcos en $(M_q^r : M_j^i)$. Además, las condiciones que exigimos a los arcos requeridos implican también una cierta simetría. Por ejemplo, para que las desigualdades Path-Bridge induzcan faceta, exigimos que se cumpla que $\left|(M_0 : M_Z)_R\right| = \left|(M_Z : M_0)_R\right|$, lo cual implica que el número de arcos en el "bridge" sea par y que el número de caminos entre M_0 y M_Z, P, sea impar.

En esta sección presentamos una nueva familia de desigualdades para el DGRP, las desigualdades 2-Path-Bridge Asimétricas, que llamamos así pues tienen una configuración similar a una desigualdad Path-Bridge con $P = 2$ pero tienen el vector de costes o coeficientes c no simétrico.

Una configuración 2-Path-Bridge Asimétrica está definida por un entero impar B, dos enteros $n_1, n_2 \geq 2$ y una partición de $V = V_R \cup V_{NR}$ en subconjuntos

$$\left\{M_0, M_Z, M_1^1, ..., M_{n_1}^1, M_1^2, .., M_{n_2}^2\right\},$$

de modo que cada R-set V_i esta contenido en en un de los subconjuntos $M_0 \cup M_Z$ ó M_i^j y, recíprocamente, cada uno de estos subconjuntos contiene, al menos, un R-set (así, todos los nodos del grafo de configuración son requeridos). Por simplicidad, identificamos M_0^1 y M_0^2 con M_0 y $M_{n_1+1}^1$ y $M_{n_2+1}^2$ con M_Z. Suponemos que los subgrafos inducidos $G(M_j^i)$ son fuertemente conexos y que los conjuntos $(M_j^i : M_{j+1}^i)$ y $(M_{j+1}^i : M_j^i)$, $i = 1, 2$, $j = 0, \ldots, n_i$ (que forman los dos caminos, que llamaremos P_1 y P_2, entre M_0 y M_Z), son no vacíos. Suponemos que $(M_0 : M_Z)$ contiene B arcos requeridos (que constituyen el "bridge" de la configuración) satisfaciendo que $|A_R(M_0 : M_Z)| = |A_R(M_Z : M_0)| + 1$. Definimos la función c de costes de esta configuración como:

- Para los arcos del "bridge": $c(M_0, M_Z) = 0,\ \ c(M_Z, M_0) = n_1(n_2 - 1)$.
- Para los arcos del camino P_1: $c(M_0, M_1^1) = n_2 - 1$ y $c(M_1^1, M_0) = 0$, mientras que $c(M_j^1, M_{j+1}^1) = 0$ y $c(M_{j+1}^1, M_j^1) = n_2 - 1$ para $j \geq 1$.
- Para los arcos del camino P_2: $c(M_0, M_1^2) = n_1$ y $c(M_1^2, M_0) = 0$, mientras que $c(M_j^2, M_{j+1}^2) = 0$ y $c(M_{j+1}^2, M_j^2) = n_1$ para $j \geq 1$.
- El coste de un arco (si existe) uniendo dos nodos del mismo camino es el coste del camino más corto entre ellos usando arcos del propio camino.
- El coste de los arcos (si existen) uniendo nodos en diferentes caminos se obtiene mediante un proceso llamado "lifting" secuencial: ordenamos estos arcos de manera arbitraria $a_1, \ldots, a_h$ y, para cada $i = 1, \ldots, h$, definimos c_{a_i} como el valor máximo para el que existe un tour para el DGRP en $G_{\mathcal{C}}$ de coste $n_1(n_2 - 1) + n_1 n_2$ que usa únicamente arcos del esqueleto y de $\{a_1, ..., a_i\}$.

La partición $\mathbf{B} = \left\{M_0, M_Z, M_1^1, ..., M_{n_1}^1, M_1^2, .., M_{n_2}^2\right\}$ y la función c definen un grafo de configuración $G_{\mathcal{C}}$ cuyo esqueleto se muestra en la Figura 4.16. En este grafo hay dos caminos, P_1 y P_2, de M_0 a M_Z, cada uno de ellos formado por n_i+2 nodos y n_i+1 pares de arcos opuestos. La desigualdad Path-Bridge Asimétrica correspondiente a esta configuración es:

$$\sum_{(i,j)\in A} c_{ij} x_{ij} \geq n_1(n_2 - 1) + n_1 n_2 \tag{4.13}$$

donde $c_{ij} = c(M_q^r, M_s^t)$ para todo $(i,j) \in (M_q^r, M_s^t)$, $c_{ij} = 0$ para todo $(i,j) \in A(M_q^r)$.

Notar que en el caso particular $n_2 = 1$ la desigualdad (4.13) anterior resulta $n_1 x(\delta^+(M_1^2)) \geq n_1$, que es una desigualdad de conectividad (4.2) con $S = M_1^2$.

Teorema 4.5.1

Las desigualdades 2-Path-Bridge Asimétricas (4.13) *son válidas para el DGRP(G).*

Demostración: Denotemos por $F(x) \geq n_1(n_2 - 1) + n_1 n_2$ a la desigualdad 2-

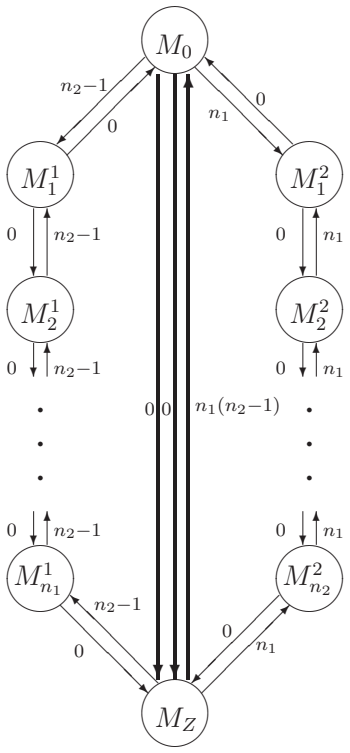

Figura 4.16: *Grafo de configuración de una desigualdad 2-Path-Bridge Asimétrica.*

Path-Bridge Asimétrica (4.13) en el grafo de configuración $G_{\mathcal{C}}$. De la definición de los coeficientes de los arcos que no están en el esqueleto (obtenidos bien por caminos más cortos o bien por "lifting" secuencial), es suficiente probar la validez para cada tour x para el DGRP en $G_{\mathcal{C}}$ que use solamente arcos del esqueleto. Sea x un tour para el DGRP de longitud mínima en el grafo de configuración $G_{\mathcal{C}}$. Si x recorre exactamente una vez cada arco requerido entre M_0 y M_Z, este tour ha de ser similar a alguno de los tours que se muestran en la figura 4.18, todos los cuales cumplen $F(x) = 2n_1(n_2 - 1) + n_1n_2$. Si x recorre de modo extra algún arco de $(M_Z : M_0)$ entonces podemos reemplazar este arco por arcos en el camino $P1$ desde M_Z hasta M_0 y obtenemos un tour para el DGRP en $G_{\mathcal{C}}$ con el mismo coste. Finalmente, si x recorre de modo extra algún arco de $(M_0 : M_z)$ entonces, dado que $\left|(M_0 : M_Z)_R\right| = \left|(M_Z : M_0)_R\right| + 1$, el recorrido de $(M_0 : M_Z)$ está desequilibrado por dos unidades y ambos caminos P_1 y P_2 tienen que ser recorridos desde M_Z hasta M_0, con un coste de $n_1(n_2 - 1) + n_1n_2$. Así, la desigualdad 2-Path-Bridge Asimétrica (4.13) es válida para DGRP($G_{\mathcal{C}}$) y, por lo tanto, también lo es para DGRP(G). ♦

Teorema 4.5.2

Las desigualdades 2-Path-Bridge Asimétricas (4.13) *inducen faceta de DGRP(G).*

Demostración: Veamos primero que inducen faceta de DGRP($G_{\mathcal{C}}$). Denotemos por $F(x) \geq 2n_1(n_2 - 1) + n_1n_2$ a la desigualdad 2-Path-Bridge Asimétrica (4.13). Sea A' el conjunto de arcos del grafo $G_{\mathcal{C}}$ que no están ni en el camino $P1$ ni en el camino $P2$. La dimensión de DGRP($G_{\mathcal{C}}$) es $|A'| + n_1 + n_2 + 3$ y éste es el número de tours afínmente independientes cumpliendo $F(x) = 2n_1(n_2 - 1) + n_1n_2$ que tenemos que encontrar.

Puede verse que, para cualquier arco $a \in A'$, existe un tour para el DGRP que recorre de modo extra el arco a, recorre los arcos requeridos y arcos de los caminos $P1$ y $P2$ y cumple la desigualdad con igualdad. Además, podemos generar $n_2 + 1$ tours similares al de la Figura 4.18(a), $n_1 + 1$ tours similares al de la Figura 4.18(b) y

el tour de la Figura 4.18(c). Todos estos tours cumplen que $F(x) = 2n_1(n_2-1)+n_1n_2$. Si expresamos estos tours como filas y los arcos como columnas, obtenemos la matriz de la Figura 4.17 (a), donde el bloque E y la matriz $D_{m\times m}$ son los de la Figura 4.17 (b) y Figura 4.17 (c), repectivamente. La matriz E tiene $n_1+1+n_2+1+1 = n_1+n_2+3$ filas, y puede probarse que es de rango completo. Por lo tanto, la desigualdad induce faceta de DGRP($G_{\mathcal{C}}$) y, por el Teorema 4.3.2 de 'lifting", también induce faceta de DGRP(G). ♦

A'	P1 P2
I	$*$
0	E

$$E = \begin{array}{|cc|cc|} \hline \multicolumn{2}{|c|}{\text{P1}} & \multicolumn{2}{c|}{\text{P2}} \\ \hline D_{n_1+1} & D_{n_1+1} & 0 & 1 \\ 0 & 1 & D_{n_2+1} & D_{n_2+1} \\ 0\ldots0 & 2\ldots2 & 1\ldots1 & 0\ldots0 \\ \hline \end{array} \qquad D_m = \begin{pmatrix} 0 & 1 & & 1 & 1 \\ 1 & 0 & & 1 & 1 \\ & & \ddots & & \\ 1 & 1 & & 0 & 1 \\ 1 & 1 & & 1 & 0 \end{pmatrix}_{m\times m}$$

Figura 4.17: *Matrices que aparecen en la demostración del Teorema 4.5.2.*

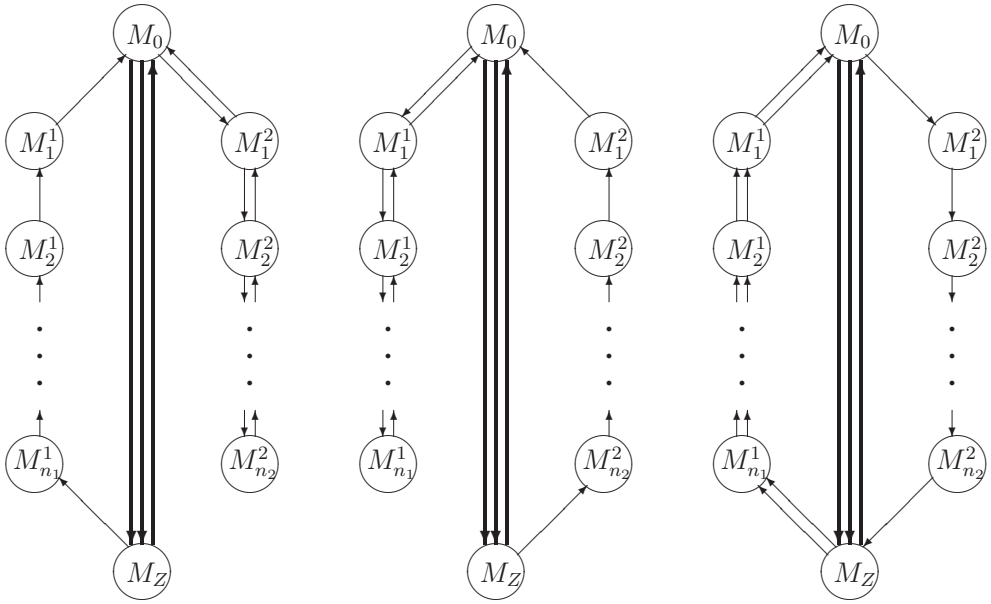

Figura 4.18: *Tours que satisfacen la desigualdad 2-Path-Bridge Asimétrica con igualdad.*

Capítulo 5

Un algoritmo de branch and cut para el SCP y el DGRP

Son Padberg y Rinaldi (1990) quienes presentan por primera vez un algoritmo de branch & cut para un problema de rutas de vehículos. Estos algoritmos están basados en la creación de un árbol de enumeración donde en cada nodo se aplica un algoritmo de planos de corte. En dicho algoritmo de planos de corte, a partir de las soluciones (no necesariamente enteras) del subproblema lineal, se aplica un algoritmo de separación para identificar desigualdades válidas que corten a la solución si ésta no es factible. Estos cortes se añaden al problema, que se vuelve a resolver mientras no se encuentren más cortes o el criterio de parada no se alcance.

5.1. Problema lineal inicial y algoritmo de planos de corte

En esta sección presentamos todos los aspectos relativos al procedimiento de planos de corte utilizado dentro del algoritmo de branch & cut para el DGRP.

5.1.1. Problema lineal inicial

El problema lineal (PL) inicial que empleamos en el nodo raíz del árbol es el siguiente:

$$\text{Minimize} \qquad \sum_{(i,j)\in A} c_{ij}x_{ij}$$

s.t.:

$$x(\delta^+(i)) = x(\delta^-(i)), \quad \forall i \in V \tag{5.1}$$

$$x(\delta^+(V_i)) \geq 1, \quad \forall V_i, \quad i \in \{1,2,...,p\} \tag{5.2}$$

$$x_{ij} \geq 1, \quad \forall (i,j) \in A_R \tag{5.3}$$

$$x_{ij} \geq 0, \quad \forall (i,j) \in A_{NR} \tag{5.4}$$

Con respecto a la formulación presentada para el DGRP, la única diferencia es que sólo hemos añadido las desigualdades de conectividad correspondientes a los R-sets $V_1, \ldots, V_p$ y hemos eliminado la restricción de integridad a las variables. De esta manera, si la solución de este problema resulta ser entera y conexa, será la solución óptima del problema original.

5.1.2. Algoritmo de planos de corte

En cada iteración del algoritmo de planos de corte se usan una serie de algoritmos de separación para hallar desigualdades que son violadas por la solución actual y que serán posteriormente añadidas a la relajación lineal del problema. El orden de aplicación de los diferentes algoritmos de separación en cada iteración es el siguiente:

- Algoritmo heurístico de separación para las desigualdades de conectividad.
- Algoritmo exacto de separación para las desigualdades de conectividad si el anterior heurístico ha obtenido menos de 20 desigualdades violadas.
- Únicamente en el nodo raíz y si no se han encontrado desigualdades de conectividad violadas, se utilizan algoritmos heurísticos para la separación de las desigualdades K-C, 2-Path-Bridge y 2-Path-Bridge asimétricas.

El algoritmo de planos de corte se aplica hasta que no se encuentra ninguna desigualdad violada nueva o se satisface el criterio de parada, al cual llamaremos *tailing-off*.

5.1.3. Tailing-off

Durante la exploración del árbol de branch & cut, un nodo puede acabar de ser estudiado por distintas razones:

- El nodo ha sido saturado porque se llegó a una solución factible.
- El valor del PL resuelto en el nodo supera la mejor cota superior conocida.
- Los algoritmos de separación no hallan ningún corte nuevo.

En este último caso se procede a la ramificación, creándose nuevos nodos a explorar. Ahora bien, mientras se sigan encontrando restricciones violadas y estas se vayan

incorporando y la solución no supere la cota superior, se seguirán resolviendo nuevos problemas lineales. En la mayoría de los casos, este proceso se alarga sin que el valor de la función objetivo mejore apenas, por lo que se hace necesario utilizar algún criterio de parada para agilizar el proceso global.

La estrategia utilizada en el algoritmo aquí descrito consiste en parar el algoritmo de planos de corte en el nodo raíz si el incremento de la función objetivo durante las últimas 20 iteraciones es menor del 0.001 %. En cualquier otro nodo, se detiene si el incremento es menor del 0.005 % en las tres últimas iteraciones o la diferencia porcentual entre la solución del problema lineal del nodo y la cota inferior global es mayor del 3 %, es decir, si $\frac{LB_i - LB}{LB} > 0.03$, donde LB es la cota inferior global y LB_i el valor de la última solución obtenida para el problema lineal del nodo que estamos estudiando.

5.1.4. Ramificación

Una vez que el algoritmo ha acabado de buscar cortes en un nodo sin haberlo saturado, sea porque ya no se encuentran más restricciones violadas o por que se haya cumplido el criterio de parada, hemos de dividir el conjunto de soluciones del nodo actual. Para ello la estrategia usada es la llamada **Strong branching**. Esta estrategia fue introducida por primera vez por Applegate et al. (2007) y consiste en escoger un conjunto de variables con valor no entero en la solución y resolver parcialmente los problemas lineales que resultarían al ramificar por cada una de dichas variables. Estos problemas no se resuelven necesariamente de forma óptima, sino que basta con realizar algunas iteraciones del algoritmo símplex para obtener una estimación del nuevo valor de la función objetivo. De esta manera, se puede estimar cuál sería el incremento en la función objetivo en el caso de ramificar por cada una de las posibles variables, eligiendo aquélla para la cual dicho incremento sea mayor.

5.2. Algoritmos de separación

A continuación presentamos los algoritmos de separación que se han implementado como parte del algoritmo de planos de corte para la identificación de desigualdades violadas de los diferentes tipos conocidos.

5.2.1. Desigualdades de conectividad

Para la separación de las desigualdades de conectividad hemos empleado dos algoritmos diferentes, uno heurístico y otro exacto.

En el algoritmo heurístico, generamos un nuevo grafo con los arcos requeridos y todos los arcos no requeridos cuya variable en la solución del subproblema lineal vale más de un cierto $\epsilon \geq 0$. Si el grafo resultante no es conexo consideramos las cortaduras definidas por sus componentes conexas para comprobar si las correspondientes restricciones de conectividad asociadas son violadas o no por la solución. Este algoritmo tiene complejidad de orden $O(|E|)$ y se ejecuta para varios valores de ϵ.

El algoritmo exacto usado para la separación de las desigualdades de conectividad se basa en construir un nuevo grafo comprimiendo cada R-set en un vértice y sustituyendo todos aquellos arcos paralelos entre componentes (en ambas direcciones) por una sola arista. A esta nueva arista se le asocia un coste igual a la suma de los valores de las variables asociadas a los arcos originales en la solución. Una vez construido el grafo, con ayuda del algoritmo propuesto por Nagamochi, Ono e Ibaraki (1994) se halla una cortadura de coste mínimo. Si el coste de ésta es menor que dos, la desigualdad de conectividad correspondiente a dicha cortadura estará violada por la solución. Este algoritmo tiene complejidad de orden $O(nm + n^2 logn)$.

5.2.2. Desigualdades K-C

El algoritmo heurístico usado para la separación de las desigualdades K-C está basado en el presentado por Corberán, Letchford y Sanchis (2001) para el problema General de Rutas (GRP). Otra versión para el mismo problema sobre grafos windy puede ser encontrada en Corberan, Plana y Sanchis (2007). En esta sección describimos una adaptación para el caso dirigido, además de una nueva versión que incorpora algunas mejoras que nos permiten trabajar con un grafo más reducido, lo cual acelera la ejecución del algoritmo.

Sea x^* una solución fraccionaria del subproblema lineal correspondiente (ver figura 5.1). El primer paso consiste en identificar los conjuntos candidatos a formar M_0 y M_K. Escogemos un R-set V_i cualquiera, y buscamos un vértice j que sea "exterior" en la solución x^*, es decir, que esté conectado en la solución x^* a otro vértice que no pertenezca a V_i. Ahora, consideremos el subgrafo inducido por los arcos no requeridos entre vértices de V_i con $x^*_{ij} > 0$ y los requeridos con $x^*_{ij} > 1$. Si dicho subgrafo contiene una única componente conexa, V_i no sirve y deberemos escoger otro R-set. Si el subgrafo no es conexo, escogemos como M_0 a la componente conexa que contiene al vértice i, y como M_K al resto del subgrafo (ver figura 5.2). Finalmente, comprobamos si M_K contiene algún vértice exterior y existe un número par de arcos requeridos entre M_0 y M_K. De no ser así, pasaremos a otro R-set.

Supongamos que hemos hallado los conjuntos M_0 y M_K. Ahora se trata de agrupar el resto del grafo en los conjuntos $M_1, ..., M_{K-1}$. Para ello, consideraremos el grafo resultante de comprimir M_0, M_K y cada R-set V_k, $k \neq i$ en un único vértice. Para simplificar la notación, denotaremos a estos vértices de la misma manera que a sus R-sets asociados. Este nuevo grafo comprimido contendrá una arista entre cada par de vértices para los que existiera algún arco no requerido entre ambos R-sets asociados en la solución x^*. Asociamos a estas aristas un peso w_{ij} resultante de sumar x^*_{st} para cada arco (s, t) existente entre las correspondientes componentes. Es

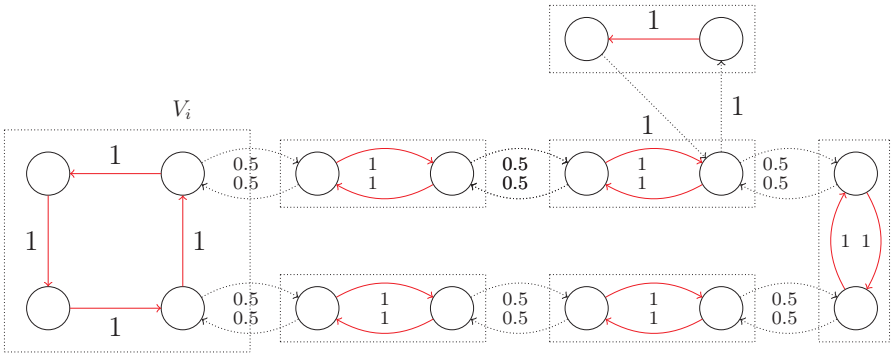

Figura 5.1: *Solución fraccionaria que viola una desigualdad K-C y R-sets.*

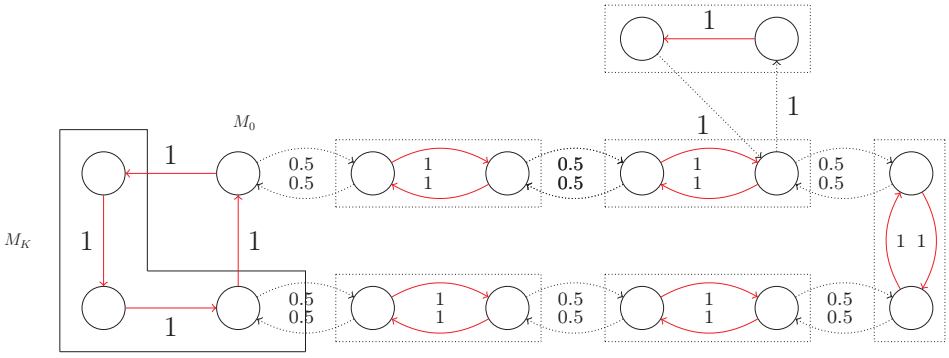

Figura 5.2: *Elección de M_0 y M_K.*

decir, sumamos los arcos paralelos sin importar su dirección. En este nuevo grafo, hallamos un árbol generador de peso máximo que no contenga ningún arco entre M_0 y M_K. Ahora calculamos el camino más corto en dicho árbol entre M_0 y M_K. A continuación comprimimos sucesivamente aquellos nodos de grado 1 del árbol (excepto M_0 y M_K) con su nodo adyacente, hasta quedarnos con un camino de M_0 a M_K. Los nodos de este grafo serán los candidatos a M_i (ver figura 5.3).

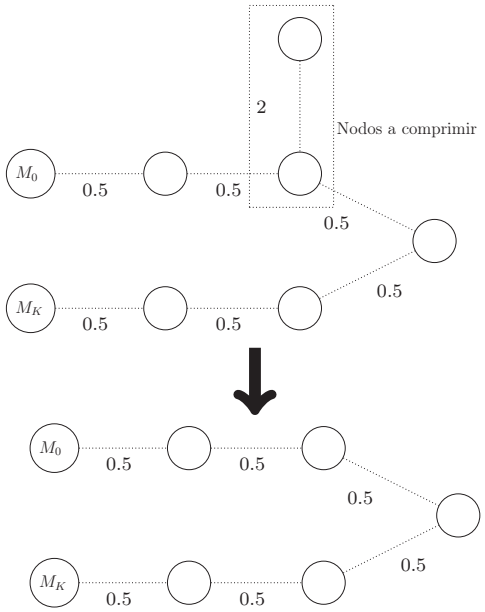

Figura 5.3: *Compresión de nodos y camino más corto entre M_0 y M_K.*

Una vez tenemos los conjuntos $M_0, ..., M_K$, comprimimos aquellos pares de conjuntos M_i, M_{i+1}, $1 \leq i \leq K-2$, para los cuales $w_{M_i M_{i+1}} > 2$, asegurándonos siempre de que nos quedamos al menos con 4 conjuntos, es decir $K \geq 3$. De esta manera conseguiremos reducir el valor del término de la izquierda de la desigualdad K-C en mayor medida que el término de la derecha. Finalmente, comprobamos si la desigualdad K-C asociada a la partición $M_0, M_1, \ldots, M_K$ resultante está violada.

Para esta tesis hemos implementado una nueva versión de este algoritmo de separa-

ción que trabaja en un grafo más reducido. Ahora, en vez de comprimir los R-sets para buscar la partición que defina la desigualdad K-C, comprimiremos las componentes inducidas por los arcos requeridos y algunos arcos no requeridos que, debido al valor que toman en la solución, consideramos que deberían tener coeficiente cero en la desigualdad para que ésta esté violada.

Consideremos el grafo inducido por los arcos requeridos de G y aquellos arcos no requeridos (i, j) para los que $x^*_{ij} \geq 2$ o $x^*_{ij} = 1$ en la solución x^* del subproblema lineal, y calculamos las componentes conexas de dicho grafo, $V^*_1, V^*_2, \ldots, V^*_s$ (ver figura 5.4). A continuación, seleccionamos una componente V^*_i e intentamos partirla en los subconjuntos M_0 y M_K de la misma manera que en el algoritmo original. Una vez obtenidos los conjuntos M_0 y M_K (figura 5.5), comprimimos cada uno de dichos conjuntos, así como cada una de las componentes V^*_j, $j \neq i$ (ver figura 5.6). A partir de ahí se procede como en el algoritmo anterior para hallar la partición $M_0, M_1, \ldots, M_K$.

Esta nueva versión del algoritmo tiene la ventaja de que trabaja en un grafo de menores dimensiones que el de la versión original, lo que lo hace computacionalmente más rápido.

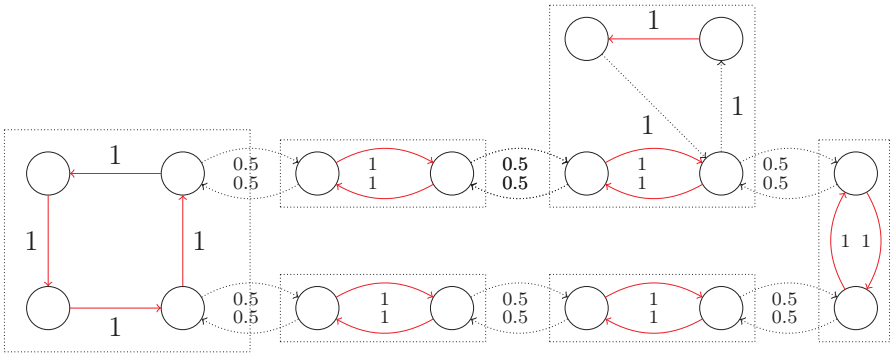

Figura 5.4: *Solución fraccionaria que viola una desigualdad K-C y componentes V^*_i.*

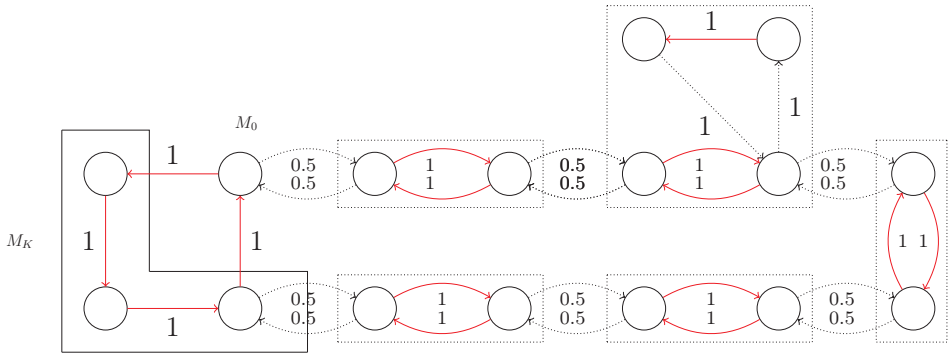

Figura 5.5: *Elección de M_0 y M_K y componentes V_j^*.*

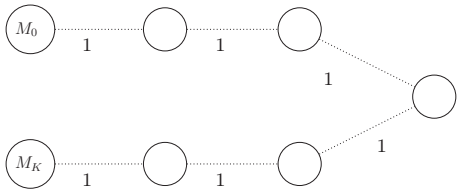

Figura 5.6: *Grafo resultante de comprimir M_0, M_K y las componentes V_j^*.*

5.2.3. Desigualdades Path-Bridge

El algoritmo descrito en esta sección sirve para identificar tanto desigualdades 2-Path-Bridge como 2-Path-Bridge asimétricas violadas. Dicho algoritmo es una generalización del descrito en la sección anterior para las desigualdades K-C.

El primer paso consiste en identificar los conjuntos candidatos a formar M_0 y M_Z. Para cada R-set V_i buscamos un vértice j que sea exterior. Ahora, consideremos el subgrafo inducido por los arcos no requeridos entre vértices de V_i con $x_{ij}^* > 0$. Si dicho subgrafo contiene una única componente conexa, V_i no sirve y deberemos escoger otro R-set. Si el subgrafo no es conexo, escogemos como M_0 a la componente conexa que contiene al vértice j, y como M_Z al resto del subgrafo (ver figura 5.8). Comprobamos si tanto M_0 como M_Z están conectados a al menos dos R-sets diferentes cada uno y si existe un número impar de arcos requeridos entre M_0 y M_Z. De no ser así,

estudiaremos otro R-set.

Una vez tengamos los candidatos a M_0 y M_Z hemos de elegir los candidatos a M_j^k. Para ello construimos un nuevo grafo G^* comprimiendo los conjuntos M_0, M_Z y cada R-set V_k en un sólo nodo. Las aristas de este grafo y sus pesos se obtienen de la misma manera que en el algoritmo para las desigualdades K-C.

En este nuevo grafo calculamos un árbol generador de máximo peso, que no contenga aristas entre M_0 y M_Z. Existe entonces un camino que une M_0 con M_Z que determinará los nodos $M_1^1, M_2^1, \ldots, M_{n1}^1$. Eliminamos los nodos, excepto M_0 y M_Z y las aristas de este camino del árbol, de manera que los conjuntos M_0 y M_Z quedan desconectados. A continuación, seleccionamos de G^* las dos aristas de mayor coste que conecten los nodos M_0 y M_Z al resto de componentes aún no seleccionadas, con lo que volvemos a obtener un camino uniendo M_0 y M_Z, que nos determina los nodos $M_0^2, M_1^2, \ldots, M_{n_2}^2$. Finalmente se van comprimiendo de forma iterativa los vértices de grado uno en el árbol (los que no formaban parte del camino) con su vértice adyacente. Si después de esto quedan vértices aislados, éstos son comprimidos con algún vértice adyacente en el grafo original G.

Una vez tengamos la partición candidata (figura 5.9), construimos las desigualdades 2-Path-Bridge y 2-Path-Bridge asimétrica asociadas y comprobamos si están violadas.

5.3. Resultados computacionales

En esta sección presentamos la instancias para el DGRP y para el SCP usadas para comprobar la eficiencia del algoritmo de branch & cut propuesto anteriormente.

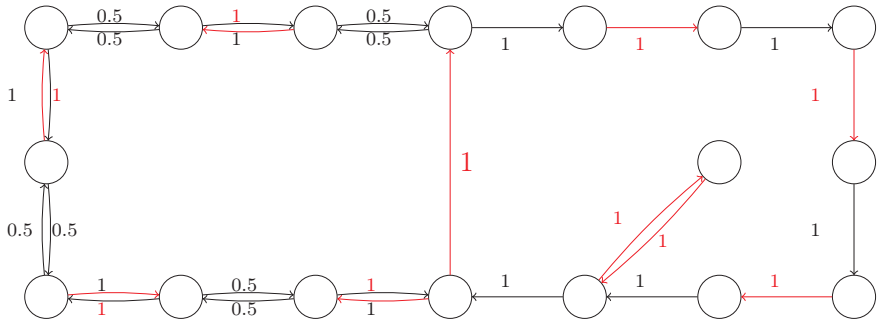

Figura 5.7: *Solución no factible que viola una desigualdad 2-Path-Bridge.*

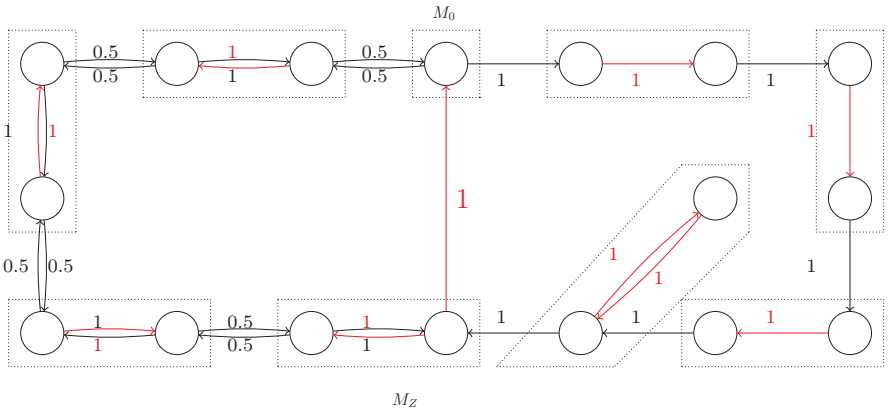

Figura 5.8: *Elección de los conjuntos M_0 y M_Z.*

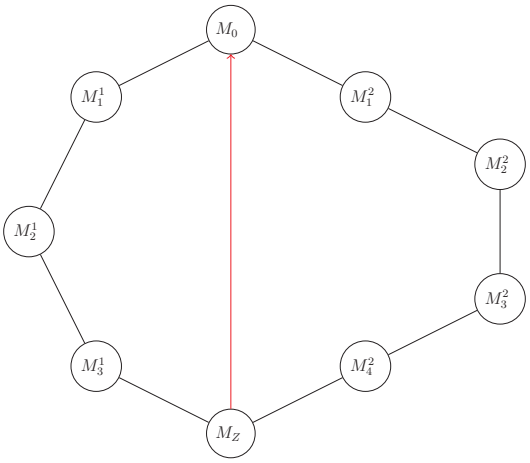

Figura 5.9: *Grafo comprimido que define una desigualdad 2PB o 2PB asimétrica.*

5.3.1. Instancias para el DGRP

Para probar la eficacia del algoritmo de branch & cut, hemos generado un primer conjunto de 36 instancias del DGRP intentando imitar la estructura de las calles de una ciudad. Primero hemos generado de manera aleatoria un número $|V|$ de vértices en una rejilla de tamaño 1000×1000. Para cada vértice v, seleccionamos d arcos a los vértices más cercanos a él y les asignamos una dirección aleatoria. El coste del arco será su distancia euclídea, y lo declaramos como requerido con una probabilidad p. Todos los vértices no incidentes con arcos requeridos son considerados vértices requeridos aislados. Por lo tanto, todas las instancias satisfacen $V = V_R$. Hemos generado una instancia para cada combinación de los siguientes valores de los parámetros: $|V| \in \{500, 750, 1000\}$, $d \in \{3, 4, 5, 6\}$ y $prob \in \{0.25, 0.5, 0.75\}$.

Para identificar las instancias, hemos incluido en el nombre de cada una de ellas los valores de los parámetros con los cuales ha sido generada. El nombre DG742, por ejemplo, significa que es una instancia del DGRP con 750 vértices, $d = 4$ y $prob = 0.25$. Las características y los resultados computacionales obtenidos para las instancias con $|V| = 500, 750$ y 1000 se muestran en las Tablas 5.1, 5.2 y 5.3, respectivamente.

En estas tablas, la columna 'Gap0(%)' muestra el gap obtenido en el nodo raíz del árbol de branch & cut, calculado como $\frac{UB-LB_0}{UB} \times 100$, donde LB_0 representa la cota inferior al finalizar el nodo raíz y UB el coste de la solución óptima (o la mejor solución hallada al finalizar el algoritmo). Las últimas dos columnas dan el número de nodos que explora el árbol de branch & cut y el tiempo empleado, en segundos.

Por lo que se puede ver, los resultados obtenidos en este conjunto de 36 instancias son muy buenos. Los gaps obtenidos al final del nodo raíz son muy ajustados y todas las instancias han sido resueltas de manera óptima en poco tiempo de computación.

Nombre	$\|V\|$	$\|A_R\|$	$\|A_{NR}\|$	$R-$sets	opt?	Gap0(%)	Nodos	Tiempo
DG532	500	218	953	286	si	0.11	3	0.58
DG535	500	443	723	117	si	0.00	1	0.20
DG537	500	639	529	31	si	0.00	1	0.14
DG542	500	282	1011	228	si	0.83	137	22.95
DG545	500	567	712	53	si	0.00	0	0.11
DG547	500	872	412	6	si	0.00	0	0.05
DG552	500	317	1072	208	si	0.00	1	0.23
DG555	500	687	720	21	si	0.00	0	0.09
DG557	500	959	409	4	si	0.00	0	0.09
DG562	500	384	1160	146	si	0.00	0	0.11
DG565	500	804	764	9	si	0.00	0	0.08
DG567	500	1132	403	1	si	0.00	0	0.11

Tabla 5.1: *Instancias del* DGRP *con* $|V|$*=500 y* $1166 \leq |A| \leq 1535$.

Nombre	$\|V\|$	$\|A_R\|$	$\|A_{NR}\|$	$R-$sets	opt?	Gap0(%)	Nodos	Tiempo
DG732	750	339	1451	422	yes	0.70	322	36.69
DG735	750	654	1150	179	yes	0.00	2	1.02
DG737	750	1003	785	35	yes	0.00	0	0.27
DG742	750	407	1497	366	yes	0.15	7	1.73
DG745	750	852	1090	82	yes	0.00	1	0.38
DG747	750	1244	642	5	yes	0.00	0	0.11
DG752	750	516	1570	292	yes	0.01	2	0.69
DG755	750	973	1114	51	yes	0.00	0	0.2
DG757	750	1537	557	2	yes	0.00	0	0.16
DG762	750	584	1784	218	yes	0.00	0	0.27
DG765	750	1204	1172	11	yes	0.00	0	0.17
DG767	750	1673	603	4	yes	0.00	0	0.19

Tabla 5.2: *Instancias del* DGRP *con* $|V|$*=750 y* $1788 \leq |A| \leq 2376$.

Nombre	$\|V\|$	$\|A_R\|$	$\|A_{NR}\|$	$R-$sets	opt?	Gap0(%)	Nodos	Tiempo
DG132	1000	464	1930	553	yes	0.89	1890	444.23
DG135	1000	870	1472	246	yes	0.05	3	1.88
DG137	1000	1315	1068	43	yes	0.00	0	0.34
DG142	1000	562	2023	466	yes	0.38	25	6.98
DG145	1000	1133	1469	116	yes	0.00	1	1.13
DG147	1000	1643	908	8	yes	0.00	0	0.22
DG152	1000	657	2197	386	yes	0.02	2	0.72
DG155	1000	1283	1501	52	yes	0.00	0	0.36
DG157	1000	2004	820	4	yes	0.00	0	0.25
DG162	1000	753	2373	306	yes	0.00	0	0.45
DG165	1000	1550	1603	26	yes	0.00	0	0.27
DG167	1000	2349	828	4	yes	0.00	0	0.30

Tabla 5.3: *Instancias del* DGRP *con* $|V|$*=1000 y* $2342 \leq |A| \leq 3177$.

Por otro lado, Blais y Laporte (2003) resuelven el DGRP transformándolo en el ATSP y resolviendo las instancias resultantes mediante el algoritmo exacto propuesto por Carpaneto, Dell'Amico y Toth (1995) para el ATSP. De esta manera consiguen resolver instancias del DGRP bastante grandes de manera óptima (ver tabla 5.4).

Con idea de comprobar el potencial de nuestro algoritmo en un conjunto de instancias de un tamaño similar a las propuestas por Blais y Loporte, hemos generado un conjunto de instancias con las mismas características. En primer lugar hemos generado de manera aleatoria grafos con 5000 vértices y 50000 arcos. Para asegurar que los grafos sean fuertemente conexos, hemos generado un ciclo Hamiltoniano no dirigido sobre todos los vértices y hemos añadido dos arcos opuestos en el grafo por cada arista del ciclo. Sobre este grafo original se ha elegido un numero determinado de vértices y de arcos de manera aleatoria para que sean los vértices y arcos requeridos. El coste de los arcos ha sido elegida también de manera aleatoria siguiendo una distribución uniforme en el intervalo $[10, 110]$. Las características de las instancias pueden verse en la Tabla 5.4, donde $|V|$ y $|A|$ son el número de vértices y arcos del

grafo original, $|V_R|$ y $|A_R|$ el número de vértices y arcos requeridos respectivamente, y $|V_{ATSP}|$ representa el número de vértices una vez transformado el grafo en una instancia del ATSP mediante el procedimiento empleado por Blais y Laporte. Hemos generado cinco instancias de cada tipo.

Para poder aplicar nuestro algoritmo a estas instancias, hemos transformado los grafos de manera que todos sus vértices sean requeridos. Para ello, hemos calculado el camino más corto entre cada par de vértices requeridos en la instancia original y añadido un arco entre estos vértices con el coste de dicho camino, para después eliminar todos los vértices no requeridos del grafo.

Aunque, muchos de estos nuevos arcos son redundantes y pueden ser eliminados, el tamaño de los grafos resultantes es enorme, lo que hace que para muchos de ellos sea imposible resolver siquiera el LP inicial. Nótese que el número de variables asociadas con los arcos no requeridos puede llegar a ser de aproximadamente nueve millones para una instancia con $|V_R| + |A_R| = 3000$. Para poder superar esta dificultad, hemos modificado nuestro algoritmo de manera que se pueda trabajar con el grafo original, conteniendo tanto vértices requeridos como no requeridos. La modificación no es trivial, e implica adaptar tanto las familias de desigualdades descritas en este capítulo como los correspondientes procedimientos de separación y su código.

Los resultados obtenidos con nuestro algoritmo sobre estas instancias se muestran en la tabla 5.4, donde se compara con los resultados obtenidos por Blais y Laporte (2003). Aunque las instancias usadas para esta tesis no son exactamente las mismas que las usadas por ellos, las características son las mismas, y pensamos que los resultados pueden ser comparables. La columna *#opt.* muestra el número de instancias resueltas de manera óptima sobre las cinco generadas, y la columna *Tiempo* presenta el tiempo medio empleado por el algoritmo de branch & cut. El tiempo mostrado para el procedimiento de Blais y Laporte incluye el tiempo utilizado en la transformación de la instancia original a una instancia ATSP, y el tiempo de

				Blais & Laporte[1]			Branch and Cut	
$\|V\|$	$\|A\|$	$\|V_R\|$	$\|A_R\|$	$\|V_{ATSP}\|$	# opt.	Tiempo	# opt.	Tiempo
5000	50000	1000	1000	2000	5/5	125.6	5/5	31.3
5000	50000	1000	1500	2500	5/5	193.6	5/5	51.8
5000	50000	1000	2000	3000	5/5	280.3	5/5	28.3
5000	50000	1000	2500	3500	4/5	374.9	5/5	21.9
5000	50000	1000	3000	4000	0/5	-	5/5	25.5
5000	50000	1500	1000	2500	5/5	183.1	5/5	37.3
5000	50000	2000	1000	3000	5/5	244.7	5/5	36.7
5000	50000	2500	1000	3500	5/5	314.5	5/5	57.4
5000	50000	3000	1000	4000	4/5	396.8	5/5	50.7
5000	50000	0	3000	3000	5/5	303.0	5/5	12.5
5000	50000	500	2500	3000	5/5	300.0	5/5	19.3
5000	50000	1500	1500	3000	5/5	269.2	5/5	32.5
5000	50000	2500	500	3000	5/5	226.1	5/5	57.8
5000	50000	3000	0	3000	4/5	273.9	5/5	347.2

Tabla 5.4: *Características de las instancias propuestas por Blais y Laporte y resultados obtenidos .*

resolución empleado ha sido limitado a cinco minutos. Nótese que los resultados correspondientes al procedimiento de Blais y Laporte han sido obtenidos sobre un ordenador Sun Ultra Sparc Station 10, que en la actualidad puede considerarse una máquina lenta. Podemos observar que nuestro algoritmo es capaz de resolver las 70 instancias óptimamente en tiempos de computación muy reducidos.

5.3.2. Instancias para el SCP

Como hemos comentado en el capítulo sobre el SCP, casi todos los estudios previos realizados sobre este problema han sido llevados a cabo desde el punto de vista de los problemas rutas por vértices, transformando las instancias del SCP en otras equivalentes del ATSP y resolviendo estas últimas. Esto ha provocado que casi todas las instancias del SCP de la Literatura se encuentren en formato ATSP, es decir, lo que se tiene es una matriz con las distancias entre los vértices. Cada uno de estos vértices representa un trabajo a realizar, es decir, un arco requerido del SCP. Por esta razón no hemos podido ejecutar nuestro algoritmo para estas instancias, así que hemos generado dos conjuntos nuevos de instancias para el SCP con el formato utilizado en los problemas de rutas por arcos.

El primer conjunto de instancias ha sido generado siguiendo el procedimiento propuesto por Srour y van de Velde (2011). Estas instancias, llamadas "drayage", intentan imitar problemas reales en el contexto del transporte de corta distancia (drayage). Según Srour y van de Velde, estos problemas se caracterizan por el hecho de que todos los trabajos provienen de o se dirigen a unos pocos terminales de carga fijos, por lo que suelen tener una estructura geométrica interesante (ver figura 5.10).

El algoritmo generador de instancias utilizado emplea un número aleatorio propuesto por Cirasella et al. (2007) y Jhonson et al. (2002), y su código detallado puede ser encontrado en Srour (2010). Cada instancia ha sido generada de la siguiente forma: se seleccionan n puntos del cuadrado $x \times x$, los cuales servirán como orígenes y destinos de los n trabajos. El generador selecciona los k primeros puntos como orígenes y los últimos m puntos como destinos. Se ha usado una estrategia "round-robin" para emparejar los orígenes con los destinos hasta obtener n trabajos. El coste de cada trabajo/arco requerido es la distancia euclídea entre el origen y el destino.

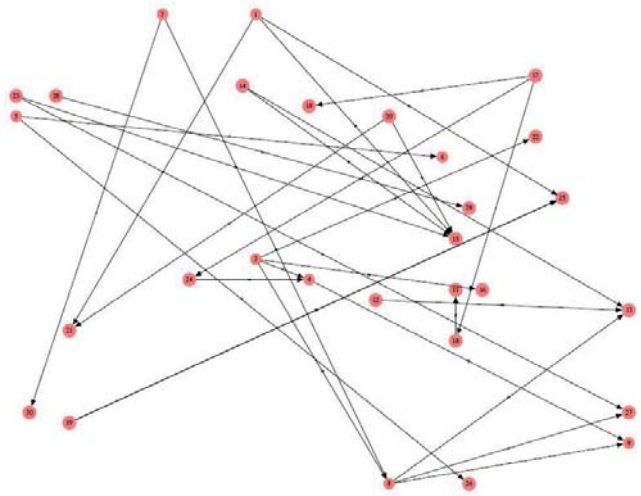

Figura 5.10: *Instancia Crane2 tipo drayage*

Hemos generado 14 instancias, cuyas características pueden verse en la Tabla 5.5 y coinciden con las características de las instancias de la tabla C1 en el trabajo de Srour y van de Velde (2011).

	$x \times x$	n	k	m	Gap0 (%)	# opt.	Tiempo
Drayage100_a	100×100	100	50	75	0.00	5/5	0,1
Drayage100_b	$10^6 \times 10^6$	100	100	65	0.07	2/2	1,2
Drayage300_a	500×500	300	150	200	0.0001	5/5	20,8
Drayage300_b	$10^6 \times 10^6$	300	300	200	0.00	2/2	4,4

Tabla 5.5: *Características de las instancias "drayage" y resultados.*

En la Tabla 5.5 podemos ver los resultados obtenidos por nuestro algoritmo de branch & cut. Podemos comprobar que se han resuelto todas las instancias de manera óptima en un tiempo computacional relativamente pequeño. Nótese también que el gap obtenido en el nodo raíz del árbol del branch & cut es muy pequeño en todas las instancias, lo que significa que el procedimiento de planos de corte es suficiente para resolver óptima o casi óptimamente la mayoría de estas instancias.

Posteriormente hemos generado nuevas instancias para el SCP definidas sobre una rejilla de la siguiente manera: dado un número x, generamos una rejilla de dimensiones $x \times x$, es decir, un grafo en el que cada punto con coordenadas enteras en el plano de dimensión $x \times x$ es un vértice, y cada vértice está unido por un arco de ida y otro de vuelta a todos los vértices que se encuentran a distancia uno. A continuación, generamos n arcos requeridos (trabajos) seleccionando para cada uno de ellos un origen de manera aleatoria y un destino aleatorio entre los vértices que distan de ese origen menos de un valor d dado. El coste de los arcos se calcula usando la distancia de Manhattan. En la Figura 5.11 podemos ver un ejemplo de instancia con una rejilla de tamaño 10×10 y 10 trabajos, donde los vértices azules son orígenes de algún trabajo, los verdes destinos, y los amarillos son a la vez origen y destino de algún trabajo.

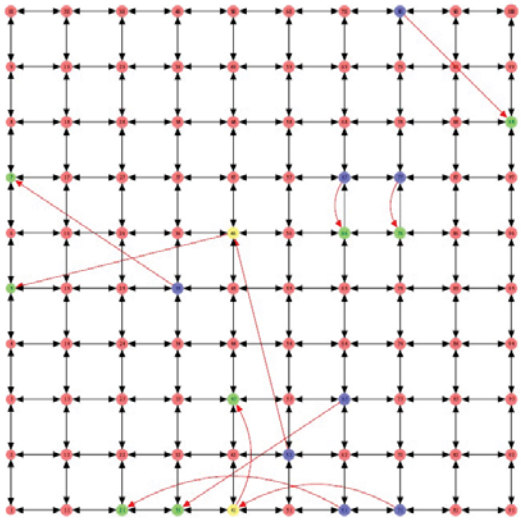

Figura 5.11: *Instancia con 10 trabajos en una rejilla* 10×10

Hemos generado instancias en una rejilla 50×50 con un número de trabajos $n \in \{100, 300, 500, 1000, 2000\}$ y distancia máxima entre origen y destino $d \in \{5, 8\}$, así como otras instancias en una rejilla 100×100 con $n \in \{500, 1000, 2000, 3000\}$ y

$d \in \{5, 8\}$. Para cada combinación de los parámetros x, n, y d hemos generado 5 instancias diferentes.

Inicialmente, probamos a resolver las instancias directamente en el grafo original, pero el elevado número de vértices no requeridos y el hecho de que todas los arcos no requeridos tuviesen coste uno hacía que las instancias fueran extremadamente difíciles de resolver, ya que se necesitaba un número muy elevado de desigualdades de conectividad para asegurar la conexión de la solución. Por ello decidimos transformar el grafo eliminando todos los vértices no requeridos y añadiendo arcos no requeridos representando los caminos más cortos entre los trabajos. Aunque el número de caminos más cortos inicialmente sea enorme, gracias a la estructura de las instancias y la métrica usada para obtener el coste de los arcos, muchos de estos caminos más cortos son redundantes y pueden ser eliminados del grafo. Por ejemplo, en una instancia con $x = 100$, $n = 500$, el número de caminos más cortos es de aproximadamente 250000, pero tras aplicar esta simplificación se queda en alrededor de 25000.

En las Tablas 5.6 y 5.7 se muestran los resultados obtenidos sobre este tipo de instancias para el SCP con rejillas 50×50 y 100×100, respectivamente. Las columnas n y d muestran el valor de los respectivos parámetros en cada conjunto de instancias. El número de instancias resueltas de manera óptima sobre el total aparece en la columna *#opt.*, mientras que la columna $Gap0(\,\%)$ hace referencia al gap medio entre la cota inferior al final del nodo raíz y la solución óptima (o la mejor solución factible conocida). Las columnas "Nodos" y "Tiempo" muestran el promedio de nodos del branch & cut explorados hasta alcanzar el óptimo (o finalizar por alcanzar el tiempo límmite) y el tiempo medio empleado en segundos. Nótese que hemos resuelto óptimamente todas las instancias con $x = 50$ en un tiempo computacional muy corto. También hemos resuelto de manera óptima 27 de las 40 instancias con $x = 100$ con un tiempo limite de una hora. Para las instancias que no hemos podido resolver de manera óptima, se muestra en la columna $Gap(\,\%)$ el gap medio entre la cota

inferior final y la mejor solución encontrada.

Conjunto	n	d	# opt.	Gap0 (%)	Nodos	Tiempo
Rejilla50_100_5	100	5	5/5	0,45	61	5,0
Rejilla50_100_8	100	8	5/5	0,10	6	1,4
Rejilla50_300_5	300	5	5/5	0,05	15,6	12,1
Rejilla50_300_8	300	8	5/5	0,00	1,8	3,7
Rejilla50_500_5	500	5	5/5	0,00	1	5,6
Rejilla50_500_8	500	8	5/5	0,00	0	2,8
Rejilla50_1000_5	1000	5	5/5	0,00	0	2,2
Rejilla50_1000_8	1000	8	5/5	0,00	0	2,1
Rejilla50_2000_5	2000	5	5/5	0,00	0	1,4
Rejilla50_2000_8	2000	8	5/5	0,00	0	1,6

Tabla 5.6: *Resultados para instancias del* SCP *en rejillas 50×50.*

Conjunto	n	d	# opt.	Gap0 (%)	Gap (%)	Nodos	Tiempo
Rejilla100_500_5	500	5	0/5	0,70	0,55	998,2	3600
Rejilla100_500_8	500	8	3/5	0,11	0,03	438,2	2033,0
Rejilla100_1000_5	1000	5	0/5	0,32	0,28	325,8	3600
Rejilla100_1000_8	1000	8	5/5	0,00	0,00	14,2	333,1
Rejilla100_2000_5	2000	5	4/5	0,01	0,00	76,6	1674,0
Rejilla100_2000_8	2000	8	5/5	0,00	0,00	0,6	110,2
Rejilla100_3000_5	3000	5	5/5	0,00	0,00	0	97,7
Rejilla100_3000_8	3000	8	5/5	0,00	0,00	0	78,1

Tabla 5.7: *Resultados para instancias del* SCP *en rejillas 100×100.*

Podemos observar que, en general, para un número fijo de trabajos las instancias tienden a ser más fáciles si el valor d usado es grande. Esto viene explicado por el diferente esfuerzo que se ha de realizar para garantizar la conectividad de la solución.

Aunque el número de R-sets de dos instancias con las mismas características pero con diferente valor de d puede ser similar, después de resolver el LP inicial, se observa que el número de componentes conexas inducidas por la solución del LP es mucho más pequeña para instancias con $d = 8$ que para instancias con $d = 5$. Una explicación de este comportamiento puede verse en la Figura 5.12, donde se muestran dos instancias con 100 trabajos en una rejilla 25×25 y diferentes valores de d, 3 y 8. El número de R-sets es 74 y 71 respectivamente, mientras que después de resolver el LP inicial el número de componentes conexas es de 17 y 5, respectivamente. La primera instancia se puede resolver óptimamente añadiendo 22 cortes de conectividad, mientras que para la segunda únicamente son necesarios 5 cortes. El mismo comportamiento puede ser observado cuando el número de trabajos se incrementa. Para la instancia mostrada en la Figura 5.13, con 500 trabajos, el número inicial de R-sets es pequeño, 41, y el número de componentes conexas después de resolver el LP inicial es de una, lo cual hace trivial su resolución.

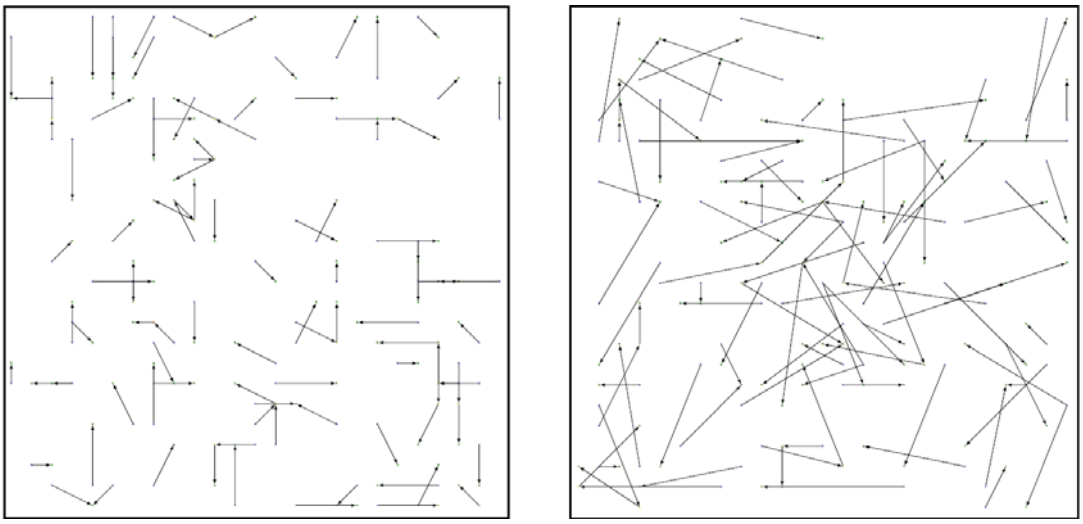

Figura 5.12: *Dos instancias del* SCP *con 100 trabajos en una rejilla 25×25,* $d = 3$ *y* $d = 8$.

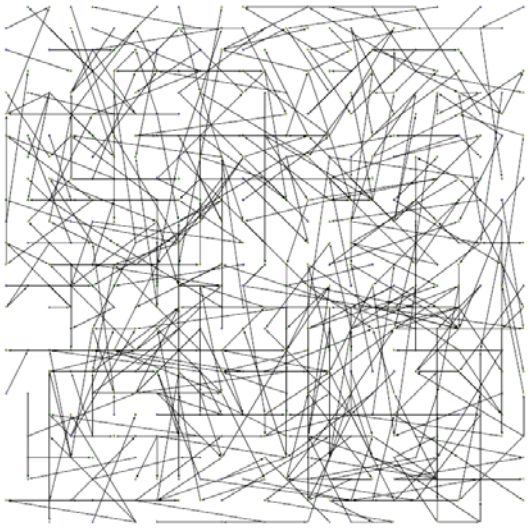

Figura 5.13: *Instancia del* SCP *con 500 trabajos en una rejilla 25×25,* $d = 8$.

Capítulo 6

El problema del Cartero Rural Generalizado en un grafo dirigido

6.1. Descripción del problema

En el tratamiento clásico de los problemas de rutas por arcos se estudia la búsqueda de rutas óptimas dado un conjunto fijo de arcos a recorrer. A veces este punto de vista, aunque haya sido el más utilizado, no es directamente aplicable a algunos problemas nuevos nacidos de casos prácticos. Éstos pueden verse como versiones algo diferentes de los problemas clásicos de rutas por arcos. De ahí la necesidad de hacer un estudio particular de estos problemas. En esta sección estudiaremos el Problema del Cartero Rural Generalizado en un grafo dirigido (Generalized Directed Rural Postman Problem, GDRPP), donde el conjunto de arcos a recorrer ya no está prefijado de antemano, sino que es una variable de decisión.

Así, el GDRPP se puede describir como el problema de encontrar una ruta de coste mínimo que atraviese algunas de las calles de una red de manera que todos los clientes

sean servidos. Un cliente es servido por un vehículo si éste pasa a una distancia menor o igual a r del cliente. Nótese que, como se ha comentado anteriormente, la diferencia del GDRPP con los problemas de rutas por arcos clásicos es que el conjunto de arcos a servir no es conocido a priori.

Este problema ha sido estudiado recientemente por Hà et al. (2013), quienes lo denominan el Close Enough Arc Routing Problem, en el contexto de una aplicación surgida en la lectura de contadores. Es una generalización del Problema del Cartero Rural Dirigido, con restricciones adicionales del Problema de Cubrimiento de Conjuntos (Set Covering Problem, SCP). En su artículo, los autores proponen una nueva formulación para el GDRPP y la comparan con una formulación introducida previamente en Hà et al. (2012) y con otra propuesta por Drexl (2007, 2013). Además de ello, proponen un algoritmo de branch & cut que proporciona muy buenos resultados computacionales.

Este problema fue descrito por primera vez por Golden, Raghavan y Wasil (2008), quienes lo denominan el Problema del Viajante Suficientemente Cercano (Close Enough TSP, CETSP). En su artículo, además de introducir y definir el problema, proponen cuatro heurísticos para resolver ocho instancias reales con una media de 900 calles y 9000 clientes cada una. El rango de lectura r utilizado ha sido de 300 y 500 pies. Los dos procedimientos usados han utilizado básicamente dos fases: inicialmente se busca de manera heurística el subconjunto de calles a atravesar, y, una vez elegido, se resuelve un DRPP con un heurístico sofisticado. Es, sin embargo, Drexl quien primero propone este problema en su tesis doctoral de 2007 al que denomina Generalized Directed Rural Postman Problem (GDRPP). Nosotros hemos adoptado aquí el nombre originalmente propuesto, aunque utilizaremos la aplicación práctica que motivó los otros trabajos para ilustrar mejor el problema.

6.1.1. Definición del Problema

El Generalized Directed Rural Postman Problem (GDRPP) se define del modo siguiente. Sea $G = (V, A)$ un grafo dirigido con conjunto de vértices V y conjunto de arcos A, y sea $c_{ij} \geq 0$ el coste de recorrer el arco $(i, j) \in A$. Dada una familia de subconjuntos de arcos $\mathbb{H} = \{H_1, \ldots, H_L\}$ (cada $H_c \subset A$ es el conjunto de arcos desde los que se puede servir al cliente c), el problema consiste en encontrar un tour de mínimo coste empezando y acabando en el depósito (vértice 1) y que recorra al menos un arco de cada subconjunto H_c. Notar que los subconjuntos H_c no tiene por qué ser disjuntos ni inducir subgrafos conexos.

6.1.2. Aplicaciones

Hace no mucho tiempo, y aún hoy en día en muchas poblaciones, la lectura de los contadores de electricidad, agua o gas se hacía puerta a puerta. En la actualidad, el envío de datos por radio frecuencia es cada día más utilizado por un elevado número de compañías y empresas de servicios. Los avances en tecnología, en particular la mejora de la precisión de los transmisores y receptores, ha hecho que el coste de obtención o envío de datos haya descendido. Esta es la razón por la que cada vez se utiliza más la tecnología de radio frecuencia (RFID) para monitorizar la transmisión de datos (ver Golden, Raghavan y Wasil, 2008). Esta tecnología permite situar un receptor en un vehículo que al pasar a una distancia menor que un cierto valor r del contador puede obtener la información. Así, dada una red de carreteras o calles donde se sitúan los contadores, el vehículo ya no tiene por qué recorrer todas las calles, sino únicamente un subconjunto de ellas de manera que se puedan leer los datos de todos los contadores. Ésta sería una de las aplicaciones más directas del GDRPP.

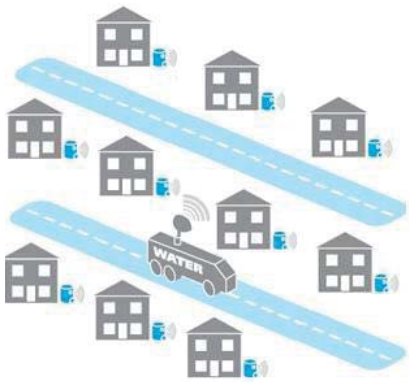

Figura 6.1: *Vehículo con RFID*

6.2. Formulación y propiedades básicas

En esta sección presentamos una formulación como problema de programación lineal entera del GDRPP, definimos el poliedro asociado a sus soluciones posibles y estudiamos algunas propiedades básicas del mismo.

6.2.1. Formulación

La notación que utilizamos para este problema es, fundamentalmente, la misma que ya describimos para el DGRP. Aquí, llamaremos "requeridos" a los arcos que pueden servir a algún cliente, y los denotamos por $A_R = H_1 \cup H_2 \cup \cdots \cup H_L$. Dado un conjunto de vértices $S \subset V$, denotaremos

- $\delta^+(S) = \{(i,j) \in A \ : \ i \in S, \ \ j \in V \setminus S\}$,
- $\delta^-(S) = \{(i,j) \in A \ : \ i \in V \setminus S, \ \ j \in S\}$,
- $A(S) = \{(i,j) \in A \ : \ i,j \in S\}$, y
- $A_R(S) = \{(i,j) \in A_R \ : \ i,j \in S\}$.

Como es habitual, escribiremos $\delta^+(i)$ en lugar de $\delta^+(\{i\})$.

Llamaremos *tour para el GDRPP* a cualquier camino cerrado que comience y termine en el depósito y que recorra, y sirva, al menos un arco de cada conjunto H_c. A cada tour para el GDRPP le asociamos un vector entero $(x, y) \in \mathbb{R}^{|\mathbb{A}|+|\mathbb{A}_\mathbb{R}|}$ en el que, para cada arco $(i, j) \in A$, x_{ij} es el número de veces que el tour recorre el arco (i, j) y, para cada arco requerido $(i, j) \in A_R$, y_{ij} toma el valor 1 si al menos un cliente es servido desde ese arco y toma el valor 0 en otro caso. Con estas variables, el GDRPP puede formularse del modo siguiente.

$$\text{Minimizar} \sum_{(i,j)\in A} c_{ij}x_{ij}$$

s.a.:

$$x(\delta^+(i)) = x(\delta^-(i)), \quad \forall i \in V \tag{6.1}$$

$$x(\delta^+(S)) \geq 1, \quad \forall S \subseteq V\backslash\{1\} : \exists\, H_c \subseteq A(S) \cup \delta(S) \tag{6.2}$$

$$x(\delta^+(S)) \geq y_{ij}, \quad \forall S \subset V\backslash\{1\} : \nexists\, H_c \subseteq A(S)\cup\delta(S),\ \forall\ (i,j) \in A_R(S) \tag{6.3}$$

$$\sum_{(i,j)\in H_c} y_{ij} \geq 1, \quad \forall H_c \in \mathbb{H} \tag{6.4}$$

$$x_{ij} \geq y_{ij}, \quad \forall (i, j) \in A_R \tag{6.5}$$

$$x_{ij} \geq 0, \quad \forall (i, j) \in A_{NR} \tag{6.6}$$

$$0 \leq y_{ij} \leq 1, \quad \forall (i, j) \in A_R \tag{6.7}$$

$$x_{ij} \text{ entero}, \quad \forall (i, j) \in A \tag{6.8}$$

$$y_{ij} \text{ entero}, \quad \forall (i, j) \in A_R \tag{6.9}$$

Las restricciones (6.1) son las ecuaciones de simetría, que aseguran que hay el mismo número de arcos entrantes y salientes en cada uno de los vértices del grafo. Las desigualdades de conectividad (6.2) y (6.3) hacen que la solución sea conexa y conexa con el depósito, aunque permiten que se tenga subtours formados sólo por arcos sin servir y sin visitar el deposito. Las restricciones (6.4), llamadas desigualdades de

servicio, garantizan que todos los clientes son servidos. La relación entre las variables x_{ij} e y_{ij} viene dada por las desigualdades de obligatoriedad (6.5), que aseguran que los arcos que están sirviendo son recorridos al menos una vez. Las restricciones (6.6) a (6.9) son las desigualdades triviales y de integridad.

Llamaremos también tour para el GDRPP a cualquier vector $(x, y) \in \mathbb{R}^{|A|+|A_R|}$ que cumpla (6.1) - (6.9). Como ya hemos dicho, esta formulación permite la existencia de tours formados por varios subtours desconectados entre sí, de modo que uno de ellos visita el depósito y sirve a todos los clientes y los otros subtours están desconectados del depósito y no sirven a ningún cliente. Sin embargo, puesto que los costes c_{ij} son no negativos, siempre habrá una solución óptima formada por un único tour.

El GDRPP puede considerarse como una combinación de dos problemas de optimización: el Problema de Cubrimiento de Conjuntos (Set Covering Problem, SCP) y el Problema General de Rutas en un grafo Dirigido (Directed General Routing Problem, DGRP):

- Dado un conjunto $E = \{e_1, \ldots, e_n\}$ (llamado conjunto base), un coste $c_j \geq 0$ asociado a cada elemento $e_j \in E$ y una familia $\mathbb{S} = \{S_1, \ldots, S_m\}$ de subconjuntos de E, el SCP consiste en encontrar un subconjunto $E' \subseteq E$ (llamado cubrimiento) que contenga, al menos, un elemento e_i de cada subconjunto S_i, con c-coste total mínimo.

- Dado un grafo dirigido $G = (V, A)$, con un coste $c_{ij} \geq 0$ asociado a cada arco $a_{ij} \in A$, y dados dos subconjuntos $A_R \subseteq A$, $V_R \subseteq V$ de arcos y vértices requeridos, respectivamente, el DGRP consiste en encontrar un tour en G de c-coste mínimo que recorra cada arco requerido y visite cada vértice requerido al menos una vez.

Veamos la relación del GDRPP con estos dos problemas. Sea $(x, y) \in \mathbb{R}^{|A|+|A_R|}$ un tour para el GDRPP en el grafo $G = (V, A)$. El conjunto de arcos $(i, j) \in A_R$ tales

que $y_{ij} = 1$ forman una solución posible (un cubrimiento) del SCP definido sobre el conjunto base $E = A_R$ y la familia de subconjuntos $\mathbb{S} = \mathbb{H}$. Si llamamos A' a este conjunto de arcos, el vector x representa una solución posible (tour) para el DGRP en el grafo G tomando como arcos requeridos a los de A' y como vértice requerido al depósito.

Recíprocamente, podemos encontrar un tour para el GDRPP en G procediendo en dos fases. Primero buscamos una solución posible del SCP definido por $E = A_R$ y $\mathbb{S} = \mathbb{H}$. Después, con el conjunto de arcos A' solución del SCP buscamos una solución posible del DGRP definido en el grafo G tomando como arcos requeridos los de A' y como vértice requerido al depósito. Obviamente, aunque las dos fases sean resueltas óptimamente, el tour para el GDRPP obtenido no tiene por qué ser óptimo.

También podemos construir un tour $(x, y) \in \mathbb{R}^{|A|+|A_R|}$ para el GDRPP tomando un tour x para el DGRP definido en el grafo G tomando como arcos requeridos todos los de $A_R = H_1 \cup \cdots \cup H_L$ y como vértice requerido al depósito, y una solución y del SCP definido con $E = A_R$ y $\mathbb{S} = \mathbb{H}$.

6.2.2. El poliedro de soluciones y propiedades básicas

Definimos GDRPP(G) como la envoltura lineal convexa de todos los tours para el GDRPP en G,

$$\text{GDRPP}(G) = conv\Big\{(x, y) \in \mathbb{R}^{|A|+|A_R|} : (x, y) \text{ es un tour para el GDRPP}\Big\}.$$

Teorema 6.2.1

GDRPP(G) es un poliedro no acotado.

Demostración: Basta ver que, si G es fuertemente conexo, entonces GDRPP(G)= $conv(T) + cone(\mathcal{Z})$, donde T es el conjunto de los tours para el GDRPP en G que no

pueden obtenerse a partir de otro tour eliminando algunos ciclos y $\mathcal{Z}$ es el conjunto de todos los ciclos en G. ♦

En la demostración del teorema siguiente utilizamos un resultado sobre la dimensión del poliedro de soluciones asociado al SCP que podemos encontrar en el trabajo de Balas and Ng (1989). Sea $\mathcal{P}$ el poliedro obtenido como la envoltura lineal convexa de todas las soluciones posibles del SCP definido sobre un conjunto base $E = \{e_1, .., e_n\}$ y una familia de subconjuntos $\mathbb{S} = \{S_1, ..., S_m\}$.

Teorema 6.2.2

$\mathcal{P}$ tiene dimensión completa n si, y sólo si, $|S_i| \geq 2 \ \ \forall\, S_i$.

Teorema 6.2.3

Si G es un grafo fuertemente conexo, dim(GDRPP(G)) $= |A| + |A_R| - |V| + 1$ *si, y sólo si,* $|H_c| \geq 2$, $\forall\, H_c \in \mathbb{H}$.

Demostración: Puede verse que exactamente $|V| - 1$ de las $|V|$ ecuaciones de simetría (6.1) son linealmente independientes. Por lo tanto, dim(GDRPP(G)) $\leq$ $|A| + |A_R| - |V| + 1$.

Si hay algún subconjunto $H_c = \{(i, j)\}$, entonces se cumple que $y_{ij} = 1$ y disminuirá en uno la dimensión, es decir $dim(GDRPP(G)) < |A| + |A_R| - |V| + 1$.

Recíprocamente, supongamos que $|H_c| \geq 2$, $\forall\, H_c \in \mathbb{H}$. Consideremos el DGRP en el grafo G tomando todos los arcos de A_R como requeridos y el depósito como vértice requerido. Como G es fuertemente conexo, dim(DGRP(G))=$|A|-|V|+1$ (ver Teorema 4.2.3), y existen $k+1$ tours para el DGRP en G afínmente independientes, $x^1, x^2, \ldots, x^{k+1}$, con k =dim(DGRP(G)). Estos tours x recorren todos los arcos de A_R y visitan el depósito, luego pueden completarse con vectores y para formar tours para el GDRPP (x, y). Consideremos el SCP con conjunto base $E = A_R$ y familia de subconjuntos $\mathbb{S} = \mathbb{H}$. Como $|H_c| \geq 2$, la dimensión del correspondiente

poliedro del SCP es $|A_R|$ (ver Teorema 6.2.2) y existen $m+1$ soluciones posibles del SCP afínmente independientes, $y^1, y^2, \ldots, y^{m+1}$, con $m = |A_R|$. Así, los vectores $(x^1, y^1), \ldots, (x^{k+1}, y^1), (x^1, y^2), \ldots, (x^1, y^{m+1})$ son $k+1+m$ tours para el GDRPP en G afínmente independientes y dim(GDRPP(G))$\geq k+m = |A| + |A_R| - |V| + 1$.

♦

En lo que sigue, supondremos que $|H_c| \geq 2$ para todo $H_c \in \mathbb{H}$. Vamos a demostrar que, bajo ciertas condiciones, algunas desigualdades de la formulación (6.1) - (6.9) del GDRPP inducen faceta del poliedro de soluciones.

Teorema 6.2.4 (Desigualdades triviales)
La desigualdad trivial (6.6)*,* $x_{ij} \geq 0$*,* $(i,j) \in A_{NR}$*, induce faceta de* GDRPP*(G) si el arco* (i,j) *no es un arco de corte de* G*.*

Demostración: Si (i,j) no es un arco de corte de G entonces $G \setminus \{(i,j)\}$ es fuertemente conexo. Consideremos el GDRPP en este grafo. Como $(i,j) \in A_{NR}$, se sigue cumpliendo que $|H_c| \geq 2$ para todo $H_c \in \mathbb{H}$, y podemos encontrar $x^1, \ldots, x^{k+1}$ tours para el GDRPP en $G \setminus \{(i,j)\}$ afínmente independientes, con $k+1 = (|A| - 1 + |A_R| - |V| + 1) + 1 = |A| + |A_R| - |V| + 1 =$dim(GDRPP($G$)). Éstos también son tours para el GDRPP en G que, como están definidos en $G \setminus \{(i,j)\}$, cumplen que $x_{ij} = 0$. Así, $x_{ij} \geq 0$ induce faceta de GDRPP(G). ♦

Teorema 6.2.5 (Desigualdades triviales)
La desigualdad trivial (6.7)*,* $y_{ij} \leq 1$*,* $(i,j) \in A_R$*, induce faceta de* GDRPP*(G).*

Demostración: Es conocido que la desigualdad $y_{ij} \leq 1$ induce faceta del poliedro de soluciones del SCP (ver los trabajos de Balas and Ng (1989) y de Sánchez et al. (1998)). Por lo tanto, existen m soluciones posibles $y^1, \ldots, y^m$ afínmente independientes, con $m = |A_R|$, cumpliendo que $y_{ij} = 1$.

Consideremos el DGRP en el grafo G tomando todos los arcos de A_R como requeridos y el depósito como vértice requerido. Como G es fuertemente conexo, dim(DGRP(G))=$|A| - |V| + 1$ (ver Teorema 4.2.3), y existen $k+1$ tours para el DGRP en G afínmente independientes, $x^1, x^2, \ldots, x^{k+1}$, con k =dim(DGRP(G)). Combinando estos vectores obtenemos $k+1+m-1 = |A|-|V|+1+|A_R|$ tours para GDRPP en G, $(x^1, y^1), \ldots, (x^{k+1}, y^1), (x^1, y^2), \ldots, (x^1, y^m)$, cumpliendo $y_{ij} = 1$. Por lo tanto, la desigualdad $y_{ij} \leq 1$ induce faceta de GDRPP(G). ♦

Teorema 6.2.6 (Desigualdades de obligatoriedad)
La desigualdad de obligatoriedad (6.5), $x_{ij} \geq y_{ij}$, $(i,j) \in A_R$, *induce faceta de* GDRPP*(G) si el arco (i,j) no es un arco de corte de G y se cumple que $|H_c| \geq 3$, para todo $H_c \in \mathbb{H}$ tal que $(i,j) \in H_c$.*

Demostración: Si (i,j) no es un arco de corte, entonces $G\backslash\{(i,j)\}$ es un grafo fuertemente conexo, y existen $x^1, ..., x^{k+1}$ tours para el GDRPP afínmente independientes, con $k = (|A| - 1) + (|A_R| - 1) - |V| + 1 = |A| + |A_R| - |V| - 1$, todos satisfaciendo $x_{ij} = 0$, $y_{ij} = 0$ y, por lo tanto, también $x_{ij} = y_{ij}$. Por otro lado, como G es fuertemente conexo, existe un camino P_{ji} desde j a i. Si añadimos al tour x_1 los arcos del camino P_{ji} más el arco (i,j), y fijamos $y_{ij} = 1$, obtenemos un nuevo tour $\overline{x}$ que cumple $x_{ij} = y_{ij} = 1$. En total tenemos $k+2 = |A|+|A_R|-|V|+1$ tours para el GDRPP en G cumpliendo $x_{ij} = y_{ij}$ y son afínmente independientes, como se muestra en la matriz siguiente:

$$\begin{pmatrix} x^1 & 0 & 0 \\ \vdots & \vdots & \vdots \\ x^{k+1} & 0 & 0 \\ * & 1 & 1 \end{pmatrix}$$

Por lo tanto, la desigualdad induce faceta de GDRPP(G). ♦

Teorema 6.2.7 (Desigualdades de servicio)

La desigualdad de servicio (6.4), $\sum_{(i,j)\in H_c} y_{ij} \geq 1,\ \forall\, H_c \in \mathbb{H}$, *induce faceta de* GDRPP*(G) si se cumple que*

- *No existe ningún* $H_q \subset H_c$, *y que*
- *Para cada* $(i',j') \in A_R$ *tal que* $(i',j') \notin H_c$, *existe* $(i,j) \in H_c$ *de manera que* $(i,j) \notin H_q\ \forall q$ *con* $(i',j') \in H_q$.

Demostración: Es conocido que la desigualdad $\sum_{(i,j)\in H_c} y_{ij} \geq 1$ induce faceta del poliedro de soluciones del SCP si, y sólo si, se cumplen las condiciones del enunciado (ver los trabajos de Balas and Ng (1989) y de Sánchez et al. (1998)). Por lo tanto, existen m soluciones posibles $y^1, \ldots, y^m$ afínmente independientes, con $m = |A_R|$, cumpliendo que $\sum_{(i,j)\in H_c} y_{ij} = 1$. Consideremos el DGRP en el grafo G tomando todos los arcos de A_R como requeridos y el depósito como vértice requerido. Como G es fuertemente conexo, dim(DGRP(G))=$|A| - |V| + 1$ (ver Teorema 4.2.3), y existen $k+1$ tours para el DGRP en G afínmente independientes, $x^1, x^2, \ldots, x^{k+1}$, con k =dim(DGRP(G)). Combinando estos vectores obtenemos $k + 1 + m - 1 = |A| - |V| + 1 + |A_R|$ tours para GDRPP en G, $(x^1, y^1), \ldots, (x^{k+1}, y^1), (x^1, y^2), \ldots, (x^1, y^m)$, cumpliendo $\sum_{(i,j)\in H_c} y_{ij} = 1$. Por lo tanto, la desigualdad $\sum_{(i,j)\in H_c} y_{ij} \geq 1$ induce faceta de GDRPP(G). ♦

Hasta ahora hemos visto cuales de las desigualdades propias de la formulación son faceta para el poliedro del GDRPP. En esta sección estudiaremos las desigualdades de conectividad e imparidad.

6.2.3. Desigualdades de conectividad

Veamos en qué casos las desigualdades de conectividad inducen faceta para el GDRPP.

Conjetura 6.2.1 (Desigualdades de conectividad)
Las desigualdades de conectividad (6.3)*,* $x(\delta^+(S)) \geq y_{ij}$, $\forall S \subset V\backslash\{1\}$ *tal que no existe* $H_c \subseteq A(S) \cup \delta(S)$ *y* $\forall\ (i,j) \in A_R(S)$*, induce faceta de* GDRPP*(G) si* $G(S)$ *y* $G(V \setminus S)$ *son fuertemente conexos y* $\forall\ H_q \in \mathbb{H}$ *se cumple que* $\left|H_q \setminus \delta(S)\right| \geq 2$.

Nota: No hemos podido demostrar esta conjetura, aunque pensamos que sí es cierta. De hecho, hemos encontrado los tours para el GDRPP en G necesarios, cumpliendo la desigualdad (6.3) con igualdad, pero sólo hemos podido demostrar que todos menos uno de ellos son afínmente independientes.

Las desigualdades de conectividad (6.2), $x(\delta^+(S)) \geq 1$, $\forall S \subseteq V\backslash\{1\}$ tal que $\exists\ H_c \subseteq A(S) \cup \delta(S)$, se pueden generalizar del modo siguiente:

$$x(\delta^+(S)) \geq 1 - y\big(H_c \cap A_R(V\backslash S)\big), \quad \forall S \subseteq V\backslash\{1\} \quad \forall\ H_c \in \mathbb{H} \tag{6.10}$$

Notar que si existe algún $H_c \subseteq A(S) \cup \delta(S)$, entonces la desigualdad (6.10) correspondiente a este H_c resulta la desigualdad (6.2), $x(\delta^+(S)) \geq 1$.

Teorema 6.2.8
Las desigualdades de conectividad (6.10)*,* $x(\delta^+(S)) \geq 1 - y\big(H_c \cap A_R(V\backslash S)\big)$, $\forall S \subseteq V\backslash\{1\}$, $\forall\ H_c \in \mathbb{H}$*, son válidas para GDRPP(G).*

Demostración: Sea (x, y) un tour cualquiera para el GDRPP en G y veamos que cumple (6.10). Si $y\big(H_c \cap A_R(V\backslash S)\big) = 0$ entonces este tour tiene que servir algún arco de $H_c \cap \big(A(S) \cup \delta(S)\big)$, para lo cual tiene que atravesar la cortadura y se cumple que $x(\delta^+(S)) \geq 1$. Si $y(H_c \cap A_R(V\backslash S)) \geq 1$ entonces la desigualdad resulta $x(\delta^+(S)) \geq 0$, que siempre se cumple. ♦

El siguiente teorema establece condiciones para que estas desigualdades induzcan faceta de GDRPP(G).

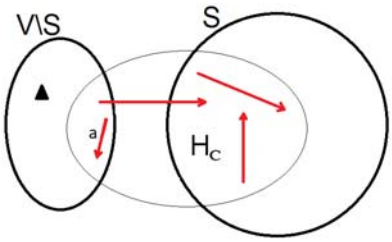

Figura 6.2: *Desigualdades de conectividad* (6.10)

Teorema 6.2.9 (Desigualdades de conectividad)

Las desigualdades de conectividad (6.10), $x(\delta^+(S)) \geq 1 - y\big(H_c \cap A_R(V\backslash S)\big)$, $\forall S \subseteq V\backslash\{1\}$ *y* $\forall\ H_c \in \mathbb{H}$, *inducen faceta de* GDRPP*(G) si* $G(S)$ *y* $G(V\backslash S)$ *son fuertemente conexos,* $\forall H_q \in \mathbb{H}$ *se cumple que* $\big|H_q \setminus \big(\delta(S) \cup (H_c \cap A_R(V\setminus S))\big)\big| \geq 2$ *y, en el caso en que* $H_c \cap A_R(V\setminus S) \neq \emptyset$ *se cumple, además, que*

(1) $\forall H_q \in \mathbb{H},\ \ H_q \cap A_R(V\setminus S) \neq \emptyset$, *y que*

(2) $\nexists\ H_q \in \mathbb{H}$ *tal que* $H_q \cap A_R(V\setminus S) \subsetneq H_c \cap A_R(V\setminus S)$.

Demostración: Supongamos primero que $H_c \cap A_R(V\setminus S) = \emptyset$, es decir $H_c \subseteq \delta(S) \cup A(S)$ y tenemos es la desigualdad de conectividad (6.2), $x(\delta^+(S)) \geq 1$. Consideremos el DGRP en el grafo G tomando todos los arcos de $A'_R = A_R \backslash \delta(S)$ como requeridos y el depósito como vértice requerido. Como $G(S)$ y $G(V \setminus S)$ son fuertemente conexos, sabemos que la desigualdad $x(\delta^+(S)) \geq 1$ induce faceta de DGRP(G) (ver Teorema 4.3.3), y existen $x^1, \ldots, x^k$, con $k = |A| - |V| + 1$, tours para el DGRP en G afínmente independientes pasando por todos los arcos de A'_R y cumpliendo $x(\delta^+(S)) = 1$. Por otro lado, puesto que para todo H_q se cumple que $\big|H_q \setminus \delta(S)\big| \geq 2$, el SCP sobre el conjunto base A'_R tiene dimensión completa (Balas and Ng, 1989), y existen $y^1, ..., y^{m+1}$, con $m = |A'_R|$, vectores afínmente independientes. Completamos estos vectores y^r con ceros en las componentes asociadas a arcos en $\delta_R(S)$ para obtener $\overline{y}^r$ y los combinamos con los vectores x anteriores para obtener $k+m = |A| - |V| + |A'_R| + 1$ tours para el GDRPP en G, $(x^1, \overline{y}^1), \ldots, (x^k, \overline{y}^1), (x^1, \overline{y}^2), \ldots, (x^1, \overline{y}^{m+1})$, afínmente independiente y cumpliendo $x(\delta^+(S)) = 1$.

Para cada arco $(i_0, j_0) \in \delta_R(S)$, y como $x(\delta^+(S)) \geq 1$ induce faceta de DGRP(G), sabemos (ver el Teorema 4.3.3) que existe un tour x^* para el DGRP que pasa por (i_0, j_0) y cumple $x(\delta^+(S)) = 1$. Así pues, definimos un nuevo tour para el GDRPP en G, (x^*, y^*) donde y^* es igual a $\overline{y}^1$ excepto en la componente correspondiente a (i_0, j_0) que ahora es $y^*_{i_0,j_0} = 1$.

Expresando como filas de una matriz estos $|A + |A_R| - |V| + 1$ tours para el GDRPP obtenemos la matriz de la Figura 6.3a, que es de rango completo, luego la desigualdad induce faceta de GDRPP(G).

Supongamos ahora que $H_c \cap A_R(V \backslash S) \neq \emptyset$ y tenemos es la desigualdad de conectividad (6.10), $x(\delta^+(S)) \geq 1 - y\big(H_c \cap A_R(V \backslash S)\big)$. Consideremos el DGRP en el grafo G tomando todos los arcos de $A'_R = A_R \setminus \big(\delta(S) \cup (H_c \cap A_R(V \backslash S))\big)$ como requeridos y el depósito como vértice requerido. Como $G(S)$ y $G(V \setminus S)$ son fuertemente conexos, sabemos que la desigualdad $x(\delta^+(S)) \geq 1$ induce faceta de DGRP(G) (ver Teorema 4.3.3), y existen $x^1, \ldots, x^k$, con $k = |A| - |V| + 1$, tours para el DGRP en G afínmente independientes pasando por todos los arcos de A'_R y cumpliendo $x(\delta^+(S)) = 1$. Por otro lado, puesto que para todo H_q se cumple que $\big|H_q \setminus \big(\delta(S) \cup (H_c \cap A_R(V \setminus S))\big)\big| \geq 2$, el SCP sobre el conjunto base A'_R tiene dimensión completa (Balas and Ng, 1989), y existen $y^1, ..., y^{m+1}$, con $m = |A'_R|$, vectores afínmente independientes. Completamos estos vectores y^r con ceros en las componentes asociadas a arcos en $\delta_R(S) \cup (H_c \cap A_R(V \setminus S)$ para obtener $\overline{y}^r$ y los combinamos con los vectores x anteriores para obtener $k + m = |A| - |V| + |A'_R| + 1$ tours para el GDRPP en G, $(x^1, \overline{y}^1), \ldots, (x^k, \overline{y}^1), (x^1, \overline{y}^2), \ldots, (x^1, \overline{y}^{m+1})$, afínmente independiente y cumpliendo $x(\delta^+(S)) = 1$ e $y(H_c \cap A_R(V \backslash S) = 0$.

Para cada arco $(i_0, j_0) \in \delta_R(S)$, y como $x(\delta^+(S)) \geq 1$ induce faceta de DGRP(G), sabemos (ver Teorema 4.3.3) que existe un tour x^* para el DGRP que pasa por (i_0, j_0) y cumple $x(\delta^+(S)) = 1$. Así pues, definimos un nuevo tour para el GDRPP en G, (x^*, y^*) donde y^* es igual a $\overline{y}^1$ excepto en la componente correspondiente a

(i_0, j_0) que ahora es $y^*_{i_0,j_0} = 1$.

Finalmente, para cada arco $(i_o, j_0) \in H_c \cap A_R(V \setminus S)$ podemos construir un tour para el GDRPP en G $(\widetilde{x}, \widetilde{y})$ tal que $\widetilde{x}(\delta^+(S)) = 0$ (ya que se si cumple la condición (1) del enunciado, se puede servir a todos los clientes desde arcos de $A(V \setminus S)$), y tal que $\widetilde{y}_{i_0 j_0} = 1,\ \widetilde{y}_{ij} = 0,\ \forall (i,j) \in H_c \cap A_R(V \setminus S) \setminus \{(i_0, j_0)\}$ (ya que si se cumple la condición (2) del enunciado, todos los clientes que se sirven desde algún arco del conjunto $H_c \cap A_R(V \setminus S) \setminus \{(i_0, j_0)\}$ pueden ser servidos también desde otros arcos de $A_R(V \setminus S)$). Así, tenemos que $\widetilde{y}(H_c \cap A_R(V \setminus S)) = 1$ y el tour $(\widetilde{x}, \widetilde{y})$ satisface la desigualdad con igualdad.

Expresando como filas de una matriz estos $|A| + |A_R| - |V| + 1$ tours para el GDRPP obtenemos la matriz de la Figura 6.3b, que es de rango completo, luego la desigualdad induce faceta de GDRPP(G). ♦

A	A'_R	$\delta_R(S)$
x^1	y^1	
$\vdots$	$\vdots$	0
x^k	y^1	
x^1	y^2	
$\vdots$	$\vdots$	0
x^1	y^{m+1}	
$*$	y^1 $\vdots$ y^1	I

A	A'_R	$\delta_R(S)$	$H_c \cap A_R(V \setminus S)$
x^1	y^1		
$\vdots$	$\vdots$	0	0
x^k	y^1		
x^1	y^2		
$\vdots$	$\vdots$	0	0
x^1	y^{m+1}		
$*$	y^1 $\vdots$ y^1	I	0
$*$	$*$	0	I

Figura 6.3: *Matrices utilizadas en la demostración del Teorema 6.2.9.*

6.2.4. Desigualdades de imparidad para el GDRPP

Las desigualdades de imparidad son una clase de desigualdades muy utilizadas en los problemas de rutas por arcos. Se basan en el hecho de que cualquier tour (camino cerrado) recorre cualquier cortadura del grafo un número par de veces. Así, si tenemos una cortadura que contiene un número impar q de arcos requeridos, cualquier tour tiene que atravesar esta cortadura al menos $q + 1$ veces. Aunque en muchos problemas de rutas por arcos sobre grafos dirigidos las desigualdades de imparidad están implicadas por las ecuaciones de simetría (por ejemplo en el DGRP, ver Sección 4.3.3), presentamos aquí una generalización de las desigualdades de imparidad que se basan no sólo en algunos arcos requeridos de la cortadura sino, también, en algunos clientes que pueden ser servidos desde otros arcos de la cortadura.

Sean $S \subset V$, $F \subseteq \delta_R(S)$ y $F^H = \{H_{n_1}, H_{n_2}, ..., H_{n_q}\}$, con $|F| + q$ impar y $q > 0$ cumpliendo (ver Figura 6.4):

- $H_{n_i} \cap H_{n_j} \cap \delta(S) = \emptyset \quad \forall n_i, n_j$.
- $F \cap H_{n_i} = \emptyset \quad \forall n_i$.
- $H_{n_i} \cap \delta(S) \neq \emptyset \quad \forall n_i \quad (H_{n_i} \setminus \delta(S)$ puede ser, o no, vacío).

Llamamos *desigualdad de imparidad* (asociada a S, F y F^H) a

$$x(\delta(S)) \geq 2y(F) - |F| + \sum_{i=1}^{q} \big(1 - 2y(H_{n_i} \setminus \delta(S))\big) + 1 \tag{6.11}$$

Notar que para los H_{n_i} tales que $H_{n_i} \subseteq \delta(S)$, desaparece en la desigualdad el término $2y(H_{n_i} \setminus \delta(S))$ y el sumando correspondiente es un 1. En particular, si esto ocurre para todos los H_{n_i} la desigualdad de imparidad resulta

$$x(\delta(S)) \geq 2y(F) - |F| + q + 1,$$

y si, además, $F = \emptyset$, entonces la desigualdad resulta

$$x(\delta(S)) \geq q + 1.$$

Notar también que, como $|F| + q$ es impar, la desigualdad (6.11) puede ser escrita también del modo siguiente:

$$x(\delta^+(S)) \geq y(F) + \frac{q + 1 - |F|}{2} - \sum_{i=1}^{q} y(H_{n_i} \backslash \delta(S)) \tag{6.12}$$

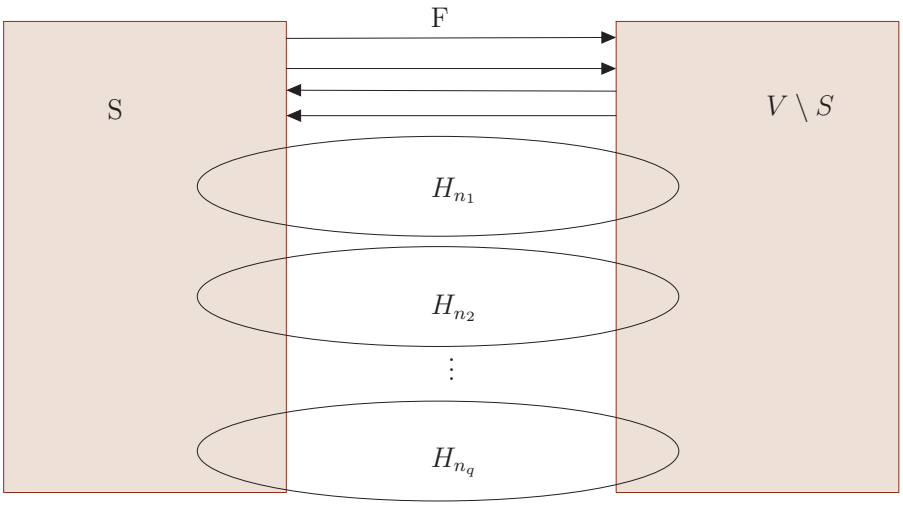

Figura 6.4: *Desigualdad de imparidad.*

Teorema 6.2.10

Las desigualdades de imparidad (6.11) *son válidas para* GDRPP*(G).*

Demostración: Sea (x, y) un tour para el GDRPP en G cualquiera. Tenemos las siguientes posibilidades:

- El tour (x, y) recorre sirviendo todos los arcos de F pero ningún arco de $H_{n_i} \backslash \delta(S)$ para ningún i. En este caso, la parte derecha de la desigualdad vale $|F| + q + 1$. Como el tour (x, y) recorre sirviendo al menos un arco de cada

H_{n_i}, $i = 1 \ldots, q$ y todos los arcos de F, este tour atraviesa la cortadura $\delta(S)$ al menos $|F| + q$ veces. Pero como $|F| + q$ es impar, en realidad la atraviesa al menos $|F| + q + 1$ veces y se cumple que $x(\delta(S)) \geq |F| + q + 1$.

- El tour (x, y) recorre sirviendo todos los arcos de F menos uno y no recorre ningún arco de $H_{n_i} \setminus \delta(S)$ para ningún i. En este caso, la parte derecha de la desigualdad vale $|F| + q - 1$. Como el tour recorre $|F| - 1$ arcos de F y al menos un arco de cada H_{n_i}, $i = 1 \ldots, q$, se cumple que $x(\delta(S)) \geq |F| - 1 + q$.

- El tour (x, y) recorre sirviendo todos los arcos de F y exactamente un arco de alguno de los conjuntos $H_{n_i} \setminus \delta(S)$. En este caso, la parte derecha de la desigualdad vale $|F| + q - 1$ y el tour atraviesa, al menos, $|F| + q - 1$ arcos de la cortadura $\delta(S)$, luego se cumple que $x(\delta(S)) \geq |F| + q - 1$.

- El tour (x, y) recorre sirviendo todos los arcos de F menos uno y exactamente un arco de alguno de los conjuntos $H_{n_i} \setminus \delta(S)$. En este caso, la parte derecha de la desigualdad vale $|F| + q - 3$ y el tour atraviesa, al menos, $|F| - 1 + q - 1$ arcos de la cortadura $\delta(S)$, luego se cumple que. $x(\delta(S)) \geq |F| + q - 2 \geq |F| + q - 3$.

- De un modo similar puede verse que en cualquier otro caso también se cumple la desigualdad. Notar que por cada arco de F que se quede sin recorrer y por cada arco de alguno de los conjuntos $H_{n_i} \setminus \delta(S)$ que se recorra sirviendo, la parte derecha de la desigualdad baja en 2 unidades. ♦

6.2.5. Desigualdades K-C

Las desigualdades K-C fueron introducidas por Corberán y Sanchis (1994) para el Problema del Cartero Rural en un grafo no dirigido, RPP. Fueron adaptadas al MGRP en Corberán, Romero y Sanchis (2003), al WGRP en Corberán, Plana y Sanchis (2007) y al DGRP en esta Tesis (Sección 4.4.1). En esta sección vamos a generalizar las desigualdades K-C para el GDRPP.

Consideremos una partición del conjunto de vértices V en $K+1$ subconjuntos $\{M_0 \cup M_K, M_1, \ldots, M_{K-1}\}$, con $K \geq 3$. Sea $\{I_1, I_2\}$ una partición del conjunto $\{1, 2, \ldots, K-1\}$ tal que (ver Figura 6.5):

- Para todo $j \in I_1$, o bien $1 \in M_j$ o bien existe un H_{m_j} tal que $H_{m_j} \cap A_R(M_j) \neq \emptyset$,
- Para todo $j \in I_2$, existe un arco requerido $a^j \in A_R(M_j)$,
- Los subgrafos inducidos $G(M_0), G(M_1), \ldots, G(M_K)$ son fuertemente conexos.

Por simplicidad en la notación, llamemos $A_R^{OK} = (M_0 : M_K)_R \cup (M_K : M_0)_R$. Sea $F \subseteq A_R^{OK}$ un conjunto de arcos y sea $F^H = \{H_{n_1}, H_{n_2}, ..., H_{n_q}\}$ un conjunto de clientes tales que:

- $|F| + q$ es par,
- $F \cap H_{n_i} = \emptyset \ \ \forall i$,
- $H_{n_i} \cap A_R^{OK} \neq \emptyset \ \ \forall i$, y
- $H_{n_i} \cap H_{n_j} \cap A_R^{OK} = \emptyset \ \ \forall i, \ \forall j$.

Supondremos que si el depósito no está en ningún conjunto M_j, $j \in I_1$, entonces está en M_0. Así, si llamamos $F(x)$ a

$$F(x) = (K-2)\Big(x(M_0 : M_K) + x(M_K : M_0)\Big) + \sum_{\substack{0 \leq i,j \leq K \\ (i,j) \neq (0,K)}} |i-j| x(M_i : M_j),$$

la desigualdad K-C para el GDRPP es:

$$F(x) \geq (K-2)\big(2y(F) - |F|\big) + (K-2)\sum_{i=1}^{q}\big(1 - 2y(H_{n_i} \setminus A_R^{OK})\big) + \\ + 2\sum_{j \in I_1}\big(1 - y(H_{m_j} \setminus A(M_j))\big) + 2\sum_{j \in I_2} y_{a^j} \qquad (6.13)$$

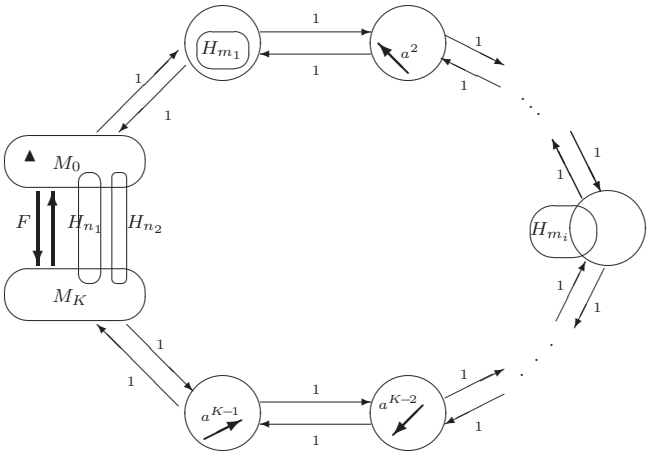

Figura 6.5: *Desigualdad K-C para el GDRPP.*

Notar que $F(x)$ es exactamente la parte izquierda de la desigualdad K-C para el DGRP (ver Sección 4.4.1):

$$F(x) \geq (K-2)|A_R^{OK}| + 2(K-1). \tag{6.14}$$

El hecho de que la desigualdad KC (6.14) sea válida implica que cualquier tour x que recorra un número par $|A_R^{OK}|$ de arcos requeridos entre M_0 y M_K y visite todos los subgrafos $G(M_0 \cup M_K)$, $G(M_1), \ldots, G(M_{K-1})$ de una estructura K-C, cumple que $F(x) \geq (K-2)|A_R^{OK}| + 2(K-1)$.

A diferencia del DGRP, en el GDRPP no es necesario recorrer cada arco de F, pues sus clientes pueden ser servidos desde otros arcos. Tampoco es necesario recorrer algún arco de cada $H_{n_i} \cap A_R^{OK}$, pues estos clientes pueden ser servidos desde arcos de $H_{n_i} \setminus A_R^{OK}$, si estos arcos existen. En el caso de que no existan, es decir, si $H_{n_i} \subseteq A_R^{OK}$, entonces sí es necesario recorrer al menos un arco de H_{n_i}. Así, si $F = \emptyset$ y si $H_{n_i} \subseteq A_R^{OK}$ para todo $i = 1, \ldots, q$, entonces todos los tours para el GDRPP en G tienen que recorrer, al menos, q arcos de A_R^{OK}. Notar que el término correspondiente de la parte derecha de la desigualdad (6.13) es $(K-2)q$, como en la K-C estándar para el DGRP.

Del mismo modo, en el GDRPP tampoco es necesario visitar todos los subgrafos $G(M_0 \cup M_K)$, $G(M_1), \ldots, G(M_{K-1})$, sino sólo aquellos $G(M_j)$, $j \in I_1$, tales que, o bien tienen al depósito o bien $H_{m_j} \subseteq A_R(M_j)$. Notar que si $I_2 = \emptyset$ y si, para cada $j \in I_1$, o bien $1 \in M_j$ o bien $H_{m_j} \subseteq A_R(M_j)$, entonces el término correspondiente de la parte derecha de la desigualdad (6.13) vale $2(K-1)$.

Así, si se cumplen todas las condiciones anteriores, $F = I_2 = \emptyset$, $H_{n_i} \subseteq A_R^{OK}\ \forall i$, y $H_{m_j} \subseteq A_R(M_j)\ \ \forall j$, la desigualdad K-C para el GDRPP (6.13) resulta

$$F(x) \geq (K-2)q + 2(K-1),$$

que es similar a la desigualdad K-C para el DGRP (6.14). Así, la desigualdad K-C para el GDRPP es una generalización de desigualdad K-C para el DGRP.

Teorema 6.2.11

Las desigualdades K-C (6.13) *son válidas para* GDRPP*(G).*

Demostración: Sea (x, y) un tour para el GDRPP en G cualquiera. Denotemos por $c_0 = (K-2)(|F|+q) + 2(K-1)$ (la parte derecha de una desigualdad K-C estándar asociada a una estructura K-C con un número $|F|+q$ par de arcos requeridos entre M_0 y M_K). Tenemos las siguientes posibilidades:

- El tour (x, y) recorre sirviendo todos los arcos de F y todos los arcos $a^j,\ j \in I_2$, no recorre ningún arco de $H_{n_i} \setminus A_{OK}^R$ para ningún n_i y no recorre ningún arco de $H_{m_j} \setminus A_R(M_j)$ para ningún m_j. En este caso, la parte derecha de la desigualdad (6.13) vale $(K-2)|F| + (K-2)q + 2|I_1| + 2|I_2| = (K-2)(|F|+q) + 2(K-1) = c_0$. Notar que si x no recorre ningún arco de $H_{n_i} \setminus A_{OK}^R$, entonces (x, y) tiene que recorrer sirviendo algún arco de $H_{n_i} \cap A_{OK}^R$. Lo mismo ocurre con $H_{m_j} \setminus A_R(M_j)$. Así, tenemos una estructura K-C y un tour x que recorre un número par $|F| + q$ de arcos entre M_0 y M_K y visita todos los subgrafos $G(M_0 \cup M_K)$, $G(M_1), \ldots, G(M_{K-1})$. Como la desigualdad K-C estándar para el DGRP es válida, x cumple que $F(x) \geq c_0$ y (x, y) cumple la desigualdad (6.13).

- El tour (x, y) recorre sirviendo todos los arcos de F menos uno y todos los arcos $a^j,\ j \in I_2$, no recorre ningún arco de $H_{n_i} \setminus A^R_{OK}$ para ningún n_i y no recorre ningún arco de $H_{m_j} \setminus A_R(M_j)$ para ningún m_j. En este caso, la parte derecha de la desigualdad (6.13) vale $(K-2)(|F|+q-2)+2(K-1) = c_0 - 2(K-2)$. El tour (x, y) recorre $|F|+q-1$ arcos entre M_0 y M_K (esto tiene un c-coste de $K-2$ por el número de tales arcos) y, como $|F|+q-1$ es impar, puede visitar todos los subgrafos $G(M_0 \cup M_K)$, $G(M_1), \ldots, G(M_{K-1})$ con un camino $G(M_0),\ G(M_1), \ldots, G(M_{K-1}),\ G(M_K),\ G(M_0)$, que tiene un c-coste de K, luego x cumple que $F(x) \geq (K-2)(|F|+q-1)+K = c_0 - 2(K-2)$ y (x, y) cumple la desigualdad (6.13).

- El tour (x, y) recorre sirviendo todos los arcos de F y todos los arcos $a^j,\ j \in I_2$, recorre sirviendo exactamente un arco de uno de los conjuntos $H_{n_i} \setminus A^R_{OK}$ para algún n_i y no recorre ningún arco de $H_{m_j} \setminus A_R(M_j)$ para ningún m_j. En este caso, la parte derecha de la desigualdad (6.13) vale $(K-2)|F|+(K-2)(q-2)+2|I_1|+2|I_2| = c_0 - 2(K-2)$. Como en el caso anterior, el tour (x, y) recorre un número $|F|+q-1$ (impar) arcos entre M_0 y M_K y visita todos los subgrafos $G(M_0 \cup M_K), G(M_1), \ldots, G(M_{K-1})$, luego el mismo razonamiento anterior nos lleva a que $F(x) \geq (K-2)(|F|+q-1)+K = c_0 - 2(K-2)$ y (x, y) cumple la desigualdad (6.13).

- El tour (x, y) recorre sirviendo todos los arcos de F, todos los arcos $a^j,\ j \in I_2$ menos uno, no recorre ningún arco de $H_{n_i} \setminus A^R_{OK}$ para ningún n_i y no recorre ningún arco de $H_{m_j} \setminus A_R(M_j)$ para ningún m_j. En este caso, la parte derecha de la desigualdad (6.13) vale $(K-2)|F|+(K-2)q+2|I_1|+2|I_2-1| = c_0 - 2$. Tenemos una estructura K-C y un tour x que recorre un número par $|F|+q$ de arcos entre M_0 y M_K y visita todos los subgrafos $G(M_0 \cup M_K)$, $G(M_1), \ldots, G(M_{K-1})$ excepto uno de ellos. Si este tour x tuviese un c-coste menor que $c_0 - 2$, añadiéndole dos arcos de c-coste 1 para conectar el subgrafo que falta obtendríamos un tour para el DGRP en la estructura K-C con c-coste menor que c_0, lo que contradice la validez de la desigualdad K-C estándar. Por

lo tanto, x cumple que $F(x) \geq c_0 - 2$ y (x, y) cumple la desigualdad (6.13).

- El tour (x, y) recorre sirviendo todos los arcos de F y todos los arcos $a^j,\ j \in I_2$, no recorre ningún arco de $H_{n_i} \setminus A^R_{OK}$ para ningún n_i y recorre exactamente un arco de $H_{m_j} \setminus A_R(M_j)$ para algún m_j. En este caso, la parte derecha de la desigualdad (6.13) vale $(K-2)|F| + (K-2)q + 2|I_1-1| + 2|I_2| = c_0 - 2$. Como en el caso anterior, el tour x recorre un número par $|F| + q$ de arcos entre M_0 y M_K y visita todos los subgrafos $G(M_0 \cup M_K)$, $G(M_1), \ldots, G(M_{K-1})$ excepto uno de ellos y el mismo razonamiento nos lleva a que $F(x) \geq c_0 - 2$, luego (x, y) cumple la desigualdad (6.13).

- De un modo similar puede verse que en cualquier otro caso también se cumple la desigualdad. Notar que por cada arco de F que se quede sin recorrer y por cada arco de alguno de los conjuntos $H_{n_i} \setminus \delta(S)$ que se recorra sirviendo, la parte derecha de la desigualdad disminuye en $2(K-2)$ unidades, y que por cada arco a^j que se quede sin recorrer y por cada arco de alguno de los conjuntos $H_{m_j} \setminus A_R(M_j)$ que se recorra sirviendo, la parte derecha de la desigualdad disminuye en 2 unidades. ♦

Un caso particular de las desigualdades K-C

No sabemos si las desigualdades K-C (6.13) definen faceta de GDRPP(G) en general. Sin embargo, hemos probado que, en un caso particular, sí inducen faceta, como indica el siguiente teorema.

Teorema 6.2.12

Consideremos una desigualdad K-C (6.13) *cumpliendo, además, las siguientes condiciones:*

- *el conjunto de arcos F es equilibrado y el conjunto de clientes $F^H = \emptyset$,*

- $H_{m_j} \subseteq A_R(M_j)$ *(o bien* M_j *contiene al depósito)* $\forall\, j \in I_1$,
- *los conjuntos* $(M_i : M_{i+1})$ *y* $(M_{i+1} : M_i)$, $i = 0, .., K-1$ *son no vacíos y no contienen arcos de* A_R,
- *los conjuntos* $(M_i : M_j)$, $|i-j| > 1$, $\{i, j\} \neq \{0, K\}$ *pueden ser o no vacíos, pero no contienen arcos de* A_R,
- *cualquier cliente* H_i *que pueda ser servido por arcos en* $F \cup \{a^j,\ j \in I_2\}$ *también puede ser servido por otro arco que no este en* A_R^{0K}.

Entonces, la desigualdad K-C para el GDRPP

$$F(x) \geq (K-2)(2y(F) - |F|) + 2|I_1| + 2\sum_{j \in I_2} y_{a^j} \tag{6.15}$$

define faceta de GDRPP(G).

Demostración: Tenemos que encontrar $|A| + |A_R| - |V| + 1$ tours (x, y) para el GDRPP en G afínmente independientes que cumplan la desigualdad (6.15) con igualdad. Dividimos el conjunto de clientes en tres tipos:

1. $C_1 = \{H_i \in \mathbb{H} :\ H_i \subseteq A(M_j)\ \ j \in I_1\}$,
2. $C_2 = \{H_i \in \mathbb{H} :\ H_i \cap (F \cup \{a^j, j \in I_2\}) \neq \emptyset\}$,
3. $C_3 = \mathbb{H} \setminus \{C_1 \cup C_2\}$.

Llamemos A_R^i a los conjuntos de los arcos que pueden servir a clientes del conjunto C_i, con $i = 1, 2, 3$, respectivamente. Notar que A_R^1, A_R^2, A_R^3 no tienen por qué formar una partición de A_R, aunque sin pérdida de generalidad supondremos que sí la forman (puesto que vamos a construir un tour para el GDRPP en G asociado a cada arco de cada A_R^i, si no formasen una partición sólo tendríamos que eliminar los tours repetidos).

Consideremos las dos instancias del SCP asociadas a los clientes de C_1 y los clientes de C_3, respectivamente. Existen vectores $y_1^1, \ldots, y_{m_1+1}^1$ y $y_1^3, \ldots, y_{m_3+1}^3$ afínmente independientes que cubren a los clientes de estos conjuntos, con $m_i = |A_R^i|$. Sea y^2 el vector con $y_a = 1$ para todo $a \in F \cup \{a^j,\ j \in I_2\}$. Entonces, el vector $y = y_1^1 + y_1^3 + y^2$ cumple que $y_a = 1 \quad \forall a \in F \cup \{a^j,\ j \in I_2\}$ y, además, que para todo cliente $H_i \in \mathbb{H}$ existe un arco $a' \in H_i$ con $y_{a'} = 1$ y que, si se cumple la última condición del enunciado, entre M_0 y M_K sólo los arcos de F tienen la componente de y igual a 1.

Consideremos la instancia del DGRP en G tomando como arcos requeridos los de $A'_R = F \cup A(M_0) \cup A(M_1) \cup \cdots \cup A(M_K)$. Como F es par y equilibrado, sabemos que $F(x) \geq (K-2)|F| + 2(K-1)$ induce faceta de DGRP(G) y existen $x^1, \ldots, x^m$ con $m = |A| - |V| + 1$ vectores afínmente independiente que pasarán al menos una vez por cada arco con $y_a = 1$, y cumplen $F(x) = (K-2)|F| + 2(K-1)$. Por lo tanto, los tours para el GDRPP $(x^1, y), ..., (x^m, y)$ cumplen la desigualdad K-C (6.15) con igualdad.

A los arcos de A_R^1 les asociamos $(x^1, y_2^1 + y^2 + y_1^3), \ldots, (x^1, y_{m_1+1}^1 + y^2 + y_1^3)$, que son $|A_R^1|$ tours para el GDRPP en G que también cumplen $F(x) = (K-2)|F| + 2(K-1)$, luego cumplen la desigualdad K-C (6.15) con igualdad.

De un modo similar, a los arcos de A_R^3 les asociamos los siguientes $|A_R^3|$ tours para el GDRPP en G: $(x^1, y_1^1 + y^2 + y_2^3), \ldots, (x^1, y_1^1 + y^2 + y_{m_3+1}^3)$ que también cumplen $F(x) = (K-2)|F| + 2(K-1)$, luego cumplen la desigualdad K-C (6.15) con igualdad.

Para los arcos de A_R^2, distinguiremos varios casos:

- Dado un arco $a^j,\ j \in I_2$, como $A(M_j)$ no contiene ningún conjunto H_i, cualquier cliente que puede ser servido con arcos de $A(M_j)$ puede ser servido por otros arcos que no estén en este conjunto. Además, como estamos exigiendo que los conjuntos $(M_i : M_{i+1})$ y $(M_{i+1} : M_i)$, $i = 0, .., K-1$

sean no vacíos, podemos construir un tour x como el que se muestra en la Figura 6.6, que recorre dos arcos opuestos entre cada par de nodos consecutivos $M_0, ..., M_{i-1}, M_{i+1}, ..., M_K$, más los de F, más los arcos requeridos de $A(M_0) \cup ... \cup A(M_{i-1}) \cup A(M_{i+1}) \cup ... \ A(M_K)$, con lo que $F(x) = (K-2)|F| + 2(K-2)$. Si definimos y como el vector que vale 1 en los arcos de F y en los arcos requeridos de $A(M_0) \cup ... \cup A(M_{i-1}) \cup A(M_{i+1}) \cup ... \ A(M_K)$, y vale cero en los demás arcos requeridos, tenemos que (x, y) es un tour para el GDRPP en G que cumple la desigualdad K-C (6.15) con igualdad y tiene $y_{a^j} = 0$. Así obtenemos $|I_2|$ tours para el GDRPP en G.

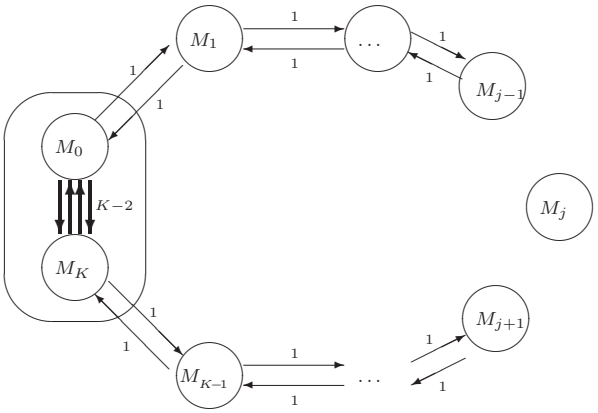

Figura 6.6: *Tour de la demostración del Teorema 6.2.12 asociado al arco $a^j, j \in I_2$.*

- Para cada arco $a \in F$, definimos un tour para el DGRP en G como en la Figura 6.7, que recorre todos los arcos requeridos de $A(M_0) \cup ... \cup A(M_K)$ y de $F \setminus \{a\}$ y, dado que $|F| - 1$ es impar, visita los conjuntos M_i con $i = 1, \ldots, K$ con un camino $G(M_0)$, $G(M_1), \ldots, G(M_{K-1})$, $G(M_K)$, $G(M_0)$. Este tour cumple que $F(x) = 2(K-1)(|F|-1) + K$. Además, el vector y que vale 1 en todos los arcos de $F \setminus \{a\}$ y en todos los arcos requeridos de $A(M_0) \cup ... \cup A(M_K)$ y vale cero en los demás arcos requeridos, y sirve a todos los clientes, luego (x, y) es un tour para el GDRPP en G que cumple la desigualdad K-C (6.15) con igualdad y tiene $y_a = 0$. De este modo hemos construido $|F|$ tours.

- Para cada arco $a \in A_R^2 \setminus F \cup \{a^j, j \in I_2\}$, existe un M_i tal que $a \in M_i$ y

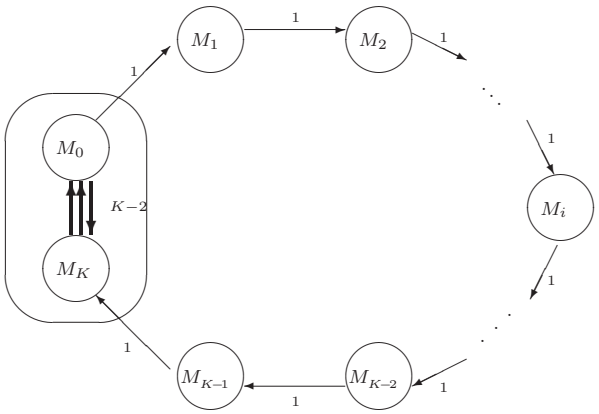

Figura 6.7: *Tour de la demostración del Teorema 6.2.12 asociado al arco $a \in F$.*

como M_i es fuertemente conexo, existe un ciclo C^a en M_i que recorre el arco a. Definimos el tour para GDRPP en G, $(x^1 + C^a, y)$ con $y_a = 1$, y sirviendo todos los arcos de $F \cup \{a^j, j \in I_2\} \cup A(M_K) \cup A(M_0) \cup (\bigcup_{i \in I_1} A(M_i))$. Este tour cumple la desigualdad K-C (6.15) con igualdad.

Así, hemos construido $|A| - |V| + 1 + |A_R|$ tours para GDRPP en G que cumplen la desigualdad K-C (6.15) con igualdad. Si los expresamos como vectores fila obtenemos la matriz de la Figura 6.8, y si restamos la primera fila a todas las demás obtenemos la matriz de la Figura 6.9, cuyos vectores son linealmente independientes, luego los $|A|-|V|+1+|A_R|$ tours para GDRPP en G anteriores son afínmente independientes y la desigualdad K-C (6.15) induce una faceta de GDRPP(G). ♦

			A_R^2		
A	A_R^1	A_R^3	F	I_2	Resto
x^1 $\vdots$ x^m	y_1^1	y_1^3	1	1	0
x^1	y_2^1 $\vdots$ $y_{m_1+1}^1$	y_1^3	1	1	0
x^1	y_1^1	y_2^3 $\vdots$ $y_{m_2+1}^3$	1	1	0
$*$	1	1	1-I	1	0
$*$	1	1	1	1-I	0
$*$	1	1	1	1	I

Figura 6.8: *Matriz 1 de la demostración del Teorema 6.2.12.*

			A_R^2		
A	A_R^1	A_R^3	F	I_2	Resto
$x^2 - x^1$ $\vdots$ $x^m - x^1$	0	0	0	0	0
0	$y_2^1 - y_1^1$ $\vdots$ $y_{m_1+1}^1 - y_1^1$	0	0	0	0
0	0	$y_2^3 - y_1^3$ $\vdots$ $y_{m_2+1}^3 - y_1^3$	0	0	0
$*$	1	1	1-I	1	0
$*$	1	1	1	1-I	0
$*$	1	1	1	1	I

Figura 6.9: *Matriz 2 de la demostración del Teorema 6.2.12.*

6.2.6. Desigualdades de Dominancia

Como hemos comentado en la Sección 6.2.1, el GDRPP puede verse como la resolución, primero, de un Problema de Cubrimiento de Conjuntos (Set Covering Problem, SCP) en el que elegimos un conjunto de arcos que sirvan a todos los clientes y la resolución, después, de un DGRP que encuentre la ruta óptima que, partiendo del depósito, atraviese aquellos arcos. Por esa razón, muchas de las desigualdades típicas del SCP pueden ser usadas para reforzar la formulación que hemos presentado del GDRPP. En esta sección presentamos una de las clases de desigualdades para el SCP ya utilizadas en Hà et al. (2013) y que nosotros también usaremos en nuestro algoritmo de branch and cut.

Dados dos arcos $(i_1, j_1),\ (i_2, j_2) \in A_R$, se dice que el primero domina al segundo si para todo cliente $H_c \in \mathbb{H}$ tal que $(i_2, j_2) \in H_c$ entonces $(i_1, j_1) \in H_c$. Las desigualdades de dominancia se definen como

$$y_{i_1 j_1} + y_{i_2 j_2} \leq 1$$

para cualquier par de arcos $(i_1, j_1),\ (i_2, j_2) \in A_R$ tales que el primero domina al segundo, o viceversa. Se basan en que, si (i_1, j_1) domina a (i_2, j_2), los tours que recorren sirviendo el arco (i_1, j_1) (cumplen que $y_{i_1 j_1} = 1$), no tienen por qué recorrer sirviendo el arco (i_2, j_2) (y pueden cumplir que $y_{i_2 j_2} = 0$). Aunque estas desigualdades no son válidas para el GDRPP(G), porque pueden haber tours con $y_{i_1 j_1} = 1$ e $y_{i_2 j_2} = 1$, sí podemos utilizarlas en nuestro algoritmo de branch and cut pues existe, al menos, un tour óptimo que las satisface.

6.3. Otra formulación para el GDRPP

La formulación que hemos utilizado en las secciones anteriores usa una variable x_{ij} para indicar el número de veces que el arco es recorrido y otra variable y_{ij} para

indicar si desde ese arco se sirve a algún cliente o no. Estamos usando $|A| + |A_R|$ variables y puede ser interesante pensar en una formulación con menos variables. En concreto, una formulación que no utilice las variables y_{ij}, sobre todo en aquellos casos en los que existe un número elevado de arcos que pueden servir clientes.

Así, vamos a asociar únicamente una variable x_{ij} a cada arco, representando el número de veces que éste es recorrido. En este caso no diferenciamos entre si el arco requerido es recorrido y servido al mismo tiempo, o no. Se entiende que si $x_{ij} \geq 1$, $(i,j) \in A_R$, el arco es servido la primera vez que es atravesado. Así, la nueva formulación que proponemos es la siguiente:

$$\text{Minimize} \quad \sum_{(i,j)\in A} c_{ij}x_{ij}$$

$$s.a:$$

$$x(\delta^+(i)) = x(\delta^-(i)), \qquad \forall i \in V \tag{6.16}$$

$$x(\delta^+(S)) \geq 1 - x(H_c \cap A(V \setminus S)), \forall S \subseteq V \setminus \{1\}, \forall H_c \in \mathbb{H} \tag{6.17}$$

$$\sum_{(i,j)\in H_c} x_{ij} \geq 1, \quad \forall H_c \in \mathbb{H} \tag{6.18}$$

$$x_{ij} \geq 0, \quad \forall (i,j) \in A \tag{6.19}$$

$$x_{ij} \quad \textit{entero}, \quad \forall (i,j) \in A \tag{6.20}$$

Las desigualdades (6.16) son las de **simetría**. En este caso sólo tenemos un tipo de desigualdades de **conectividad** (6.17), en las que se obliga a que la cortadura de cualquier conjunto S sea mayor o igual a uno si algún cliente no ha sido servido con arcos de $V \setminus S$. Finalmente, tenemos las desigualdades de **servicio** (6.18), las **triviales** (6.19) y las de integridad (6.20).

6.3.1. Desigualdades de Imparidad

Hemos propuesto unas nuevas desigualdades de imparidad y K-C que deben ser satisfechas por las soluciones de la formulación anterior. En esta nueva formulación no hay variables y que definan los arcos que son servidos, pero aún así podemos reescribir algunas desigualdades de la formulación anterior utilizando sólo variables x.

Las desigualdades de imparidad para las soluciones del tipo (x, y) venían dadas por

$$x(\delta(S)) \geq 2y(F) - |F| + \sum_{i=1}^{q}(1 - 2y(H_{n_i} \setminus \delta(S))) + 1,$$

donde $S \subset V$, $F \subseteq \delta_R(S)$ y $F^H = \{H_{n_1}, H_{n_2}, ..., H_{n_q}\}$, con $|F| + q$ impar y $q > 0$ cumpliendo:

- $H_{n_i} \cap H_{n_j} \cap \delta(S) = \emptyset \quad \forall n_i, n_j$.
- $F \cap H_{n_i} = \emptyset \quad \forall n_i$.
- $H_{n_i} \cap \delta(S) \neq \emptyset \quad \forall n_i \quad (H_{n_i} \setminus \delta(S)$ puede ser, o no, vacío).

Ahora no tiene sentido que definamos el conjunto F, ya que las variables x_{ij} pueden valer más de uno. Definimos entonces la desigualdad de imparidad como

$$x(\delta(S)) \geq \sum_{i=1}^{q}(1 - 2x(H_{n_i} \setminus \delta(S))) + 1,$$

donde $S \subset V$ y $F^H = \{H_{n_1}, H_{n_2}, ..., H_{n_q}\}$, con q impar y cumpliendo:

- $H_{n_i} \cap H_{n_j} \cap \delta(S) = \emptyset \quad \forall n_i, n_j$.

- $H_{n_i} \cap \delta(S) \neq \emptyset \ \ \forall n_i \ \ (H_{n_i} \backslash \delta(S)$ puede ser, o no, vacío).

6.3.2. Desigualdades K-C

Como en el caso anterior podemos reescribir algunas desigualdades K-C para la nueva formulación de manera que únicamente dependan de las variables x.

Consideramos una partición del conjunto de nodos V en $K+1$ subconjuntos $\{M_0 \cup M_K, M_1, \ldots, M_{K-1}\}$, $K \geq 3$, tal que los subgrafos $G(M_0)$, $G(M_1), \ldots, G(M_K)$ son fuertemente conexos (Figura 6.10). Supongamos que para todo $j \in \{1, \ldots, K-1\}$, o bien $1 \in M_j$ o bien existe un H_{m_j} tal que $H_{m_j} \cap A_R(M_j) \neq \emptyset$ (si el depósito no está en ningún conjunto M_j, $j \in \{1, \ldots, K-1\}$, entonces supondremos que está en M_0). Por simplicidad en la notación, llamemos $A_R^{OK} = (M_0 : M_K)_R \cup (M_K : M_0)_R$. Sea $F^H = \{H_{n_1}, H_{n_2}, ..., H_{n_q}\}$ un conjunto de clientes tales que:

- $q = |F^H|$ es par,
- $H_{n_i} \cap A_R^{OK} \neq \emptyset \ \ \forall i$, y
- $H_{n_i} \cap H_{n_j} \cap A_R^{OK} = \emptyset \ \ \forall i, \ \forall j$.

Llamamos desigualdades K-C para la formulación del GDRPP sólo con las variables x a

$$F(x) \geq (K-2) \sum_{i=1}^{q} \left(1 - 2x(H_{n_i} \setminus A_R^{OK})\right) + 2 \sum_{j=1}^{K-1} \left(1 - x(H_{m_j} \setminus A(M_j))\right) \qquad (6.21)$$

donde, de nuevo, $F(x)$ es exactamente la parte izquierda de la desigualdad K-C para el DGRP (ver Sección 4.4.1).

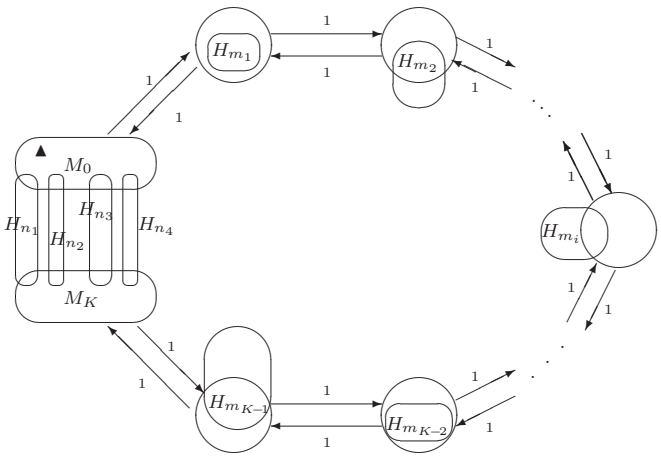

Figura 6.10: *Desigualdad K-C para el GDRPP formulado sólo con las variables x.*

Capítulo 7

Un algoritmo de branch and cut para el GDRPP

En el Capítulo 5 hemos presentado un algoritmo de tipo branch & cut para el DGRP. En este capítulo presentamos otro para el Problema Generalizado del Cartero Rural en un grafo dirigido (GDRPP). El esquema que seguiremos es muy parecido al utilizado en el Capítulo 5 y haremos referencia a él cuando la metodología usada sea la misma.

7.1. Problema lineal inicial y algoritmo de planos de corte

En esta sección presentamos todos los aspectos relativos al procedimiento de planos de corte utilizado dentro del algoritmo de branch & cut para el GDRPP.

7.1.1. Problema inicial

Como ya hemos comentado, un algoritmo de branch & cut comienza con una primera relajación lineal de un problema lineal entero. El problema lineal (PL) inicial que resolvemos en el nodo raíz del árbol es el siguiente:

$$\text{Minimize} \qquad \sum_{(i,j)\in A} c_{ij}x_{ij}$$

s.a.:

$$\begin{aligned}
x(\delta^+(0)) &\geq 1 && (7.1)\\
x(\delta^+(i)) - x(\delta^-(i)) &= 0, \quad \forall i \in V && (7.2)\\
x(\delta^-(H_c)) &\geq 1, \quad \forall H_c \in \mathbb{H} && (7.3)\\
\sum_{(i,j)\in H_c} y_{ij} &\geq 1, \quad \forall H_c \in H && (7.4)\\
x_{ij} &\geq y_{ij}, \quad \forall (i,j) \in A_R && (7.5)\\
y_{i_1j_1} + y_{i_2j_2} &\leq 1 \text{ si } (i_1,j_1) \text{ domina a } (i_2,j_2) && (7.6)\\
x_{ij} &\geq 0, \quad \forall (i,j) \in A_{NR} && (7.7)\\
0 \leq y_{ij} &\leq 1 \qquad \forall (i,j) \in A_R && (7.8)
\end{aligned}$$

Este PL incluye una desigualdad que obliga a la solución a estar conectada con el depósito (7.1), las ecuaciones de simetría (7.2) y una desigualdad de conectividad por cada conjunto H_c (7.3). También incluye a las desigualdades de servicio (7.4), obligatoriedad (7.5) y las desigualdades triviales (7.7, 7.8). Por último, hemos añadido las desigualdades de dominancia (7.6), que no estaban incluidas en la formulación original, y se ha relajado la restricción de integridad de las variables x e y. Así, y como en el caso para el DGRP, sólo si la solución resulta ser entera y conexa será la solución óptima del problema original.

7.1.2. Algoritmo de planos de corte

La metodología usada en cada iteración del algoritmo de planos de corte es muy similar a la del procedimiento usado para el DGRP. Los pasos del algoritmo son los siguientes:

- Se ejecuta un algoritmo heurístico de separación para las desigualdades de conectividad.
- Si el heurístico anterior no ha encontrado ninguna desigualdad de conectividad violada, aplicamos un algoritmo exacto de separación.
- Finalmente, si no se han encontrado desigualdades de conectividad violadas, se utilizan algoritmos heurísticos para la separación de las desigualdades de imparidad y K-C.

El algoritmo de planos de corte es aplicado hasta que no se encuentra ninguna desigualdad violada nueva o se satisface un criterio de parada (*tailing-off*).

7.1.3. Cota superior

A menudo, encontrar una solución factible únicamente a través del árbol de ramificación resulta un proceso lento. Así, y con idea de mejorar la eficiencia del algoritmo de branch & cut, hemos usado un heurístico que proporcione una buena cota superior que nos permita saturar el mayor número de nodos posibles.

El heurístico utilizado se basa en:

- Resolver un problema de set covering para seleccionar un conjunto de arcos A' que cubra (sirva) a todos los clientes.

- Dado A', se resuelve una instancia del DGRP en la que ese subconjunto de arcos define el conjunto de arcos requeridos.

Más concretamente, consideremos la solución fraccionaria (x^*, y^*) del PL resuelto en cualquier iteración del procedimiento de planos de corte. Comenzando por aquellos arcos a con mayor valor y_a^*, seleccionamos un subconjunto de arcos, $A' \subseteq A_R$, que sirvan a todos los clientes. Para la elección de los arcos, además, se tiene en cuenta el número de clientes aún no servidos que el arco considerado puede servir y el grado de entrada y de salida de sus vértices finales en el grafo inducido por los arcos ya seleccionados. Una vez elegido A', se utiliza el algoritmo de branch & cut propuesto para el DGRP para buscar un tour de mínimo coste que atraviese todos sus arcos y visite el depósito.

Este algoritmo heurístico se ejecuta cada 100 iteraciones del algoritmo de planos de corte realizadas en el nodo cero del árbol. Fuera del nodo cero, el heurístico se aplica cada 20 nodos (si se han explorado menos de 200 nodos), cada 50 (cuando se han explorado entre 200 y 500) y, finalmente, cada 200 nodos (si se han explorado más de 500).

7.1.4. Tailing-off

Como criterio de parada se ha usado el incremento de la función objetivo durante las últimas iteraciones. Si en las últimas 20 iteraciones ese incremento es menor del 0.00001 % en el nodo raíz, detenemos la ejecución del algoritmo de planos de corte y pasamos a ramificar. En cualquier otro nodo del árbol de ramificación, la decisión de parar se toma si el incremento en las 5 últimas iteraciones es menor del 0.00005 %.

7.1.5. Ramificación

Volvemos a usar la estrategia **strong branching** para elegir la variable a ramificar. Para la elección del nodo a estudiar utilizamos la estrategia **best bound**, es decir, una vez se ha saturado un nodo, el siguiente a estudiar es aquél con mejor valor de la función objetivo.

7.2. Algoritmos de separación

En esta sección presentamos los procedimientos que hemos utilizado para la resolución de los problemas de separación correspondientes a las distintas familias de desigualdades descritas en el capítulo anterior.

7.2.1. Desigualdades de conectividad

El algoritmo usado para la separación de las desigualdades de conectividad es el siguiente. Se genera primero el grafo inducido por los arcos con valor $y_a \geq \varepsilon$ o $x_a \geq \varepsilon$, donde ε es un parámetro. Si el grafo resultante no es conexo consideramos las cortaduras definidas por sus componentes débilmente conexas $C_1, ..., C_q$ para comprobar si las correspondientes restricciones de conectividad son violadas o no por la solución.

Se estudian únicamente las cortaduras de las componentes C_i que contengan al menos un arco requerido, es decir un arco a tal que $y_a \geq \varepsilon$. Como se ha visto, hemos encontrado distintos tipos de restricciones de conectividad. Según las características de la componente conexa considerada, usaremos unas u otras. Los casos a tener en

cuenta son los siguientes:

- Que exista un $H_c \in \mathbb{H}$ tal que $H_c \subseteq \delta(C_i) \cup A(C_i)$. En este caso se comprueba si la desigualdad $x(\delta^-(C_i)) \geq 1$ está violada. Si lo está, se añade la restricción al problema.
- Que no exista tal H_c. Buscamos entonces el arco $a_{max} \in A(C_i)$ con valor máximo de y, y_{max}, y chequeamos la violación de la desigualdad $x(\delta^-(C_i)) \geq y_{max}$, y buscamos el cliente H_c tal que $y(H_c \cap A(V \setminus C_i))$ es mínimo y chequeamos la violación de la desigualdad $x(\delta^+(C_i)) \geq 1 - y(H_c \cap A(V \setminus C_i))$.

El algoritmo exacto de conectividad consiste en resolver un problema de flujo máximo entre el depósito y cada vértice de un arco requerido, para buscar las cortaduras de peso mínimo que puedan violar las desigualdades de conectividad. Una vez encontrada la cortadura mínima $\delta(S)$, para un conjunto S de vértices, procedemos como hacíamos en el heurístico a comprobar si se violan o no las diferentes desigualdades de conectividad.

7.2.2. Desigualdades de imparidad

Para buscar las desigualdades de imparidad violadas por la solución (x^*, y^*), construimos el grafo G^* inducido por los arcos que cumplen $x^*_a - y^*_a \geq \varepsilon$, si el arco es requerido, y $x^*_a \geq \varepsilon$, si no lo es. Sean C_1,...,C_k sus componentes débilmente conexas. Calculamos la cortadura de cada una de ellas y estudiamos aquellas que son R-impares, es decir, aquellas que cumplen que $y^*(\delta(S))$ es impar.

Diremos que una componente conexa es del tipo $2n+1$, si se cumple que $2n+0.75 \leq y^*(\delta(C_i)) \leq 2n + 1.25$, y definimos $Tipo = 2n + 1$. Elegimos los arcos a tales que $x^*_a \approx 1$, que van a definir el conjunto F. Sean ahora c_1,...,c_l los clientes que pueden ser servidos por los arcos de la cortadura de C_i y que no pueden ser servidos por

arcos de F, de manera que los conjuntos H_{c_i} sean disjuntos y tengan el menor valor $y^*(H_{c_i} \setminus \delta(C_i))$. Si se cumple que el número de tales clientes es igual a $Tipo - |F|$, entonces construimos la desigualdad de imparidad y comprobamos si se viola o no.

7.2.3. Desigualdades K-C

El algoritmo heurístico usado para la separación de las K-C es similar al utilizado para la separación de las correspondiente versión de desigualdades K-C para el DGRP. Dada una solución fraccionaria (x^*, y^*), tomamos como arcos requeridos aquellos con $y_a^* \geq \varepsilon$. Los arcos requeridos definen los R-sets. Para cada R-set probamos a dividirlo en M_0 y M_K de modo que entre ellos haya un valor de y^* par (aproximadamente) y buscamos los arcos que definen el conjunto F y los conjuntos H_c que definen F^H de un modo similar al descrito anteriormente para las desigualdades de imparidad. Una vez encontrada la estructura K-C chequeamos si está violada la desigualdad K-C correspondiente.

7.3. Resultados computacionales

En esta sección presentamos las instancias del GDRPP utilizadas para comprobar la eficiencia del algoritmo de branch & cut propuesto. El algoritmo ha sido programado en $C++$ y usa Cplex 12.4. Se ha habilitado la opción de Cplex que le permite generar y añadir cortes $0 - 1/2$ violados. Hemos fijado un tiempo limite de dos horas y se ha ejecutado en un PC Intel Core i7 a 3.40 GHz con 16 Gb RAM.

7.3.1. Instancias para el GDRPP

Puesto que Hà et al. (2013) han propuesto un algoritmo de branch & cut para este problema (al que recordemos se refieren como el Close Enough Arc Routing Problem), era imprescindible comparar nuestro algoritmo con el suyo. Así pues, cuando ha sido posible, hemos utilizado sus mismas instancias.

El primer conjunto de instancias utilizadas por Hà et al. (2013) ha sido generado en la forma siguiente. Los vértices han sido seleccionados aleatoriamente en el cuadrado unidad. Los arcos se han generado también de manera aleatoria intentando imitar una red de carreteras y su coste corresponde a la distancia euclídea entre sus vértices. El número de clientes viene dado por la expresión $q = m * t$, donde m es el número de arcos y t es un parámetro que toma valores en $\{0.5, 1, 5, 10\}$, y los clientes han sido elegidos aleatoriamente en el cuadrado unidad. Para elegir los arcos requeridos se ha usado la distancia euclídea a los clientes, es decir, si un cliente está a una distancia menor que $r \in \{150, 200\}$ de un arco, éste puede servirlo y es declarado requerido.

La Tabla 7.1 muestra las características de las instancias. El nombre ce200-0.5, por ejemplo, representa $r = 200$ y $t = 0.5$. $|V|$ y $|A|$ son, respectivamente, el número de vértices y arcos del grafo y $|\mathbb{H}|$ es el número de clientes. $Gap0$ es el gap en el nodo raíz del árbol de ramificación, *Resueltas* da el número de instancias resueltas, $Nodos$ es el número de nodos explorados del árbol y $Tiempo$ es el tiempo de computación en segundos. Hay 5 instancias por cada conjunto de parámetros. Los valores mostrados en la tabla corresponden a la media de los obtenidos para las 5 instancias.

Se puede ver que el branch & cut ha podido resolver todas las instancias en un tiempo razonablemente pequeño, así como que el gap en el nodo 0 es muy pequeño. Al parecer, la dificultad de las instancias depende fundamentalmente de la proporción

de clientes con respecto al número de arcos del grafo. Cuanto más clientes hay, más arcos serán considerados como arcos requeridos y la instancia es más fácil de resolver. En lo que respecta a r, las instancias parecen ser más difíciles de resolver cuanto menor es su valor. Esto también parece tener que ver con la cantidad de arcos que se consideran requeridos: cuando r es pequeño, menor es el conjunto de arcos requeridos y más difícil la instancia.

La Tabla 7.2 muestra las medias de los cortes añadidos para las cinco instancias probadas de cada tipo. La última columna (Z-H) proporciona el promedio de los cortes $0 - 1/2$ añadidos por Cplex. Observemos que las desigualdades que más se añaden al problema son las de conectividad e imparidad, y que su número disminuye al aumentar r y el número de clientes.

	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Gap0 (%)	Resueltas	Nodos	Tiempo
ce200-0.5	500	1000	500	0,65	5	148,8	245,7
ce200-1	500	1000	1000	0,28	5	47,8	88,6
ce200-5	500	1000	5000	0,08	5	16,8	28,3
ce200-10	500	1000	10000	0,11	5	12,2	20,3
ce150-0.5	500	1500	750	0,49	5	1025,8	830,5
ce150-1	500	1500	1500	0,38	5	1924,2	1235,5
ce150-5	500	1500	7500	0,12	5	30,8	49,2
ce150-10	500	1500	15000	0,13	5	43,2	50,1

Tabla 7.1: *Resultados computacionales en las instancias "ce".*

En la Tabla 7.3 se comparan los resultados obtenidos por el B&C de Hà et al. (2013) con los obtenidos por el B&C propuesto por nosotros. Hay que señalar que

	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Conectividad	Paridad	K-C	Z-H
ce200-0.5	500	1000	500	9892,0	1311,6	2,8	175,0
ce200-1	500	1000	1000	4985,0	997,8	24,0	165,6
ce200-5	500	1000	5000	1093,0	668,0	6,8	121,8
ce200-10	500	1000	10000	429,8	583,4	3,0	124,8
ce150-0.5	500	1500	750	8118,2	2405,2	39,4	413,4
ce150-1	500	1500	1500	4871,8	1897,0	23,2	371,8
ce150-5	500	1500	7500	403,4	745,8	6,2	166,6
ce150-10	500	1500	15000	412,2	734,6	3,6	173,6

Tabla 7.2: *Cortes añadidos en las instancias "ce".*

su algoritmo ha sido ejecutado en un PC con CPU a 2.4 GHz con 6 Gb de RAM y utilizando también las desigualdades $0-1/2$ de Cplex, razón por la que las usamos nosotros también. La columna "Gap0" muestra ahora la mejora producida en el Gap en el nodo cero por nuestro algoritmo respecto del obtenido por el algoritmo de Hà et al. Podemos observar que nuestro algoritmo es capaz de resolver un número mayor de instancias y que mejora el Gap en el nodo cero obtenido por el otro algoritmo en hasta dos puntos en algunos de los casos.

				Ha et al.		Nuestros resultados		
	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Resueltas	Tiempo	Resueltas	Tiempo	Gap0 (%)
ce200-0.5	500	1000	500	3	4155,6	5	245,7	2,54
ce200-1	500	1000	1000	4	2447,8	5	88,6	2,11
ce200-5	500	1000	5000	5	315,1	5	28,3	0,87
ce200-10	500	1000	10000	5	82,3	5	20,3	0,79
ce150-0.5	500	1500	750	0	7202,9	5	830,5	2,04
ce150-1	500	1500	1500	2	4499,9	5	1235,5	1,43
ce150-5	500	1500	7500	5	154,5	5	49,2	0,47
ce150-10	500	1500	15000	5	205,9	5	50,1	0,26

Tabla 7.3: *Comparación con los resultados de Hà et al. (2013)*

El segundo conjunto de instancias utilizado también ha sido propuesto por Hà et al. (2013). Se trata de instancias definidas sobre grafos mixtos. A partir de dos instancias del RPP sobre un grafo mixto, MB537 y MB547, propuestas por Corberán et al. (2006), Hà et al. las trasforman en instancias del GDRPP sustituyendo primero cada arista por dos arcos paralelos con el mismo coste y aplican, a continuación, el procedimiento descrito para las instancias "ce". La única diferencia con el anterior método de generación de instancias es que ahora r viene determinado por la longitud media de todos los arcos de grafo. Para cada grafo y para cada valor de t, se han generado 5 instancias. El grafo mixto de la instancia original MB357 tiene 500 vértices, 364 aristas y 476 arcos, mientras que el grafo de MB547 tiene 500 vértices, 351 aristas y 681 arcos.

La Tabla 7.4 muestra los resultados obtenidos por nuestro algoritmo en las instancias del GDRPP definidas sobre grafos mixtos. Como en el caso anterior, se observa que el algoritmo es capaz de resolver casi todas las instancias. De nuevo los gaps obtenidos en el nodo raíz son muy pequeños y los tiempos de computación razonables.

	Gap0 (%)	Resueltas	Nodos	Tiempo
MB0537-0.5	0,16	5/5	14,6	456,3
MB0537-1	0,20	5/5	36,8	368,5
MB0537-5	0,19	5/5	34,0	223,8
MB0537-10	0,18	5/5	73,8	233,4
MB0547-0.5	1,17	4/5	2685,8	2854,3
MB0547-1	0,78	5/5	1297,6	1515,2
MB0547-5	0,19	5/5	176,8	232,1
MB0547-10	0,18	5/5	45,4	131,5

Tabla 7.4: *Resultados sobre las instancias del GDRPP definidas sobre grafos mixtos.*

En la Tabla 7.5 podemos ver la comparación de los resultados obtenidos por nuestro algoritmo con los resultados del algoritmo de Ha et al. (2013) sobre las instancias definidas en grafos mixtos. Nuestro B&C es capaz de resolver un número mayor de instancias y, respecto al gap en el nodo cero, podemos comprobar que en las instancias con menos clientes llegamos a conseguir una mejora de hasta 5 puntos, aunque su procedimiento obtiene un mejor gap cuando las instancias tienen un mayor número de clientes. Nótese la mayor dificultad de las instancias del GDRPP definidas sobre grafos mixtos.

El último conjunto de instancias procede de instancias para el problema del Cartero Rural en un grafo no dirigido usadas en Corberán, Plana y Sanchis (2007) (UR500, UR750, UR1000). Para convertirlos en grafos dirigidos cada arista ha sido transformada en un par de arcos opuestos con el mismo coste. En cuanto a los clientes, en este caso los hemos generado de forma distinta: hemos creado un cliente por cada R-set del grafo original (así en este caso los conjuntos H_c son disjuntos). Hay doce instancias en cada grupo. Las instancias denominadas UR500 tienen $298 \leq |V| \leq 499$, $597 \leq |A| \leq 1526$ y $1 \leq |\mathbb{H}| \leq 99$. Las instancias UR750 tienen $452 \leq |V| \leq 749$, $915 \leq |A| \leq 2314$ y $1 \leq |\mathbb{H}| \leq 140$. Finalmente, las del conjunto UR1000 tienen $605 \leq |V| \leq 1000$, $2289 \leq |A| \leq 3083$ y $1 \leq |\mathbb{H}| \leq 204$.

En la Tabla 7.6 podemos ver las características medias del último conjunto de instancias para el GDRPP y los resultados obtenidos. El branch & cut propuesto es capaz de resolver todas las instancias del tipo UR500, 10 de 12 de UR750 y únicamente 2 de UR100. De hecho, en una de las instancias UR100 ni siquiera es capaz de encontrar una solución posible en el tiempo límite fijado de dos horas. Aun así, el gap en el nodo cero (con respecto al óptimo o a la mejor solución encontrada) y el gap final es pequeño en todos los casos. Si nos fijamos en el tiempo de computación, parece ser que el hecho de que los clientes sean disjuntos influye en la dificultad de las instancias. Hay que señalar, sin embargo, que el algoritmo y los procedimientos de separación no habían considerado esta posibilidad, pues estaban diseñados para

				Ha et al.		Nuestros resultados		
	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Resueltas	Tiempo	Resueltas	Tiempo	Gap0 (%)
MB0537-0.5	500	1204	400	0	7200,7	5	456,3	5,7
MB0537-1	500	1204	800	0	7201,5	5	368,5	5,7
MB0537-5	500	1204	4000	3	3937,8	5	223,8	0
MB0537-10	500	1204	8000	5	2418,3	5	233,4	-1
MB0547-0.5	500	1383	520	0	7201,1	4	2854,3	5
MB0547-1	500	1383	1040	0	7202,2	5	1515,2	3,5
MB0547-5	500	1383	5200	4	1639,9	5	232,1	-0,3
MB0547-10	500	1383	10400	5	756,2	5	131,5	-1

Tabla 7.5: *Comparación con los resultados de Ha et al. en las instancias "MB".*

	$\|V\|$	$\|A\|$	$\|\mathbb{H}\|$	Gap0 (%)	Gap (%)	Resueltas	Nodos	Tiempo
UR500	446,0	2257,8	35,3	0,24	0,00	12/12	320,2	790,0
UR750	665,7	3396,8	55,7	0,32	1,18	10/12	1590,0	3247,0
UR1000	882,2	4580,8	74,8	3,58	4,34	2/12	1262,0	6016,3

Tabla 7.6: *Resultados en instancias definidas sobre grafos no dirigidos con clientes disjuntos.*

la resolución de instancias en las que los clientes podían tener elementos comunes. Lo mismo se puede decir de la cota superior. El heurístico descrito en el apartado 7.1.3 no funcionó bien en este caso y prácticamente la resolución de estas instancias se hizo sin disponer de una cota superior. Pensamos que una implementación más apropiada para el caso de clientes disjuntos conduciría a mejores soluciones en este caso.

7.3.2. Resultados con la formulación alternativa

En el Capítulo 6 proponíamos, además de la formulación con variables x e y, una formulación con un único conjunto de variables x, y adaptábamos las desigualdades

de conectividad, imparidad y K-C a esta formulación. Hemos adaptado en el mismo sentido el branch & cut descrito anteriormente. Básicamente, lo que se ha hecho es reescribir los algoritmos de separación para adaptarlos a las nuevas desigualdades. Para comprobar y comparar las posibilidades que tiene esta formulación con respecto a la primera, hemos elegido el primer conjunto de instancias para el GDRPP propuesto por Hà et al. (2013).

En la Tabla 7.7 podemos ver los resultados obtenidos con la formulación que utiliza variables x únicamente (F2). Estos resultados los hemos comparado con los que habíamos obtenido con la primera formulación (F1). Se puede observar que los resultados obtenidos con la formulación F2 son bastante peores que los obtenidos con la F1 en lo que se refiere a óptimos encontrados, gap y tiempos de computación. Sin embargo, queremos señalar que la comparación presentada no ha sido muy justa en lo que a la formulación F2 se refiere. La razón es que no hemos estudiado a fondo las posibilidades de F2, únicamente hemos adaptado las desigualdades y los algoritmos que habíamos desarrollado para F1 a la nueva formulación. Pensamos que un estudio más profundo de la formulación F2 puede conducir a la obtención de mejores resultados y, por lo tanto, éste no debería ser el punto final para esta nueva formulación.

				F1			**F2**		
	|V|	|A|	|$\mathbb{H}$|	Resueltas	Tiempo	Gap0 (%)	Resueltas	Tiempo	Gap0 (%)
ce200-0,5	500	1000	500	5/5	245,7	0,65	5/5	1079,92	1,81
ce200-1	500	1000	1000	5/5	88,6	0,28	5/5	856,48	1,41
ce200-5	500	1000	5000	5/5	28,3	0,08	5/5	187,96	0,82
ce200-10	500	1000	10000	5/5,	20,3	0,11	5/5	137,48	1,17
ce150-0,5	500	1500	750	5/5	830,5	0,49	1/5	6025,43	2,74
ce150-1	500	1500	1500	5/5	1235,5	0,38	2/5	4036,92	2,32
ce150-5	500	1500	7500	5/5	49,2	0,12	5/5	331,83	0,99
ce150-10	500	1500	15000	5/5	50,1	0,13	4/5	1739,64	1,26

Tabla 7.7: *Comparación de las dos formulaciones para el GDRPP.*

Conclusiones

En esta tesis hemos estudiado tres problemas importantes de Rutas por Arcos: el Problema General de Rutas en un grafo dirigido (Directed General Routing Problem, DGRP), su caso particular el Problema de la Grúa (Stacker Crane Problem, SCP) y el Problema del Cartero Rural Generalizado en un grafo dirigido (Generalized Directed Rural Postman Problem, GDRPP).

En primer lugar hemos presentado un algoritmo metaheurístico para el SCP que ha sido probado en instancias de tamaño mediano proporcionando buenos resultados.

A continuación hemos estudiado el DGRP, que es un problema más general y que contiene como casos particulares a muchos otros problemas de rutas por arcos, incluido el SCP. Así, todos los resultados obtenidos para el DGRP se pueden aplicar al SCP. A diferencia de la mayoría de los trabajos sobre Problemas de Rutas por Arcos, el estudio del DGRP se ha realizado sobre el grafo original en lugar de hacerlo sobre un grafo transformado que no contiene vértices que no sean incidentes con arcos requeridos. Hemos estudiado el poliedro asociado al conjunto de soluciones posibles del DGRP y hemos encontrado las siguientes desigualdades que definen faceta del mismo:

- desigualdades triviales,
- desigualdades de obligatoriedad,

- desigualdades de conectividad,
- desigualdades K-C,
- desigualdades Honeycomb,
- desigualdades Path-Bridge y 2-Path-Bridge Asimétricas.

También hemos probado un teorema de "lifting", que afirma que todas las desigualdades que inducen faceta del DGRP en un grafo más reducido, llamado grafo de configuración, pueden extenderse al grafo original definiendo también faceta del DGRP en éste último.

Para la resolución del DGRP hemos diseñado e implementado un algoritmo de "branch and cut". Dicho algoritmo ha sido probado en un gran número de instancias del DGRP y del SCP. Algunas de estas instancias han sido generadas *ex profeso* para el correspondiente problema, como es el caso de las instancias de tipo Grid para el SCP, mientras que otras han sido tomadas directamente de la Literatura, como las de Blais y Laporte (2003) para el DGRP. Consideramos que los resultados obtenidos son muy buenos, pues hemos resuelto óptimamente instancias de hasta 50000 arcos y 3000 arcos requeridos.

A diferencia del DGRP, donde cada arco requerido ha de ser obligatoriamente recorrido, en el GDRPP cada arco requerido pertenece a uno o a varios subconjuntos de arcos (llamados clientes), y hay que recorrer al menos un arco de cada subconjunto. Para este problema hemos hecho también un estudio poliédrico, encontrando las siguientes desigualdades que definen faceta del poliedro asociado:

- desigualdades triviales,
- desigualdades de obligatoriedad,
- desigualdades de conectividad,

- desigualdades de servicio,
- desigualdades de imparidad y
- desigualdades K-C.

Cabe destacar que, para las desigualdades de conectividad, imparidad y K-C hemos presentado diferentes versiones. Hemos diseñado e implementado otro algoritmo de branch and cut para resolver este problema, que emplea un algoritmo heurístico para conseguir mejores cotas superiores, basado en la resolución del DGRP mencionado anteriormente. Mediante este algoritmo hemos resuelto óptimamente algunas instancias con más de 1300 arcos y 10000 clientes en pocos segundos y hemos alcanzado "gaps" muy buenos en las instancias más difíciles. Aunque casi todas las instancias con las que hemos trabajado han sido las usadas por Hà et al. (2012), también hemos generado un nuevo un conjunto de instancias en las que los conjuntos de arcos asociados a cada cliente son disjuntos.

El algoritmo está basado en una formulación del problema con dos tipos de variables, x e y, representando el número de veces que se recorre cada arco y si éste es servido o no, respectivamente. Como alternativa, hemos propuesto una nueva formulación para el GDRPP usando únicamente las variables x y hemos adaptado el algoritmo de branch and cut para comparar su eficiencia con respecto a la formulación inicialmente propuesta. Sin embargo, los resultados computacionales indican que esta nueva formulación es peor que la original.

Como trabajo futuro nos planteamos mejorar el algoritmo de branch & cut para el GDRPP para el caso en que los clientes estén definidos por conjuntos disjuntos de arcos. También pretendemos profundizar en el estudio de la formulación para el GDRPP sólo con las variables x. Finalmente, queremos estudiar los problemas abordados en esta tesis en el caso en que tengamos una flota de K vehículos. Pretendemos llevar a cabo un estudio de los poliedros asociados a sus conjuntos de

soluciones, adaptando las desigualdades descritas en esta tesis y buscando nuevas desigualdades. Los conocimientos adquiridos en este estudio servirán de base para la implementación de un algoritmo de branch and cut.

Bibliografía

[1] D. L. Applegate, R. E. Bixby, V. Chvàtal, and W. J. Cook. *The Traveling Salesman Problem: A Computational Study (Princeton Series in Applied Mathematics)*. Princeton University Press, Princeton, NJ, USA, 2007.

[2] A. Assad, M. Ball, L. Bodin, and B. Golden. Routing and scheduling of vehicles and crews. *Computers & Operations Research*, 10:63–211, 1983.

[3] A.A. Assad and B.L. Golden. *Arc Routing Methods and Applications*. Handbooks of Operations Research and Management Science 8. North Holland, 2000.

[4] P. Augerat, J.M. Belenguer, E. Benavent, Á. Corberán, D. Naddef, and G. Rinaldi. Computational results with a branch-and-cut code for the capacitated vehicle routing problem. Technical Report RR949-M, ARTEMIS-IMAG, France, 1995.

[5] A. Bachem and M. Grötschel. New aspects of polyhedral theory. *Modern Applied Mathematics: Optimization and Operations Research (B. Korte, eds). North Holland, Amsterdam*, pages 51–106, 1982.

[6] E. Balas and M. Ng. On the set covering polytope I: All the facets with coefficients in {0,1,2}. *Mathematical Programming*, 43:57–69, 1989.

[7] M.O. Ball and M.J. Magazine. Sequencing of insertions in printed circuit board assembly. *Operations Research*, 36:192–201, 1988.

[8] M. Bazaraa and J. Jarvis. *Linear Programming and Network Flows*. Wiley, New York, 1977.

[9] J.M. Belenguer, E. Benavent, N. Labadi, C. Prins, and M. Reghioui. Split delivery capacitated arc routing problem: Lower bound and metaheuristic. *Transportation Science*, 44:206–220, 2010.

[10] E. Benavent, V. Campos, Á. Corberán, and E. Mota. Problemas de rutas por arcos. *Qüestió*, 7:479–490, 1983.

[11] E. Benavent, Á. Corberán, E. Piñana, I. Plana, and J.M. Sanchis. A scatter search algorithm for the windy rural postman problem. *Computers & Operations Research*, 32:3111–3128, 2005.

[12] E. Benavent, A. Carrotta, Á. Corebrán, J.M Sanchis, and D. Vigo. Lower bounds and heuristics for the windy rural potman problem. *European Journal of Operational Research*, 176:855–869, 2007.

[13] E. Benavent, Á. Corberán, and J.M. Sanchis. A metaheuristic for the min−max windy rural postman problem with k vehicles. *Computational Management Science*, 7:269–287, 2010.

[14] G. Berbeglia, J.F. Cordeau, I. Gribkovskaia, and G. Laporte. Static pickup and delivery problems: a classification scheme and survey. *TOP*, 15:1–31, 2007.

[15] C. Berge. *Graphs and Hypergraphs*. North Holland, Amsterdam, 1973.

[16] M. Blais and G. Laporte. Exact solution of the generalized routing problem through graph transformations. *Journal of the Operational Research Society*, 54:67–77, 2003.

[17] L. Bodin and B. Golden. Classification in vehicle routing and scheduling. *Networks*, 11:97–108, 1981.

[18] L. Bondy and U. Murty. *Graph Theory with Applications*. American Elsevier, New York, 1976.

[19] V. Campos and J.V. Savall. A computational study of several heuristics for the DRPP. *Computational Optimization and Applications*, 4:67–77, 1995.

[20] G. Carpaneto, M. Dell'Amico, and P. Toth. Exact solution of large scale, asymmetric traveling salesman problems. *ACM Transactions on Mathematical Software*, 21:395–409, 1995.

[21] S. Chopra and G. Rinaldi. The graphical asymmetric traveling salesman polyhedron: Symmetric inequalities. *SIAM J. Discrete Math.*, 9:602–624, 1996.

[22] N. Christofides. The optimum traversal of a graph. *Omega*, 1:719–732, 1973.

[23] N. Christofides. *Graph Theory. An Algorithmic Approach.* Academic Press, New York, 1975.

[24] N. Christofides. Worst-case analysis of a new heuristic for the traveling salesman problem. Technical report, Graduate School of Industrial Administration, Carnegie-Mellon University, 1976.

[25] N. Christofides, V. Campos, Á. Corberán, and E. Mota. An algorithm for the rural postman problem. Technical report, IC. OR. 81. 5., Imperial College, London, 1981.

[26] N. Christofides, E. Benavent, V. Campos, Á. Corberán, and E. Mota. *An Optimal Method for the Mixed Postman Problem.* Lectures Notes in Control and Information Sciences 59. Springer, Berlin, 1984.

[27] N. Christofides, V. Campos, Á. Corberán, and E. Mota. An algorithm for the rural postman problem on a directed graph. *Mathematical Programming Study*, 26:155–166, 1986.

[28] V. Chvátal. *Linear Programming.* Freeman, New York, 1983.

[29] J. Cirasella, D.S. Johnson, L.A. McGeoch, and W. Zhang. Traveling salesman problem: Algorithms, instance generators, and tests. *ALENEX*, 2001:32–59, 2007.

[30] Á. Corberán and J.M. Sanchis. A polyhedral approach to the rural postman problem. *European Journal of Operational Research*, 79:95–114, 1994.

[31] Á. Corberán and J.M. Sanchis. The general routing problem polyhedron: Facets from the RPP and GTSP polyhedra. *European Journal of Operational Research*, 108:538–550, 1998.

[32] Á. Corberán, R. Martí, and A. Romero. Heuristics for the mixed rural postman problem. *Computers & Operations Research*, 27:183–203, 2000.

[33] Á. Corberán, A.N. Letchford, and J.M. Sanchis. A cutting plane algorithm for the general routing problem. *Mathematical Programming*, 90:291–316, 2001.

[34] Á. Corberán, R. Martí, and J.M. Sanchis. A grasp heuristic for the mixed Chinese postman problem. *European Journal of Operational Research*, 142: 70–80, 2002.

[35] Á. Corberán, A. Romero, and J.M. Sanchis. The mixed general routing problem polyhedron. *Mathematical Programming*, 96:103–137, 2003.

[36] Á. Corberán, G. Mejía, and J.M. Sanchis. New results on the mixed general routing problem. *Operations Research*, 53:363–376, 2005.

[37] Á. Corberán, E. Mota, and J.M. Sanchis. A comparison of two different formulations for arc routing problems on mixed graphs. *Computers & Operations Research*, 33:3384–3402, 2006.

[38] Á. Corberán and C. Prins. Recent results on arc routing problems: An annotated bibliography. *Networks*, 56:50–69, 2010.

[39] Á. Corberán, I. Plana, and J.M Sanchis. Zigzag inequalities: A new class of facet-inducing inequalities for arc routing problems. *Mathematical Programming*, 108:79–96, 2006.

[40] Á. Corberán, I. Plana, and J.M Sanchis. A branch & cut algorithm for the windy general routing problem and special cases. *Networks*, 49:245–257, 2007.

[41] Á. Corberán, I. Plana, and J.M Sanchis. The windy routing polyhedron: A global view of many known arc routing polyhedra. *SIAM Journal on Discrete Mathematics*, 22:606–628, 2008.

[42] Á. Corberán, M. Oswald, I. Plana, G. Reinelt, and J.M. Sanchis. New results on the windy postman problem. *Mathematical Programming*, 132:309–332, 2012.

[43] G. Cornuèjols, J. Fonlupt, and D. Naddef. The traveling salesman problem on a graph and some related inequalities. *Mathematical Programming*, 33:1–27, 1985.

[44] G. Dantzig. *Linear Programming*. Freeman, New York, 1963.

[45] M. Drexl. *On some generalized routing problems*. PhD thesis, Rheinisch-Westfalishe Technische Hochschule Aachen, 2007.

[46] M. Drexl. On the generalized directed rural postman problem. *Journal of the Operational Research Society*, doi:10.157/jors.2013.60, 2013.

[47] M. Dror. *Arc Routing: Theory, Solutions and Applications*. Kluwer Academic Publishers, 2000.

[48] J. Edmonds. The Chinese postman problem. *Operations Research*, 13 Supplement 1:B73, 1965.

[49] J. Edmonds. Paths, trees and flowers. *Canadian Journal of Mathematics*, 17: 449–467, 1965.

[50] J. Edmonds and E.L. Johnson. Matching, Euler tours and the Chinese postman. *Mathematical Programming*, 5:88–124, 1973.

[51] H.A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems. part I: The Chinese postman problem. *Operations Research*, 43:231–242, 1995a.

[52] H.A. Eiselt, M. Gendreau, and G. Laporte. Arc routing problems. part II: The rural postman problem. *Operations Research*, 43:399–414, 1995b.

[53] E. Fernández, O. Meza, R. Garfinkel, and M. Ortega. On the undirected rural postman problem: Tight bounds based on a new formulation. *Operations Research*, 51:281–291, 2003.

[54] B. Fleischmann. A cutting plane procedure for the traveling salesman problem on road networks. *European Journal of Operational Research*, 21:147–172, 1985.

[55] B. Fleischmann. A new class of cutting planes for the symmetric traveling salesman problem. *Mathematical Programming*, 40:225–246, 1988.

[56] H. Fleischner. *Eulerian Graphs and Related Topics. Part I.* North-Holland, Amsterdam, 1990.

[57] H. Fleischner. *Eulerian Graphs and Related Topics. Part II.* North-Holland, Amsterdam, 1991.

[58] G.N. Frederickson. Approximation algorithms for some postman problems. *Journal of the Association for Computing Machinery*, 26:538–554, 1979.

[59] G.N. Frederickson, M.S. Hecht, and C.E. Kim. Aproximation algorithms for some routing problems. *Journal on Computing*, 7:178–193, 1978.

[60] R. Garfinkel and G. Nemhausser. *Integer Programming.* Wiley, New York, 1972.

[61] G. Ghiani and G. Laporte. A branch-and-cut algorithm for the undirected rural postman problem. *Mathematical Programming*, 87:467–481, 2000.

[62] B. Golden, S. Raghavan, and E. Wasil. *The Vehicle Routing Problem: Lastest Avances and New Challenges.* Springer, 2008.

[63] M. Grötschel. Developments in combinatorial optimization. In *Perspectives in Mathematics. Anniversary of Oberwolfach*, pages 249–294, Basel, 1984. Birkhäuser Verlag.

[64] M. Grötschel and O. Holland. Solving matching problems with linear programming. *Mathematical Programming*, 3:243–259, 1985.

[65] M. Grötschel and M.W. Padberg. *Polyhedral Theory.* J. Wiley & Sons, 1985.

[66] M. Grötschel and Z. Win. A cutting plane algortihm for the windy postman problem. *Mathematical Programming*, 55:339–358, 1992.

[67] M. Grötschel, M. Jünger, and G. Reinelt. A cutting plane for the linear ordering problem. *Operations Research*, 32:1195–1220, 1984.

[68] H. Gün. *Polyhedral structures and efficient algorithms for certain classes of directed rural postman problems.* PhD thesis, University of Maryland, USA, 1993.

[69] M-H. Hà, N. Bostel, A. Langevin, and L-M. Rousseau. An exact algorithm for close enough traveling salesman problem. In *Proceedings of the 1st International Conference on Operations Research and Enterprise Systems*, pages 233–238, 2012.

[70] M-H. Hà, N. Bostel, A. Langevin, and L-M. Rousseau. Solving the close enough arc routing problem. *Networks*, doi:10.1002/net.21525, 2013.

[71] F. Harary. *Graph Theory.* Addison Wesley, Reading, MA, 1969.

[72] R. Hassin and S. Khuller. Z-approximations. *Journal of Algorithms*, 41:429–442, 2001.

[73] A. Hertz, G. Laporte, and P. Nanchen. Improvement procedures for the undirected rural postman problem. *INFORMS Journal on Computing*, 11:53–62, 1999.

[74] K. Hoffman and M. Padberg. Lp-based combinatorial problem solving. *Annals of Operations Research*, 4:145–194, 1985.

[75] D. S. Johnson, G. Gutin, L. A. McGeoch, A. Yeo, W. Zhang, and A. Zverovitch. *Experimental analysis of heuristics for the ATSP*. Kluwer Academic Publishers, Dordrecht, 2002.

[76] L. Khachian. A polynomial algorithm for linear programming. *Soviet Math. Doklady*, 20:191–194, 1979.

[77] G. Laporte. Modeling and solving several classes of arc routing problems as traveling salesman problems. *Computers & Operations Research*, 24:1057–1061, 1997.

[78] J.K. Lenstra and A. Rinnooy-Kan. On the general routing problem. *Networks*, 6:273–280, 1976.

[79] J.K. Lenstra and A. Rinnooy-Kan. Complexity of vehicle routing and scheduling problems. *Networks*, 11:221–227, 1981.

[80] A.N. Letchford. New inequalities for the general routing problem. *European Journal of Operational Research*, 96:317–322, 1997.

[81] A.N. Letchford. The general routing problem: a unifying framework. *European Journal of Operational Research*, 112:122–133, 1999.

[82] T.M. Liebling. *Graphteorie in Plannungs- und Tourenproblemen*, volume 21. Springer-Verlag, 1970.

[83] H.R. Lourenço, O. Martin, and T. Stützle. *Handbook of metaheuristics*, chapter Iterated Local Search, pages 321–353. Kluwer, 2002.

[84] E. Minieka. The Chinese postman problem for mixed networks. *Management Science*, 25:643–648, 1979.

[85] N. Mladenovic and P. Hansen. Variable neighborhood search. *Computers & Operations Research*, 24:1097–1100, 1997.

[86] D. Naddef and G. Rinaldi. The symmetric traveling salesman polytope and its graphical relaxation: Composition of valid inequalities. *Mathematical Programming*, 51:359–400, 1991.

[87] H. Nagamochi, T. Ono, and T. Ibaraki. Implementing an efficient minimum capacity cut algorithm. *Mathematical Programming*, 67:297–324, 1994.

[88] G. Nemhausser and L. Wolsey. *Integer and Combinatorial Optimization*. Wiley, New York, 1988.

[89] Y. Nobert and J.C. Picard. An optimal algorithm for the mixed Chinese postman problem. *Networks*, 27:95–108, 1996.

[90] C.S. Orloff. A fundamental problem in vehicle routing. *Networks*, 4:35–64, 1974.

[91] M. Padberg and G. Rinaldi. A branch and cut algorithm for the resolution of larg-scale symmetric traveling salesman. *Mathematical Programming*, 47: 19–36, 1990.

[92] M.W. Padberg and M. Rao. Odd minimum cut-sets and b-matchings. *Mathematics of Operations Research*, 7:67–80, 1982.

[93] C. Papadimitriou. On the complexity of edge traversing. *Journal of the ACM*, 23:544–554, 1976.

[94] C. Papadimitriou. The Euclidean traveling salesman problem is NP-complete. *Mathematical Programming*, 14:312–324, 1977.

[95] C. Papadimitriou. Polytopes and complexity. *Progress in Combinatorial Optimization*, pages 295–305, 1984.

[96] I. Plana. *El Problema General de Rutas Con Viento*. PhD thesis, Universidad de Valencia, 2005.

[97] W.R. Pulleyblank. *Polyhedral Combinatorics*. Springer Berlin Heidelberg, 1983.

[98] B. Raghavachari and J. Veerasamy. A 3/2-approximation algorithm for the mixed postman problem. *SIAM Journal on Discrete Mathematics*, 12:425–433, 1999.

[99] G. Reinelt and D.O. Theis. Transformation of facets of the general routing problem polytope. *SIAM Journal on Optimization*, 16:220, 2005.

[100] G. Reinelt and D.O. Theis. On the general routing polytope. *Discrete Applied Mathematics*, 156:368–384, 2008.

[101] A. Romero. *El problema del cartero rural en un grafo mixto*. PhD thesis, Universidad de Valencia, 1997.

[102] J.M. Sanchis. *El poliedro del problema del cartero rural*. PhD thesis, Universidad de Valencia, 1990.

[103] J.V. Savall. *Resultados poliédricos y algoritmos aproximados para el DRPP*. PhD thesis, Universidad de Valencia, 1990.

[104] A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, Chichester, 1986.

[105] M. Sánchez-García, M.I. Sobrón, and B. Vitoriano. On the set covering polytope: Facets with coefficients in $\{0, 1, 2, 3\}$. *Annals of Operations Research*, 81:343–356, 1998.

[106] J. Srour. *Dissecting drayage: an examination of structure, information, and control in drayage operations*. PhD thesis, Erasmus Universiteit, Rotterdam, 2010.

[107] J. Srour and S. van de Velde. Are stacker crane problems easy? A statistical study. *Computers & Operations Research*, 40:674–690, 2011.

[108] D. O. Theis. *Polyhedra and algorithms for the General Routing Problem*. PhD thesis, Ruprecht Karls University of Heidelberg, 2005.

[109] Z. Win. On the windy postman problem on Eulerian graphs. *Mathematical Programming*, 44:97–112, 1989.

[110] L. Zhang. Simple heuristics for some variants of the traveling salesman problem. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 1175–1178, 1992.

[111] L. Zhang and W. Zheng. Genetic coding for solving both the stacker crane problem and its k-variant. In *IEEE International Conference on Systems, Man and Cybernetics*, pages 1061–1066, 1995.