CIRCUITOS EULERIANOS OPTIMOS EN GRAFOS NO DIRIGIDOS : EL PROBLEMA DEL CARTERO RURAL.

Memoria presentada por D.Angel Corberán Salvador para optar al grado de Doctor en Ciencias Matemáticas.
Realizada bajo la dirección de D.Marco Antonio López Cerdá , Catedrático de Investigación Operativa de la Facultad de Matemáticas de la Universidad de Valencia.

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

Nuestro agradecimiento a todas las personas que nos han ayudado y estimulado, y en particular al personal del Centro de Cálculo de la Universidad Politécnica de Valencia, sin cuya amable colaboración no habría sido posible una parte importante de este trabajo.

INDICE

en la práctica , siendo sustituido por dos algoritmos heurísticos destinados a la obtención de buenos multiplicadores.Demostramos una serie de resultados que caracterizan a un subconjunto del conjunto de multiplicadores óptimos y se relaciona dicha caracterización con el procedimiento heurístico para su obtención.

En la Sección 4 proporcionamos un algoritmo heurístico para la obtención de soluciones posibles del RPP. El algoritmo se basa en el cálculo de un SST y en la resolución de un problema de 1-matching de coste mínimo. A partir de la solución posible obtenida con el algoritmo es posible, en algunos casos, encontrar otra solución cuyo coste esté mas próximo del valor óptimo del problema. El valor de esta solución será utilizado como cota superior en el algoritmo de enumeración de la siguiente Sección.

En la Sección 5 describimos las partes fundamentales de las que consta el algoritmo de Branch and Bound diseñado para el RPP: estrategia de separación(partición) en subproblemas, estrategia de ramificación (selección de subproblemas) y saturación de nudos.

Finalmente, en la Sección 6 ilustramos con un ejemplo el funcionamiento del algoritmo descrito en la Sección 5. Dicho algoritmo ha sido probado con una colección de 24 problemas de diversos tamaños. Grafos de hasta 84 vértices, 180 aristas y 74 aristas "requeridas" han podido ser resueltos óptimamente en tiempos inferiores a 300 segundos de CPU. La Sección concluye con los resultados obtenidos para estos problemas y con una breve valoración de los mismos.

I N T R O D U C C I O N

CONCEPTOS PREVIOS

1. TEORIA DE GRAFOS

En el siguiente resumen, de definiciones y resultados básicos de la Teoría de Grafos, seguimos fundamentalmente los textos de Christofides (1975) y Bondy and Murty (1976).

Otros textos de referencia obligada son Berge (1962) y Harary (1969).

1.1 Definiciones y conceptos básicos

Un *grafo* G es una colección de puntos ó vértices $i_1, i_2, \ldots\ldots, i_n$ (que representamos por N), y una colección de líneas $\ell_1, \ell_2, \ldots\ldots, \ell_m$ (representadas por el conjunto A) que unen todos, o algunos de esos puntos. Representamos el grafo G por el par (N,A).

Cuando las líneas de A tienen una dirección, lo que usualmente se representa por una flecha, reciben el nombre de *arcos* y el grafo resultante el de *grafo dirigido* .

Si las líneas no tienen dirección, reciben el nombre de *aristas* y el grafo es *no dirigido*. Un grafo que contenga aristas y arcos recibe el nombre de *grafo mixto*.

Una arista puede representarse por medio de los dos vértices que une; a dichos vértices se les denomina *vértices terminales*.

Una arista cuyos vértices terminales coinciden recibe el nombre de *bucle*.

Un grafo múltiple o *multigrafo* es aquel para el que pueden existir varias aristas entre dos vértices dados.

Un grafo se llama *simple* si no tiene bucles y no

existe más de una arista entre cualquier par de vértices.

Una *cadena* o *camino* en un grafo no dirigido es una sucesión de aristas $(\ell_1,\ell_2,\ldots,\ell_q)$ en la que cada arista ℓ_i, excepto quizás la primera y la última, está conectada a las aristas ℓ_{i-1} , y ℓ_{i+1} por sus dos vértices terminales.

Puede también representarse por la sucesión de vértices que son utilizados.

Una cadena *simple* es aquella que no utiliza la misma arista más de una vez; y una cadena *elemental* es la que no utiliza el mismo vértice más de una vez.

Existe, a menudo, un número c_ℓ asociado con una arista ℓ. Estos números reciben el nombre de costes, longitudes ó pesos de las aristas. El coste de una cadena o camino es la suma de los pesos de las aristas que lo forman.

Un *circuito* en un grafo no dirigido ó *ciclo* es un camino en el que los vértices inicial y final coinciden.

Un circuito elemental que pasa a través de todos los vértices de un grafo recibe el nombre de *circuito hamiltoniano*.

Un circuito simple que atraviesa todas las aristas (ó arcos) de un grafo recibe el nombre de *circuito (tour) euleriano*.

En un grafo no dirigido, el *grado de un vértice*, $d(i)$ ó d_i, se define como el número de aristas que son incidentes con él.

Teorema 1 : La suma de los grados de los vértices de un grafo G es dos veces el número de aristas.

complementario de N_0 en N), entonces el conjunto de aristas de G cuyos vértices terminales pertenecen uno a N_0 y el otro a $\bar{N}_0$ recibe el nombre de *conjunto de corte* de G.

1.2 Arboles

Un grafo es *acíclico* si no tiene ciclos.

Un *árbol* es un grafo acíclico y conexo.

Teorema 2 : Las siguientes afirmaciones son equivalentes para un grafo $G=(N,A)$, donde $|N|=n$.

1) G es un árbol.
2) Cada par de vértices de G están unidos por un único camino elemental.
3) G es un grafo conexo con n vértices y $n-1$ aristas.
4) G es un grafo acíclico con n vértices y $n-1$ aristas.
5) G es acíclico y si cualesquiera par de vértices no adyacentes de G son unidos por una arista ℓ, entonces $G \cup \ell$ tiene exactamente un ciclo.

Un *árbol generador* de un grafo G se define como un grafo parcial de G que forma un árbol. Puede también definirse como un grafo parcial conexo minimal (respecto del número de aristas) de G.

Arbol generador de mínimo peso (SST) de un grafo G es el de menor peso ó coste de los árboles generadores de G, cuando hay costes asociados con sus aristas.

Existen algoritmos eficientes para encontrar el SST de un grafo, entre ellos los más conocidos son los de Krus

kal y Prim. (Christofides (1975), páginas 135-139).

El algoritmo de Kruskal consiste básicamente en:

Step 1: Empezar con un grafo completamente desconectado T de n vértices.

Step 2: Ordenar las aristas de G en orden creciente según el coste.

Step 3: Empezando desde el principio de esta lista (con las aristas de menor coste) añadir aristas a T siempre - que esta adición no cierre un circuito en T.

Step 4: Repetir la step 3 hasta que hayan sido añadidas $(n-1)$ aristas.

Entonces T es el árbol generador de mínimo peso del grafo G.

1.3 Caminos más cortos

Para un grafo dado $G=(N,A)$, con costes definidos - sobre sus aristas, el *problema del camino más corto* es el - de encontrar el camino más corto desde un vértice de partida especificado $i \in N$ a un vértice final especificado $j \in N$, si tal camino existe.

Este problema puede generalizarse al problema de encontrar los caminos más cortos entre todos los pares de vértices.

Existen algoritmos muy sencillos y eficientes para calcular los caminos más cortos entre dos vértices dados, entre un vértice y todos los demás y entre todos los pares de vértices. La elección entre ellos depende del planteamiento - deseado y de los costes de las aristas (si son positivos,nega

tivos y nulos ó únicamente no negativos).

Así el algoritmo de Dijkstra es el más eficiente para calcular el camino más corto entre dos vértices específicos si los costes de las aristas son no negativos.

Cuando los costes pueden ser también negativos - tenemos los algoritmos de Ford, Moore y Bellman.

Para calcular los caminos más cortos entre todos los pares de vértices podemos utilizar repetidamente el algoritmo de Dijkstra, ó bién utilizar los algoritmos de Floyd, - Murchland ó Dreyfus, que pueden también aplicarse en el caso - de costes negativos.

Para una descripción detallada de los algoritmos, ver Christofides (1975), Bondy y Murty (1976), Goudran y Minoux (1979), Dreyfus y Law (1978) y Lawler (1976)

1.4 Matchings (Acoplamientos)

Un *matching* de un grafo no dirigido $G=(N,A)$ es - un subconjunto M del conjunto de aristas de A escogido de forma que no hayan dos aristas cualesquiera de M que sean adyacentes (es decir que tengan un vértice terminal común).

El problema del *Matching mínimo*, es el problema de encontrar un matching perfecto (es decir, aquél en que todo vértice está acoplado por medio de una arista del matching a cualquier otro vértice) y que tenga coste mínimo.

Este problema puede plantearse como un problema lineal entero:

$$\text{Min} \sum_{\ell \in A} c_\ell \xi_\ell$$

sujeto a:

$$\sum_{\ell \in A} a_{i\ell}\xi_{\ell} = 1 \qquad \forall i \in N$$

$$\xi_{\ell} = 0,1$$

donde $[a_{i\ell}]$ es la matriz de incidencia:

$$a_{i\ell} = \begin{cases} 1 \text{ si el vértice } i \text{ es incidente con la} \\ \quad \text{arista } \ell. \\ 0 \text{ en otro caso.} \end{cases}$$

y donde $\xi_{\ell}=1$ (ó 0) dependiendo de si ℓ está (ó no) en el matching.

Existen buenos algoritmos para este problema, en el sentido de que el número de operaciones que realizan es una función polinómica del número n de vértices del grafo.

Aunque el crecimiento en el tiempo de computación es teóricamente $O[n^4]$, en la práctica es apreciablemente menor $O[n^3]$. Estos algoritmos, muy estudiados y elaborados, pueden encontrarse en Christofides (1975) y fundamentalmente en Lawler (1976), quién dedica dos capítulos de su libro a los problemas de matching y sus aplicaciones.

2. LA RELAJACION LAGRANGIANA

El éxito de la Relajación Lagrangiana, se deriva de la observación del hecho de que un gran número de problemas difíciles pueden ser considerados como problemas sencillos - complicados por un, relativamente pequeño, conjunto de restricciones. Si se dualizan estas restricciones, se obtiene un problema Lagrangiano que es fácil de resolver. El valor óptimo - de éste problema es una cota inferior del valor óptimo del - problema original. El problema Lagrangiano puede utilizarse - en lugar de la relajación lineal para obtener cotas en un algoritmo de Branch and Bound.

Aunque los métodos Lagrangianos ya se utilizaron en Optimización Discreta antes de 1970, no es sino en 1970 - cuando "nace" la aproximación Lagrangiana tal como hoy se utiliza, cuando Held y Karp utilizaron un problema lagrangiano - basado en los árboles generadores de mínimo peso para obtener un buen algoritmo para el TSP. Y es en 1974 cuando Geoffrion da el nombre a esta aproximación:Relajación Lagrangiana.

A continuación resumimos algunos de los principales aspectos de la teoría y de la práctica de la Relajación - Lagrangiana. Nos basamos fundamentalmente en Fisher (1981) y Geoffrion (1974). En Shaphiro (1979) se encuentra un resumen muy completo del uso de la Relajación Lagrangiana en problemas de Optimización Discreta.

En general, un Problema de Programación Lineal - Entera tiene la forma:

(P) $\quad \underset{x \geq 0}{\text{Min}} \quad cx$

sujeto a: $\quad Ax \geq b$

$Bx \geq d$

x_j entero $\quad j \in I$

Donde c, b y d son vectores; y A y B matrices de dimensiones apropiadas.

Definimos la *Relajación Lagrangiana* de (P) respecto a las restricciones $Ax \geq b$ y a un *Vector de Multiplicadores* λ, no negativos, como el Problema:

(PR_λ) $\quad \underset{x \geq 0}{\text{Min}} \quad cx + \lambda(b - Ax)$

sujeto a: $\quad Bx \geq d$

x_j entero $\quad j \in I$

Si las restricciones relajadas fueran igualdades (o sea, de la forma $Ax = b$), el vector λ podría tomar también valores negativos, cumpliéndose exactamente igual todos los Teoremas que veremos a continuación.

En este apartado seguiremos la siguiente notación:

(X) es un Problema de Optimización.

$v(X)$ es su valor óptimo.

$F(X)$ es el conjunto de soluciones posibles de (X).

$(\bar{X})$ es la Relajación Lineal de (X).

Además $\bar{\lambda}$ denotará el vector de variables duales correspondientes a $Ax \geq b$ en la solución óptima del Problema relajado lineal-

mente $(\bar{P})$.

Supondremos además que las restricciones $Bx \geqslant d$ incluyen cotas superiores para todas las variables, por lo que (P) o cualquiera de sus relajaciones consideradas aquí, será siempre acotado.

Teorema 3

Para todo $\lambda \geqslant 0$, se cumple que $v(PR_\lambda) \leqslant v(P)$.

Por lo tanto, para cualquier vector de multiplicadores $\lambda \geqslant 0$, $v(PR_\lambda)$ es una cota inferior sobre el óptimo del problema original.

La utilidad de la Relajación Lagrangiana (PR_λ) reside en que este problema sea más facilmente resoluble que (P), y si es posible, por un algoritmo polinomial.

La mejor cota inferior que podríamos obtener de (PR_λ) correspondería a un vector de multiplicadores $\lambda \geqslant 0$ que fuera óptimo para el problema:

$$(D) \qquad \max_{\lambda \geqslant 0} \; v(PR_\lambda)$$

Teorema 4

Supondremos que el problema (P) es posible.

a) $v(\bar{P}) \leqslant v(D) \leqslant v(P)$

b) $v(\bar{P}) \leqslant v(PR_{\bar{\lambda}})$

c) Si para un $\lambda \geqslant 0$, tenemos un vector x que satisface:

1) x es la solución óptima de (PR_λ).

2) $Ax \geqslant b$

3) $\lambda(b-Ax)=0$

entonces x es una solución óptima de (P).

El apartado a) del teorema nos dice que la mejor cota inferior obtenida por la Relajación Lagrangiana (PR_λ) es por lo menos tan buena como la obtenida por la Relajación Lineal $(\bar{P})$.

Además, por b), si resolvemos la Relajación Li-

neal ($\bar{P}$) y elegimos como $\lambda \geq 0$ las variables duales óptimas $\bar{\lambda} \geq 0$ correspondientes a las restricciones $Ax \geq b$, la solución de - ($PR_{\bar{\lambda}}$) puede mejorar la cota obtenida por la Relajación Lineal.

Por último en c) se dan las condiciones suficientes para que la solución óptima de (PR_λ) lo sea también del - problema (P). Si las restricciones relajadas son igualdades - $Ax=b$, el apartado 3) se cumplirá si se cumple el 2) (que en - este caso exigiría que sea $Ax=b$).

Cuando se cumplen las condiciones 1) y 2) pero - no 3), diremos que x es una solución ε-óptima, con $\varepsilon=\lambda(Ax-b) \geq 0$ En este caso $v(PR_\lambda)+\varepsilon$ es una cota superior para $v(P)$.

Diremos que el problema (PR_λ) tiene la *Propiedad de Integralidad* si y sólo si se cumple:

$$v(PR_\lambda) = v(\overline{PR}_\lambda) \qquad \forall \lambda \geq 0$$

Teorema 5

Si ($\bar{P}$) es posible y (PR_λ) tiene la propiedad de integralidad, entonces:

$$v(D) = v(\bar{P})$$

Por lo tanto si se da la Propiedad de Integralidad en (PR_λ), como máximo podremos obtener, utilizando la Relajación Lagrangiana (PR_λ), una cota inferior igual a la proporcionada por la Relajación Lineal. La ventaja de utilizar la Relajación Lagrangiana consiste en que la cota inferior pueda ser obtenida en un tiempo de computacional menor, o incluso que la Relajación Lineal sea intratable debido a que posea un gran número de restricciones.

Método del Subgradiente.

Resolver el problema $(\mathcal{D})$ completamente suele ser en la práctica demasiado costoso computacionalmente, por lo - que normalmente se intenta obtener simplemente unos multiplicadores $\lambda \geq 0$, aunque no óptimos, que se acerquen lo más posible a serlo.

El método más utilizado y que ha dado los mejores resultados en la práctica para calcular unos buenos multiplicadores, $\lambda \geq 0$, es el Método del Subgradiente que será el que utilizaremos en este trabajo.

En este Método se parte de unos multiplicadores iniciales λ° que pueden ser todos nulos y se genera una sucesión de multiplicadores λ^k de acuerdo con la fórmula recursiva:

$$\lambda^{k+1} = \text{Max } \{0,\ \lambda^k + \theta_k (b - Ax^k)\}$$

Donde x^k es una solución óptima de (PR_{λ^k}) y $\theta_k > 0$ es la *longitud del paso*.

La justificación teórica de este Método viene dada por el siguiente teorema:

Teorema 6

Si $\{\theta_k\}$ cumple las condiciones:

$$\lim_{k \to \infty} \theta_k = 0 \qquad \theta_k > 0 \quad \forall k$$

$$\sum_{k=1}^{\infty} \theta_k = +\infty$$

entonces se cumple que $\lim_{k \to \infty} v(PR_{\lambda^k}) = v(\mathcal{D})$.

O sea que mediante una selección adecuada de los θ_k se puede asegurar la convergencia de $v(PR_\lambda k)$ a $v(D)$.

Sin embargo en la práctica se utiliza un número finito de iteraciones, más o menos grande, por lo que no se puede asegurar que se alcance $v(D)$.

Para la elección de la longitud de paso θ_k, el método más utilizado es hacer:

$$\theta_k = \frac{\alpha_k (z^* - v(PR_\lambda k))}{\| b - Ax^k \|^2}$$

Donde α_k es un escalar $0 < \alpha_k \leq 2$, z^* es una cota superior sobre $v(P)$ y $\|\ \ \|$ indica la norma euclídea.

Uno de los incovenientes del Método del Subgradiente es que no puede asegurar que $v(PR_\lambda k)$ sea creciente.

Para α_k se suele utilizar la estrategia de dividirlo por 2, si en un determinado número de iteraciones $v(PR_\lambda k)$ no se ha incrementado.

3. NOTAS SOBRE LA TEORIA DE LA NP-COMPLETITUD

Los fundamentos de la teoría de la *NP*-completitud se encuentran en Cook (1971). Desde entonces esta teoría ha conocido un rápido desarrollo y una gran notoriedad que se justifica sobre todo por la importancia de algunos de sus resultados. Un desarrollo riguroso de esta teoría se encuentra por ejemplo en Garey and Johnson (1979)

3.1 Definiciones básicas

En matemáticas, la palabra "problema" se utiliza con diversos significados. Aquí, un *problema* será una cuestión general que debe ser respondida, y que, usualmente, contiene varios parámetros (ó variables), cuyos valores se dejan sin especificar. Un problema está definido al proporcionar una descripción general de todos sus parámetros y especificar qué propiedades debe satisfacer la respuesta ó *solución* (óptima). Si se especifican ciertos valores para todos los parámetros del problema, obtenemos un *ejemplo* del problema.

Los *algoritmos* son procedimientos generales, paso a paso, para resolver problemas.

Un algoritmo se dice que *resuelve* un problema si el algoritmo produce una solución (óptima) para cualquier ejemplo I de P. El objetivo es encontrar el algoritmo "más eficiente" para resolver un problema.

Hay definidas varias medidas de la complejidad para intentar recoger aspectos sobre la eficiencia de los algoritmos. Nosotros nos centraremos en la llamada *complejidad* del *tiempo* (time complexity), puesto que los requerimientos de tiempo son los más importantes al juzgar la eficiencia de un algoritmo. El tiempo que requiere un algoritmo para resolver un ejemplo depende de su "tamaño", que refleja la cantidad de datos de entrada para describir este ejemplo. Esta medida se formaliza -

considerando un *esquema de codificación* (encoding scheme) que asocia ejemplos del problema a series de símbolos que los describen. En los computadores reales estas series consisten normalmente de ceros y unos. El *tamaño* ó *longitud de entrada* (input lenght) del ejemplo se define como la longitud de esa serie de simbolos. Tal serie puede utilizarse para entrar el ejemplo en una máquina real (ó hipotética). Antes de ver algunos resultados importantes sobre el "tamaño" de los ejemplos de un problema, debemos estar seguros de que los resultados no cambiarán sustancialmente si tomamos esquemas de codificación ligeramente diferentes y máquinas ligeramente diferentes. Las máquinas más utilizadas en el análisis teórico son la Turing y la RAM que en cierto sentido son equivalentes. Para nósotros, es suficiente considerar un computador real en lugar de una máquina Turing.

Dado un esquema de codificación y un modelo de máquina, la *función de complejidad del tiempo* (time complexity function) $f: N \to N$ de un algoritmo, expresa sus requerimientos de tiempo proporcionando, para cada tamaño posible o longitud de entrada $n \in N$, el tiempo máximo necesario para resolver un problema de ese tamaño por el algoritmo.

Los requerimientos de tiempo se miden contando el número de "pasos elementales" que el algoritmo realiza para resolver un ejemplo. Informalmente, operaciones tales como la adición, multiplicación, comparación, etc. son consideradas pasos elementales (tengamos en cuenta que si trabajamos con números grandes, las operaciones con dichos números necesita-

rían más de un paso elemental).

Ahora debemos definir la "eficiencia". Diremos - que una función $f(n)$ es $O(g(n))$, i.e. f es de orden g, si existe una constante c tal que $|f(n)| \leq c|g(n)|$ para todos los enteros $n \geq 0$. Un *algoritmo de tiempo polinomial* es un algoritmo - cuya función de complejidad del tiempo $f(n)$ es $O(p(n))$ para - algún polinomio $p: N \to N$. Cualquier algoritmo cuya función de complejidad del tiempo no pueda acotarse por un polinomio, recibe el nombre de *superpolinomial* ó *algoritmo de tiempo exponencial* . Puesto que las funciones exponenciales crecen mucho más rápidamente que las polinomiales, entonces diremos - que un algoritmo es *bueno* ó *eficiente* si funciona en tiempo - polinomial.

La naturaleza fundamental de esta distinción entre algoritmos, fué discutida por primera vez por Cobham - (1964) y Edmonds(1965 b). Edmonds,equiparó algoritmos de tiempo polinomial con "buenos" algoritmos y conjeturó que ciertos problemas de programación entera no podrían resolverse por esos "buenos" algoritmos. Existe un gran acuerdo respecto a - que un problema no está "bien resuelto" hasta que se conoce - un algoritmo en tiempo polinomial para él. Un problema, pues, diremos que es *intratable* si es tan difícil que ningún algoritmo en tiempo polinomial puede posiblemente resolverlo.

3.2 Las clases P y NP

Para distinguir entre problemas fáciles y difíciles, analizaremos *problemas de decisión* que son problemas -

que tienen únicamente dos posibles soluciones, "si" ó "no".

Los problemas de optimización pueden convertirse en problemas de decisión en la forma siguiente: dado un problema combinatorial de minimización y sea B un número (cota - superior), el problema de decisión correspondiente es "¿Existe una solución posible cuyo valor sea como máximo B?.

La clase de todos aquellos problemas de decisión para lo que existe un algoritmo de tiempo polinomial, se llama la *clase P*. Hay muchos problemas de decisión que no son conocidos como pertenecientes a *P*; por ejemplo, "¿Un grafo es - hamiltoniano?"

Podemos definir informalmente la clase *NP* en términos de lo que se llaman *algoritmos no deterministas* ,que - son aquéllos compuestos por dos etapas diferenciadas, la primera es una etapa de suposición (guessing stage) y otra de verificación (checking stage). Dado un ejemplo *I*, la primera etapa "supone" alguna estructura *S*. Consideramos *I* y *S* el input para la etapa de verificación, que procede a trabajar de una manera normal;y eventualmente se detiene con contestación "si", eventualmente se detiene con contestación "no" ó sigue la computación sin detenerse.

Diremos que un *algoritmo no determinista resuelve* un problema de decisión π en tiempo polinomial si existen dos polinomios $p_\pi(n)$ y $q_\pi(n)$ tal que se cumplen las dos propiedades siguientes para todos los ejemplos *I* de π.

1) Si la respuesta a *I* es "si" entonces existe una estructura *S* que satisface:

a) si n es la longitud de entrada de I entonces S puede ser codificado en tiempo $O(p_\pi(n))$,

b) si I y S se utilizan como el input de la etapa de verificación y n' es la longitud de entrada de S entonces la contestación será "si" en tiempo $O(q_\pi(n+n'))$.

2) Si la respuesta a I es "no", entonces no existe una estructura que lleve a la etapa de verificación a contestar "si" cuando se utilizan I y S como input.

Informalmente, *la clase* NP se define como la clase de todos los problemas de decisión que pueden resolverse por algoritmos no deterministas de tiempo polinomial.

Es obvio que $P \subseteq NP$, y parece evidente que $P \neq NP$, pero éste es uno de los problemas abiertos más importantes en matemáticas.

3.3 Problemas NP-completos

Si P es distinto de NP, entonces la distinción entre P y NP-P es importante y significativa. Todos los problemas en P pueden resolverse con algoritmos de tiempo polinomial, mientras que todos los problemas en NP-P son intratables

Cook (1971) introdujo una clase de problemas de decisión que son los más difíciles en NP. Supongamos que tenemos dos problemas de decisión π y π'.

Una *transformación polinomial* es un algoritmo que a partir de un ejemplo dado de π produce en tiempo polinomial un ejemplo de π' tal que se cumple que: para cada ejemplo

I de π, la respuesta a I es "si" si y solamente si la respuesta a la transformación de I (como un ejemplo de π') es "si". Diremos que un problema de decisión π es *NP-completo* si $\pi \in NP$ y todos los problemas en NP pueden ser transformados polinomialmente a π.

Todo problema π NP-completo cumple la siguiente propiedad: si π puede resolverse en tiempo polinomial entonces todos los problemas NP-completos pueden resolverse en tiempo polinomial; i.e. si π es NP-completo y si $\pi \in P$ entonces $P = NP$ Esto justifica la afirmación de que los problemas NP-completos son los problemas más difíciles de la clase NP.

Teorema

El Problema del Cartero Chino en grafos mixtos (MCPP) es NP-completo. Incluso cuando los costes son los mismos, el grafo es planar y el grado máximo de cada vértice es 3.

Papadimitriou (1976)

Teorema

El problema del Cartero Rural en un grafo no dirigido (RPP) es NP-completo.

Incluso cuando el coste de todas las aristas del grafo es 1

El Problema del Cartero Rural en un grafo dirigido (DRPP) es NP-completo. Incluso cuando el coste de todos los arcos del grafo es 1.

Lenstra y Rinooy Kan (1976).

S E C C I O N 1

CIRCUITOS EULERIANOS
Y PROBLEMAS RELACIONADOS

En esta sección nos remontamos a los orígenes de la Teoría de Grafos, cuando Euler en 1736 planteó el Problema de los Puentes de Konigsberg y proporcionó las condiciones para la existencia en un grafo no dirigido de un circuito que atraviese cada arista del grafo exactamente una vez. Establecemos las condiciones para la existencia de tales circuitos, Eulerianos, en grafos dirigidos y mixtos y damos un algoritmo para trazar dicho circuito, si existe, en un grafo no dirigido.

Un problema directamente relacionado con el de la existencia de un circuito Euleriano en un grafo G es el Problema del Cartero Chino (CPP)que consiste en, dado un grafo G con costes no negativos asociados a las aristas, encontrar un circuito que atraviese cada arista de G al menos una vez y con el mínimo coste. Discutimos la resolución exacta de este problema en el caso de grafos dirigidos y no dirigidos y comentamos la dificultad del problema en el caso mixto.

El problema del Cartero Rural (RPP) es una generalización del CPP en el sentido de que el circuito debe atravesar al menos una vez las aristas de un subconjunto dado de aristas, no necesariamente todas, del grafo original. Exponemos los orígenes del RPP,sus posibles aplicaciones, comentamos algunas de sus características y discutimos una serie de problemas relacionados con él.

1.1 GRAFOS Y CIRCUITOS EULERIANOS

Se considera que la Teoría de Grafos comienza en 1736 cuando Euler estableció un problema conocido como el Problema de los Puentes de Konigsberg. La ciudad de Konigsberg - (ahora Kaliningrado) está construida sobre las dos orillas - del rio Pregel y sobre dos islas en el río. Las orillas y las dos islas están unidas por siete puentes tal como se expresa en el mapa de la figura 1.a

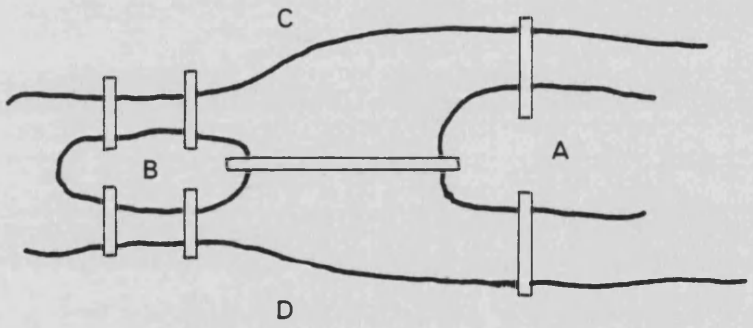


Figura 1.a: Mapa de Konigsberg

El problema planteado era si era posible partir de cualquiera de las cuatro áreas donde se asentaba la ciudad, cruzar cada puente exactamente una vez y volver al punto de - partida.

Si representamos cada orilla y cada isla por un vértice y cada puente por una arista que los une, el mapa de la ciudad puede representarse por el grafo (multigrafo) de la figura 1.b

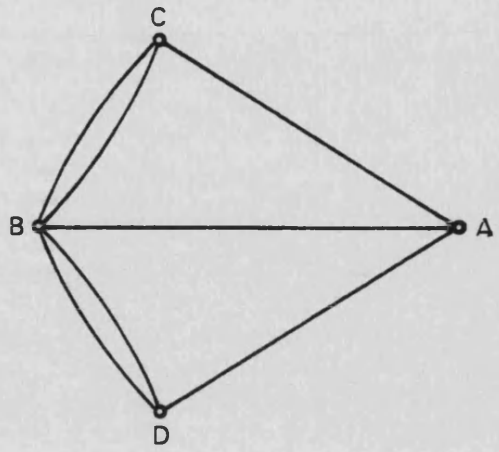


Figura 1.b: Grafo del Problema de los Puentes de Konigsberg

El problema consiste, ahora, en si el grafo de la figura 1.b contiene o nó un circuito euleriano. En general, dado un grafo G, el problema es la existencia o nó de un circuito euleriano (un circuito que atraviese cada arista del grafo exactamente una vez). Si tal circuito existe, se dice que el grafo es Euleriano.

Euler concluyó que el grafo no contenía un circuito Euleriano y resolvió el problema sobre la existencia de tales circuitos mediante el siguiente teorema:

Teorema 1: Un grafo (multigrafo) conexo, no dirigido, contiene un circuito Euleriano si, y solamente si, todos los vértices del grafo tienen grado par.

El siguiente teorema caracteriza la existencia de circuitos eulerianos sobre grafos dirigidos.

Teorema 2: Un grafo (multigrafo) $G=(N,A)$ dirigido y conexo -

contiene un circuito Euleriano si, y solamente si, los grados de entrada $d_t(i)$ y los grados de salida $d_0(i)$ de los vértices satisfacen:

$$d_t(i)=d_0(i) \qquad \forall i \in N$$

Teorema 3: Un grafo mixto $G=(N, A \cup L)$ contiene un circuito Euleriano si, y solamente sí,

a) G es conexo

b) Cada nudo de G es incidente con un número par de arcos y aristas conjuntamente.

c) Para cada subconjunto $S \subseteq N$, la diferencia entre el número de arcos dirigidos desde S a $\bar{S}$ y el número de arcos dirigidos desde $\bar{S}$ a S es menor que o igual al número de aristas uniendo S y $\bar{S}$.

(Ford y Fulkerson (1962))

Existen varias reglas y algoritmos muy sencillos para trazar un circuito Euleriano (si existe) en un grafo no dirigido. Estos algoritmos pueden extenderse fácilmente al caso de grafos dirigidos. Ver Edmondsy Johnson (1973).

El más conocido es eldeFleury (Bondy y Murty, 1976) que consiste en:

1. Escoger un vértice arbitrario i_0 y hacer $w_0=i_0$
2. Supongamos que hemos escogido el camino simple $w_i=i_0 \ell_1 i_1 \ldots \ell_i i_i$ Entonces escoger una arista ℓ_{i+1} del conjunto $A-\{\ell_1, \ell_2, \ldots, \ell_i\}$ de forma que:
   i) ℓ_{i+1} sea incidente con i_i
   ii) a menos que no haya otra alternativa, ℓ_{i+1} no sea una

arista de corte de $G_i = G - \{\ell_1, \ell_2, \ldots, \ell_i\}$ (una arista ℓ_i de G es una arista de corte de G sii no está contenida en algún ciclo de G)

3. Parar cuando la etapa 2 ya no pueda ser ejecutada.

1.2 EL PROBLEMA DEL CARTERO CHINO (CPP)

Un problema directamente relacionado con el de la existencia de un circuito Euleriano en un grafo G es el siguiente: Si las aristas de G tienen costes no negativos, encontrar un circuito que atraviese cada arista de G al menos una vez y de forma que el coste total del recorrido sea mínimo.

Obviamente, si G contiene un circuito Euleriano, éste circuito es óptimo puesto que atraviesa cada arista exactamente una vez. El problema anterior es conocido como el *Problema del Cartero Chino* (CPP) en reconocimiento a que fué planteado por primera vez por Kwan Mei-Ko en 1960. Su solución tiene un gran número de posibles aplicaciones, como la recogida de basuras, reparto de leche o correo, inspección de tendidos electricos, líneas de teléfono o redes de ferrocarril, etc.

El problema,que surgió al diseñar un diagrama para una ruta de un cartero, fué planteado por Kwan Mei-Ko en los siguientes términos: " un cartero debe cubrir su ruta asignada antes de volver a la oficina. El problema es encontrar el recorrido mas corto para el cartero". El problema se reduce al siguiente: " Dado un grafo conexo en el plano, tenemos que dibujar un grafo contínuo(permitiendo las repeticiones) partiendo de un punto dado y volviendo a él, minimizando el número de aristas repetidas."

El método de solución propuesto por Mei-Ko consiste en: a partir de cualquier solución posible dada, ir realizando mejoras sucesivas en ella modificando los costes de las aris

tas y descubriendo ciclos de coste negativo. El inconveniente principal de éste método, como señaló J.Edmonds, es que no resulta evidente la detección de ciclos negativos en un grafo - no dirigido.

En 1969,R.Bellman y K.L.Cooke formulan el problema en el caso en que todos los costes valen 1, al que denominan Problema de los Puentes de Koningsberg Generalizado, como un problema de Programación Dinámica. Este método de solución require tiempos de computación que crecen exponencialmente con el número de aristas del grafo.

El problema general con costes arbitrarios en las aristas fué formulado y resuelto como un problema de matching ,con un subproblema de caminos mas cortos, por Edmonds(1965), Busacker y Saaty(1965), Christofides (1973) y Edmonds y Johnson (1973).

Aunque las técnicas de solución para los casos - dirigido y no dirigido tienen ciertas semejanzas, un caso no- es transformable en otro y por lo tanto debe desarrollarse la solución en forma distinta para cada caso.

1.2.1 El CPP para grafos no dirigidos

Sea $G=(N,A)$ un grafo no dirigido, donde N es un conjunto de n vértices y A es un conjunto de m aristas que pueden ser atravesadas en cualquier dirección. Consideraremos dos casos:

Caso a: El grafo G es par (todos sus vértices son incidentes con un número par de aristas)

La solución óptima es un circuito Euleriano (Teor. 1)

Caso b: El grafo G no es par

En este caso el problema puede formularse como:

$$\text{Min} \sum_{\ell \in A} c_\ell x_\ell \qquad (1.0)$$

suj. a:

$$\sum_{\ell \in A} a_{i\ell}(1+x_\ell) \equiv 0 \pmod 2 \quad \forall i \in N \qquad (1.1)$$

$$x_\ell \geq 0 \text{ , enteras} \qquad \forall \ell \in A \qquad (1.2)$$

donde $[a_{i\ell}]$ representa la matriz de incidencia, definida por:

$$a_{i\ell} = \begin{cases} 1 \text{ si la arista } \ell \text{ es incidente con } i \\ 0 \text{ en otro caso} \end{cases}$$

y la variable entera x_ℓ representa el número de veces extra que la arista ℓ será atravesada, siendo c_ℓ el coste de recorrer la arista $\ell \in A$. En términos del cartero que debe repartir correo, x_ℓ es el número de veces que debe atravesar una calle sin efectuar reparto. Los costes c_ℓ se suponen no negativos, de forma que $x_\ell = 0$, $\forall \ell \in A$ es una solución óptima cuando todos los vértices tienen grado par.

Las restricciones (1.1) pueden escribirse como:

$$\sum_{\ell \in A} a_{i\ell} x_\ell + \sum_{\ell \in A} a_{i\ell} - 2w_i = 0 \qquad \forall i \in N \qquad (1.1')$$

$$w_i \geq 0 \text{ , enteras} \qquad \forall i \in N \qquad (1.3)$$

Notar que $\sum_{\ell \in A} a_{i\ell}$ es el grado del vértice i en G

La solución al problema definido por (1.0), (1.1'), (1.2) y (1,3) corresponde a la resolución de un problema de 1-matching mínimo en el conjunto de vértices impares

de N (para una discusión completa de éste problema, ver Edmonds y Johnson (1973)).

Utilizando la solución óptima del problema anterior, construimos un nuevo grafo $G^*=(N,A^*)$ que contiene $1+x_\ell$ copias de cada arista $\ell \in A$. El grafo G^* es par y por lo tanto contiene un circuito Euleriano, que corresponde a una solución óptima en el grafo original G.

1.2.2 El CPP para grafos dirigidos (DCPP)

Consideremos ahora un grafo dirigido $G=(N,A)$, donde N es un conjunto de n vértices y A un conjunto de m arcos. El grafo debe ser fuertemente conexo para asegurar que para cualquier par de vértices distintos i y j, existe al menos un camino desde i a j.

Como en el caso no dirigido, consideraremos dos casos:

Caso a: El grafo G es simétrico ($\forall i \in N$, el n°de arcos que entran en i, $d_t(i)$, es igual al n°de arcos que salen de i, $d_0(i)$).

Entonces la solución óptima es un circuito Euleriano.

Caso b: El grafo G es no simétrico.

El problema puede formularse como:

$$\text{Min} \sum_{(i,j)\in A} c_{ij}x_{ij} \tag{1.4}$$

suj. a:

$$\sum_j (1+x_{ij})b_{ij} = \sum_j (1+x_{ji})b_{ji} \quad \forall i \in N \tag{1.5}$$

$$x_{ij} \geq 0 \text{ , enteras} \quad \forall (i,j) \in A \tag{1.6}$$

donde $B=[b_{ij}]$ está definida como:

$$b_{ij} = \begin{cases} 1 \text{ si el arco } (i,j)\in A \\ 0 \text{ en otro caso} \end{cases}$$

y x_{ij} representa el número de veces que el arco (i,j) se repite en un circuito óptimo del DCPP. Entonces $1+x_{ij}$ será el número de veces que el arco (i,j) ha sido atravesado en un circuito óptimo. Los costes c_{ij} se suponen no negativos, de forma que $x_{ij}=0\ \forall(i,j)\in A$ es una solución óptima cuando todos los vértices son simétricos.

Las restricciones (1.5) pueden escribirse como:

$$\sum_j x_{ij}b_{ij} - \sum_j x_{ji}b_{ji} = \mathcal{D}(i)\ \forall i\in N \qquad (1.5')$$

donde $\mathcal{D}(i)=d_t(i)-d_0(i)$.

El problema definido por (1.4),(1.5') y (1.6) es un problema de flujos de mínimo coste. Consideremos:

1) Los vértices $i\in N$ con $\mathcal{D}(i)<0$ como sumideros con demanda $-\mathcal{D}(i)$
2) Los vértices $i\in N$ con $\mathcal{D}(i)>0$ como fuentes con oferta $\mathcal{D}(i)$

Este problema puede resolverse añadiendo una superfuente conectada a todas las fuentes y un supersumidero conectado a todos los sumideros. La capacidad de cada arco dirigido hacia el supersumidero será la demanda de su vértice inicial; la capacidad de cada arco que sale de la superfuente será igual a la oferta de su vértice final. Las restantes capacidades son infinitas. Resolvemos, entonces, un problema de flujos de coste mínimo que satisfaga todas las ofertas y demandas.

1.2.3 El CPP para grafos mixtos (MCPP)

Consideremos el caso ahora de un grafo mixto $G=(N,A,L)$, donde L es un conjunto de aristas y A un conjunto de arcos. Un arco debe ser atravesado únicamente según su dirección, mientras que una arista puede ser recorrida en cualquiera (o ambas) de sus direcciones.

Una condición necesaria y suficiente para la existencia de solución al MCPP es la no existencia de un subconjunto propio de vértices S tal que cada arco que una un vértice $i \in S$ con otro $j \in \bar{S}$, esté dirigido desde i hacia j, y no existan aristas entre S y $\bar{S}$. De la misma forma que en el caso dirigido, ésta condición se satisface si exigimos que el grafo sea fuertemente conexo.

Consideraremos que un grafo mixto es par si todos sus vértices son incidentes con un número par de aristas y arcos (conjuntamente) y que es simétrico si los grados de entrada y salida de cada vértice son iguales (independientemente de las aristas incidentes con dicho vértice).

Distinguiremos tres casos:

Caso a: El grafo G es par y simétrico.

La solución es un circuito Euleriano.

Caso b: El grafo G es par pero no simétrico.

Este caso está óptimamente resuelto transformando el grafo original y planteándolo como un problema de flujos de coste mínimo:

En primer lugar, se construye el grafo G_d a partir de G, asignando una dirección arbitraria a cada arista $\ell \in L$

Calculamos en G_d los grados de entrada $d_t(i)$ y de sali

da $d_0(i)$ de cada vértice $i \in N$ y hacemos $D(i)=d_I(i)-d_0(i)$.

Construimos ahora el grafo $G'=(N,A')$ en la forma:

a) por cada arco $(i,j) \in A$ incluimos un arco (i,j) en A' con capacidad infinita y coste igual al de (i,j) en G.

b) por cada arista $\ell=(i,j) \in L$, creamos dos arcos (i,j) y (j,i) en A'. Cada uno de esos arcos tiene capacidad infinita y coste igual al de la arista en G.

c) por cada arista $\ell=(i,j) \in L$, creamos un arco en A' cuya dirección sea la contraria a la asignada a dicha arista al construir el grafo G_d. Estos arcos se llamarán arcos artificiales y tendrán coste 0 y capacidad dos.

Se resuelve sobre G' un problema de flujos de coste mínimo con las ofertas y demandas dadas por los valores de $D(i)$. La solución a éste problema produce una solución óptima al MCPP.

Para una discusión completa de este caso ver Minieka (1978) y Edmonds y Johnson(1973) (aunque el algoritmo propuesto por éstos últimos contiene un pequeño error corregido por Frederickson (1979)).

Caso c: El grafo G no es par.

No existe para este caso una técnica de solución eficiente. Papadimitriou (1976) demuestra que el problema del MCPP es NP-completo. Incluso cuando el grafo es - planar, tiene los mismos costes y los grados de los - vértices son menores o iguales a tres, el problema sigue siendo NP-completo.

En 1979, Kappauf y Koehler proporcionan una formulación del MCPP como un PLE y establecen una correspondencia uno a uno entre los puntos extremos del poliedro de la relajación lineal del MCPP y un conjunto de lo que llaman Grafos Eulerianos Asignados. Como ellos mismos afirman, la búsqueda de los puntos extremos de un poliedro puede ser, en el mejor de los casos, inefi ciente,y computacionalmente intratable en el peor de ellos. Además comentan que el MCPP es bastante degene rado y normalmente muchas bases corresponderán a un - mismo punto extremo. Moverse ,pues, de un punto extre mo a otro puede ser difícil.

Tambien en 1979, Minieka sugiere el planteamiento del problema como el de un problema de flujos con ganancias que podría ser resuelto por Programación Lineal Entera. Este problema se resolvería en un grafo trans formado del original en donde por cada arista debería mos añadir 2 vértices y 5 arcos más. Minieka dice que aunque no elegante, es al menos un posible método de solución.

1.3 EL PROBLEMA DEL CARTERO RURAL (RPP)

El Problema del Cartero Rural surge como una generalización del CPP al exigir que el circuito atraviese al menos una vez cada arista de un subconjunto dado de aristas del grafo original.

1.3.1 Orígenes

El RPP se plantea por primera vez en Orloff (1974) quien lo enuncia en los siguientes términos: El Problema del Cartero Rural es el problema de encontrar un circuito de coste mínimo que atraviese cada arista de un subconjunto requerido de aristas de un grafo.

Orloff considera que tanto el RPP como el Problema del Agente Viajero (TSP) son casos particulares de un problema al que denomina Problema General de Rutas (GRP) y que define en la forma siguiente: El GRP es el problema de encontrar un circuito de coste mínimo que atraviese cada arista de un subconjunto requerido de aristas del grafo y que visita cada vértice de un subconjunto requerido de vértices del grafo. (Obviamente el CPP, el RPP y el TSP tanto en grafos dirigidos como en grafos no dirigidos son casos particulares del GRP).A continuación formula el GRP como un PLE y construye un algoritmo del tipo de B&B basado en la eliminación de subcircuitos.

Lenstra y Rinooy Kan (1976) demuestran que es incorrecta una transformación que parecía hacer posible la transformación de un TSP a un GRP más sencillo de resolver.Pero el error principal es la formulación planteada para el GRP

que nosotros consideramos incorrecta. Veamos cuál es la formulación propuesta por Orloff:

Consideremos un grafo $G=(N,A)$ no dirigido; sea $R \subseteq A$ el conjunto de aristas requeridas para ser atravesadas, sea N_R el conjunto de vértices incidentes con aristas de R y sea Q el conjunto de vértices requeridos en G. (Suponemos que $Q \cap N_R = \phi$). Sea E^* el subconjunto de vértices de N_R de grado impar, $T = E^* \cup Q$ Sea $S=\{S_1, S_2, \ldots, S_F\}$ el conjunto de componentes desconectadas en (N_R, R). Todos los nudos $j \in Q$ están desconectados en $G^* = (Q \cup N_R, R)$ pero no se les considera como subcircuitos. Supone que no existen inicialmente subcircuitos en el grafo G^*, por lo tanto todos los S_i tendrán al menos dos vértices impares cada uno. Se construye el grafo completo $F(T_G)$ (el grafo completo formado por los vértices de T y todos los caminos más cortos en G que los conectan). Se define c'_{ij}=coste del cmc en $G, i \in T, j \in T, i \neq j$ y $c'_{ii} = \infty \ \forall i \in T$.

El problema lo formula como:

$$\text{Min} \sum_{i=1}^{|T|-1} \sum_{j=i+1}^{|T|} c'_{ij} x_{ij}$$

sujeto a:

$$\sum_{i=1}^{j-1} x_{ij} + \sum_{i=j+1}^{|T|} x_{ji} = 1 \qquad \forall j \in E^*$$

$$\sum_{i=1}^{j-1} x_{ij} + \sum_{i=j+1}^{|T|} x_{ji} = 2 \qquad \forall j \in Q$$

No se permiten subcircuitos

$x_{ij} = 0$ ó 1

x_{ij} *existe para* $j > i$ *solamente*

Veamos que ocurre en el siguiente grafo:

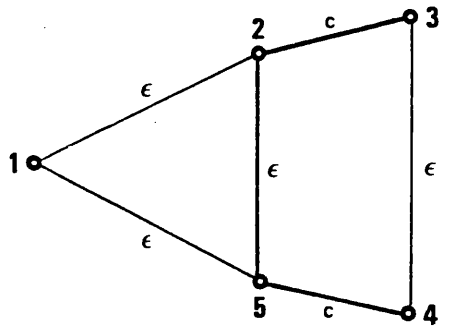


Figura 1.c: Grafo original

En este grafo, consideremos el GRP donde $Q=\{1\}$, $R=\{(2,3),(2,5),(4,5)\}$ y $c,\varepsilon > 0$. Construimos $E^*=\{3,4\}$ $T=\{1,3,4\}$ y el grafo completo $F(T_G)$, que será:

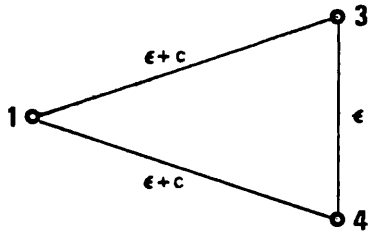


Figura 1.d: Grafo completo $F(T_G)$

El GRP en este caso se plantearía como:

$$\text{Min} \left\{ (c+\varepsilon)x_{13} + (c+\varepsilon)x_{14} + \varepsilon x_{34} \right\}$$

$$x_{13} + x_{34} = 1$$

$$x_{14} + x_{34} = 1$$

$$x_{13} + x_{14} = 2$$

no se permiten subcircuitos

$$x_{13}, x_{14}, x_{34} = 0,1$$

La solución al problema planteado sería: $x_{13}=x_{14}=1, x_{34}=0$, lo que nos proporcionaría la siguiente solución al GRP:

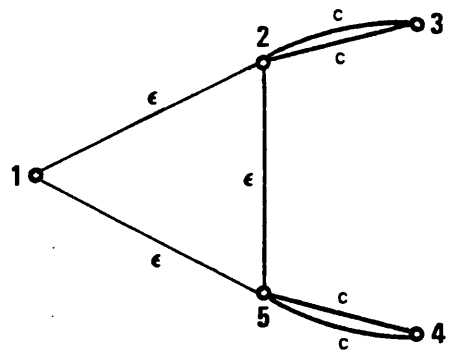


Figura 1.e: Solución a la formulación del GRP

que obviamente no es óptima, tal como demuestra la figura siguiente:

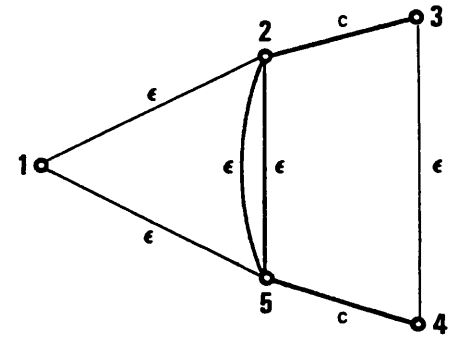


Figura 1.f: Contraejemplo

eligiendo valores apropiados para ε y c (siempre que ε < c).

1.3.2 Definición y Características

Consideremos un grafo conexo, no dirigido $G=(N,A)$ donde N es un conjunto de n vértices y A es un conjunto de m aristas. Sea $c_\ell \geq 0$ el coste de la arista ℓ. Dado un subconjunto de aristas requeridas $A_R \subseteq A$ $(A_R \neq \phi)$, el Problema del Cartero Rural es el de encontrar un circuito de coste mínimo tal que cada arista de A_R sea atravesada al menos una vez. Si $A_R=A$, tenemos el Problema del Cartero Chino que ya comentamos en el apartado 1.2.

El RPP es un problema de Optimización Combinato-

rial que puede considerarse como una generalización del Problema del Agente Viajero (TSP) en el sentido de que un TSP puede convertirse en un RPP en la forma siguiente:

Se reemplaza cada ciudad i del TSP por una arista requerida (i',i'') del RPP con coste cero. Cada arista (i,j) del TSP es reemplazada por una arista (i',j') en el RPP con el mismo coste pero no requerida. Las aristas no requeridas en la solución del RPP forman una solución del TSP.

Una transformación similar a la anteriormente propuesta fué la utilizada para demostrar la NP-completitud del RPP (Lenstra y Rinooy Kan (1976)).

Si se especifica un vértice de partida para el cartero, en el sentido de que el cartero debe partir de, y volver a, ese vértice (por ejemplo la Oficina de Correos), y éste es uno de los vértices incidentes con alguna arista requerida, no se requiere ninguna transformación y la solución al RPP satisfará la condición expuesta. Si, en cambio, ese vértice de partida no es incidente con aristas requeridas, podemos realizar una transformación similar a la expuesta para el TSP y así asegurarnos que el circuito solución al RPP pase por dicho vértice.

El propósito de ésta memoria es el estudio del RPP y el diseño de un algoritmo exacto para su resolución. La solución proporcionada por nuestro algoritmo será el grafo Euleriano de coste mínimo que contiene cada arista requerida al menos una vez. El problema de encontrar el circuito Euleriano en dicho grafo es muy sencillo y ya fué comentado en el apartado 1.1.

1.3.3 Aplicaciones y Problemas Relacionados

El Problema del Cartero Rural forma parte de una clase importante de problemas de gran incidencia práctica:los Problemas de Rutas (Routing Problems) que pueden clasificarse en Problemas de Rutas sobre Vértices (Node Routing), sobre arcos o aristas (Arc Routing) o sobre vértices y arcos (General Routing), y cada uno de éstos tipos de problemas a su vez se subdividiría en dos, según se consideren restricciones de capacidades (de los vehículos) o no.

La importancia que en los últimos años han ido adquiriendo los problemas de Routing y, en nuestro caso, los de Routing sobre arcos o aristas, está justificada por la posibilidad de aplicarlos a un número cada vez mayor de problemas prácticos. Las aplicaciones de éste tipo de problemas incluyen la limpieza y riego de las calles, recogida de basuras, reparto de leche o correo, rutas de autobuses escolares, pulverización con sal de las carreteras para prevenir la formación de hielo, la inspección y mantenimiento de tendidos eléctricos, líneas de teléfono, conducciones de gas o petroleo,redes de ferrocarril,etc.

Una prueba de la importancia de este tipo de problemas y de la atención cada vez mayor que están recibiendo es el International Workshop on the Routing and Scheduling of Vehicles and Crews que se celebró en 1979 en la Universidad de Maryland (USA), cuyos trabajos han sido recogidos en un número especial de la revista Networks en el volumen correspondiente a 1981.

La mayor parte de los trabajos realizados en problemas de Rutas consisten en el diseño de algoritmos heurísticos que proporcionen una buena solución posible (bajo ciertas suposiciones sobre las características del grafo como la no existencia de arcos y la de que el grafo sea planar) y que satisfagan una serie de condiciones relativas a la especifidad de cada problema. Así, por ejemplo, Beltrami y Bodin (1974) consideran algunos problemas relativos a la recogida de basuras y desarrollan procedimientos heurísticos que fueron aplicados en las ciudades de Nueva York y Washington; Male y Liebman (1978) describen también un esquema eficiente de rutas para la recogida de basuras que fué aplicado en la ciudad de Knoxville (Tenn. USA); Stern y Dror (1979) elaboran un algoritmo para el problema, relacionado con el k-CPP, de diseñar las rutas para los lectores de contadores eléctricos, y lo aplican en la ciudad de Beersheva (Israel), en donde lograron un 40% de reducción en el número de rutas de trabajo.

La extensión natural del CPP es el k-CPP, que consiste en encontrar un conjunto de k circuitos que partan de un mismo vértice y que conjuntamente recorran todas las aristas del grafo con el mínimo coste. Frederickson,Hecht y Kim (1978) demuestran que es un problema NP-completo y proporcionan un algoritmo aproximado para su resolución cuya cota en el peor de los casos es de $2-^{1}/_{k}$. Christofides (1973) propone un algoritmo heurístico para este problema cuando se le imponen restricciones de capacidad sobre los circuitos. Tambien Golden y Wong (1981) construyen un algoritmo de aproximación para este

último caso y demuestran que es NP-completo; problema que denominan Problema del Cartero Chino Capacitado y que consideran un caso particular del Problema de Routing Capacitado sobre Arcos (CARP).

S E C C I O N 2

TRANSFORMACIONES Y FORMULACION DEL RPP

En ésta Sección definimos el Problema del Cartero Rural sobre un grafo G, conexo y no dirigido, y realizamos ciertas transformaciones sobre G con objeto de simplificar su estructura y formulación. Para ello, a partir del grafo G_R, inducido en G por el subconjunto de aristas requeridas A_R, construimos el grafo completo G_C' añadiendo a G_R una arista entre cualquier par de vértices, con coste igual al del camino más corto entre ellos (calculado en el grafo original G). Demostramos que un circuito en G_C' que atraviese cada arista de A_R al menos una vez produce un circuito solución al RPP en G, y que cada solución óptima del RPP en G_C' produce una solución óptima del RPP en G con el mismo coste. Es posible simplificar el grafo G_C' y obtener un grafo G_C con las mismas soluciones al RPP y una estructura mucho más reducida.

Formulamos el Problema del Cartero Rural sobre el grafo G_C como un PLE y demostramos ciertas propiedades de ésta formulación: cotas superiores e inferiores para todas las variables del problema, redundancia de algunas de las restricciones en presencia de otras,... Finalmente se da la formulación definitiva del problema que será la base de las sucesivas Secciones de ésta Memoria.

2.1 TRANSFORMACIONES SOBRE EL GRAFO ORIGINAL

Con objeto de poder simplificar la estructura del problema y su formulación, como un problema de PLE, vamos a realizar previamente algunas transformaciones sobre el grafo original G. Sea $G=(N,A)$ un grafo conexo y no dirigido, donde N es un conjunto de n vértices y A es un conjunto de m aristas. Sea c_ℓ el coste asociado a la arista $\ell \in A$,siendo $c_\ell \geq 0$. El Problema del Cartero Rural consiste en, dado un subconjunto de aristas "requeridas" $A_R \subseteq A$ ($A_R \neq \emptyset$), encontrar un circuito de coste mínimo que atraviese cada arista de A_R al menos una vez. El coste del circuito es la suma de los costes de sus aristas.

2.1.1 Construcción del grafo completo $G'_C = (N_R, A_R \cup A'_S)$

Sea $G_R = (N_R, A_R)$ el grafo inducido por A_R , donde N_R es el subconjunto de vértices de N incidentes con, al menos, una arista de A_R. A partir de G_R podemos construir el grafo completo, que representaremos por G'_C. Para ello bastará con añadir a G_R una arista entre cada par de vértices de N_R , exigiendo que el coste de dicha arista sea igual al coste del camino más corto entre sus vértices (calculado en el grafo original). El grafo completo resultante será $G'_C=(N_R, A_R \cup A'_S)$, donde A'_S es el conjunto de aristas añadidas. Notar que en G'_C hay una arista adicional en paralelo con cada arista de A_R . La figura 2.a nos muestra el grafo original G donde el

conjunto de aristas requeridas A_R está representado por líneas de trazo continuo.

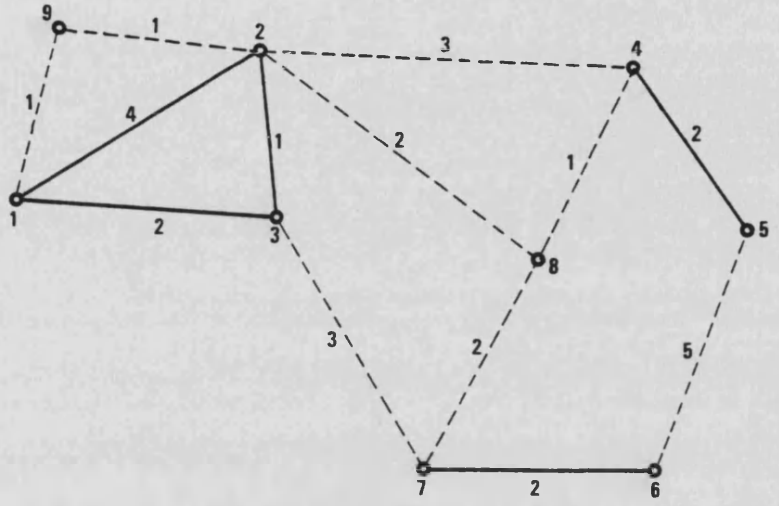


Figura 2.a : El grafo original

La figura 2.b nos muestra el grafo resultante completo G'_C.

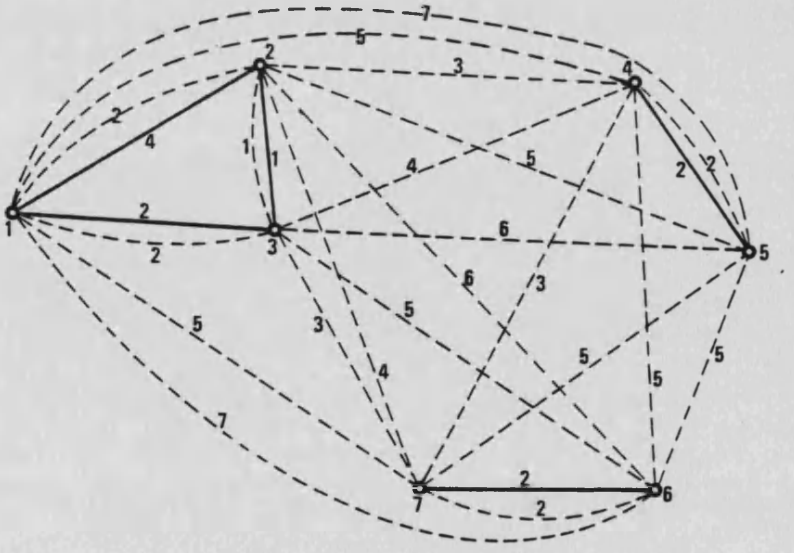


Figura 2.b : El grafo G'_C

La transformación realizada ofrece ciertas ventajas. El grafo inducido por las aristas requeridas A_R , G_R , está formado por un conjunto finito de componentes conexas $C_1, C_2, \ldots, C_k$,disconexas entre sí. Un circuito solución al RPP deberá, pues, utilizar aristas del conjunto $A-A_R$ para conectar dichas componentes. Toda solución posible al RPP utilizará aristas no requeridas y como, cualquier circuito cumple que el grado de todos sus vértices es par, una condición que se debería exigir al formular el problema como un PLE es que el grado de todos los vértices del circuito sea par. Y no sólo el de los vértices de N_R , sino también el de aquellos vértices de $N-N_R$ incidentes con las aristas de $A-A_R$ utilizadas por la solución. La ventaja que ofrece la transformación descrita anteriormente es que trabajamos en un grafo G_C' donde, el conjunto de vértices está constituido únicamente por aquellos que son incidentes con aristas de A_R . Y dado que $N_R \subseteq N$, tendremos que $|N_R| \leq |N|$, por lo tanto el número de restricciones que deberemos imponer será menor.

Vamos,ahora, a demostrar que cualquier solución al Problema del Cartero Rural sobre el grafo G_C' corresponde a una solución sobre el grafo original y que una solución óptima del RPP sobre G_C' corresponde a una solución óptima del mismo coste sobre G .Veamos la relación existente entre los conjuntos de soluciones posibles al RPP sobre los grafos G y G_C'.

Proposición 2.1

Toda solución posible al RPP sobre G'_C proporciona una solución posible al RPP sobre el grafo original G del mismo coste.

Demostración:

Toda solución al RPP sobre el grafo G'_C es un circuito que atraviesa las aristas de A_R al menos una vez. Si descomponemos las aristas $\ell=(i,j) \in A'_S$, $i,j \in N_R$, que aparecen en el circuito solución sobre G'_C ,en los caminos más cortos $(i,\alpha_1),(\alpha_1,\alpha_2),\ldots,(\alpha_k,j)$ con $\alpha_1,\alpha_2,\ldots,\alpha_k \in N$,que las generaron, tendremos un circuito solución al RPP sobre el grafo G y del mismo coste.

Proposición 2.2

Toda solución óptima al RPP sobre el grafo G'_C proporciona una solución óptima al RPP sobre el grafo G ,del mismo coste.

Demostración:

Sea x_1 una solución óptima al RPP sobre el grafo G'_C de coste $z(x_1)$; sabemos, por la proposición anterior, que x_1 proporciona una solución posible x_2 al RPP sobre G del mismo coste, es decir $z(x_1)=z(x_2)$. Vamos a comprobar que x_2 es una solución óptima al RPP sobre G .

Supongamos que no fuera cierto; es decir, existe una solución x^* al RPP en G de coste $z(x^*)<z(x_2)$. A partir de la solución x^*, construimos el grafo $G^*=(N_R,A^*)$, donde A^*está formado por las aristas de A_R(tantas veces como aparezcan en

x^*) y una arista $\ell \in A^*$, por cada camino μ_{ij} $(i,j \in N_R)$ en x^* formado exclusivamente por aristas de $A-A_R$, de coste igual al del camino más corto en G entre $i \in N_R$ y $j \in N_R$. Obviamente este grafo G^* corresponde a un circuito x sobre el grafo G_C' de coste $z(x) \leq z(x^*)$ y por lo tanto $z(x) < z(x_1)$. Lo que contradice la hipótesis de que x_1 era una solución óptima sobre G_C'.

2.1.2 Grafo simplificado $G_C = (N_R, A_R \cup A_S)$

En el apartado anterior hemos discutido la conveniencia de construir el grafo G_C' y de resolver el RPP sobre dicho grafo en lugar de hacerlo sobre el grafo original G. Un problema con el que nos encontramos ahora es la magnitud del cardinal de A_S' ($|A_S'|$). Recordemos que el grafo G_C' es completo y por lo tanto si $|N_R| = n_r$, entonces $|A_S'| = \binom{n_r}{2}$. Como la dificultad del problema depende en gran medida del número de aristas no requeridas en el grafo, vamos a simplificar el grafo G_C' y obtener un nuevo grafo $G_C = (N_R, A_R \cup A_S)$ que tenga las mismas soluciones al RPP y menor número de aristas no requeridas.

A partir del grafo $G_C' = (N_R, A_R \cup A_S')$, obtenemos el grafo $G_C = (N_R, A_R \cup A_S)$, eliminando:

a) Todas las aristas $(i,j) \in A_S'$ tales que $c_{ij} = c_{ik} + c_{kj}$ para algún k, y
b) Una de las dos aristas en paralelo si ambas tienen el mismo coste.

La figura 2.c nos muestra el grafo G_C correspondiente al grafo G_C' de la figura 2.b

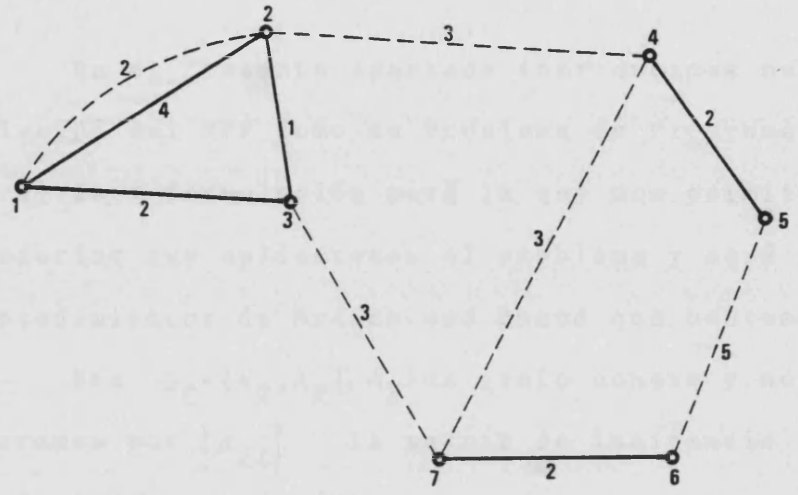


Figura 2.c : El grafo G_C

Al igual que en el apartado anterior debemos establecer las relaciones existentes entre las soluciones posibles y óptimas al RPP sobre los grafos G_C' y G_C . Estas relaciones vienen expresadas en la siguiente Proposición, cuya demostración se omite debido a que es consecuencia directa de la construcción del grafo G_C (en donde $A_S \subseteq A_S'$).

Proposición 2.3

Cualquier solución posible al RPP sobre G_C es una solución posible al RPP sobre G_C', y cualquier solución óptima al RPP sobre G_C lo es también sobre el grafo G_C' .

Comentario 2.1

Como aplicación de las Proposiciones 2.1, 2.2 y 2.3 , podemos definir el Problema del Cartero Rural como el problema de encontrar un circuito, sobre el grafo $G_C = (N_R, A_R \cup A_S)$, de coste mínimo que atraviese cada arista de A_R al menos una vez.

2.2 FORMULACION DEL RPP COMO UN PLE

En el presente apartado introducimos nuestra propia formulación del RPP como un Problema de Programación Lineal Entera. Esta formulación será la que nos permita calcular la cota inferior que aplicaremos al problema y será la base de los procedimientos de Branch and Bound que usaremos.

Sea $G_C=(N_R, A_R \cup A_S)$ un grafo conexo y no dirigido. Representaremos por $[a_{il}]$ la matriz de incidencia del grafo, $i \in N_R$, $l \in A_R \cup A_S$, es decir:

$$a_{il} = \begin{cases} 1 & \text{si la arista } l \text{ es incidente con } i \\ 0 & \text{en otro caso} \end{cases}$$

2.2.1 Definición de las variables y Formulación

Definición 2.1 : definimos x_l como el número de veces que la arista $l \in A_R$ (arista requerida) se repite en un circuito óptimo del RPP sobre el grafo G_C . Entonces $1+x_l$ será el número de veces que la arista l ha sido utilizada en un circuito óptimo del RPP.

Definición 2.2 : definimos y_l como el número de veces que la arista $l \in A_S$ (arista no requerida) aparece en un circuito óptimo del RPP sobre el grafo G_C .

Sean $C_1, C_2, \ldots\ldots, C_k$ las componentes conexas, disconexas entre sí, del grafo inducido por A_R en G_C. Representaremos tambien por C_i el conjunto de vértices de la componente i . La familia de las C_i, $i=1 \ldots k$, la representaremos por F .

Si $V \subset F$ es una subfamilia de F , utilizaremos $N(V)$ para representar el conjunto de todos los vértices de los elementos de V, es decir : $N(V) = \bigcup_{C_i \in V} C_i$

El RPP puede formularse como :

$$(P_1) \qquad \text{Min} \left\{ \sum_{\ell \in A_R} c_\ell x_\ell + \sum_{\ell \in A_S} c_\ell y_\ell + \sum_{\ell \in A_R} c_\ell \right\} \tag{0}$$

sujeto a :

$$\sum_{\ell \in A_R} a_{i\ell}(1+x_\ell) + \sum_{\ell \in A_S} a_{i\ell} y_\ell \equiv 0 \pmod 2 \qquad \forall i \in N_R \tag{1}$$

$$\sum_{\ell \in K_t} y_\ell \geq 2 \qquad \forall K_t = \left\{ (i,j) \in A_S \mid i \in N(V_t), j \in N(\bar{V}_t), V_t \subset F \right\} \tag{2}$$

$$\begin{aligned} x_\ell &\geq 0 \quad \text{, enteras} \qquad \forall \ell \in A_R \\ y_\ell &\geq 0 \quad \text{, enteras} \qquad \forall \ell \in A_S \end{aligned} \tag{3}$$

donde por K_t representaremos un conjunto de aristas de corte de G_C.

Veamos cuál es el significado de cada conjunto de restricciones del problema (P_1) :

Las restricciones del tipo (1) expresan la condición de paridad de los grados de los vértices, puesto que el primer miembro de dichas restricciones nos da el número de aristas, tanto requeridas como no requeridas, así como sus repeticiones, que son incidentes con un vértice $i \in N_R$.

Las restricciones del tipo (2), que llamaremos de conectividad, expresan la condición de conexión entre las componentes conexas $C_1, \ldots, C_k$ inducidas por A_R en G_C (en

el sentido de que el cartero debe entrar y salir de cada componente), evitando la formación de subcircuitos de la forma, por ejemplo, siguiente:

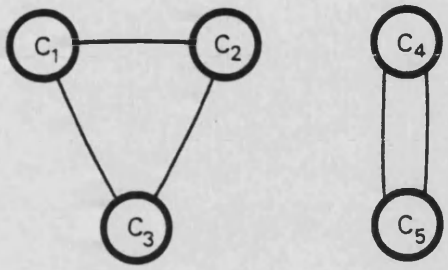


Figura 2.d : Subcircuitos no permitidos

Las restricciones (3) son las condiciones de no negatividad de las variables, y la función objetivo (0) intenta minimizar el peso de las aristas que deben aparecer y repetirse para cumplir las condiciones impuestas. Recordemos que se suponen no negativos los costes de las aristas.

Vamos a demostrar la equivalencia de este problema con el RPP.

Teorema 2.4

El Problema lineal entero (P_1) es equivalente al RPP definido sobre el grafo G_C.

Demostración:

Para demostrarlo veremos que existe una correspondencia biunívoca entre las soluciones posibles de (P_1) y los circuitos solución al RPP sobre G_C. Además esta correspondencia asocia soluciones que tienen el mismo coste en sus problemas respectivos.

1) Veamos en primer lugar que a toda solución posible (x,y) de (P_1) le podemos asociar un circuito que atraviesa, al menos

una vez, las aristas de A_R y que tiene el mismo coste.

A partir del grafo $G_R=(N_R,A_R)$, construimos el grafo $G^*=(N_R,A_R \cup A^*)$ en la forma siguiente: añadimos a G_R tantas copias de la arista $\ell \in A_R$ como indique el valor de la variable x_ℓ en la solución de (P_1) y tantas apariciones de la arista $\ell \in A_S$ como indique y_ℓ en la solución a (P_1).

Veamos que el grafo G^* es un circuito. Para ello, será necesario comprobar que todos los vértices en G^* tienen grado par y que G^* es conexo. Puesto que la solución (x,y) satisface las restricciones (1) $,\forall i \in N_R$, tenemos que todos los vértices de G^* serán incidentes con un número par de aristas, luego tendrán grado par. Además, si el grafo G^* fuera no conexo, y teniendo en cuenta que las componentes $C_1,.....,C_k$ del grafo inducido G_R son conexas, eso implicaría que $\exists V_t \subset F$(F es la familia de los C_i $i=1....k$)de forma que entre los vértices de V_t y de $\overline{V}_t$ no existiría ninguna arista. Lo cual es absurdo puesto que la solución (x,y) a (P_1) cumple las restricciones (2).

Hemos visto,pues, que a partir de una solución posible de (P_1) podemos construir un grafo G^* (un circuito) que contiene las aristas de A_R al menos una vez y que, por construcción, la suma de los costes de todas sus aristas es igual al coste de la solución (x,y) de (P_1).

2) Tenemos que demostrar en segundo lugar, que a todo circuito solución del RPP sobre G_C , le podemos asociar un par (x,y) que es una solución posible de (P_1) con el mismo coste.

Dado el circuito C^* solución del RPP sobre el grafo G_C, vamos a asignar los siguientes valores a las variables x_ℓ, $\ell \in A_R$ e y_ℓ, $\ell \in A_S$:

$$\forall \ell \in A_R \qquad x_\ell = \left\{ \text{n°apariciones de la arista } \ell \text{ en } C^* \right\} - 1$$

$$\forall \ell \in A_S \qquad y_\ell = \left\{ \text{n°apariciones de la arista } \ell \text{ en } C^* \right\}$$

Comprobaremos que (x,y) definido por la asignación anterior, es una solución posible de (P_1).

Se satisfacen las restricciones (1) para cualquier vértice $i \in N_R$. Si ésto no fuera así, significaría que algún vértice $j \in N_R$ del circuito C^* es incidente con un número impar de aristas, lo que es absurdo. Las restricciones del tipo (2) también se cumplen para cualquier conjunto de aristas de corte. En efecto, si existiera un conjunto K_t tal que

$$\sum_{\ell \in K_t} y_\ell \leqslant 1$$

ésto implicaría que partiendo de un vértice cualquiera i, $i \in N(V_t)$ no podríamos llegar a otro vértice $j \in N(\overline{V}_t)$ ó bien, que utilizando una arista para llegar a $j \in N(\overline{V}_t)$ no podríamos volver al vértice inicial $i \in N(V_t)$, lo cual es absurdo teniendo en cuenta que C^* es un circuito que pasa por todas las aristas de A_R al menos una vez.

Para ver que un circuito solución al RPP sobre G_C y la solución posible de (P_1) asociada tienen el mismo coste, basta considerar que el coste de un circuito es la suma de los costes de sus aristas.

2.2.2 Simplificación en la formulación

En la formulación (P_1) del RPP, que hemos visto en el apartado anterior, hemos formalizado matemáticamente

las condiciones que, como se vió en el Teorema 2.4, debían caracterizar a los circuitos solución al RPP. Sin embargo, puesto que el método que vamos a utilizar para resolver el RPP es un método de Branch & Bound, es de especial importancia el cálculo de una buena cota inferior. Entendiendo por buena que se aproxime lo más posible al valor óptimo del problema y que sea sencilla y rápida de calcular.

Con este objetivo, hemos intentado caracterizar mejor las restricciones que definirán el subproblema cuya resolución nos proporcionará la cota inferior, simplificando la formulación de (P_1) en la forma que establece la siguiente proposición.

Proposición 2.5

En presencia de las restricciones (1), las restricciones del tipo (2) son equivalentes a las restricciones

$$\sum_{\ell \in K_t} y_\ell \geq 1 \qquad \forall K_t = \left\{ (i,j) \in A_S \mid i \in N(V_t), j \in N(\bar{V}_t), V_t \subset F \right\} \qquad (2')$$

Demostración :

Obviamente (2)----> (2')

Veamos ahora que (1) y (2')----> (2)

Para demostrar esta implicación vamos a probar que en un grafo no dirigido $G=(N,A)$ conexo y par, dado un subconjunto cualquiera de vértices $N_1 \subset N$, el número de aristas entre N_1 y $N-N_1$ es par. En efecto, representaremos por $G_1=<N_1>$ el subgrafo de G inducido por los vértices de N_1. Sabemos que en cualquier grafo la suma de los grados de los vértices es un número par, por

lo tanto : $$\sum_{i \in N_1} d_{G_1}(i) \equiv 0 \pmod 2$$

donde $d_G(i)$ representa, en general, el grado del vértice i respecto del grafo G.

Si denotamos por p el número de aristas entre N_1 y $N-N_1$ ($p \neq 0$, por ser G conexo), como por hipótesis el grafo G es par, tenemos : $$\sum_{i \in N_1} d_G(i) = \sum_{i \in N_1} d_{G_1}(i) + p \equiv 0 \pmod 2$$

de donde concluimos que p debe ser un número par.

Luego no puede ocurrir, aplicando el resultado anterior, que exista un conjunto de aristas de corte K_t tal que $\sum_{\ell \in K_t} y_\ell = 1$; con lo que queda demostrada la proposición.

Comentario 2.2

Aplicando la Proposición anterior, podemos formular el problema (P_2) , equivalente a (P_1), como el definido por la función objetivo (0) y los conjuntos de restricciones (1), $(2')$ y (3).

En éste problema (P_2) vamos a introducir nuevas variables enteras $w_i, i \in N_R$ a fin de convertir las restricciones (1) en las ecuaciones $(1')$ y (4) :

$$\sum_{\ell \in A_R} a_{i\ell}(1+x_\ell) + \sum_{\ell \in A_S} a_{i\ell} y_\ell - 2w_i = 0 \qquad \forall i \in N_R \qquad (1')$$

$$w_i \geq 0, \text{ enteras } \forall i \in N_R \qquad (4)$$

Las siguientes proposiciones nos van a proporcionar cotas para los valores de las variables del problema en cualquier solución óptima del RPP.

Proposición 2.6

Las variables asociadas con una solución óptima del RPP están acotadas en la forma siguiente:

$$0 \leq x_\ell \leq 1 \quad \text{y enteras} \quad \forall \ell \in A_R$$
$$0 \leq y_\ell \leq 2 \quad \text{y enteras} \quad \forall \ell \in A_S \qquad (3')$$

Demostración :

La demostración es evidente puesto que si no fuera cierto, los valores de x_ℓ e y_ℓ podrían ser reducidos de 2 en 2 sin perder la posibilidad como solución. Y, puesto que $c_\ell \geq 0 \quad \forall \ell \in A_R \cup A_S$, el valor de la función objetivo disminuiría o permanecería igual.

Comentario 2.3

La proposición anterior es una extensión del resultado de Edmonds y Johnson para el Problema del Cartero Chino (CPP) en un grafo no dirigido. En el CPP ($A_R = A$, i.e: hay que atravesar todas las aristas del grafo), Edmonds y Johnson demuestran que el número máximo de repeticiones de una arista es una. Nosotros extendemos este resultado (permitir una repetición de una arista requerida es equivalente a permitir dos apariciones de la misma) al caso de las aristas del conjunto A_S, aquellas aristas que no han de ser necesariamente atravesadas.

Proposición 2.7

Las variables w_i, $i \in N_R$ satisfacen la condición:

$$d(i) \leq w_i \leq d_{G_C}(i) \quad \text{y enteras} \quad \forall i \in N_R \qquad (4')$$

donde:

$$d(i) = \begin{cases} 1/2\ d_{G_R}(i) & \text{si el vértice } i \text{ es par en } G_R \\ 1/2\left[1+d_{G_R}(i)\right] & \text{en otro caso} \end{cases}$$

Demostración:

A partir de las restricciones (1') y (3') , tenemos: (a) $w_i \geq 1/2 \sum_{\ell \in A_R} a_{i\ell} = 1/2\ d_{G_R}(i)$

Además, considerando que w_i debe tomar valores enteros, tenemos:

$w_i \geq 1/2\ d_{G_R}(i)$, si $d_{G_R}(i)$ es par

y

$w_i \geq 1 + \left\lfloor 1/2\ d_{G_R}(i) \right\rfloor$, si $d_{G_R}(i)$ es impar

donde por $\lfloor x \rfloor$ representamos la parte entera por defecto de x.

(b) $w_i \leq 1/2 \left[\sum_{\ell \in A_R} 2a_{i\ell} + \sum_{\ell \in A_S} 2a_{i\ell} \right] = \sum_{\forall \ell \in A_R \cup A_S} a_{i\ell} = d_{G_C}(i)$.

Entonces el RPP puede formularse como:

$$(P_R) \qquad \text{Min} \sum_{\ell \in A_R} c_\ell x_\ell + \sum_{\ell \in A_S} c_\ell y_\ell + \sum_{\ell \in A_R} c_\ell \qquad (0)$$

sujeto a :

$$\sum_{\ell \in A_R} a_{i\ell}(1+x_\ell) + \sum_{\ell \in A_S} a_{i\ell} y_\ell - 2w_i = 0 \qquad \forall i \in N_R \qquad (1')$$

$$\sum_{\ell \in K_t} y_\ell \geq 1 \qquad \forall K_t = \left\{ (i,j) \in A_S \mid i \in N(V_t), j \in N(\bar{V}_t), V_t \subset F \right\} \qquad (2')$$

$$0 \leq x_\ell \leq 1 \qquad \forall \ell \in A_R$$

$$0 \leq y_\ell \leq 2 \qquad \forall \ell \in A_S \qquad (3')$$

$$d(i) \leq w_i \leq d_{G_C}(i) \qquad \forall i \in N_R \qquad (4')$$

donde la expresión para $d(i)$ viene dada en la proposición 2.7

Comentario 2.4

Es importante señalar que la formulación definitiva (P_R) dada al RPP es una generalización de la planteada por Edmonds y Johnson para el CPP.

En efecto, si $A_S=\phi$ ($A_R=A$ y $N_R=N$), a partir de (P_R), obtendríamos el siguiente problema formulado sobre el grafo original $G=(N,A)$

$$\text{Min} \quad \sum_{\ell \in A} c_\ell x_\ell + \sum_{\ell \in A} c_\ell$$

$$\sum_{\ell \in A} a_{i\ell}(1+x_\ell) - 2w_i = 0 \qquad \forall i \in N$$

$$x_\ell = \{0,1\} \text{ entera } \forall \ell \in A$$

$$d(i) \leq w_i \leq d_G(i) \text{ entera } \forall i \in N$$

que coincide con el problema formulado para el CPP, y que Edmonds y Johnson demuestran que es un caso especial del problema de matching (acoplamiento) general.

Comentario 2.5

Una vez planteado (P_R) como la formulación definitiva del RPP sobre el grafo G_C, veamos cuáles son sus dimensiones en cuanto a número de variables y número de restricciones. Consideremos el grafo original $G=(N,A)$. A partir del subconjunto de aristas $A_R \subset A$, formamos el grafo inducido $G_R=(N_R,A_R)$ que consiste en k componentes conexas y a partir de este grafo,

mediante las transformaciones descritas en los apartados 2.1.1 y 2.1.2 , obtenemos el grafo $G_C=(N_R, A_R \cup A_S)$.

El número de variables del problema viene dado por $|A_R|+|A_S|+|N_R|$ y el número de restricciones por $2*|N_R|+ +2^{k-1} -1+|A_R|+|A_S|$.

S E C C I O N 3

COTAS INFERIORES PARA EL RPP

Vamos a desarrollar, en la Sección 5, un procedimiento exacto de solución para el RPP basado en los métodos de Branch and Bound. En cada nudo del árbol de enumeración calcularemos cotas inferiores al problema, resolviendo problemas relacionados con él que son relajaciones de (P_R). Como en todos los métodos de Branch & Bound, la calidad de las cotas calculadas será de gran importancia en la efectividad del algoritmo.

Observando la formulación del RPP dada en (P_R), identificamos fácilmente tres conjuntos distintos de restricciones:

(a) las restricciones de integridad sobre los valores de las variables, dadas en $(3')$ y $(4')$,

(b) las restricciones $(2')$ de conectividad de la solución, y

(c) las restricciones $(1')$ de paridad de los grados de los vértices.

Relajando, en una u otra forma, cualquiera de estos conjuntos de restricciones, obtendremos un subproblema de (P_R) cuya solución proporcionará una cota inferior al valor óptimo de nuestro problema.

En esta Sección, discutiremos brevemente las dos primeras relajaciones del problema, las referentes a (a) y (b), y centraremos la atención en aquélla que resulta de relajar las restricciones $(1')$ y que es la que ha proporcionado los mejores resultados en nuestro problema.

3.1 LA RELAJACION LINEAL

La forma más usual de obtener una cota inferior en cualquier problema de Programación Lineal Entera consiste en utilizar la relajación lineal.

Esto quizás sea debido a diversos motivos. En primer lugar que hasta hace relativamente poco tiempo no se conocía otro método para obtener dichas cotas. (Recordemos que el Método de Optimización del Subgradiente aunque estudiado en su aspecto teórico por Polyak(1967), Demyanov(1966), Geoffrion(1974),Held,Wolfe y Crowder(1974) y otros, no empieza a aplicarse con éxito a los problemas de Programación Matemática hasta 1971 cuando lo utilizaron Held y Karp en relación con el Problema del Agente Viajero). En segundo lugar, hay que tener en cuenta que la mayor parte de los paquetes comerciales de Programación Matemática (por ejemplo el FMPS de UNIVAC y el MPSX de IBM) sólo contemplan esta posibilidad de cálculo de cotas inferiores en un PLE.

En nuestro problema, si relajamos las restricciones de integridad, es decir, si sustituimos las restricciones (3´) y (4´) en la formulación de (P_R) por:

$$0 \leq x_\ell \leq 1 \qquad \forall \ell \in A_R$$
$$0 \leq y_\ell \leq 2 \qquad \forall \ell \in A_S$$
$$d(i) \leq w_i \leq d_{G_C}(i) \qquad \forall i \in N_R$$

obtendremos un problema lineal, que representaremos por $(\overline{P}_R)$, que puede ser resuelto aplicando el Método del SIMPLEX.

Aplicando éste método en algunos ejemplos, hemos llegado a la conclusión de que el procedimiento no es computacionalmente satisfactorio en nuestro problema. Veamos, por ejemplo, el comportamiento de esta cota inferior en uno de los problemas test más sencillos que se estudiaron (problema 11). Se trata de un problema cuyo grafo simplificado G_C tiene 9 vértices, 7 aristas requeridas, 7 aristas no requeridas y 3 componentes conexas. La cota inferior proporcionada por la solución óptima de $(\overline{P}_R)$ se quedó a un 4.3% del valor óptimo del problema y tardó unos 3 segundos de CPU en ser calculada, utilizando el paquete de Programación Lineal de la casa UNIVAC; lo que es excesivo para el resultado obtenido y para el tamaño del problema. Y más teniendo en cuenta que el problema fué resuelto óptimamente por nuestro algoritmo en 0.139 segundos de CPU (tiempo empleado en calcular una cota superior y una cota inferior que dió el valor de la solución óptima).

La explicación para esta diferencia de comportamientos se basa en el hecho de que tanto el número de variables como el de restricciones es alto en $(\overline{P}_R)$. Mientras que, como veremos, la cota inferior que desarrollamos en nuestro algoritmo se basa en el cálculo de tres problemas diferentes e independientes, dos de asignación de valores según los signos de los coeficientes de las variables y otro que se basa en el cálculo de un SST en un grafo reducido con k vértices.

Los resultados obtenidos, mayor distancia de la cota inferior proporcionada por la relajación lineal del problema (P_R) al óptimo y mayor tiempo de computación, nos han

llevado a abandonar esta relajación e investigar otros métodos para calcular una buena cota inferior al RPP.

3.2 LA RELAJACION BASADA EN EL 1-MATCHING PROBLEM

Consideremos la formulación (P_R) del RPP dada por $(0),(1'),(2'),(3')$ y $(4')$. Representaremos por λ_t los multiplicadores de Lagrange asociados con las restricciones $(2')$ y consideremos la relajación lagrangeana de esas restricciones. El problema resultante será :

$$(P_M(\lambda)) \quad \text{Min} \quad \sum_{\ell \in A_R} c_\ell x_\ell + \sum_{\ell \in A_S} c_\ell y_\ell + \sum_t \lambda_t \Big(1 - \sum_{\ell \in K_t} y_\ell\Big) + \sum_{\ell \in A_R} c_\ell \qquad (0')$$

sujeto a:

$$\sum_{\ell \in A_R} a_{i\ell}(1+x_\ell) + \sum_{\ell \in A_S} a_{i\ell} y_\ell - 2w_i = 0 \quad \forall i \in N_R \qquad (1')$$

$$\begin{aligned} &0 \leq x_\ell \leq 1 \quad \text{enteras} \quad \forall \ell \in A_R \\ &0 \leq y_\ell \leq 2 \quad \text{enteras} \quad \forall \ell \in A_S \end{aligned} \qquad (3')$$

$$d(i) \leq w_i \leq d_{G_C}(i) \quad \text{enteras} \quad \forall i \in N_R \qquad (4')$$

El problema relajado $(P_M(\lambda))$ consiste en, con el mínimo coste, conseguir que todos los vértices del conjunto N_R tengan grado par. Si los costes modificados por los multiplicadores λ_t son no negativos, este problema se puede reducir a un problema equivalente de 1-matching (1-acoplamiento) mínimo definido sobre un nuevo grafo G_M . Grafo cuyo conjunto de vértices será el de los vértices impares, con respecto a G_R, de G_C, y que tendrá una arista entre cada par de vértices con coste igual al del camino más corto entre ellos calculado en G .

A partir de la solución al problema relajado, se pueden identificar las restricciones del tipo $(2')$ que se vio

lan y que pueden, por lo tanto, ser relajadas vía un multiplicador de Lagrange λ_t como se expresa en (0') para obtener la función objetivo con costes modificados $\bar{c}_\ell = c_\ell - \lambda_t \quad \forall \ell \in K_t$. De esta forma, las restricciones (2') se construyen y relajan de manera iterativa.

Si $(P_M(\lambda))$ es el problema definido por (0'), (1'), (3') y (4') para un vector de multiplicadores dado λ, y si $V(P_M(\lambda))$ es el valor asociado a la solución óptima de $(P_M(\lambda))$ nosotros deseamos escoger aquel $\bar{\lambda}$ de forma que:

$$V(P_M(\bar{\lambda})) = \max_{\lambda} \left[V(P_M(\lambda)) \right]$$

La forma usual de resolver el problema de maximización anterior es utilizando el Método del Subgradiente, ó alguna variante del mismo. Computacionalmente, este método es incómodo puesto que requiere la solución de un gran número de problemas $(P_M(\lambda))$ y la identificación en cada uno de ellos de las restricciones del tipo (2') que se violan (hay $2^{k-1} - 1$ restricciones de este tipo). Para evitar algunas de estas dificultades, Balas y Christofides (1981) introducen el concepto de Lagrangiano Restringido, imponiendo a los multiplicadores λ_t la condición de que la solución al problema $(P_M(\lambda))$ sea la misma que para $(P_M(\lambda=0))$; con lo que, y entre otras ventajas, el problema $(P_M(\lambda))$ sólo necesita ser resuelto una vez.

Aunque hemos descrito un procedimiento para calcular las cotas inferiores a (P_R) a partir de la solución de $(P_M(\lambda))$, nos encontramos con que dicho procedimiento tiene

dos inconvenientes importantes:

1) Hemos visto que la solución al problema $(P_M(\lambda))$, dado λ, viene dada por la resolución de un problema de 1-matching sobre cierto grafo G_M . Hay que tener en cuenta, pues, que para la construcción del grafo G_M hay que calcular todos los caminos más cortos entre los vértices de grado impar (respecto de G_R) del grafo G_C, para lo cual existen algoritmos eficientes que requieren un tiempo de computación $O\left[|N_R|^3\right]$ .Además,para resolver el problema de 1-matching de coste mínimo en el grafo G_M hay algoritmos cuyo tiempo de computación es $O\left[|N_R|^4\right]$ (aunque en la práctica el crecimiento pueda ser menor).

   En muchos de los problemas que hemos resuelto, el cardinal de N_R es un número grande, entre 40 y 50 vértices, lo que hace que el tiempo necesario para calcular una cota inferior resolviendo un problema del tipo $(P_M(\lambda))$ sea bastante elevado.

2) Para el valor inicial $\lambda_t=0, \forall t$, la solución al problema relajado será, en general de la forma indicada en la figura 3.a.

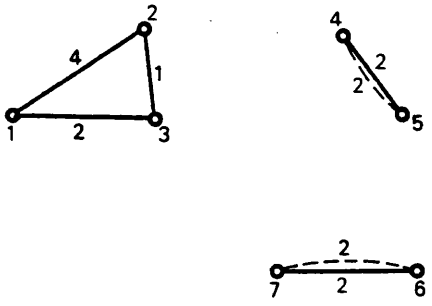


Figura 3.a: Solución a $(P_M(\lambda))$ del grafo de la figura 2.c

Solución que obviamente está muy lejos de la solución óptima del RPP. Para tratar de impedir una solución del tipo descrito en la figura 3.a es para lo que se calculan valores de los multiplicadores λ_t, uno por cada restricción de corte incumplida y se incorporan a la función objetivo. Pero para ello hay que identificar cada vez todos esos cortes incumplidos, sin saber hasta haberlos determinado si se incumplen o no. El número de conjuntos de aristas de corte crece exponencialmente con el número de componentes conexas en el grafo G_R.

Estas dos razones, fundamentalmente, son las que nos han hecho abandonar esta relajación del problema como la que proporcione la cota inferior al RPP.

3.3 LA RELAJACION BASADA EN EL SST

En la relajación anterior, las únicas restricciones consideradas (aparte de las de integridad y cotas sobre los valores de las variables) para calcular la cota inferior eran las del tipo (1'), que exigen que cada vértice $i \in N_R$, sea incidente en cualquier solución óptima al RPP con un número par de aristas (requeridas o no). En otras palabras, que se cumplan las condiciones para la existencia de circuitos eulerianos. Vimos que una de las principales desventajas de dicha relajación era que la solución al problema estaba, en general, formada por varios circuitos eulerianos disconexos entre sí.

Para evitar ésto, vamos a considerar las restricciones (2') que nos garantizan la conexión de todas las componentes generadas por el conjunto de aristas A_R.

Consideremos el problema (P_R) y sea $(P_R(u))$ el problema obtenido a partir de (P_R) relajando lagrangeanamente las restricciones (1'), de paridad sobre los grados de los vértices de N_R, utilizando multiplicadores $u_i, i \in N_R$.

El problema $(P_R(u))$ tiene la siguienta expresión:

$$(P_R(u)) \qquad \text{Min} \left\{ \sum_{\ell \in A_R} (c_\ell - u_{i_\ell} - u_{j_\ell}) x_\ell + \sum_{\ell \in A_S} (c_\ell - u_{i_\ell} - u_{j_\ell}) y_\ell + \right.$$

$$\left. + \sum_{i \in N_R} u_i \left(2w_i - \sum_{\ell \in A_R} a_{i\ell}\right) \right\} + \sum_{\ell \in A_R} c_\ell$$

sujeto a:

$$\sum_{\ell \in K_t} y_\ell \geq 1 \qquad \forall K_t = \left\{ (i,j) \in A_S \mid i \in N(V_t), j \in N(\bar{V}_t), V_t \subset F \right\}$$

$$0 \leq x_\ell \leq 1 \text{ ,enteras} \qquad \forall \ell \in A_R$$
$$0 \leq y_\ell \leq 2 \text{ ,enteras} \qquad \forall \ell \in A_S \qquad (3')$$

$$d(i) \leq w_i \leq d_{G_C}(i) \text{ ,enteras} \qquad \forall i \in N_R \qquad (4')$$

donde i_ℓ y j_ℓ son los vértices incidentes con la arista ℓ.

3.3.1 Resolución del Problema $(P_R(u))$

Para un vector de multiplicadores dado $u=(u_i, i \in N_R)$, veamos cual es el procedimiento para calcular $(P_R(u))$.

Podemos observar que en el problema $(P_R(u))$ hay tres problemas independientes, cada uno de ellos con un conjunto de variables diferentes. En efecto, denotaremos por $(P_R^1(u))$ al problema de :

$$(P_R^1(u)) \qquad \text{Min} \quad \sum_{\ell \in A_R} (c_\ell - u_{i_\ell} - u_{j_\ell})\, x_\ell$$

sujeto a:

$$0 \leq x_\ell \leq 1 \text{ ,enteras} \qquad \forall \ell \in A_R \qquad (3'a)$$

Por $(P_R^2(u))$ representaremos el problema de :

$$(P_R^2(u)) \qquad \text{Min} \quad \sum_{i \in N_R} u_i \,(2w_i - \sum_{\ell \in A_R} a_{i\ell})$$

sujeto a :

$$d(i) \leq w_i \leq d_{G_C}(i) \text{ ,enteras} \qquad \forall i \in N_R \qquad (4')$$

Y por último, $(P_R^3(u))$ será :

$(P_R^3(u))$ Min $\sum_{\ell \in A_S} (c_\ell - u_{i_\ell} - u_{j_\ell})\, y_\ell$

sujeto a:

$$\sum_{\ell \in K_t} y_\ell \geq 1 \qquad \forall K_t = \left\{ (i,j) \in A_S \mid i \in N(V_t), j \in N(\overline{V}_t), V_t \subset F \right\}$$

$$0 \leq y_\ell \leq 2 \quad \text{,enteras} \quad \forall \ell \in A_S \qquad (3'b)$$

La solución óptima al problema $(P_R(u))$ vendrá dada por las soluciones óptimas de cada uno de sus tres subproblemas:

Proposición 3.1

La solución óptima al problema $(P_R^1(u))$ viene dada por la siguiente asignación de valores a las variables

$x_\ell = 1$ si $c_\ell - u_{i_\ell} - u_{j_\ell} \leq 0$

$x_\ell = 0$ en otro caso

Demostración :

Es evidente teniendo en cuenta las cotas de las variables y la condición de integridad expresadas en las restricciones $(3'a)$

Como resultado directo de la Proposición anterior, que luego utilizaremos, podemos establecer el siguiente:

Corolario 3.1

Para cualquier vector u de multiplicadores, se cumple que $V(P_R^1(u)) \leq 0$. Y además,

$V(P_R^1(u)) = 0$ sii $c_\ell - u_{i_\ell} - u_{j_\ell} \geq 0 \qquad \forall \ell \in A_R$

Proposición 3.2

La solución óptima al problema $(P_R^2(u))$ viene dada por la siguiente asignación de valores a las variables w_i:

$$w_i = d_{G_C}(i) \qquad \text{si} \quad u_i \leq 0$$

$$w_i = d(i) = \begin{cases} 1/2\ d_{G_R}(i) & \text{si } i \text{ es par en } G_R \text{ y } u_i > 0 \\ 1/2\left(1+d_{G_R}(i)\right) & \text{si es impar y } u_i > 0 \end{cases}$$

Demostración:

Es evidente teniendo en cuenta las cotas de las variables y las condiciones de integridad expresadas en las restricciones

Corolario 3.2

Para cualquier vector u de multiplicadores que satisfaga $u_i \geq 0, \forall i \in N_R$, se tiene que:

$$0 \leq V(P_R^2(u)) \leq \sum_{i \in N_R} u_i$$

Concretamente :

$$V(P_R^2(u)) = \sum_{i \in N_R^o} u_i$$

donde N_R^o representa el subconjunto de N_R de vértices impares en el grafo G_R ($N_R^o = \{ i \in N_R \mid d_{G_R}(i) \equiv 1 \pmod 2 \}$).

Demostración:

En efecto, si $u_i \geq 0 \ \forall i \in N_R$, para aquellos vértices $i \in N_R$ tales que su multiplicador asociado u_i sea positivo, por la Proposición 3.2, tenemos $w_i = d(i)$, y teniendo en cuenta que

$$\sum_{l \in A_R} a_{il} = d_{G_R}(i)$$

tenemos que:

si $i \in N_R^o$, $w_i = 1/2(1+d_{G_R}(i))$, de donde $2w_i - \sum_{\ell \in A_R} a_{i\ell} = 1$

si $i \in N_R - N_R^o$, $w_i = 1/2\ d_{G_R}(i)$, de donde $2w_i - \sum_{\ell \in A_R} a_{i\ell} = 0$

Por lo tanto,

$$0 \leq \sum_{i \in N_R^o} u_i = V(P_R^2(u)) \leq \sum_{i \in N_R} u_i$$

El problema $(P_R^3(u))$, definido en las variables $y_\ell, \ell \in A_S$, es el problema de encontrar un árbol generador de mínimo peso (SST) en un grafo condensado del grafo G_C :

Definición 3.1

Llamaremos grafo condensado $\tilde{G}_C = (\tilde{N}, \tilde{A})$, a un grafo donde:

1) un vértice $i \in \tilde{N}$ corresponde a una componente C_i de G_R
2) una arista $(i,j) \in \tilde{A}$ existe sii existe alguna arista $(i',j') \in A_S$ tal que $i' \in C_i$, $j' \in C_j$.

El coste de la arista (i,j) en $\tilde{A}$ es :

$$\tilde{c}_{ij} = \min_{\substack{i' \in C_i \\ j' \in C_j}} \{ \bar{c}_{i'j'} \}$$

donde por $\bar{c}_{i'j'}$ representamos los costes modificados por los multiplicadores u_i $(\bar{c}_{i'j'} = c_{i'j'} - u_{i'_\ell} - u_{j'_\ell})$

La figura 3.b nos muestra el grafo condensado correspondiente al grafo G_C de la figura 2.c

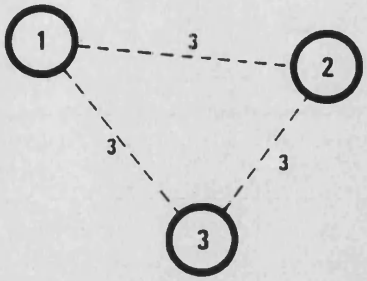


Figura 3.b: Ejemplo de grafo condensado

Veamos cómo la solución de $(P_R^3(u))$ viene dada por la resolución de un árbol generador de mínimo peso en $\tilde{G}_C$:

Teorema 3.3

Los valores óptimos de las variables $y_\ell, \ell \in A_S$ para $(P_R^3(u))$, se obtienen resolviendo el problema de encontrar un SST en el grafo condensado $\tilde{G}_C$, haciendo:

$y_\ell = 1 \quad \forall \ell \in A_S$ que corresponden a aristas en el SST con costes $\tilde{c}_\ell > 0$

$y_\ell = 2 \quad \forall \ell \in A_S \mid \tilde{c}_\ell \leq 0$ independientemente de si corresponden a aristas en el SST o no

$y_\ell = 0$ en otro caso

Demostración:

La demostración se deduce a partir de la definición de árbol generador en $\tilde{G}_C$: es un grafo parcial conexo minimal de $\tilde{G}_C$, donde minimal está utilizado en el sentido de que ningún subconjunto de aristas, contenido en el anterior, formaría un subgrafo parcial conexo de $\tilde{G}_C$. Y el SST es un árbol generador de mínimo peso, es decir, el SST en el grafo $\tilde{G}_C$ re-

presenta la forma de conectar los k vértices de $\tilde{G}_C$ (las componentes de G_C) con el mínimo coste.

Teniendo en cuenta que las variables $y_\ell, \ell \varepsilon A_S$, deben tomar valores enteros y menores o iguales que 2, a aquellas variables asociadas con aristas de coste modificado $\bar{c}_\ell \leq 0$ debemos asignarles el valor máximo que pueden tomar ($y_\ell = 2$) en orden a minimizar la función objetivo. Y puesto que las aristas que aparecen en el SST, con coste positivo, son la forma de conectar $\tilde{G}_C$ con el mínimo coste, haremos $y_\ell = 1$ para aquellas aristas $\ell \varepsilon A_S$ correspondientes a aristas en el SST con costes $\tilde{c}_\ell > 0$.

Corolario 3.3

Para cualquier vector u de multiplicadores tal que $\bar{c}_\ell \geq 0 \ \forall \ell \varepsilon A_S$, se cumple que :

$$V(P_R^3(u)) = V(SST)$$

donde por $V(SST)$ representamos el coste del árbol generador de mínimo peso en el grafo $\tilde{G}_C$.

Demostración:

Es consecuencia directa del Teorema anterior en el caso particular en el que la elección de los multiplicadores u_i sea tal que los costes modificados $\bar{c}_\ell = c_\ell - u_{i_\ell} - u_{j_\ell}$ sean todos no negativos. Así, por definición del grafo condensado $\tilde{G}_C$, todas las aristas de $\tilde{A}$ tendrán costes no negativos y el valor de la solución al problema $(P_R^3(u))$ será el correspondiente a asignar el valor 1 a todas aquellas variables asociadas con las aristas $\ell \varepsilon A_S$ que corresponden a aristas de $\tilde{A}$ que aparecen en la solución del SST.

Comentario 3.1

Las proposiciones 3.1 y 3.2 y el teorema 3.3 proporcionan el procedimiento para calcular la solución óptima a cada uno de los subproblemas en los que hemos dividido $(P_R(u))$. La unión de todas ellas dará la solución óptima del problema global.

Mediante la resolución de $(P_R(u))$ disponemos de una cota inferior para el RPP. Esta cota inferior tiene la ventaja de que es muy sencilla y rápida de calcular, puesto que, dado un vector de multiplicadores y calculados los costes modificados por ellos, se basa en dos problemas de asignación de valores a las variables y en el cálculo de un SST en un grafo con k vértices.

La gran facilidad de cálculo de esta cota inferior lleva asociada una no muy grande proximidad de su valor al de la solución óptima. Esta cota será mejorada por varios métodos descritos en próximos apartados.

3.3.2 Cálculo del vector u de multiplicadores

En el apartado anterior, hemos visto que dado un vector u, cualquiera, de multiplicadores asociados a los vértices de N_R, la resolución del problema $(P_R(u))$ proporciona una cota inferior al valor óptimo del problema (P_R).

En este apartado vamos a discutir el problema de la elección de ese vector de multiplicadores con el objetivo de maximizar el valor de $(P_R(u))$. Es decir, se trata de encontrar el vector u^* que satisfaga:

$$V(P_R(u^*)) = \max_{u} (V(P_R(u)))$$

Resolver este problema resulta en la práctica demasiado costoso y lo que se intenta generalmente es obtener unos buenos multiplicadores lo más próximos posible a los óptimos. El procedimiento que seguimos para intentar mejorar la asignación inicial de valores nulos a los multiplicadores u_i es un procedimiento heurístico que se describe a continuación y que proporciona buenos valores para los multiplicadores:

Algoritmo Heurístico para el cálculo del vector u

Consideremos el grafo $G_C=(N_R, A_R \cup A_S)$, donde $N_R=\{i_1, i_2, \ldots, i_p\}$, $p=|N_R|$; el algoritmo consiste en el siguiente proceso iterativo:

STEP 0 (Inicialización)

Tomar $\delta=1$. Hacer $\bar{c}(i,j)=c(i,j) \quad \forall (i,j)\in A_R \cup A_S$

Ir al Step 1

STEP 1 (Cálculo de multiplicadores)

Si $d_{G_R}(i_\delta)\equiv 0 \pmod 2$. Hacer $u(i_\delta)=0$ e ir al Step 4

Si $d_{G_R}(i_\delta)\equiv 1 \pmod 2$. Ir al Step 2

STEP 2 (Cálculo de multiplicadores)

$$u(i_\delta) = \min_{\substack{j\in N_R \\ j\neq i_\delta}} \{ \bar{c}(i_\delta, j) \}$$

Ir al Step 3

STEP 3 (Actualización de costes)

$$\bar{c}(i_\delta, j) \longleftarrow \bar{c}(i_\delta, j) - u(i_\delta) \qquad \forall j\in N_R,\ j\neq i_\delta$$

Ir al Step 4

STEP 4 (Fin)

Si $\delta=p$. Fin

Si $\delta\neq p$. Hacer $\delta \longleftarrow \delta+1$ .Ir al Step 1 .

El algoritmo descrito se basa en lo siguiente:

a) que los multiplicadores tomen valores no negativos, a fin de que las variables w_i tomen el valor de su cota inferior en la resolución de $(P_R^2(u))$;

b) que cualquier asignación de un valor positivo al multiplicador u_i, correspondiente a un vértice $i\varepsilon N_R$ de grado par respecto a G_R, no produciría ningún incremento en el valor de la función objetivo (en este caso $2w_i-\sum_{\ell\varepsilon A_R} a_{i\ell}=0$), mientras que disminuiría los costes de las aristas incidentes con el vértice, i, pudiendo disminuir el valor de la solución al SST de $(P_R^3(u))$; y

c) que los costes modificados de las aristas, $\bar{c}_\ell=c_\ell-u_{i_\ell}-u_{j_\ell}$, se conserven no negativos para que así el valor $V(P_R^3(u))$ sea no negativo y $V(P_R^1(u))=0$.

Ilustraremos el funcionamiento del algoritmo heurístico descrito anteriormente con el siguiente ejemplo:

Ejemplo 3.1

Consideremos el ejemplo de grafo $G_C=(N_R, A_R \cup A_S)$ dado en la figura 3.c, donde las aristas de A_R están representadas mediante un trazo continuo.

En este ejemplo, la asignación de valores a los multiplicadores $u_i (i=1,\ldots,13)$ proporcionada por la aplicación del algoritmo sería:

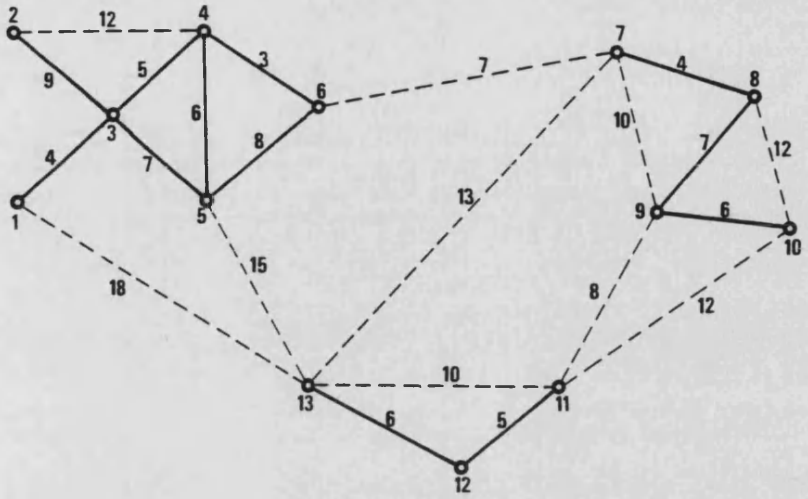


Figura 3.c: Grafo G_C

$u_1=4$ $u_2=9$ $u_3=0$ $u_4=3$ $u_5=3$ $u_6=0$ $u_7=4$

$u_8=0$ $u_9=0$ $u_{10}=6$ $u_{11}=5$ $u_{12}=0$ $u_{13}=5$

El mismo grafo G_C de la figura 3.c con los costes modificados, será :

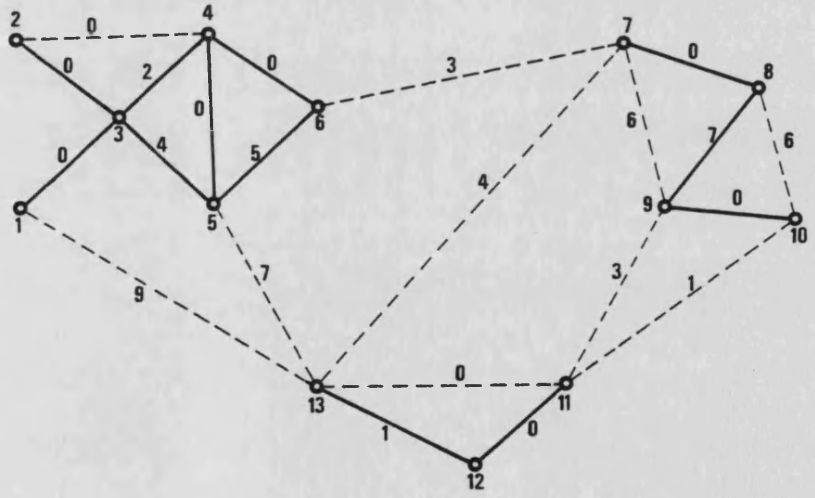


Figura 3.d: Grafo G_C con costes modificados

Este grafo nos permite interpretar a los multiplicadores u_i como la penalización efectuada a los costes de las aristas para intentar la aparición o duplicación de algunas

de ellas y así conseguir la paridad de los grados de los vértices, tal como se expresa en la figura siguiente:

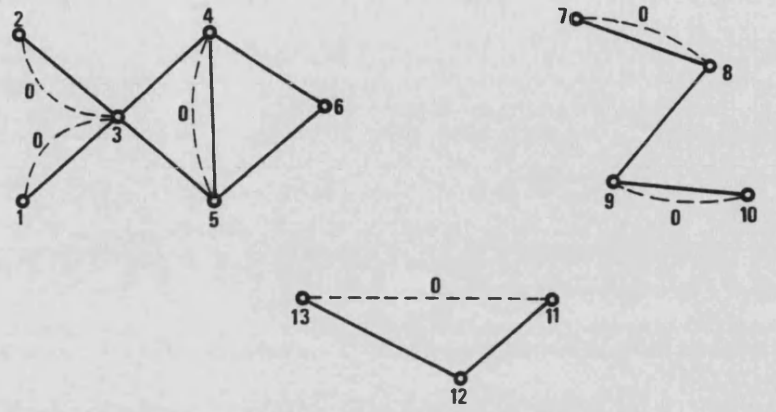


Figura 3.e: Finalidad de los multiplicadores u_i

Comentario 3.2

Una vez descrito el algoritmo para el cálculo de los multiplicadores $u_i, i \in N_R$, hemos visto las ideas generales a las que se ajusta su funcionamiento y un ejemplo que nos ha permitido ilustrar su cálculo e interpretación.

Vamos a ver a continuación tres teoremas que caracterizan a un subconjunto de multiplicadores óptimos del problema de $\underset{u}{\text{Max}}\ V(P_R(u))$ y veremos su relación con el resultado obtenido por nuestro algoritmo.

Teorema 3.4

Existe al menos un vector de multiplicadores óptimo, u, que satisface $u_i \geq 0 \quad \forall i \in N_R$.

Demostración:

Vamos a demostrarlo por reducción al absurdo : supondremos que cualquier vector óptimo de multiplicadores tiene componentes negativas.

Sea $\bar{u}$ un vector de multiplicadores, óptimo del problema de $\underset{u}{\text{Max}}\ V(P_R(u))$. Sabemos que $\exists i \in N_R : \bar{u}_i < 0$

En general, el vértice $i \in N_R$ será incidente con:

a) $\ell_1, \ldots, \ell_p \in A_R$ $\begin{cases} \ell_1, \ldots, \ell_q & \text{tienen } \bar{c}_{\ell_k} = c_{\ell_k} - \bar{u}_{j_{\ell_k}} - \bar{u}_i < 0 \\ \ell_{q+1}, \ldots, \ell_p & \text{" } \quad \bar{c}_{\ell_k} \geq 0 \quad \forall k \in \{q+1, \ldots, p\} \end{cases}$

b) $\ell_{p+1}, \ldots, \ell_s \in A_S$ $\begin{cases} \ell_{p+1}, \ldots, \ell_r & \text{tienen } \bar{c}_{\ell_k} < 0 \quad \forall k \in \{p+1, \ldots, r\} \\ \ell_{r+1}, \ldots, \ell_s & \text{" } \quad \bar{c}_{\ell_k} \geq 0 \quad \forall k \in \{r+1, \ldots, s\} \end{cases}$

Definimos un nuevo vector ,u^* , de multiplicadores en la forma siguiente:

$$\begin{cases} u^*_i = 0 \\ u^*_j = \bar{u}_j \quad \forall j \in N_R, j \neq i \end{cases}$$

y demostraremos que $V(P_R(u^*)) \geq V(P_R(\bar{u}))$

Veamos la relación existente entre $\bar{c}_\ell$ y c^*_ℓ, los costes modificados por $\bar{u}$ y u^*, $\forall \ell \in A_R \cup A_S$:

1) $\forall \ell \in A_R \cup A_S \mid \ell$ no es incidente con el vértice i :

$$\bar{c}_\ell = c_\ell - \bar{u}_{j_\ell} - \bar{u}_{p_\ell} = c_\ell - u^*_{j_\ell} - u^*_{p_\ell} = c^*_\ell$$

2) $\forall \ell \in A_R \cup A_S \mid \ell$ es incidente con el vértice i :

a) Si $\bar{c}_\ell < 0 \rightarrow c_\ell - \bar{u}_{j_\ell} - \bar{u}_i < 0 \rightarrow c_\ell - \bar{u}_{j_\ell} = c^*_\ell < \bar{u}_i < 0 \rightarrow c^*_\ell < 0$

b) Si $\bar{c}_\ell > 0$, no tenemos información sobre el signo de c^*_ℓ.

Vamos a hacer un análisis del valor de $V(P_R(u^*))$ en el peor de los casos. Esto ocurrirá cuando todas las aristas incidentes con i que tenían $\bar{c}_\ell \geq 0$, tienen nuevos costes modificados $c^*_\ell \leq 0$. Es el peor de los casos puesto que por la Proposición 3.1 : $x_\ell = 1$ si $c_\ell \leq 0$ y $x_\ell = 0$ en otro caso ; por el Teorema 3.3 ; $y_\ell = 2$ si $c_\ell \leq 0$ e $y_\ell = 1$ ó 0 según estén en el SST sobre el grafo $\tilde{G}_C$ o no.

Entonces, resumiendo:

$$\bar{c}_\ell = c^*_\ell \qquad \forall \ell \in A_R \cup A_S \mid a_{i\ell} = 0$$

$$\bar{c}_\ell < 0 \;\rightarrow\; c^*_\ell < 0 \quad \forall \ell \in \{\ell_1, \ldots, \ell_q\} \cup \{\ell_{p+1}, \ldots, \ell_r\}$$

$$\bar{c}_\ell \geq 0 \;\rightarrow\; \text{supondremos (en el peor de los casos)}$$

$$c^*_\ell \leq 0 \quad \forall \ell \in \{\ell_{q+1}, \ldots, \ell_p\} \cup \{\ell_{r+1}, \ldots, \ell_s\}$$

Puesto que el único cambio que hemos realizado ha sido en el valor de un multiplicador (u^*_i sustituye a $\bar{u}_i$), el cambio en la función objetivo de los problemas $(P_R(\bar{u}))$ y $(P_R(u^*))$ se centrará en las asignaciones de valores a las variables asociadas con las aristas incidentes con el vértice i (únicas aristas cuyo coste cambia); mientras que una parte de la función objetivo, que representaremos por K, permanecerá constante en el valor de la solución de ambos problemas. Representaremos por $V_i(P_R(u))$ el valor de los términos de la función objetivo que dependen del multiplicador u_i (al resolver $P^2_R(u)$) y de los costes modificados de las aristas requeridas incidentes con i (al resolver $(P^1_R(u))$). Es decir, todos los términos en $V(P^1_R(u))$ y $V(P^2_R(u))$ que cambian al sustituir $\bar{u}_i$ por u^*_i. Así:

$$(1) \qquad \begin{aligned} V(P_R(\bar{u})) &= V_i(P_R(\bar{u})) + K + V(P^3_R(\bar{u})) \\ V(P_R(u^*)) &= V_i(P_R(u^*)) + K + V(P^3_R(u^*)) \end{aligned}$$

en donde:

$$V_i(P_R(\bar{u})) = \sum_{k=1}^{q} \bar{c}_{\ell_k} + \bar{u}_i(2w_i - d_{G_R}(i)) = \sum_{k=1}^{q} \bar{c}_{\ell_k} + \bar{u}_i(2s - p)$$

$$(2) \qquad (\text{si } \bar{u}_i < 0 \;\rightarrow\; w_i = d_{G_C}(i) = s) \text{ , y}$$

$$V_i(P_R(u^*)) = \sum_{k=1}^{q} c^*_{\ell_k} + \sum_{k=q+1}^{p} c^*_{\ell_k} + u^*_i(2w_i - d_{G_R}(i)) =$$

$$= \sum_{k=1}^{q} c^*_{\ell_k} + \sum_{k=q+1}^{p} c^*_{\ell_k} \qquad (u^*_i = 0)$$

Estudiemos ahora la relación existente entre $V(P_R^3(\bar{u}))$ y $V(P_R^3(u^*))$:

Representaremos por K_1 el valor en $V(P_R^3(u))$ proporcionado por la asignación $y_\ell = 2 \ \forall \ell \in A_S \mid a_{i\ell} = 0$ y $\bar{c}_\ell < 0$. K_1 será constante en el valor óptimo de ambos problemas ($\bar{c}_\ell = c_\ell^* \ \forall \ell \in A_R \cup A_S \mid a_{i\ell} = 0$). Puesto que todas las aristas no requeridas, incidentes con i, que tenían $\bar{c}_\ell < 0$ tienen como se ha visto $c_\ell^* < 0$, la asignación $y_\ell = 2$ para éstas aristas será otro término, de distinto valor, para $V(P_R^3(\bar{u}))$ y $V(P_R^3(u^*))$.

Construimos ahora los grafos condensados, $\tilde{G}_C(\bar{u})$ y $\tilde{G}_C(u^*)$, a partir de los grafos G_C con costes modificados por los multiplicadores $\bar{u}$ y u^*, respectivamente. Vamos a construir los grafos $\tilde{\tilde{G}}_C(\bar{u})$ y $\tilde{\tilde{G}}_C(u^*)$ comprimiendo en un solo vértice todos aquellos vértices de $\tilde{N}$ que están conectados mediante aristas de coste modificado $\bar{c}_\ell < 0$ (y por lo tanto $c_\ell^* < 0$). Los grafos $\tilde{\tilde{G}}_C(\bar{u})$ y $\tilde{\tilde{G}}_C(u^*)$ tienen exactamente la misma estructura salvo en lo que afecta a los costes de las aristas incidentes con el vértice que corresponde a la nueva componente que contiene al vértice i; tengamos en cuenta que en $\tilde{\tilde{G}}_C(\bar{u})$ no hay aristas de coste negativo, mientras que en $\tilde{\tilde{G}}_C(u^*)$ sí las hay (aquellas aristas tales que $\bar{c}_\ell \geq 0$ y $c_\ell^* \leq 0$).

Si representamos por $V(\overline{SST})$ el coste del SST solución al grafo $\tilde{\tilde{G}}_C(\bar{u})$ y por $V(SST^*)$ lo mismo en $\tilde{\tilde{G}}_C(u^*)$, tendremos:

$$
\begin{aligned}
V(P_R^3(\bar{u})) &= K_1 + 2 \sum_{k=p+1}^{n} \bar{c}_{\ell_k} + V(\overline{SST}) \\
V(P_R^3(u^*)) &= K_1 + 2 \sum_{k=p+1}^{n} c_{\ell_k}^* + V(SST^*)
\end{aligned}
\qquad (3)
$$

Obviamente : $V(SST^*) = 2\sum_{k=r+1}^{s} c^*_{\ell_k} + K_2$ (4)

donde K_2 es el coste,no negativo,óptimo de conectar el grafo $\approx$
$G_C(u^*)$ una vez fijadas las aristas ℓ_k, $k\in\{r+1,..,s\}$ $(c^*_{\ell_k}\leq 0)$
Y tenemos que : $V(\overline{SST}) \leq \sum_{k=r+1}^{s} \bar{c}_{\ell_k} + K_2$ $(\bar{c}_{\ell_k}\geq 0 \quad k\in\{r+1,..,s\})$ (5

Entonces, sustituyendo (2),(3) y (4) en la expresión (1), tenemos:

$$V(P_R(u^*)) = \sum_{k=1}^{q} c^*_{\ell_k} + \sum_{k=q+1}^{p} c^*_{\ell_k} + K + K_1 + 2\sum_{k=p+1}^{r} c^*_{\ell_k} + 2\sum_{k=r+1}^{s} c^*_{\ell_k} + K_2$$

$$V(P_R(\bar{u})) \leq \sum_{k=1}^{q} \bar{c}_{\ell_k} + \bar{u}_i(2s-p) + K + K_1 + 2\sum_{k=p+1}^{r} \bar{c}_{\ell_k} + \sum_{k=r+1}^{s} \bar{c}_{\ell_k} + K_2$$

de donde :

$$V(P_R(u^*)) - V(P_R(\bar{u})) \geq \sum_{k=1}^{q} (c^*_{\ell_k} - \bar{c}_{\ell_k}) + \sum_{k=q+1}^{p} c^*_{\ell_k} - \bar{u}_i(2s-p) + \sum_{k=p+1}^{r} (c^*_{\ell_k} - \bar{c}_{\ell_k}) + \sum_{k=r+1}^{s} c^*_{\ell_k} + \sum_{k=r+1}^{s} (c^*_{\ell_k} - \bar{c}_{\ell_k}) =$$

(puesto que $c^*_{\ell_k} = c_{\ell_k} - \bar{u}_{j\ell_k} \rightarrow c^*_{\ell_k} - \bar{c}_{\ell_k} = \bar{u}_i$)

$$= q\bar{u}_i + \sum_{k=q+1}^{p} c^*_{\ell_k} - \bar{u}_i(2s-p) + 2(r-p)\bar{u}_i + \sum_{k=r+1}^{s} c^*_{\ell_k} + (s-r)\bar{u}_i = (-s-p+r+q)\bar{u}_i + \sum_{k=q+1}^{p} c^*_{\ell_k} + \sum_{k=r+1}^{s} c^*_{\ell_k}$$

y puesto que $\bar{c}_{\ell_k} = c^*_{\ell_k} - \bar{u}_i \geq 0 \qquad k \in \{q+1,..p\} \cup \{r+1,...,s\}$,

$$c^*_{\ell_k} \geq \bar{u}_i$$

Luego:

$$V(P_R(u^*)) - V(P_R(\bar{u})) \geq$$

$$\geq (-s-p+r+q)\bar{u}_i + (p-q)\bar{u}_i + (s-r)\bar{u}_i = 0.$$

Hemos demostrado pues que el valor asociado con multiplicador u^* es por lo menos igual al asociado con el multiplicador $\bar{u}$. Como este razonamiento puede aplicarse para todos aquellos vértices $i \in N_R$ cuyo multiplicador asociado $\bar{u}_i$ sea negativo, hemos demostrado el Teorema.

Podemos decir también que el conjunto :

$$\bar{U}_1 = \left\{ \bar{u} \mid \bar{u} \geq 0 \text{ y } V(P_R(\bar{u})) = \max_u (V(P_R(u))) \right\} \neq \phi \quad \text{si}$$

$$\bar{U}_0 = \left\{ \bar{u} \mid V(P_R(\bar{u})) = \max_u (V(P_R(u))) \right\} \neq \phi$$

Teorema 3.5

En el subconjunto $\bar{U}_1$ de vectores de multiplicadores de Lagrange existe al menos un vector u^* que satisface :

$$u^*_i \leq \min_{j \neq i} \{ c_{ij} \} \quad \forall i$$

Demostración:

Vamos a demostrarlo por reducción al absurdo: supondremos que cualquier vector óptimo de multiplicadores $\bar{u} \in \bar{U}_1$ tiene al menos una componente $\bar{u}_i > \min_{i \neq j} \{c_{ij}\}$

Seguiremos la misma notación que en el Teorema 3.5 y utilizaremos un razonamiento similar al allí seguido.

El vértice $i \in N_R$, será, en general, incidente con:

a) $\ell_1,\ldots,\ell_p \in A_R$ $\begin{cases} \ell_1,..,\ell_q & \text{tienen } \bar{c}_{\ell_k} = c_{\ell_k} - \bar{u}_{j_{\ell_k}} - \bar{u}_i < 0 \\ \ell_{q+1},..,\ell_p & " \quad \bar{c}_{\ell_k} \geq 0 \quad \forall k \in \{q+1,\ldots,p\} \end{cases}$

b) $\ell_{p+1},..,\ell_s \in A_S$ $\begin{cases} \ell_{p+1},..,\ell_r & \text{tienen } \bar{c}_{\ell_k} < 0 \quad \forall k \in \{p+1,\ldots,r\} \\ \ell_{r+1},..,\ell_s & " \quad \bar{c}_{\ell_k} \geq 0 \quad \forall k \in \{r+1,\ldots,s\} \end{cases}$

Definimos un nuevo vector de multiplicadores u^*, en la forma siguiente:

$$\begin{cases} u_i^* = 0 \\ u_j^* = \bar{u}_j \quad \forall j \in N_R,\ j \neq i \end{cases}$$

Obviamente, $u^* \geq 0$ y $u_i^* \leq \min_{j \neq i}\{c_{ij}\}$, ya que $c_{ij} \geq 0$

Demostraremos que $V(P_R(u^*)) \geq V(P_R(\bar{u}))$

Veamos la relación existente entre $\bar{c}_\ell$ y c_ℓ^* $\forall \ell$:

1) $\forall \ell \in A_R \cup A_S \mid a_{i\ell} = 0$: $\bar{c}_\ell = c_\ell^*$

2) $\forall \ell \in A_R \cup A_S \mid a_{i\ell} = 1$:

   a) $\bar{c}_\ell \geq 0 \rightarrow c_\ell - \bar{u}_{j_\ell} \geq \bar{u}_i \geq 0 \rightarrow c_\ell^* \geq 0$

   b) $\bar{c}_\ell < 0$, no tenemos información sobre el signo de c_ℓ^*

De la misma forma que en el Teorema 3.4, vamos a hacer un análisis del valor de $V(P_R(u^*))$ en el peor de los casos. Esto sucederá cuando todas las aristas incidentes con i que tenían coste $\bar{c}_\ell < 0$, siguen teniendo ahora $c_\ell^* < 0$. (Recordemos que $c_\ell^* \leq \bar{c}_\ell$ $\forall \ell \mid a_{i\ell} = 1$).

Resumiendo:

$\bar{c}_\ell = c_\ell^*$ $\quad \forall \ell \in A_R \cup A_S \mid a_{i\ell} = 0$

$\bar{c}_\ell > 0 \rightarrow c_\ell^* \geq 0$ $\quad \forall \ell \in \{\ell_{q+1},..,\ell_p\} \cup \{\ell_{r+1},..,\ell_s\}$

$\bar{c}_\ell < 0 \rightarrow$ supondremos (en el peor de los casos)

$c_\ell^* < 0$ $\quad \forall \ell \in \{\ell_1,..,\ell_q\} \cup \{\ell_{p+1},..,\ell_r\}$

Siguiendo la notación y los comentarios del Teo-

rema 3.4, llegamos a :

$$V(P_R(\bar{u})) = V_i(P_R(\bar{u}))+K+V(P_R^3(\bar{u})) =$$

$$(1) \qquad = V_i(P_R(\bar{u}))+K+2\sum_{k=p+1}^{r}\bar{c}_{\ell_k}+K_1+V(\overline{SST})$$

$$V(P_R(u^*)) = V_i(P_R(u^*))+K+V(P_R^3(u^*)) =$$

$$= V_i(P_R(u^*))+K+2\sum_{k=p+1}^{r}c^*_{\ell_k}+K_1+V(SST^*)$$

Y puesto que: $\bar{c}_\ell = c^*_\ell \quad \forall \ell \mid a_{i\ell}=0$; $\bar{c}_\ell < c^*_\ell \quad \forall \ell \mid a_{i\ell}=1$ $\rightarrow$ $\bar{c}_\ell \leqslant c^*_\ell \quad \forall \ell \in A_R \cup A_S$

tenemos que: $V(\overline{SST}) \leqslant V(SST^*)$. Con lo que a partir de (1)

$$V(P_R(u^*)) - V(P_R(\bar{u})) \geqslant V_i(P_R(u^*)) - V_i(P_R(\bar{u})) +$$

$$(2) \qquad +2\sum_{k=p+1}^{r}c^*_{\ell_k} - 2\sum_{k=p+1}^{r}\bar{c}_{\ell_k}$$

En nuestro caso: $V_i(P_R(u^*)) = \sum_{k=1}^{q} c^*_{\ell_k} + u^*_i(2w_i - d_{G_R}(i)) = \sum_{k=1}^{q} c^*_{\ell_k}$

y

$$(3) \qquad V_i(P_R(\bar{u})) = \sum_{k=1}^{q}\bar{c}_{\ell_k} + \bar{u}_i(2w_i - d_{G_R}(i))$$

y además, puesto que $c^*_{\ell_k} - \bar{c}_{\ell_k} = \bar{u}_i$,tenemos a partir de (2) y (3)

$$V(P_R(u^*)) - V(P_R(\bar{u})) \geqslant q\bar{u}_i + 2(r-p)\bar{u}_i - \bar{u}_i(2w_i - d_{G_R}(i)) =$$

$$= (2r-2p+q)\bar{u}_i - \bar{u}_i(2w_i - d_{G_R}(i)) \qquad (4)$$

Sabemos que al ser $\bar{u}_i > \min_j\{c_{ij}\}$, existirá al menos una arista $(i,k) \mid \bar{c}_{ik} < 0$. Luego $(r-p)+q \geqslant 1$ (5)

Vamos a distinguir dos casos:

a) El vértice i es par respecto al grafo G_R $\rightarrow$ $2w_i - d_{G_R}(i) = 0$

Por (4) y (5):

$$V(P_R(u^*))-V(P_R(\bar{u})) \geq (2r-2p+q)\bar{u}_i = (r-p)\bar{u}_i+(r-p-q)\bar{u}_i \geq$$

$$\geq (r-p+q)\bar{u}_i > 0$$

(ya que $(r-p+q)\geq 1$ y $\bar{u}_i>0$)

b) El vértice i es impar respecto al grafo $G_R \rightarrow 2w_i-d_{G_R}(i)=1$

Por (4) y (5):

$$V(P_R(u^*))-V(P_R(\bar{u})) \geq (2r-2p+q-1)\bar{u}_i = (r-p+q-1)\bar{u}_i +$$

$$+ (r-p)\bar{u}_i \geq 0$$

Puesto que el razonamiento que hemos seguido puede extenderse a todas aquellas componentes del vector $\bar{u}$ que no satisfagan $\bar{u}_i \leq \min_j\{c_{ij}\}$, tendremos un nuevo vector de multiplicadores $u^* \varepsilon \bar{U}_1$ que cumple la tesis; lo que es absurdo, y el teorema queda demostrado.

Diremos entonces que el conjunto de multiplicadores:

$$\bar{U}_2 = \left\{\bar{u} \mid \bar{u}\geq 0,\ \bar{u}_i \leq \min_j\{c_{ij}\} \text{ y } V(P_R(\bar{u})) = \max_u V(P_R(u))\right\} \neq \phi$$

si $\bar{U}_0 \neq \phi$. (Si $\bar{U}_0 \neq \phi \rightarrow \bar{U}_1 \neq \phi \rightarrow \bar{U}_2 \neq \phi$).

Teorema 3.6

En el subconjunto $\bar{U}_2$ de vectores de multiplicadores óptimos, existe al menos un vector u^* que satisface que los costes modificados, $c_{ij}-u_i^*-u_j^*$, son no negativos.

Demostración:

Lo demostraremos por reducción al absurdo. Supondremos que no existe una solución óptima que cumpla que los costes reducidos de las aristas sean no negativos.
Dada, pues, cualquier solución $\bar{u} \varepsilon U_2$, existirá, al menos, una

arista $\ell_0=(i,j)\in A_R \cup A_S$ tal que $\bar{c}_{\ell_0}=c_{\ell_0}-\bar{u}_{i\ell_0}-\bar{u}_{j\ell_0}<0$.

A partir de $\bar{u}$, construiremos otro vector u^*, tal que satisfaga $0\leq u^*_i \leq \min_{\forall j\neq i}\{c_{ij}\}$ y que dejará los costes modificados no negativos, de forma que $V(P_R(u^*))\geq V(P_R(\bar{u}))$, con lo que llegaremos a una contradicción y habremos demostrado el Teorema.

Puesto que $\bar{c}_{\ell_0}=c_{\ell_0}-\bar{u}_{i\ell_0}-\bar{u}_{j\ell_0}<0$, tenemos que : $\bar{u}_{i\ell_0}>c_{\ell_0}-\bar{u}_{j\ell_0}\geq 0$ (ya que $\bar{u}_i\leq\min\{c_{ij}\}$, $\forall i\in N_R$); donde para facilitar la notación y puesto que nos centraremos en el vértice i, representaremos siempre a i_{ℓ_0} como i, por lo que:

(1) $$\bar{u}_i > c_{\ell_0}-\bar{u}_{j\ell_0}\geq 0$$

El vértice $i\in N_R$, será, en general, incidente con las siguientes aristas:

a) la arista $\ell_0\in A_R\cup A_S$ que satisface $\bar{c}_{\ell_0}<0$

b) las aristas $\ell_1,.....,\ell_q \in A_R$, de las cuales

i) $\ell_1,.....,\ell_p$ tienen coste modificado negativo

ii) $\ell_{p+1},...,\ell_q$ " " " no negativo

c) las aristas $\ell_{q+1},...,\ell_s \in A_S$, de las cuales

i) $\ell_{q+1},...,\ell_r$ tienen coste modificado negativo

ii) $\ell_{r+1},...,\ell_s$ " " " no negativo

A partir del vector $\bar{u}$ de multiplicadores, construimos el vector u^*, en la forma siguiente:

(2) $$\begin{cases} u^*_j = u_j \quad \forall j\neq i \\ u^*_i = \min\left\{ c_{\ell_0}-\bar{u}_{j\ell_0}\,;\, c_\ell-\bar{u}_{j\ell},\ \ell\in\{\ell_1,..,\ell_p\}\,;\, c_\ell-\bar{u}_{j\ell},\ \ell\in\{\ell_{q+1},..,\ell_r\}\right\}\end{cases}$$

donde por j_ℓ representamos el otro vértice(distinto de i)incidente con la arista ℓ.

1) Vamos a demostrar que u^* cumple $0 \leq u_j^* \leq \min_{k \neq j}\{c_{jk}\}$ $\forall j \in N_R$.

En efecto, puesto que $u_j^* = \bar{u}_j$ $\forall j \neq i$ y $\bar{u} \in \bar{U}_2$,

$$0 \leq u_j^* \leq \min_k \{c_{jk}\} \quad \forall j \neq i \ ;$$

veamos que también se cumple para u_i^* :

por hipótesis $\bar{u}_{j_\ell} \leq c_\ell$ $\forall \ell$ y por (2), tenemos $u_i^* \geq 0$;

por (2): $u_i^* \leq c_{\ell_0} - \bar{u}_{j_{\ell_0}} \rightarrow u_i^* < \bar{u}_i$ (por (1)) y por lo tanto,

$$0 \leq u_i^* < \min_j \{c_{ij}\}$$

2) Vamos a demostrar que los nuevos costes modificados, c_ℓ^* , de todas las aristas incidentes con i son ahora no negativos.

En efecto, puesto que $u_j^* = \bar{u}_j$ $\forall j \neq i$ y $u_i^* < \bar{u}_i$,
para todas las aristas incidentes con i cuyo coste modificado $\bar{c}_\ell$ era no negativo, tendremos:

$c_\ell^* = c_\ell - u_i^* - \bar{u}_{j_\ell} > c_\ell - \bar{u}_i - \bar{u}_{j_\ell} = \bar{c}_\ell \geq 0$ $\forall \ell \in \{\ell_{p+1}, \ldots, \ell_q, \ell_{r+1}, \ldots, \ell_s\}$

y para aquéllas que tenían coste $\bar{c}_\ell < 0$, por la definición de u^*, dada en (2), tendremos:

$c_\ell^* = c_\ell - \bar{u}_{j_\ell} - u_i^* \geq 0$ $\forall \ell \in \{\ell_0, \ell_1, \ldots, \ell_p, \ell_{q+1}, \ldots, \ell_r\}$

Luego con la definición dada en (2) para el multiplicador del vértice i, uno de los incidentes con aristas de coste modificado negativo, pasamos a tener las aristas incidentes con i con nuevo coste reducido no negativo.

Este procedimiento puede extenderse a todas las aristas del grafo con coste reducido negativo, sin más que aplicar la definición (2) para calcular un nuevo multiplicador en uno de los vértices incidentes con alguna de dichas aristas.

3) Demostraremos, por último, que $V(P_R(u^*)) \geq V(P_R(\bar{u}))$

Para ello distinguiríamos dos casos: si la arista $\ell_0 \in A_R$ ó $\ell_0 \in A_S$. Nosotros lo demostraremos únicamente en el caso en que $\ell_0 \in A_R$, ya que en el otro caso la demostración sería idéntica sin más que multiplicar por 2 el término $\bar{c}_{\ell_0}$ que aparecerá en la expresión de $V(P_R(\bar{u}))$.

Puesto que el único cambio que hemos hecho ha sido en el valor de un multiplicador ($\bar{u}_i$ por u_i^*), el cambio en el valor de la función objetivo de ambos problemas se centrará en las asignaciones de valores referidas a las variables asociadas con las aristas incidentes con el vértice i , cuyos costes cambiarán al cambiar el multiplicador; mientras que una parte de la función objetivo, que representaremos por K, permanecerá constante en la solución de ambos problemas.

Representaremos por $V_i(P_R(u))$ el valor de los términos de la función objetivo que dependen del valor de u_i y de los costes modificados de las aristas requeridas incidentes con i. Es decir, todos aquellos términos de $V(P_R^1(u))$ y de $V(P_R^2(u))$ que cambian en $V(P_R(u))$ al sustituir $\bar{u}_i$ por u_i^*.

Así:
$$\begin{aligned} V(P_R(\bar{u})) &= V_i(P_R(\bar{u})) + K + V(P_R^3(\bar{u})) \\ V(P_R(u^*)) &= V_i(P_{R_p}(u^*)) + K + V(P_R^3(u^*)) \end{aligned} \qquad (3)$$

en donde:
$$V_i(P_R(\bar{u})) = \bar{c}_{\ell_0} + \sum_{k=1} \bar{c}_{\ell_k} + \bar{u}_i\left(2w_i - \sum_{\ell \in A_R} a_{i\ell}\right)$$

ya que $\bar{c}_\ell < 0 \quad \forall \ell \in \{\ell_0, \ell_1, \ldots, \ell_p\}$, mientras que:

$$V_i(P_R(u^*)) = 0 + \quad 0 \quad + u_i^*\left(2w_i - \sum_{\ell \in A_R} a_{i\ell}\right)$$

ya que todas las aristas incidentes con i tienen los costes modificados $c^*_\ell \geq 0$,

y donde:

$$V(P^3_R(\bar{u})) = 2\sum_{k=q+1}^{r} \bar{c}_{\ell_k} + K_1 + V(\overline{SST})$$

y

$$V(P^3_R(u^*)) = \quad 0 \quad + K_1 + V(SST^*)$$

donde K_1 representa el valor en $V(P^3_R(u))$ proporcionado por la asignación de valores $y_\ell = 2 \ \forall \ell \in A_S \mid a_{i\ell} = 0$ y $\bar{c}_\ell < 0$. Este término K_1, será constante en el valor óptimo de ambos problemas ($\bar{c}_\ell = c^*_\ell \ \forall \ell \mid a_{i\ell} = 0$). Y $V(\overline{SST})$ y $V(SST^*)$ tienen el mismo significado que en la demostración del Teorema 3.4

Para comprobar que $V(P_R(\bar{u})) \leq V(P_R(u^*))$,

a) Veamos que $V(SST^*) \geq V(\overline{SST})$

En efecto, al cambiar $\bar{u}_i$ por u^*_i, tenemos que por ser $\bar{u}_i > u^*_i \geq 0$ y $\bar{u}_j = u^*_j \ \forall j \neq i$:

$\bar{c}_\ell = c_\ell - \bar{u}_{k_\ell} - \bar{u}_{j_\ell} = c_\ell - u^*_{k_\ell} - u^*_{j_\ell} = c^*_\ell \ \forall \ell \in A_R \cup A_S$ no incidente con i

$\bar{c}_\ell = c_\ell - \bar{u}_{j_\ell} - \bar{u}_i < c_\ell - u^*_{j_\ell} - u^*_i = c^*_\ell \ \forall \ell \in A_R \cup A_S$ incidente con i

Luego $\bar{c}_\ell \leq c^*_\ell \ \forall \ell \in A_R \cup A_S$, por lo que el árbol generador de mínimo peso calculado en el grafo con costes correspondientes a c^*_ℓ, tendrá un coste como mínimo igual al del SST calculado con los costes modificados $\bar{c}_\ell$.

b) Veamos que $V(P_R(\bar{u})) \leq V(P_R(u^*))$

Tanto $\bar{u}_i$ como u^*_i son no negativos, por lo tanto w_i tomará el valor de su cota inferior $d(i)$.

Distinguiremos dos casos según el grado del vértice i respecto a G_R.

i) El vértice i es par respecto a G_R:

entonces $2w_i - \sum_{\ell \in A_R} a_{i\ell} = 2(1/2 d_{G_R}(i)) - d_{G_R}(i) = 0$, con lo que

$$V(P_R(\bar{u})) - V(P_R(u^*)) \leq \bar{c}_{\ell_0} + \sum_{k=1}^{p} \bar{c}_{\ell_k} + 2 \sum_{k=q+1}^{r} \bar{c}_{\ell_k} \leq \bar{c}_{\ell_0} < 0$$

ii) El vértice i es impar respecto a G_R:

entonces $2w_i - \sum_{\ell \in A_R} a_{i\ell} = 2(1/2(1+d_{G_R}(i))) - d_{G_R}(i) = 1$, y

$$V_i(P_R(\bar{u})) = \bar{c}_{\ell_0} + \sum_{k=1}^{p} \bar{c}_{\ell_k} + \bar{u}_i \quad , y$$

$$V_i(P_R(u^*)) = u_i^*$$

Distinguiremos tres subcasos, según el valor de u_i^* se haya alcanzado para cada una de las tres expresiones en el mínimo de (2):

α) $u_i^* = c_{\ell_0} - \bar{u}_{j_{\ell_0}}$

$$V(P_R(\bar{u})) - V(P_R(u^*)) \leq 2 \sum_{k=q+1}^{r} \bar{c}_{\ell_k} + \bar{c}_{\ell_0} + \sum_{k=1}^{p} \bar{c}_{\ell_k} + \bar{u}_i - u_i^* \leq$$

$$\leq \bar{c}_{\ell_0} + \bar{u}_i - u_i^* = c_{\ell_0} - \bar{u}_i - \bar{u}_{j_{\ell_0}} + \bar{u}_i - u_i^* = 0$$

β) $u_i^* = c_\ell - \bar{u}_{j_\ell}$ para algún $\ell \in \{\ell_1, \ldots, \ell_p\} \subset A_R$

$(\bar{c}_\ell = c_\ell - \bar{u}_i - \bar{u}_{j_\ell} < 0)$

$$V(P_R(\bar{u})) - V(P_R(u^*)) \leq 2 \sum_{k=q+1}^{r} \bar{c}_{\ell_k} + \bar{c}_{\ell_0} + \sum_{k=1}^{p} \bar{c}_{\ell_k} + \bar{u}_i - u_i^* \leq$$

$$\leq \bar{c}_{\ell_0} + \bar{c}_\ell + \bar{u}_i - u_i^* = \bar{c}_{\ell_0} + c_\ell - \bar{u}_{j_\ell} - \bar{u}_i + \bar{u}_i - u_i^* = \bar{c}_{\ell_0} < 0$$

γ) $u_i^* = c_\ell - \bar{u}_{j_\ell}$ para algún $\ell \in \{\ell_{q+1}, \ldots, \ell_r\} \subset A_R$

$(\bar{c}_\ell = c_\ell - \bar{u}_i - \bar{u}_{j_\ell} < 0)$

entonces:

$$V(P_R(\bar{u}))-V(P_R(u^*)) \leq 2\sum_{k=q+1}^{r}\bar{c}_{\ell_k}+\bar{c}_{\ell_0}+\sum_{k=1}^{p}\bar{c}_{\ell_k}+\bar{u}_i-u_i^* \leq$$

$$\leq \bar{c}_{\ell_0}+\bar{u}_i+2\bar{c}_\ell-u_i^*=\bar{c}_{\ell_0}+\bar{u}_i+2c_\ell-2\bar{u}_{j_\ell}-2\bar{u}_i-u_i^* =$$

$$= \bar{c}_{\ell_0}+2c_\ell-2\bar{u}_{j_\ell}-\bar{u}_i-u_i^*=\bar{c}_{\ell_0}+\bar{c}_\ell < 0.$$

Siempre se cumple que $V(P_R(\bar{u})) \leq V(P_R(u^*))$.

Hemos llegado a una contradicción, por lo que el Teorema queda demostrado.

Comentario 3.3

Los tres teoremas anteriores nos proporcionan una caracterización importante para algunos vectores óptimos de multiplicadores.

Una lectura conjunta de los tres teoremas nos asegura lo siguiente: si el conjunto de multiplicadores óptimos es no vacío, existe ,al menos, un vector u de multiplicadores que satisface:

a) $u_i \geq 0 \quad \forall i \in N_R$

b) $u_i \leq \min_{j \neq i} \{c_{ij}\}$

c) $\bar{c}_{ij}=c_{ij}-u_i-u_j \geq 0 \quad \forall (i,j) \in A_R \cup A_S$

El vector de multiplicadores, obtenido mediante la aplicación del algoritmo heurístico descrito, se construye haciendo crecer lo más posible el valor de $(P_R(u))$ y de forma que satisfaga las tres condiciones anteriores. A pesar de esto, el heurístico no proporciona el vector de multiplicadores óptimo; pero su paralelismo con los resultados de los teoremas anteriores nos 'garantiza' una buena proximidad al óptimo.

Esta es una de las razones que 'justifican' el mal funcionamiento del Método del Subgradiente, para intentar $\underset{u}{\text{Max}}\ V(P_R(u))$ que comentamos a continuación.

Método del Subgradiente

Para resolver el problema de $\underset{u}{\text{Max}}\ (V(P_R(u)))$, al menos de forma aproximada, existen diversos métodos entre los que cabe destacar el método del Subgradiente, que nosotros hemos ensayado y cuyo funcionamiento está brevemente descrito en la introducción.

El Método del Subgradiente es un procedimiento iterativo que partiendo de unos multiplicadores iniciales, que pueden ser todos nulos y que nosotros hicimos iguales a los proporcionados por el algoritmo heurístico descrito, va generando una sucesión de vectores de multiplicadores u^k, y por lo tanto una sucesión de problemas $(P_R(u^k))$, cuyo límite, si se cumplen determinadas condiciones, alcanzaría el valor máximo de $(P_R(u))$.

En nuestro problema concreto, el Método del Subgradiente no produjo los resultados esperados. Utilizamos diversas estrategias eligiendo escalares α_k iniciales para determinar la longitud del paso y los dividimos por 2 cada cierto número de iteraciones en las que el valor del problema relajado no se había incrementado, hasta llegar a un valor que nosotros fijamos en 0.125 ó 0.0625 según los casos.

Las figuras siguientes representan la aplicación del Método del Subgradiente para varios problemas. En todos (salvo en el problema de la figura 3.g) se realizaron 100 ite

raciones. Las abcisas representan las iteraciones 1,10,20,.. ..,100 y las ordenadas, el mejor valor obtenido por el Subgradiente en cada grupo de 10 iteraciones.

Representamos también el valor óptimo del RPP correspondiente a cada uno de los problemas y el valor de la cota inferior definitiva que explicaremos en los apartados siguientes.

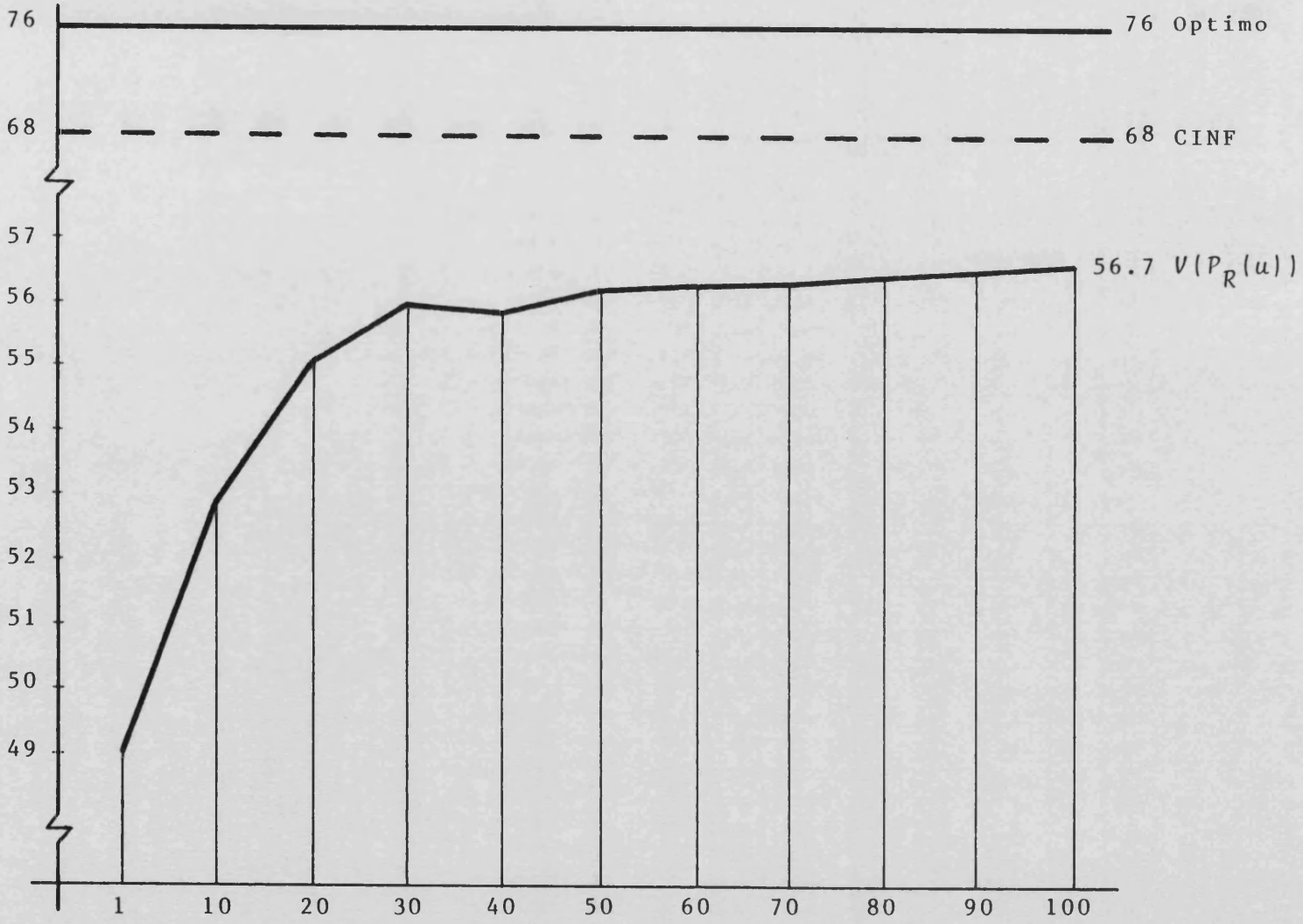


Figura 3.f: Problema 1

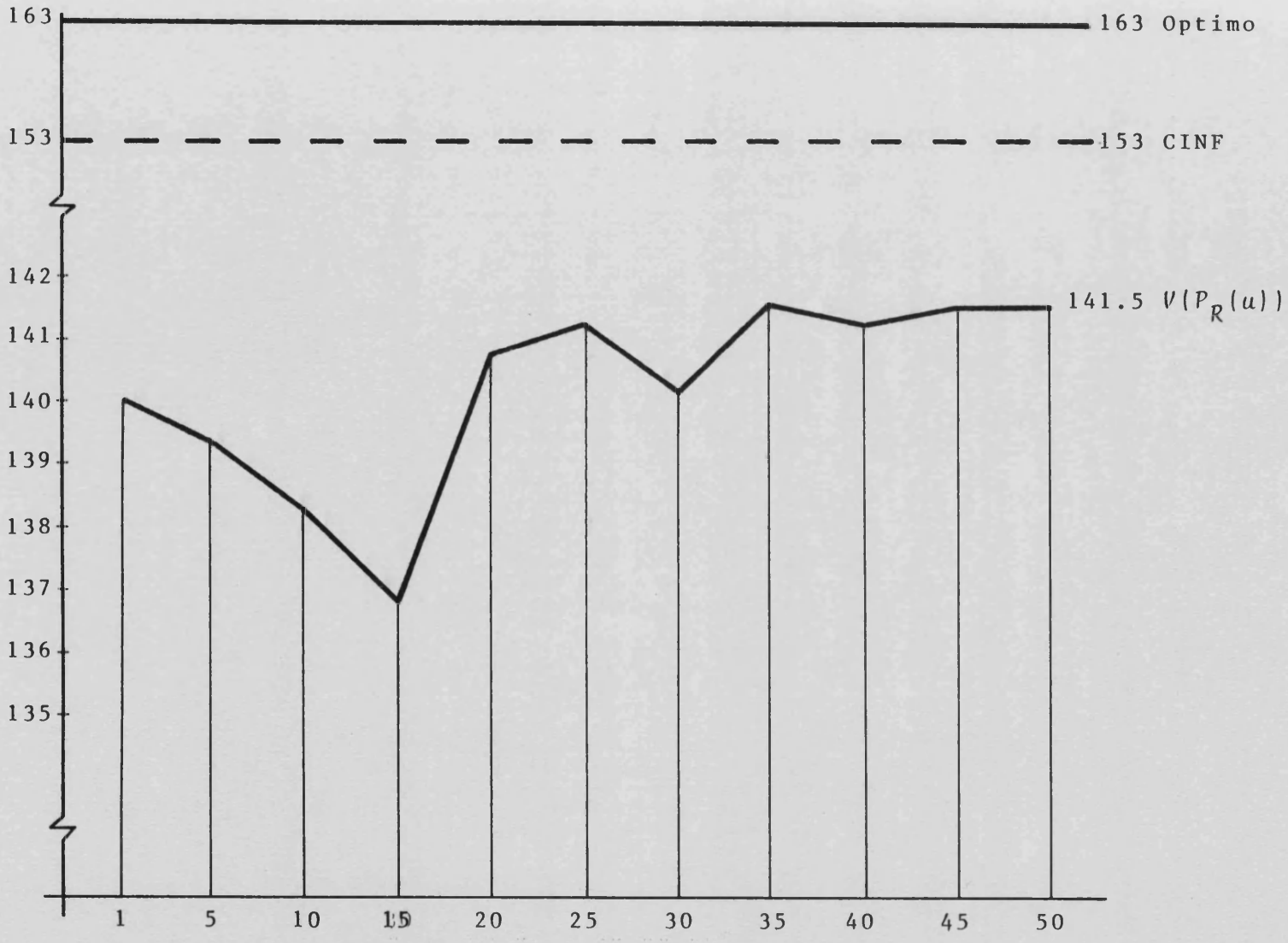


Figura 3.g: Problema 2

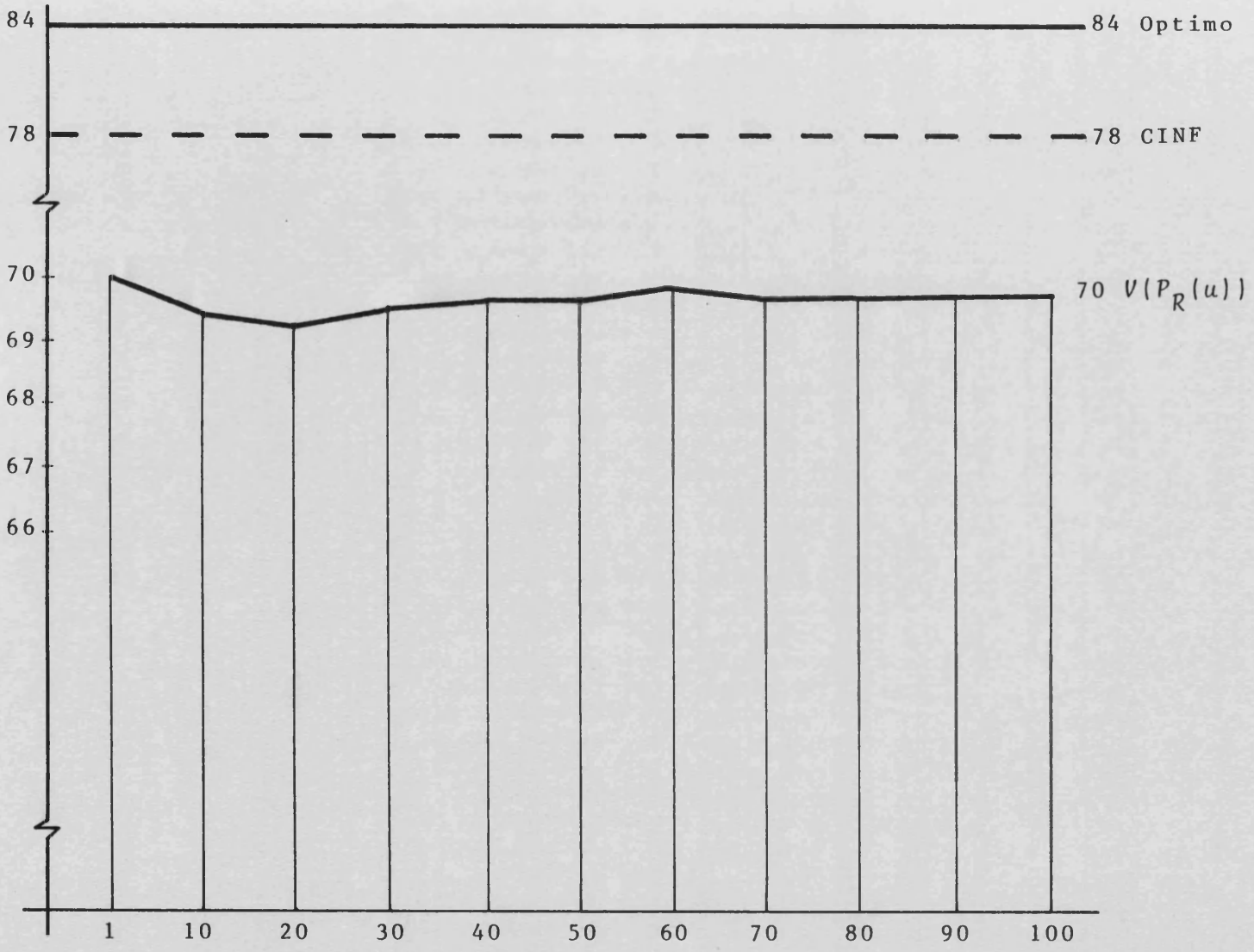


Figura 3.h: Problema 4

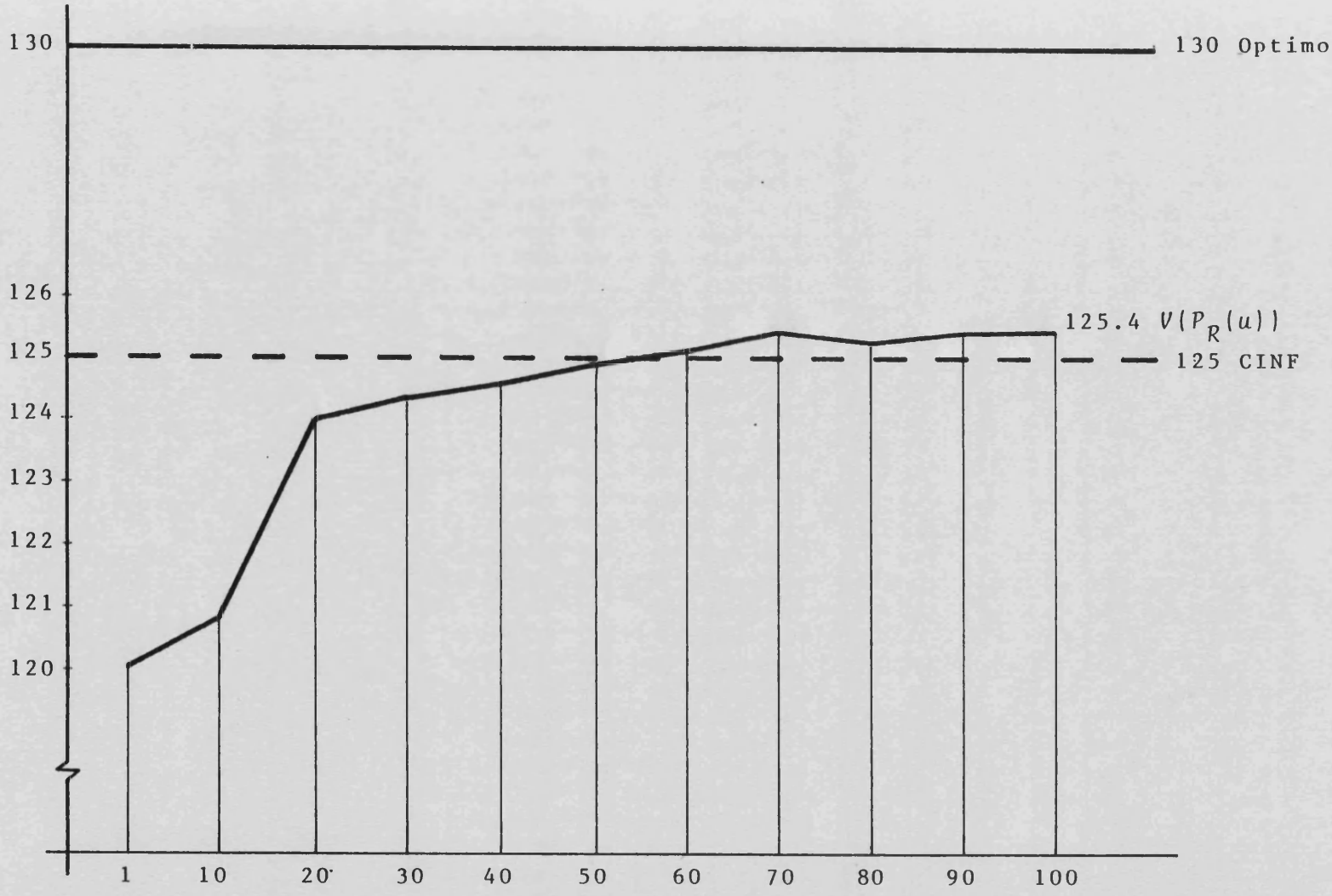


Figura 3.i: Problema 7

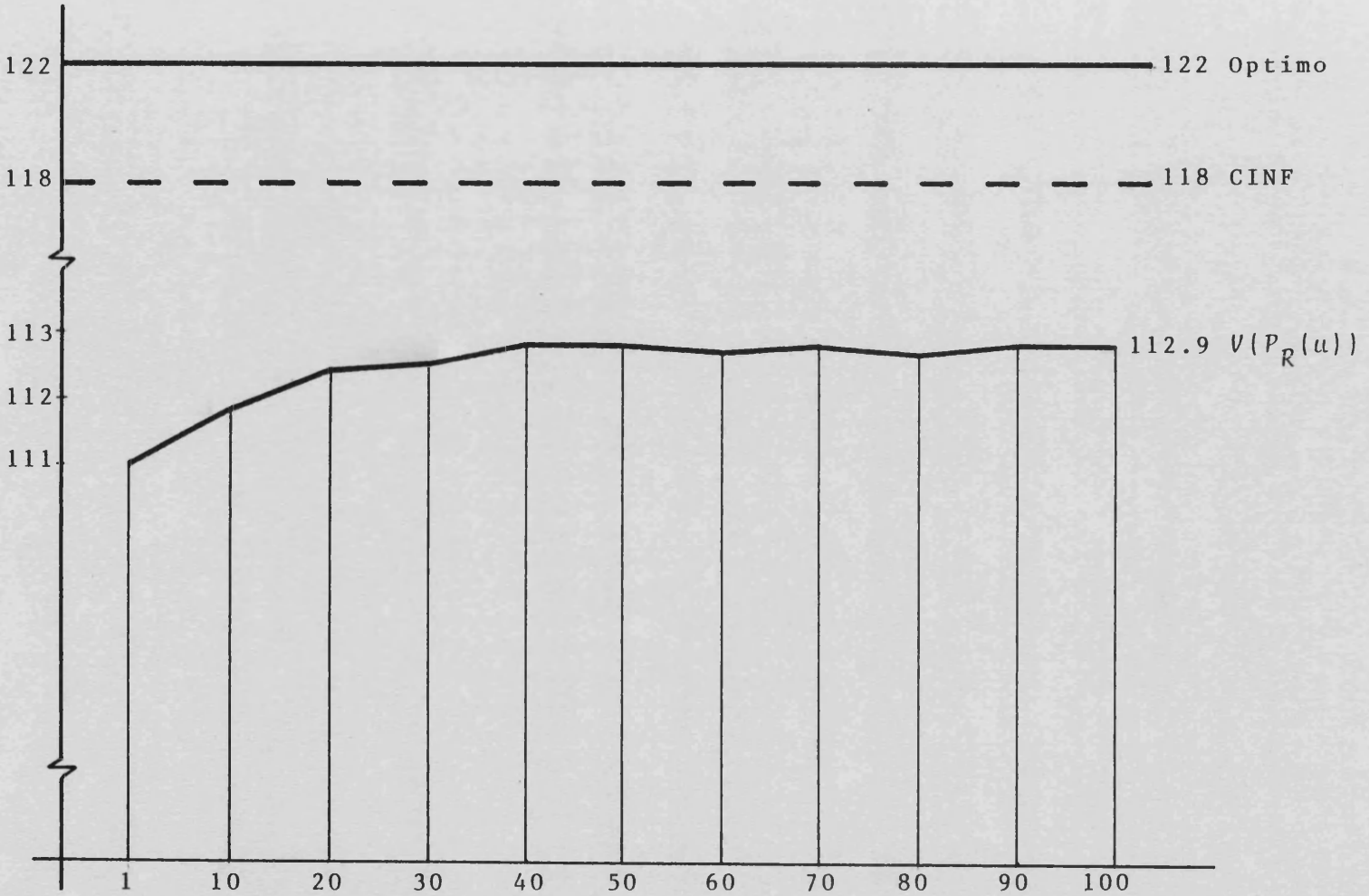


Figura 3.j: Problema 8

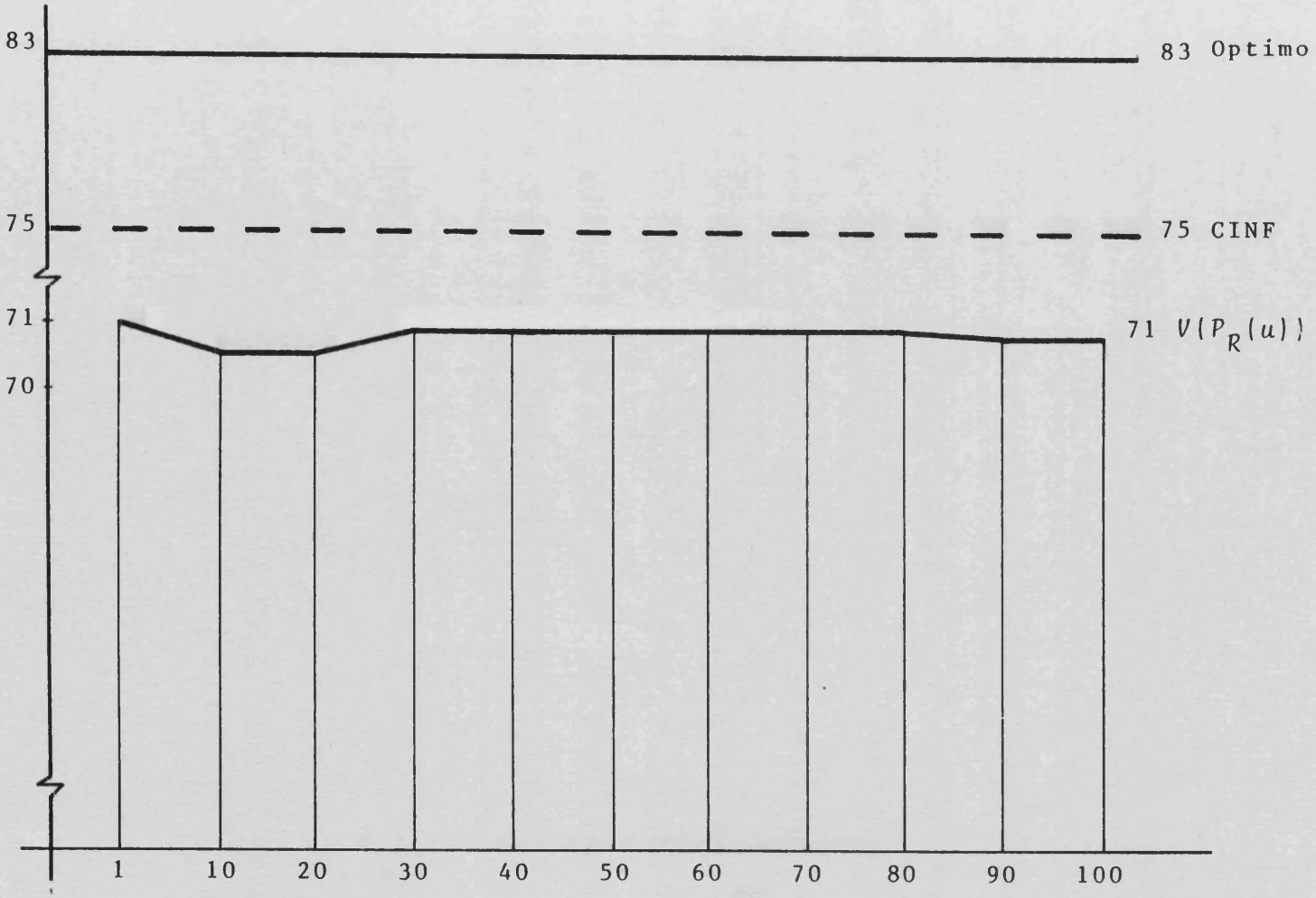


Figura 3.k: Problema 9

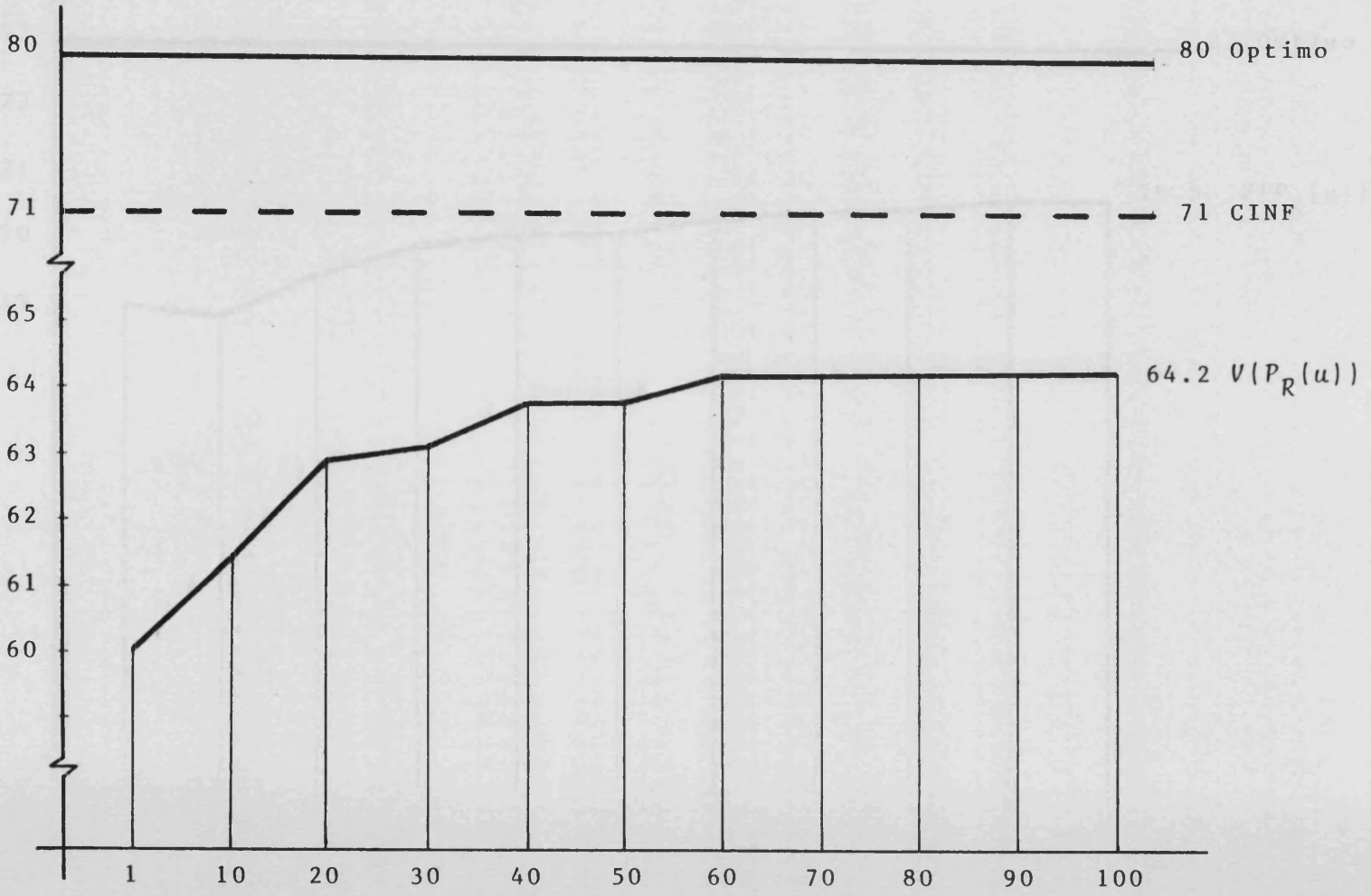


Figura 3.1: Problema 10

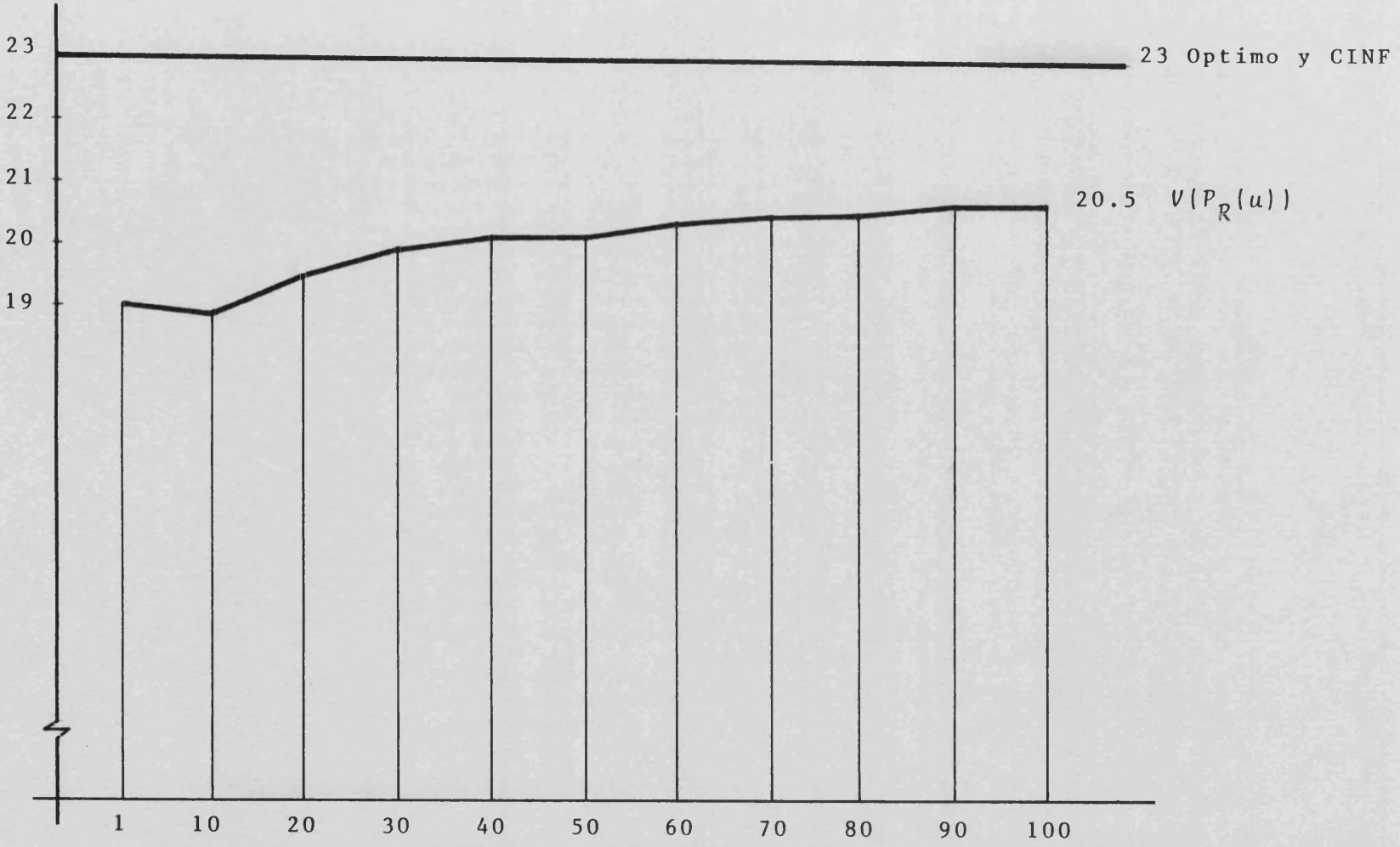


Figura 3.m: Problema 11

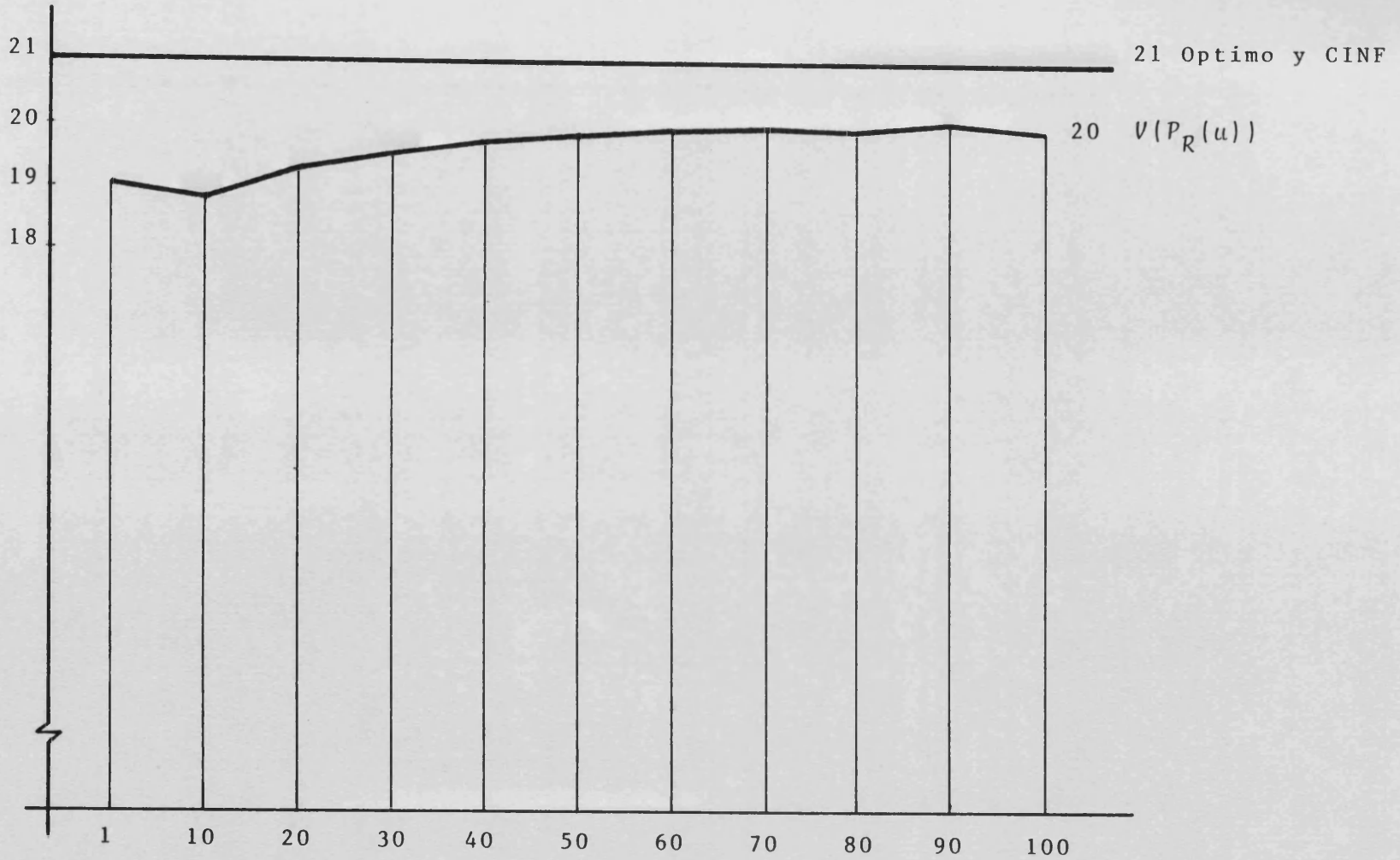


Figura 3.n: Problema 12

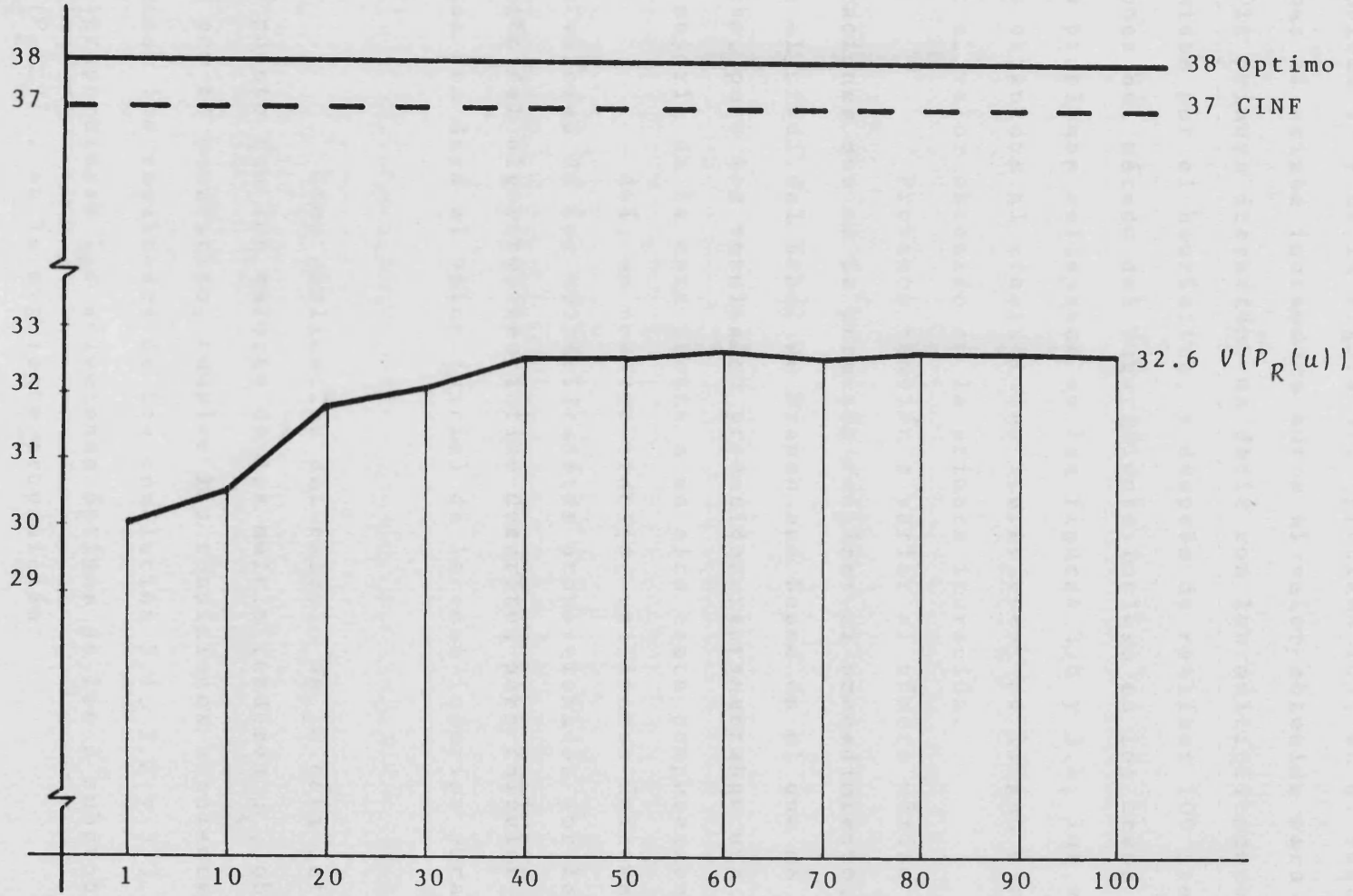


Figura 3.o: Problema 13

Como se puede observar en los gráficos anteriores, salvo en el caso de la figura 3.f (correspondiente al problema 1) y de la figura 3.1 (problema 10), en el resto apenas si existe incremento entre el valor obtenido para $V(P_R(u))$ en la primera iteración, es decir con los multiplicadores obtenidos por el heurístico, y después de realizar 100 iteraciones del método del Subgradiente.Incluso en los casos de los problemas reflejados en las figuras 3.h y 3.k, los valores obtenidos al efectuar las iteraciones, no llegan a superar el valor obtenido en la primera iteración.

Probamos también a variar el número máximo de iteraciones que se le permitía realizar al procedimiento, según el nivel del árbol de Branch and Bound en el que se encontraba, pero los resultados producidos representaban una ligera mejoría de la cota frente a un alto coste computacional.

Así, en nuestro trabajo, utilizamos únicamente los valores de los multiplicadores proporcionados por la aplicación del algoritmo heurístico descrito, para calcular $V(P_R(u))$, que nos dará el valor inicial de la cota inferior para el RPP.

Como explicación del cálculo de la cota inferior, y puesto que los valores de los multiplicadores u_i, obtenidos por el heurístico, cumplen las condiciones siguientes, resumimos los resultados de los corolarios 3.1, 3.2 y 3.3, que nos proporcionan las soluciones óptimas de los 3 subproblemas de $(P_R(u))$, en la siguiente proposición:

Proposición 3.7

Si el vector u de multiplicadores se elige de forma que:

a) $u_i \geq 0 \quad \forall i \in N_R$,

b) $\bar{c}_\ell = c_\ell - u_{i_\ell} - u_{j_\ell} \geq 0 \quad \forall \ell \in A_R \cup A_S$

Entonces,

$$V(P_R(u)) = V(SST) + \sum_{i \in N_R^o} u_i + \sum_{\ell \in A_R} c_\ell$$

donde, recordemos, $V(SST)$ representa el coste del SST en el grafo $\tilde{G}_C$ y N_R^o es el subconjunto de N_R, de vértices impares respecto a las aristas requeridas.

Comentario 3.4

De ahora en adelante, $V(P_R(u))$, va a representar el valor del problema relajado $(P_R(u))$, donde u es el vector de multiplicadores obtenido utilizando el heurístico descrito (a menos que explícitamente se diga otra cosa). Esto es importante porque nos permite asegurar que la solución de $(P_R^3(u))$ está formada exclusivamente por $k-1$ aristas, de coste reducido no negativo, que forman un SST de mínimo peso en un grafo condensado $\tilde{G}_C$. Y porque, por la proposición 3.5, tenemos un método sencillo y rápido de calcular el valor de $(P_R(u))$, como ilustramos a continuación:

Ejemplo 3.2: Cálculo de $V(P_R(u))$

Partiendo del grafo G_C, con costes reducidos, de la figura 3.c, construimos el grafo condensado $\tilde{G}_C$ que será el siguiente:

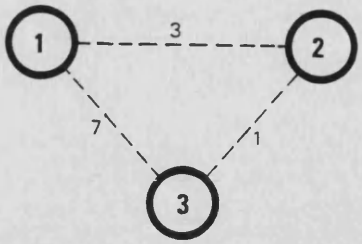


Figura 3.p: Grafo condensado

Resolviendo el problema de hallar un SST en el grafo $\tilde{G}_C$ de la figura 3.p, formado por las aristas de coste 1 y 3, tendremos que la solución a $(P_R^3(u))$, vendrá dada por:

$y_{10,11}=1 \quad ; \quad y_{6,7}=1 \quad ; \quad y_\ell=0 \quad \forall \ell \in A_S-\{(10,11),(6,7)\}$

Así pues el valor de la cota inferior, será:

$$V(P_R(u))=V(SST)+\sum_{i\in N_R^\circ} u_i+\sum_{\ell\in A_R} c_\ell = 4+39+70= 113$$

3.3.2 Mejora de la Cota Inferior: CINF1

Vamos a desarrollar dos procedimientos para mejorar la cota inferior a partir del valor obtenido en $V(P_R(u))$. El primero de ellos será estudiado en este apartado y el valor que proporcione lo representaremos por CINF1.

El procedimiento para calcular CINF1 viene dado en la siguiente proposición:

Proposición 3.8

Si representamos por SSTMAX el coste de la arista de mayor peso en el SST solución de $\tilde{G}_C$, tenemos que:

$$V(P_R(u)) + SSTMAX \leq V(P_R)$$

Demostración:

Demostración:

Teniendo en cuenta que la solución al RPP es un circuito, esa solución óptima debe producir un circuito en el grafo condensado $\tilde{G}_C$. Y un circuito, en un grafo con k vértices ($|\tilde{N}|=k$), debe contener al menos k aristas. Puesto que la solución al problema $(P_R(u))$ es un árbol de mínimo peso en $\tilde{G}_C$ con $k-1$ aristas, a la solución de $(P_R(u))$ le podemos incorporar una arista extra.

Veamos como la arista extra que podemos añadir a la solución de $(P_R(u))$ tiene un coste como mínimo igual al de la arista de mayor peso en el árbol. En efecto, teniendo en cuenta el funcionamiento del algoritmo de búsqueda del SST en un grafo (descrito en la Introducción), consideremos la última arista que se incorpora al árbol; ésa última arista la representaremos por ℓ^* y su coste por SSTMAX (SSTMAX ≥ 0).

La arista ℓ^* es elegida por el algoritmo para formar parte del SST en $\tilde{G}_C$ porque es la arista de mínimo coste entre S_t y $\bar{S}_t$, $S_t \subset \tilde{N}$, $\bar{S}_t = \tilde{N} - S_t$, de forma que

$$\bar{c}_{\ell^*} = \min \left\{ \bar{c}_\ell \,, \forall \ell \in K_t = (S_t, \bar{S}_t) \right\} \quad \text{, luego:}$$

$$\sum_{\ell \in K_t} y_\ell = 1 = y_{\ell^*} \qquad K_t = (S_t, \bar{S}_t)$$

Y puesto que cualquier circuito debe cumplir que el número de aristas en cada corte del tipo $(S_t, \bar{S}_t)$ debe ser al menos 2, sabemos que el coste de la arista extra que podemos añadir, será al menos el coste de la arista de menor peso en el corte $(S_t, \bar{S}_t)$, ésto es, el coste de ℓ^*. Queda probado que $V(P_R(u)) +$ SSTMAX sigue siendo una cota inferior, que representaremos

por CINF1, al valor óptimo del RPP.

Comentario 3.5

En la demostración anterior hemos insistido en el hecho de que podíamos añadir a la solución del SST una arista extra (puesto que toda solución al RPP debe contener al menos k aristas de conexión) y luego hemos encontrado el coste que podíamos añadir a $V(P_R(u))$ para que continuara siendo una cota inferior de $V(P_R)$. No hemos partido únicamente del hecho de que en cada corte, definido por cada arista del SST, deben haber al menos dos aristas porque podríamos llegar a la conclusión, evidentemente falsa, de que al valor $V(P_R(u))$ le podemos añadir la suma de los costes de todas las aristas del SST y constituir una cota inferior del problema.

Ejemplo 3.3

Como aplicación de la Proposición 3.6 al Ejemplo 3.2, tenemos que la nueva cota inferior será :

CINF1 = 113+3 = 116

donde 3 es el coste de la arista de mayor peso en la solución al SST sobre el grafo $\tilde{G}_C$ de la figura 3.p

3.3.3 Mejora de la Cota Inferior CINF2

El otro procedimiento para mejorar el valor de la Cota Inferior, a partir de $V(P_R(u))$, se basa en lo siguiente: en presencia de las restricciones (1'), la 2-conectividad de la solución a (P_R) está implícita por las restantes restricciones de (P_R). Sin embargo, en el problema relajado $P_R(u)$

la 2-conectividad de la solución en el grafo $\tilde{G}_C$ no está garantizada y, por lo tanto, las restricciones

$$\sum_{\ell \in K_t} y_\ell \geq 2 \quad \forall \tilde{K}_t = \left\{ (i,j) \in \tilde{A} \mid i \in S_t, j \in \bar{S}_t, S_t \subset \tilde{N} \right\} \quad (5)$$

pueden añadirse explícitamente al conjunto de restricciones de $(P_R(u))$, con objeto de incrementar la cota.

En particular, si en la solución al problema del SST definido en $\tilde{G}_C$, puede detectarse una restricción t del tipo (5) que sea violada, esta restricción t puede relajarse mediante un multiplicador de Lagrange λ_t e incorporarse a la función objetivo de $(P_R(u))$.

El problema relajado, que representaremos por $(P_R(u,\lambda))$, será:

$$(P_R(u,\lambda)) \quad \text{Min} \left\{ \sum_{\ell \in A_R} (c_\ell - u_{i_\ell} - u_{j_\ell}) x_\ell + \sum_{\ell \in A_S} (c_\ell - u_{i_\ell} - u_{j_\ell}) y_\ell + \right.$$

$$\left. + \sum_{i \in N_R} u_i \left(2w_i - \sum_{\ell \in A_R} a_{i\ell}\right) + \sum_t \lambda_t \left(2 - \sum_{\ell \in K_t} y_\ell\right) + \sum_{\ell \in A_R} c_\ell \right\}$$

sujeto a:

$$\sum_{\ell \in K_t} y_\ell \geq 1 \quad \forall K_t = \left\{ (i,j) \in A_S \mid i \in N(V_t), j \in N(\bar{V}_t), V_t \subset F \right\}$$

$$0 \leq x_\ell \leq 1 \text{ , enteras} \quad \forall \ell \in A_R$$

$$0 \leq y_\ell \leq 2 \text{ , enteras} \quad \forall \ell \in A_S$$

$$d(i) \leq w_i \leq d_{G_C}(i) \text{ , enteras} \quad \forall i \in N_R$$

En la práctica,no vale la pena iterar con objeto de encontrar los mejores multiplicadores λ_t(por ejemplo utilizando el Método del Subgradiente), pero sí utilizar algún procedimiento para fijar a priori los valores de λ_t. Un procedimiento heurístico de este tipo ha sido desarrollado por Balas

y Christofides (1981) para el Problema del Agente Viajero. Y se basa en el concepto de lagrangeano restringido que ya fué comentado en el apartado 3.2 de esta sección.

El procedimiento utilizado para calcular la cota inferior es:

i) Encontrar buenos valores de los multiplicadores u(utilizando el algoritmo descrito en el apartado 3.3.2) en el sentido de maximizar el valor $V(P_R(u))$.

ii) A partir de los valores de y_ℓ obtenidos en i), identificar cortes del tipo (5), calcular los multiplicadores asociados a dichos cortes e incorporarlos a la función objetivo de $(P_R(u))$, tal como aparece en $(P_R(u,\lambda))$.

Veamos una proposición que nos caracteriza todos los cortes del tipo (5) que se violan por la solución a $(P_R^3(u))$.

Proposición 3.9

A cada arista del SST en el grafo $\tilde{G}_C$, le corresponde una restricción del tipo (5), 2-conectividad, que se viola y viceversa.

Demostración:

En efecto, al ser los costes de las aristas de $\tilde{G}_C$ no negativos ($\tilde{c}_\ell \geq 0 \quad \forall \ell \in \tilde{A}$) y minimizar la función objetivo de $(P_R^3(u))$, todas las restricciones a dicho problema se cumplirán en la forma:

$$\sum_{\ell \in \tilde{K}_t} y_\ell = 1 \qquad \forall \tilde{K}_t \qquad (6)$$

Cada arista del SST en $\tilde{G}_C$, define una restricción del tipo (6) y por lo tanto una restricción del tipo (5) que se viola.

Para demostrar que a cada restricción del tipo (5) que se viola le corresponde una arista del SST, bastará comprobar que como máximo hay $k-1$ restricciones que se violan (ya que hay $k-1$ aristas en el SST). Tomemos una restricción del tipo:

$$\sum_{\ell \in \tilde{K}_t} y_\ell \geq 1 \qquad \text{dado} \quad \tilde{K}_t$$

que no corresponda a una arista del árbol; ésto significa que puesto que el SST en $\tilde{G}_C$ es conexo debe cumplirse la restricción anterior, pero en la forma $\sum y_\ell \neq 1$. Luego se debe cumplir que $\sum y_\ell \geq 2$ y por lo tanto la 2-conectividad. Hay pues, como máximo, tantas restricciones del tipo (5) que se violan como aristas hay en el SST.

Describimos a continuación el algoritmo que nos proporcionará los valores de los multiplicadores λ_t:

Algoritmo Heurístico para el cálculo del vector λ

Basándonos en la Proposición 3.9, el algoritmo consiste en el siguiente proceso iterativo:

STEP 0 (Inicialización)

Sea $\tilde{A}_T=\{\ell_1,\ell_2,\ldots,\ell_{k-1}\}$ el subconjunto de $\tilde{A}$ formado por las aristas que constituyen la solución al SST en el grafo $\tilde{G}_C$.

Tomar $t=1$. Hacer $\bar{\bar{c}}_\ell=\bar{c}_\ell \quad \forall \ell \in \tilde{A}$.

Ir al Step 1

STEP 1 (Identificación de la restricción incumplida y cálculo del multiplicador asociado)

A partir de ℓ_t, construir el corte $\tilde{K}_t=(S_t,\bar{S}_t)$, donde $i_{\ell_t}\in S_t$, $j_{\ell_t}\in\bar{S}_t$.

Hacer $\lambda_t = \min_{\ell\in\tilde{K}_t} \{\bar{\bar{c}}_\ell\}$

Ir al Step 2

STEP 2 (Actualización de costes)

Hacer $\bar{\bar{c}}_\ell \leftarrow \bar{\bar{c}}_\ell - \lambda_t \quad \forall \ell\in\tilde{K}_t$

Ir al Step 3

STEP 3 (Fin)

Si $t=k-1$. Fin

Si $t\neq k-1$, hacer $t \leftarrow t+1$ y volver a Step 1

Ilustraremos el funcionamiento de este algoritmo con el siguiente ejemplo:

Ejemplo 3.4

En la figura 3.q representamos el grafo condensado $\tilde{G}_C$ correspondiente al ejemplo 3.1

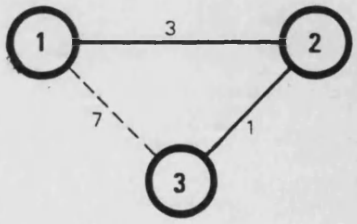


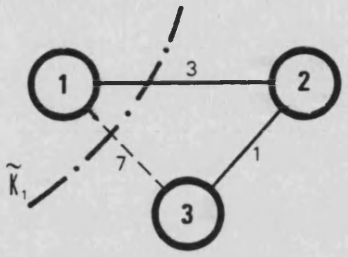
Figura 3.q : Grafo $\tilde{G}_C$ del ejemplo 3.1

donde las aristas dibujadas con trazo contínuo corresponden a las aristas de la solución al SST encontrada en el Ejemplo 3.2

Vamos a aplicar el algoritmo descrito:

$\tilde{A}_T = \{(1,2),(2,3)\}$

tomamos la primera arista: $(1,2)$, esta arista tiene asociada un corte que representaremos por $\tilde{K}_1$ y donde se incumple una restricción del tipo (5)

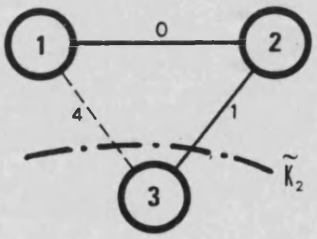


calculamos el multiplicador λ_1 asociado:

$$\lambda_1 = \min \{3,7\} = 3 \qquad \text{y}$$

actualizamos los costes: $4 \leftarrow 7$; $0 \leftarrow 3$

tomamos la segunda arista: $(3,2)$, e identificamos el corte asociado $\tilde{K}_2$



calculamos $\lambda_2 = \min \{1,4\} = 1$ y

actualizamos los costes: $3 \leftarrow 4$; $0 \leftarrow 1$.

El grafo resultante será

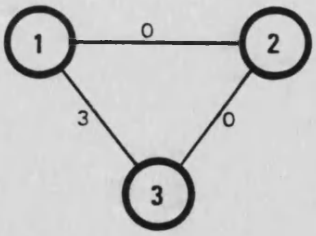


Figura 3.r : Grafo resultante

La cota inferior obtenida a partir de la resolución de $P_R(u,\lambda)$ tendrá un valor, CINF2, que en el caso del ejemplo 3.1, será:

$$\text{CINF2} = \sum_{i \in N_R^o} u_i + V(\text{SST}) + 2 \sum_t \lambda_t + \sum_{\ell \in A_R} c_\ell =$$

CINF2 = 39+ 0+2*4+70= 117

donde $V(SST) = 0$, como resulta de hallar el SST en el grafo con costes modificados (por los multiplicadores u_i y λ_t) de la figura 3.r.

3.3.4 Estudio comparativo de CINF1 y CINF2

En esta Sección hemos discutido las posibles relajaciones en el problema (P_R) para obtener cotas inferiores al valor óptimo del problema. Hemos valorado todos los casos y hemos insistido en la discusión del obtenido al relajar las restricciones de paridad sobre los grados de los vértices. A partir de esta relajación hemos construido dos cotas inferiores, CINF1 y CINF2, que son las utilizadas en el procedimiento Branch & Bound para encontrar la solución óptima al RPP.

Es importante resaltar que no hay razones teóricas ni prácticas, tal como han sido construidas, para asegurar que una cota inferior funciona mejor que otra. Depende de las características particulares de cada problema tal como puede apreciarse en la Tabla 3.1.

En la tabla se dan los valores óptimos del RPP en 24 problemas, así como los valores obtenidos de CINF1 y CINF2 para dichos casos. Podemos observar que no hay una cota inferior que supere en todos los casos a la otra; ésto depende del número de aristas no requeridas entre las componentes, de sus costes y, en gran medida, de la estructura del grafo condensado $\tilde{G}_C$ y de la forma que tiene el SST calculado en dicho grafo.

PROBLEMA	1	2	3	4	5	6	7	8	9	10	11	12
OPTIMO	76	163	102	84	129	102	130	122	83	80	23	21
CINF1	68	153	99	78	121	91	125	118	75	69	23	21
CINF2	68	148	101	77	108	92	125	113	69	71	23	21

PROBLEMA	13	14	15	16	17	18	19	20	21	22	23	24
OPTIMO	38	209	445	203	112	148	263	398	366	621	480	405
CINF1	37	191	399	193	99	139	246	364	341	579	465	386
CINF2	33	193	398	186	94	132	246	373	339	591	464	388

Tabla 3.1: Comportamiento de la Cota Inferior

En el procedimiento de obtención de CINF2, hay que calcular un árbol generador de mínimo peso en $\tilde{G}_C$, con costes modificados $\bar{c}_\ell = c_\ell - u_{i_\ell} - u_{j_\ell}$, para determinar las restricciones del tipo (5) incumplidas y así calcular los valores de los multiplicadores λ_t asociados a las mismas. Es, en esta etapa del cálculo de CINF2, cuando conocemos el valor de SSTMAX y, por lo tanto, sin esfuerzo computacional extra, el valor de - CINF1.

Nuestra estrategia será, pues, calcular el valor de CINF1, y sólo si esta cota inferior no satura el nudo en el que nos encontramos del árbol de enumeración, a partir de ella, calcularemos el valor de CINF2. Utilizaremos la expresión CINF para representar el máximo de los valores de CINF1 y CINF2.

Un aspecto importante, que merece ser señalado, es el hecho de que nuestra cota inferior requiere muy poco esfuerzo computacional para ser calculada y su valor crece rápidamente a medida que descendemos en el árbol de enumeración. Lo que compensa el relativo alejamiento (un 6% por término medio). Esta es una de las razones por las que se hace innecesario, incluso contraproducente en términos de tiempo de CPU, la utilización del Método del Subgradiente para intentar incrementarla. Nuestra experiencia práctica nos ha mostrado que el - tiempo utilizado para realizar un cierto número k de iteraciones del Subgradiente era aproximadamente el utilizado para estudiar k nudos del árbol de B&B; número de nudos superior al que necesitaba nuestra cota para saturar la rama del árbol en la que nos encontrábamos.

S E C C I O N 4

COTA SUPERIOR PARA EL RPP

El Problema del Cartero Rural (RPP) es un problema NP-completo como demostraron Lenstra y Rinooy Kan (1976). Así, cualquier método de solución exacta para el RPP tiene un índice de crecimiento del tiempo de computación exponencial. Es importante, por lo tanto, encontrar métodos aproximados de solución cuyo índice de crecimiento del tiempo de computación sea polinomial y que se compruebe experimentalmente su buen funcionamiento.

Por otra parte, en un algoritmo general de Branch and Bound, la saturación de un nudo se produce cuando la cota inferior en dicho nudo supera o iguala el valor de la mejor solución conocida para el problema. Obviamente, la saturación de nudos se acelerará si el valor de la cota superior (el de la mejor solución posible conocida) se ajusta lo máximo posible al valor de la solución óptima del problema.

Existen diversos métodos para intentar obtener buenas cotas superiores. En algunos casos podemos conocer a priori una buena solución posible; por ejemplo, cuando el problema de PLE es el modelo de un sistema que ya funciona y sobre el que se quieren realizar ciertas mejoras. En otros, las soluciones posibles se van obteniendo en algunos de los nudos del árbol y la mejor es la que pasa a ser la cota superior. Otro método para encontrar buenas soluciones posibles es el uso de algoritmos aproximados o heurísticos; es decir, técnicas que no garantizan la proximidad al óptimo (en algunos casos sí se garantiza por medio del Worst Case Analysis que está em

pezando a ser objeto generalizado de estudio) pero que son - rápidas y ,por lo tanto, poco costosas computacionalmente.

Como dice Robert Garfinkel (1972) : ' Very little is lost if a heuristic fails and a great deal is gained if a good feasible solution is found '.

4.1 UN HEURISTICO PARA EL RPP

Hemos desarrollado un algoritmo heurístico para encontrar buenas soluciones posibles para el RPP. Se basa en la resolución de dos problemas bien conocidos : el de encontrar un árbol generador de mínimo peso (SST) en un grafo, y en resolver un problema de 1-matching sobre un conjunto de vértices de grado impar, para los que existen algoritmos eficientes con índice de crecimiento del tiempo de computación polinomial.

A partir del grafo original $G=(N,A)$, realizamos las transformaciones de la Sección 2 para obtener el grafo - $G_C=(N_R, A_R \cup A_S)$. Construimos el grafo reducido $\tilde{G}_C$, tal como se vió en la definición 3.1.

Representaremos por T la solución al SST sobre el grafo $\tilde{G}_C$ y sea T^+ el conjunto de aristas de G_C correspondientes a las aristas T de $\tilde{G}_C$. Consideremos el conjunto de - aristas $A_R \cup T^+$ y representaremos por N_R^o el conjunto de vértices de G_C que tienen grado impar respecto a las aristas de - $A_R \cup T^+$.

Sea $<N_R^o>$ el grafo inducido en G_C' (grafo completo) por el conjunto N_R^o. Es decir, el grafo cuyo conjunto de vértices es N_R^o y cuyo conjunto de aristas está formado por aristas entre cualquier par de vértices y con coste el del camino más corto entre ellos calculado en el grafo original G. Resolvemos el problema de 1-matching de mínimo coste en el - grafo $<N_R^o>$, y representamos por M el conjunto de aristas en

este matching.

Proposición 4.1

La unión del conjunto de aristas M en el matching, el conjunto T^+ y el conjunto A_R forma una solución posible para el RPP.

Demostración:

Vamos a hacer la siguiente asignación de valores a las variables:

$$y_\ell = 1 \quad \forall \ell \in T^+ \qquad \text{y} \qquad y_\ell = 0 \quad \forall \ell \in A_S - T^+$$

Evidentemente, esta asignación satisface las restricciones del tipo (2') por la definición de árbol generador.

Definimos ahora $M_R = \{\ell \in M \cap A_R \subset A_R\}$ y $M_S = \{\ell \in M \cap A_S \subset A_S$ y hacemos:

$$x_\ell = 1 \quad \forall \ell \in M_R \quad \text{y} \quad x_\ell = 0 \quad \forall \ell \in A_R - M_R$$

$$y'_\ell = 1 + y_\ell \quad \forall \ell \in M_S \quad \text{y} \quad y'_\ell = y_\ell \quad \forall \ell \in A_S - M_S$$

Puesto que las variables y_ℓ satisfacen las restricciones (2') también lo harán las variables y'_ℓ

Por definición del problema de 1-matching sobre los vértices de $< N_R^o >$, las variables y'_ℓ y x_ℓ satisfacen las restricciones de paridad (1). Y ya que las variables toman valores enteros y menores o iguales que su cota superior (tengamos en cuenta que los costes son no negativos y que una arista puede aparecer una vez como máximo en el SST y una vez, también como máximo, en la solución del problema de 1-matching), la asignación de valores dada es una solución del problema (P_2). Por lo tanto es una solución posible para el RPP.

Para ilustrar el funcionamiento de este algoritmo heurístico, veamos el siguiente ejemplo.

Ejemplo 4.1

Sea el grafo G_C correspondiente al problema 15:

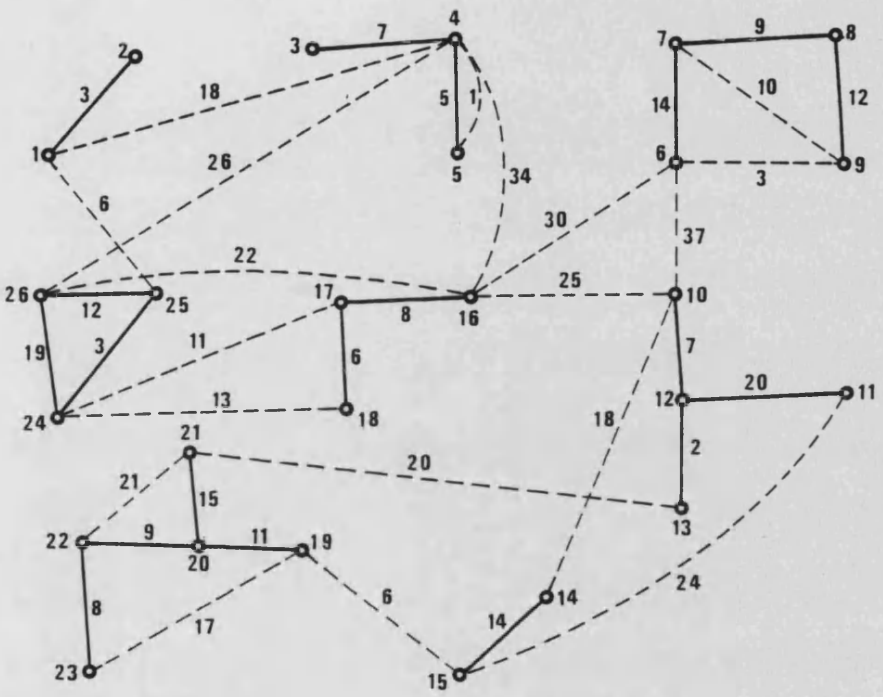


Figura 4.a: Grafo G_C del problema 15

Construimos a continuación el grafo $\tilde{G}_C$ y calculamos el SST que representamos por líneas de trazo contínuo:

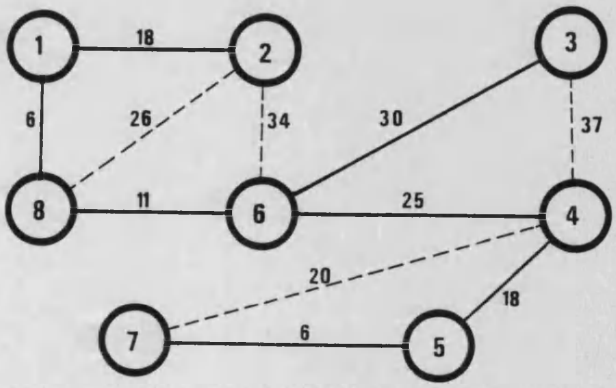


Figura 4.b: SST en el grafo $\tilde{G}_C$

La solución posible determinada por el conjunto de aristas $A_R \cup T^+ \cup M$ está representada en la figura siguiente, donde las aristas de A_R están dibujadas con trazo contínuo, las de T^+ con trazo discontínuo y las de M con puntos:

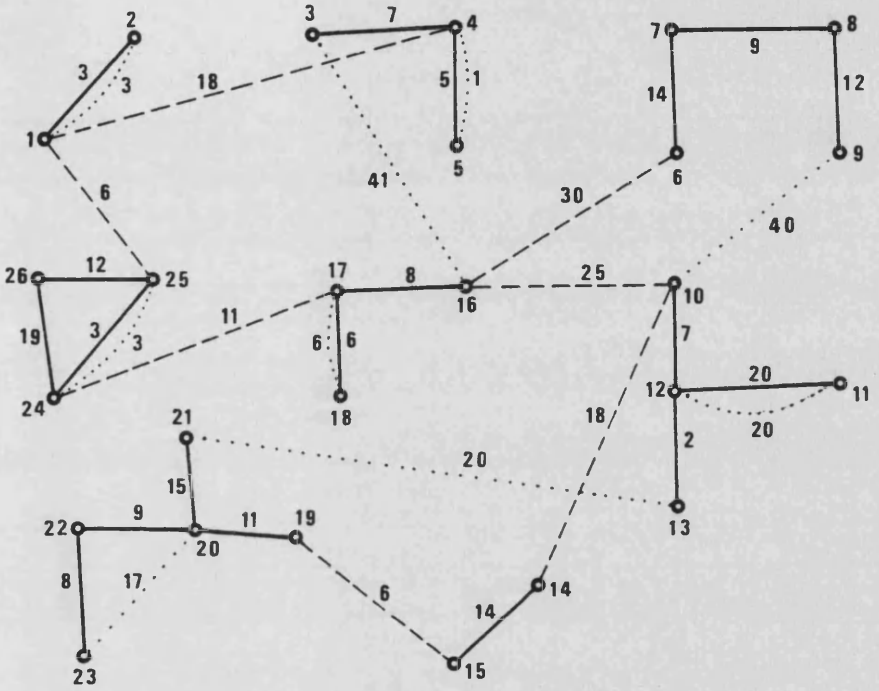


Figura 4.c: Grafo inducido por $A_R \cup T^+ \cup M$

Esta solución posible al RPP definido sobre el grafo del problema 15, tendrá un coste que viene dado por:

$$V = \sum_{\ell \in A_R} c_\ell + \sum_{\ell \in SST} c_\ell + \sum_{\ell \in M} c_\ell = 184 + 114 + 151 = 449$$

Comentario 4.1

El algoritmo descrito es un algoritmo polinomial ya que como hemos visto consta de dos fases: el cálculo de un SST y el de un matching perfecto de mínimo coste. Existen buenos algoritmos, $O(k^2)$, para hallar el SST de un grafo con k vértices, descritos en Christofides (1975). Uno de los mejo-

res algoritmos conocidos para el cálculo de matchings de mínimo coste es el desarrollado por Lawler (1976) y cuyo índice de crecimiento es $O(n^3)$.

Otro aspecto que merece ser tomado en cuenta es el hecho de que el SST que calculamos en nuestro heurístico es un SST especial, en el sentido de que, entre todas aquellas aristas, del mismo coste, candidatas para formar parte del árbol, se elige aquélla que satisface más restricciones del tipo (*1*)(restricciones de paridad sobre los grados de los vértices).

4.2 COTA SUPERIOR PARA EL RPP

Hemos comentado, en la introducción de esta Sección, la importancia que tiene en cualquier procedimiento de Branch and Bound el conocimiento de una buena cota superior para el problema.

Aunque el algoritmo heurístico descrito en el apartado 4.1 es óptimo en dos etapas, el SST es la solución óptima al problema de conectar un grafo,y el matching de mínimo coste es la forma óptima de hacer que todos los vértices tengan grado par, no tiene por qué ser un procedimiento óptimo en su conjunto.Así pues, el algoritmo descrito no proporciona,en general, la solución óptima al RPP y es susceptible de ser mejorado. En muchos casos puede mejorarse utilizando el procedimiento descrito por la siguiente proposición:

Proposición 4.2

En el grafo inducido por la solución del algoritmo heurístico, si existen dos aristas, $(i,j)\varepsilon T^+$ y $(j,k)\varepsilon M$, de forma que al sustituirlas por la arista (i,k) el grafo permanece conexo y se cumple que $c_{ik} < c_{ij}+c_{jk}$, entonces,el grafo resultante inducido por el conjunto de aristas:

$$A_R \cup T^+ \cup M \cup \{(i,k)\} - \{(i,j),(j,k)\}$$

es una nueva solución posible al RPP de menor coste.

Demostración:

Es evidente teniendo en cuenta que al realizar

la sustitución de (i,j) y (j,k) por la arista (i,k), el grafo resultante sigue siendo conexo por hipótesis. Además los grados de los vértices permanecen inalterados salvo el del vértice j que disminuye exactamente en dos unidades, con lo que el grafo sigue siendo par. Tenemos pues una nueva solución posible al RPP de coste menor a la proporcionada por el algoritmo heurístico.

Ilustraremos este procedimiento de mejora de la cota superior mediante el siguiente ejemplo:

Ejemplo 4.2

En el grafo de la figura 4.c, solución posible al RPP del problema 15, podemos sustituir la arista $(17,24)\varepsilon T^+$ y la arista $(17,18)\varepsilon M$ por la $(18,24)$, obteniendo así una nueva solución posible para el RPP con coste menor; concretamente la mejora será de 11+6-13=4 unidades.

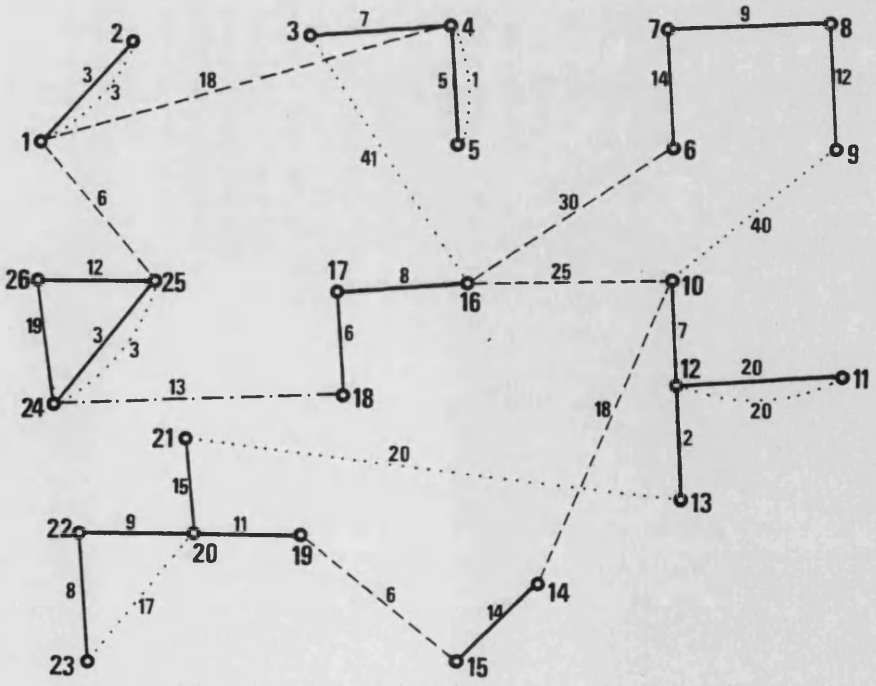


Figura 4.d: Solución posible mejorada

La nueva solución posible tendrá un coste de -449-4=445 unidades. En este caso dicho valor coincide con el valor de la solución óptima del problema.

Comportamiento de la Cota Superior

Utilizaremos como cota superior para el RPP, en el nudo 0 del árbol de enumeración, el resultado obtenido al aplicar el procedimiento de mejora, proporcionado por la Proposición 4.2, a la solución posible encontrada por el algoritmo heurístico del apartado 4.1. A este resultado lo designaremos por CSUP.

El comportamiento de esta cota superior viene dado en la tabla siguiente, donde en 12 del total de 24 problemas, la cota superior proporcionó el valor óptimo y que en media está a un 1.3% del valor de la solución óptima.

PROBLEMA	1	2	3	4	5	6	7	8	9	10	11	12
OPTIMO	76	163	102	84	129	102	130	122	83	80	23	21
CSUP	76	164	102	84	135	107	130	122	84	80	23	22
%	0	0.6	0	0	4.6	4.9	0	0	1.2	0	0	4.7
PROBLEMA	13	14	15	16	17	18	19	20	21	22	23	24
OPTIMO	38	209	445	203	112	148	263	398	366	621	480	405
CSUP	38	212	445	203	116	148	280	400	372	632	480	411
%	0	1.4	0	0	3.5	0	6.4	0.5	1.6	1.7	0	1.5

Tabla 4.1: Comportamiento de la Cota Superior

S E C C I O N 5

ALGORITMO DE BRANCH AND BOUND PARA EL RPP

El procedimiento de Branch and Bound es un método muy útil para resolver problemas de optimización combinatorial y problemas de programación lineal entera en general. Proporciona una metodología para encontrar una solución óptima - haciendo una enumeración parcial de las soluciones posibles - del problema.

Los primeros en sugerir un método del tipo Branch and Bound para resolver problemas de programación lineal entera fueron Land y Doig en 1960. El nombre, sin embargo, se debe a Little y otros, quienes, en 1963, proporcionan la primera demostración experimental del método, aplicada al problema del agente viajero (TSP).

En esta sección, presentamos una descripción del algoritmo utilizado, del tipo de Branch and Bound, para obtener una solución óptima del Problema del Cartero Rural (RPP) en un grafo no dirigido.

Para ello, discutiremos las partes fundamentales de las que consta el procedimiento y que se centran en:

1) Estrategia de separación (partición) en subproblemas
2) Estrategia de ramificación (selección de subproblemas)
3) Saturación de nudos

5.1 ESTRATEGIA DE SEPARACION EN SUBPROBLEMAS

La estrategia de separación particiona el conjunto de soluciones posibles del problema original en dos o más subconjuntos. Cada subconjunto en la partición es el conjunto de soluciones posibles de un problema obtenido al imponer restricciones adicionales al problema original. Estos problemas reciben el nombre de problemas candidatos y pueden, a continuación, ser saturados (en el sentido de que ha finalizado su estudio y queda excluído en lo sucesivo) o no. Si un problema candidato no ha sido saturado, es posible aplicar la estrategia de separación a cualquier problema candidato y generar, - dos o más, subproblemas candidatos. Y así sucesivamente.

En nuestro problema la estrategia de separación se realiza utilizando las variables asociadas a las aristas - del conjunto A_S-I. Donde I es el subconjunto de aristas de A_S cuyo vértice inicial y final pertenecen a la misma componente conexa; lo denominaremos conjunto de aristas no requeridas internas. Así, A_S-I será el conjunto de aristas no requeridas - cuyos vértices incidentes pertenecen a distintas componentes y que llamaremos aristas de conexión.

Si representamos por S_p el conjunto de soluciones posibles al problema en el nudo p, entonces, la separación se realiza con las variables y_ℓ, $\ell \varepsilon A_S$-I, en la forma siguiente:

$$S_p \cap \{y_\ell \geq 1\} \quad \text{y} \quad S_p \cap \{y_\ell = 0\} \text{ para algún } \ell \varepsilon A_S - I$$

El nudo p ha sido separado en dos nudos sucesores añadiendo las restricciones opuestas $y_\ell \geq 1$ e $y_\ell = 0$. Lo cual significa que en la solución de uno de los subproblemas, la arista ℓ debe aparecer al menos una vez, mientras que en el otro subproblema, dicha arista no debe ser utilizada por la solución.

Obviamente cualquier solución posible debe cumplir una u otra restricción, con lo que nos aseguramos que la estrategia de separación está bien definida.

Notemos que en algunos casos es posible limitar el desarrollo de un nudo, en el sentido de que sólo tenga un sucesor; estos casos están estudiados en las proposiciones siguientes:

Proposición 5.1

Sea $V_p(P_R(u))$ el valor de la cota inferior obtenida al resolver el problema $(P_R(u))$, tal como se estudió en el apartado 3.3.1 de la Sección 3, en un nudo cualquiera p del árbol de ramificación. Sea una arista $\ell \in A_S - I$, $i_\ell \in S$ y $j_\ell \in \bar{S} = N_R - S$, y sea

$$\hat{c} = \max_{\ell' \neq \ell} \{\bar{c}_{\ell'} = c_{\ell'} - u_{i_{\ell'}} - u_{j_{\ell'}} \mid \ell' = (i_{\ell'}, j_{\ell'}) \in (S, \bar{S})\}$$

Supondremos $\bar{c}_\ell - \hat{c} \geq 0$. Entonces:
si $\bar{c}_\ell - \hat{c} \geq \text{CSUP} - V_p(P_R(u)) \rightarrow y_\ell = 0$ (la arista ℓ no está en la solución), donde CSUP representa el valor de la mejor solución al RPP conocida hasta el nudo p.

Demostración

En efecto, vamos a separar el subproblema defini

do en el nudo p , en dos nuevos subproblemas, añadiendo las restricciones $\{y_\ell \geq 1\}$ y $\{y_\ell = 0\}$.

Denotaremos por $p+1$ el nudo que corresponde al subproblema definido en el nudo p junto a la restricción $\{y_\ell \geq 1\}$ y veremos que la cota inferior en este nudo por lo menos igua la el valor de la cota superior.

Por la proposición 3.7, tenemos:

$$V_p(P_R(u)) = V_p(SST) + \sum_{i \in N_R^o} u_i + IR_p$$

donde por IR_p denotamos la suma de los costes de las aristas de A_R y los costes de todas las aristas, inicialmente de A_S, que han sido fijadas en el árbol de Branch and Bound a estar en la solución, en la rama desde el nudo 0 hasta el nudo p.

En el nudo $p+1$ (fijando la arista ℓ a aparecer en la solución) podemos mantener la asignación de los valores de los multiplicadores u_i dada en el nudo p (siguen siendo posibles en el sentido de que continúan satisfaciendo las condiciones de la proposición 3.7). Y vamos a analizar el peor de los casos para $V_{p+1}(P_R(u))$,que será cuando al fijar la arista ℓ , los vértices incidentes con ella i_ℓ y j_ℓ pasan a ser pares, con lo que los coeficientes de u_{i_ℓ} y de u_{j_ℓ} serán 0.

Entonces,

$$V_{p+1}(P_R(u)) = V_{p+1}(SST) + \sum_{\substack{i \in N_R^o \\ i \neq i_\ell \\ i \neq j_\ell}} u_i + IR_p + c_\ell$$

Obviamente, se cumplirá que: $V_p(SST) \leq V_{p+1}(SST) + \hat{c}$

por definición de $\hat{c}$.

Así pués,

$$V_{p+1}(P_R(u)) \geq IR_p + c_\ell + \sum_{i \in N_R^o} u_i - u_{i_\ell} - u_{j_\ell} + V_p(SST) -$$

$$-\hat{c} = V_p(P_R(u)) + c_\ell - u_{i_\ell} - u_{j_\ell} - \hat{c} = V_p(P_R(u)) + \bar{c}_\ell - \hat{c}$$

y puesto que por hipótesis $\bar{c}_\ell - \hat{c} \geq CSUP - V_p(P_R(u))$, tendremos:

$$V_{p+1}(P_R(u)) \geq CSUP$$

lo que nos asegura que si fijamos la arista ℓ , el nudo correspondiente será saturado; ésto es equivalente a afirmar que en el nudo p se debe cumplir $y_\ell = 0$.

Proposición 5.2

En cualquier nudo p, sea $\ell \in A_S - I$, $i_\ell \in S$ y $j_\ell \in \bar{S} = N_R - S$ y sea $c^* = \min_{\ell' \neq \ell} \{\bar{c}_{\ell'} / \ell' \in (S, \bar{S})\}$

Supondremos $c^* - \bar{c}_\ell \geq 0$. Entonces

si $c^* - \bar{c}_\ell \geq CSUP - CINF_p \rightarrow y_\ell = 1$ (la arista ℓ está en la solución).

Donde $CINF_p$ es el valor de cualquier cota inferior obtenida, en la sección 3, en el nudo p, y $\bar{c}_\ell$ representa el coste de la arista ℓ modificado por los multiplicadores correspondientes a la cota inferior utilizada (u_i ó / y λ_t)

Demostración

Al igual que en la proposición anterior, vamos a separar el subproblema definido en el nudo p. Consideraremos el subproblema definido en el nudo $p+1$ como el que resulta de añadir la restricción $\{y_\ell = 0\}$ al subproblema en el nudo p. Demostraremos que la cota inferior en el nudo $p+1$ por lo menos iguala al valor de la cota superior.

En efecto, en el nudo $p+1$, podemos asignar (como mínimo)los mismos valores a los multiplicadores (u_i ó| y λ_t) que en el nudo p. La suma de los costes de las aristas requeridas, y los de aquellas aristas que han sido fijadas a lo - largo de la rama desde el nudo 0 al nudo p, permanece constante al pasar del nudo p al $p+1$.

Pero en lo referente a $V_p(SST)$ y $V_{p+1}(SST)$, se - tiene que:

$$V_{p+1}(SST) = V_p(SST) + c^* - \bar{c}_\ell$$

ya que la solución en el nudo $p+1$ no puede contener a la arista ℓ y así,para conectar S y $\bar{S}$,el SST debe contener a la arista cuyo coste es c^*, lo cual nos lleva a que:

$$CINF_{p+1} \geq CINF_p + c^* - \bar{c}_\ell \geq CSUP$$

Por lo tanto el nudo $p+1$ sería saturado; lo que es equivalente a afirmar que,para tal arista $\ell \varepsilon A_S - I$, se debe cumplir que $y_\ell = 1$ en el nudo p.

Comentario 5.1

Si en un nudo p, encontramos una arista ℓ que cumpla las condiciones de la proposición 5.1, podemos asegurar - que tal arista no estará en la solución a dicho subproblema - ni en la de sus sucesores. Esto se puede interpretar en la - forma siguiente:si separamos el subproblema definido en el nudo p utilizando la variable y_ℓ, el nudo p sólo tendrá un subproblema sucesor.

Una interpretación similar puede hacerse para la proposición 5.2.

Elección de la variable de ramificación

En el apartado anterior hemos visto que la separación de un subproblema en dos nuevos subproblemas se hace utilizando variables asociadas con las aristas de $A_S - I$.

La eficiencia de un algoritmo de Branch and Bound depende, en gran medida, de la estrategia seguida para elegir la variable sobre la que vamos a realizar la separación en - subproblemas en cada nudo del árbol. Dichas variables reciben el nombre de variables de ramificación ó variables de particionamiento.

En nuestro algoritmo hemos probado dos estrategias de elección de la variable de ramificación:
El principal objetivo de ambas estrategias es el de escoger - un conjunto de aristas $L \subseteq A_S - I$, durante el proceso de particionamiento en subproblemas, de forma que obtengamos un grafo conexo, inducido por el conjunto de aristas $A_R \cup L$, tan pronto como sea posible.

(i) En la primera estrategia la variable de ramificación corresponde a la arista de mayor coste en la solución al problema del árbol generador de mínimo peso en el grafo $\tilde{G}_C$ correspondiente a ese nudo.

(ii) En la segunda estrategia, intentamos que todos los vértices del grafo $\tilde{G}_C$ tengan grado 2 (considerando únicamente las aristas en L). Entre todas las aristas candidatas, escogemos, como segundo criterio, aquella arista que cuando se añade a $A_R \cup L$ satisface las restricciones de paridad, del tipo (1'), para tantos vértices de $\tilde{G}_C$ como sea posible.

La estrategia (i) se basa primero en conseguir la conectividad del grafo y entonces en el coste de las aristas; mientras que la estrategia (ii) se basa primero en conseguir la conectividad y entonces en satisfacer tantos requerimientos de paridad, en las componentes y en los vértices, como sea posible.

Ilustramos las estrategias (i) y (ii) con el siguiente ejemplo:

Ejemplo 5.1

Dado el grafo $\tilde{G}_C$ de la figura 5.a correspondiente al problema (original) definido en el nudo 0 del árbol de Branch and Bound

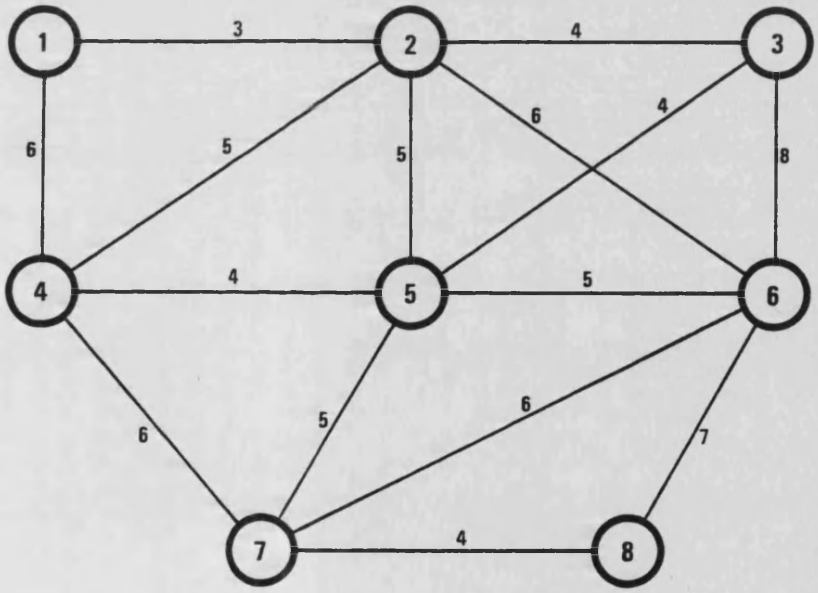


Figura 5.a: grafo $\tilde{G}_C$

Una posible aplicación de la estrategia (i) nos llevará a la elección de las variables de ramificación, que en el nudo del árbol - donde han sido todas fijadas, formarían el grafo de la figura 5.b

Observamos que en el grafo de la figura 5.b hay 4 vértices incidentes con una única arista (en la solución deben ser incidentes con al

menos dos) y un vértice que es incidente con 4 aristas (con - dos sería suficiente para semejarse a un circuito). Intuitivamente, por lo tanto, cabe suponer que esta estructura estará - bastante alejada de una solución óptima al RPP en G_C.

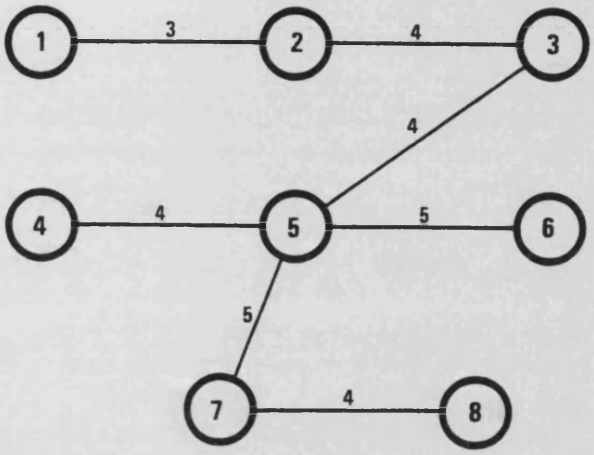


Figura 5.b: Aplicación de la estrategia (i) a $\tilde{G}_C$

Estas desventajas desaparecen utilizando la estrategia (ii) que nos muestra la figura 5.c

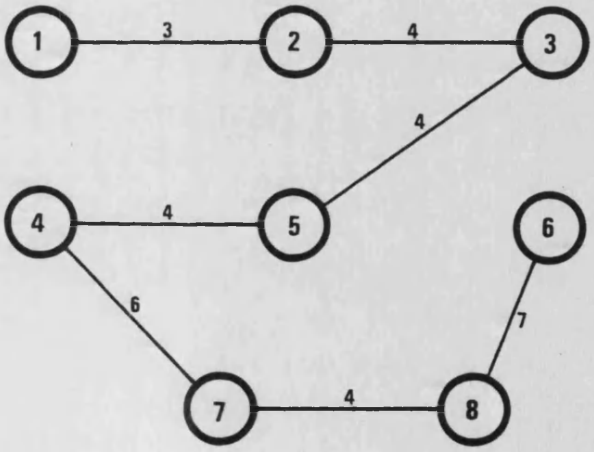


Figura 5.c: Estrategia (ii) aplicada a $\tilde{G}_C$

Donde podemos observar que todos los vértices, - salvo dos, son incidentes con exactamente dos aristas. Esta estructura es mucho más parecida a lo que, intuitivamente, pensamos será un circuito solución al RPP. De hecho, esta estrategia

permite encontrar rapidamente la solución óptima al problema con lo que se favorece la saturación de nudos mediante la cota inferior.

En la práctica computacional, la estrategia (ii) se ha demostrado mucho más eficiente que la estrategia (i). El tiempo empleado en la solución de los problemas utilizando la estrategia (ii) ha sido prácticamente la octava parte del empleado utilizando la primera estrategia.

Finitud del algoritmo de Branch and Bound

La finitud de nuestro algoritmo de Branch and Bound está garantizada por la estrategia de separación utilizada.

Puesto que la separación en subproblemas se realiza utilizando las aristas del conjunto A_S-I, y $|A_S- I|<\infty$, cualquier rama del árbol de enumeración no tendrá más de $|A_S-I|$ nudos.

Otro razonamiento análogo al que se utiliza en los métodos para problemas generales de PLE es el siguiente: Las variables y_ℓ cumplen que $0\leq y_\ell\leq 2 \quad \forall \ell\in A_S-I$, entonces el número total de posibles valores para las variables y_ℓ, $\ell\in A_S-I$, vendrá acotado por:

$$U(A_S-I) = \prod_{1}^{|A_S-I|} (1+2) = 3^{|A_S-I|}$$

puesto que la estructura del árbol es un particionamiento sucesivo del conjunto

$$Y(A_S-I) = \{y_\ell \mid 0\leq y_\ell\leq 2 \quad , \ell\in A_S-I\}$$

tenemos que el algoritmo no puede examinar más nudos que el

número dado por todos los elementos $U(A_S-I)$ del conjunto $V(A_S-I)$.

5.2 ESTRATEGIA DE RAMIFICACION

En cualquier etapa del proceso de enumeración, el conjunto de nudos que no han sido saturados ni particionados en subproblemas, constituye el conjunto de los llamados vértices vivos. El proceso acaba cuando dicho conjunto es vacío. LLamamos estrategia de ramificación a la regla de selección de los nudos (elección del nudo no saturado a investigar).

Existen varias estrategias de ramificación, las dos más usuales son la LLB (Least Lower Bound) y la LIFO (Last In First Out).

La estrategia LLB consiste en investigar el nudo vivo con menor valor de la cota inferior. Este criterio posee dos desventajas importantes: no garantiza que encontremos pronto una buena solución posible, y "salta" de un nudo a otro del árbol con el consiguiente problema de almacenamiento de los datos de los nudos vivos.

La estrategia LIFO consiste en pasar a investigar un nudo sucesor del que se está investigando si éste no se satura. Aunque puede encontrar un buen número de soluciones posibles de coste mayor a la mejor conocida y por lo tanto inútiles, esta estrategia carece de las desventajas anteriormente mencionadas para la estrategia LLB.

En nuestro trabajo hemos utilizado la estrategia LIFO. Esta nos proporciona, junto a la elección de la variable de ramificación descrita en el apartado anterior, un método muy útil para encontrar soluciones posibles rápidamente y

por lo tanto para saturar nudos. Esto viene reflejado en la siguiente proposición:

Proposición 5.3

En cualquier nudo, cuando han sido fijadas $k-1$ aristas por el árbol de enumeración, encontramos una solución posible al RPP y por lo tanto el nudo será saturado.

Demostración

En efecto, consideramos el grafo G_M cuyo conjunto de aristas está formado únicamente por las $k-1$ aristas fijadas y por el conjunto de aristas requeridas A_R.

Este grafo es ahora conexo, debido a la elección de las variables de ramificación.

Sea N_M^o el conjunto de todos los vértices de grado impar en G_M y sea $<N^o_M>$ el grafo inducido en G_C por el conjunto N_M^o. Si resolvemos un problema de 1-matching mínimo en $<N_M^o>$,obtendremos un grafo conexo en donde todos los vértices son pares y que contiene a todas las aristas de A_R. Dicho grafo será una solución posible al RPP. El nudo será, pués, saturado.

Ejemplo 5.2

Veamos la aplicación de la estrategia LIFO y la proposición anterior en un grafo con las componentes en la figura siguiente 5.d, donde los nudos subrayados indican que han sido saturados. En este caso han sido saturados al hallar en ellos soluciones posibles del RPP,ya que tanto el nudo $k-1$ como el $k+1$,co

rresponden a nudos en los que el procedimiento de enumeración ha fijado ya $k-1$ aristas yalos que se puede aplicar la proposición 5.3

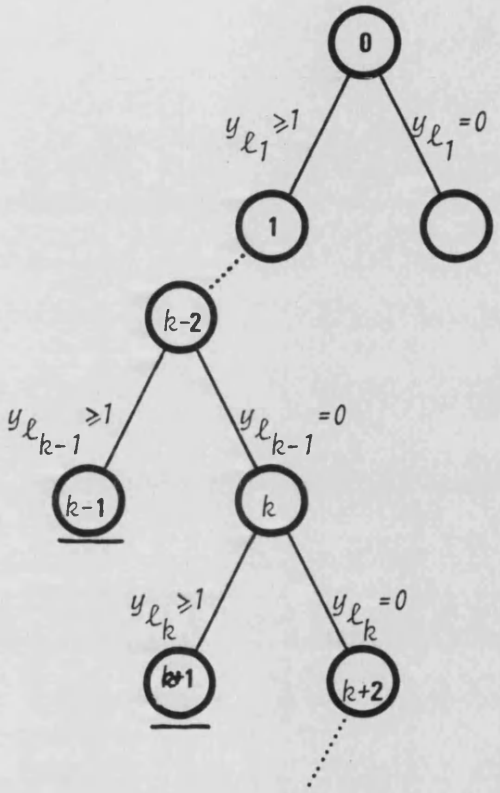


Figura 5.d: Estrategia LIFO

Comentario 5.2

La utilización en nuestro algoritmo de la estrategia de ramificación descrita anteriormente, proporciona la ventaja adicional de no tener que calcular en cada nudo p del árbol los multiplicadores $u_i, i \in N_R$, para obtener el valor de la cota inferior. Podemos obtenerlos reactualizando los valores obtenidos para dichos multiplicadores en el nudo precedente, obteniendo así un considerable ahorro de cálculo y por lo tanto de tiempo de computación.

5.3 SATURACION DE NUDOS

En nuestro algoritmo específico de Branch and - Bound, al igual que en el caso general aplicado a cualquier problema de PLE, la saturación de los nudos del árbol de enumeración viene dada cuando se cumple una de las tres condiciones siguientes:

a) Al resolver el subproblema en un nudo p, obtenemos una solución óptima a dicho subproblema y por lo tanto una solución posible para el problema original.

b) El subproblema definido en el nudo p es imposible.

c) La cota inferior obtenida en un nudo p tiene un valor al - menos igual al de la mejor solución posible conocida.

El caso a) se produce cuando, utilizando las estrategias descritas de elección de la variable a ramificar y de- elección del nudo a investigar, se satisfacen las condiciones de la Proposición 5.3. Al encontrar una solución posible actualizamos el valor de la cota superior CSUP como el mínimo entre el valor proporcionado por el heurístico descrito en la Sección 4 mejorado y los valores correspondientes a las soluciones posibles halladas durante el proceso de enumeración. De ahí la - importancia de la elección de las estrategias citadas.

El caso b) ocurre cuando durante el proceso de ramificación hemos eliminado de la solución todas las aristas incidentes con una componente o todas las pertenecientes a algún conjunto de aristas de corte. En estos casos, obviamente, el - subproblema definido en dicho nudo será imposible.

El caso c) refleja la importancia que tiene tanto el conseguir un valor lo mayor posible para la cota inferior, como el que el valor inicial de la cota superior sea lo más ajustado posible al valor de la solución óptima.

En muchos casos es innecesario el cálculo de una cota inferior en un determinado nudo. Esto viene reflejado en las siguientes proposiciones:

Notación 5.1

Representaremos por $CINF_p$ y $CINF_{p+1}$ los valores de la cota inferior obtenida en los nudos p y $p+1$ respectivamente.

Designaremos por L_p el subconjunto de aristas de A_S-I que han sido fijadas (a aparecer en la solución) en el camino desde el nudo 0 al nudo p en el árbol de enumeración. G_p será el grafo inducido por el conjunto de aristas $A_R \cup L_p$ y N_p^o será el subconjunto de N_R de vértices impares en G_p

Llamaremos *cola* a cualquier nudo del árbol de B&B donde hayan sido fijadas $k-1$ aristas y, por lo tanto, donde encontramos una solución posible al RPP.

Proposición 5.4

Sea el nudo $p+1$ el sucesor del nudo p al fijar la arista $\ell'=(i',j') \in A_S-I$. Entonces:

$$CINF_{p+1} \geq CINF_p + \mathcal{D}$$

donde $\mathcal{D}$ viene dado por la siguiente expresión:

$$\mathcal{D}=\begin{cases} c_{\ell'}+u_{p+1}(i')+u_{p+1}(j') & \text{si } i' \text{ y } j' \text{ son impares en } G_{p+1} \\ c_{\ell'}-u_p(i')-u_p(j') & \text{si } i' \text{ y } j' \text{ son pares en } G_{p+1} \\ c_{\ell'}+u_{p+1}(i')-u_p(j') & \text{si } i' \text{ es impar y } j' \text{ par en } G_{p+1} \end{cases}$$

y donde: $u_{p+1}(i) = u_p(i) \qquad i \neq i', j'$

$$u_{p+1}(i) = u_p(i) + \min_{j \neq i} \{\bar{c}_{ij}\} \qquad i = i', j'$$

Demostración:

En el nudo p, el valor de la cota inferior vendrá dado por:

$$CINF_p = \sum_{\ell \in A_R \cup L_p} c_\ell + \sum_{i \in N^o_p} u_p(i) + V_p(SST) + 2 \sum_{t \in K^p_t} \lambda_t$$

Al fijar la arista $\ell' = (i', j')$, en el nudo $p+1$, la cota inferior será:

$$CINF_{p+1} = \sum_{\ell \in A_R \cup L_{p+1}} c_\ell + \sum_{i \in N^o_{p+1}} u_{p+1}(i) + V_{p+1}(SST) + 2 \sum_{t \in K^{p+1}_t} \lambda_t$$

donde $L_{p+1} = L_p \cup \{\ell'\}$.

Obviamente, se cumplirá que :

$$V_{p+1}(SST) + 2 \sum_{t \in K^{p+1}_t} \lambda_t \leq V_p(SST) + 2 \sum_{t \in K^p_t} \lambda_t$$

con lo que

$$CINF_{p+1} - CINF_p \leq c_{\ell'} + \sum_{i \in N^o_{p+1}} u_{p+1}(i) - \sum_{i \in N^o_p} u_p(i)$$

Distinguiremos tres casos:

a) $i' \in N^o_{p+1}$ y $j' \in N^o_{p+1}$ (ambos son impares en G_{p+1})

entonces: $i', j' \notin N^o_p \rightarrow N^o_{p+1} = N^o_p \cup \{i'\} \cup \{j'\}$ con lo que

$$CINF_{p+1} - CINF_p \leq c_{\ell'} + \sum_{i \in N^o_p} u_{p+1}(i) + u_{p+1}(i') + u_{p+1}(j') - \sum_{i \in N^o_p} u_p(i) =$$

$$= c_{\ell'} + u_{p+1}(i') + u_{p+1}(j')$$

(la última igualdad se cumple por definición de u_{p+1})

b) $i' \notin N^o_{p+1}$ y $j' \notin N^o_{p+1}$ (ambos son pares en G_{p+1})

entonces: $i', j' \in N^o_p \rightarrow N^o_p = N^o_{p+1} \cup \{i'\} \cup \{j'\}$ luego

$$CINF_{p+1} - CINF_p \leq c_{\ell'} + \sum_{i \in N^o_{p+1}} u_{p+1}(i) - \sum_{i \in N^o_{p+1}} u_p(i) - u_p(i') - u_p(j') =$$

$$= c_{\ell'} - u_p(i') - u_p(j')$$

c) $i' \in N^o_{p+1}$ y $j' \notin N^o_{p+1}$

entonces: $i' \notin N^o_p$, $j' \in N^o_p$ $\rightarrow$ $N^o_{p+1} \cup \{j'\} = N^o_p \cup \{i'\}$ luego

$$CINF_{p+1} - CINF_p \leq c_{\ell'} + \sum_{i \in N^o_{p+1}} u_{p+1}(i) - \sum_{i \in N^o_{p+1}} u_p(i) - u_p(j') +$$

$$+ u_{p+1}(i') = c_{\ell'} + u_{p+1}(i') - u_p(j')$$

Así, pues, queda demostrada la proposición.

Proposición 5.5

Sea el nudo $p+1$ una cola. Sea el nudo $p+2$ un sucesor del nudo p al imponer la restricción $\{y_{\ell'} = 0, \ell' = (i', j') \in A_S - I\}$

Se cumple entonces que:

$$CINF_{p+2} \geq CINF_p + \mathcal{D}(i')\left[u_{p+2}(i') - u_p(i')\right] + \mathcal{D}(j')\left[u_{p+2}(j') - u_p(j')\right]$$

donde $\mathcal{D}(i) = 1$ si el vértice i es impar en G_{p+2} y $\mathcal{D}(i) = 0$ en otro caso, y donde, al igual que en la proposición anterior, los u_{p+2} se definen como los u_{p+1} en función de los u_p

Demostración:

Distinguiremos dos casos:

a) $u_{p+2}(i') = u_p(i')$ y $u_{p+2}(j') = u_p(j')$

En este caso la demostración es evidente puesto que el conjunto de soluciones posibles del problema definido en el nudo $p+2$, S_{p+2}, satisface:

$$S_{p+2} \subseteq S_p \quad \rightarrow \quad CINF_{p+2} \geq CINF_p$$

b) $u_{p+2}(i') - u_p(i') > 0$ o/y $u_{p+2}(j') - u_p(j') > 0$

Por la definición de los multiplicadores $u_p(i)$ y $u_{p+2}(i)$, tenemos que el coste modificado de la arista ℓ' en el nudo p debía ser 0 ($\bar{c}_{\ell'} = 0$). Entonces:

$$CINF_p = \sum_{\ell \in A_R \cup L_p} c_\ell + \sum_{i \in N_p^o} u_p(i) + V_p(SST) + 2\lambda_p =$$

$$= \sum_{\ell \in A_R \cup L_p} c_\ell + \sum_{i \in N_p^o} u_p(i)$$

ya que por la estrategia de ramificación, y teniendo en cuenta la hipótesis de que el nudo $p+1$ (al fijar la arista ℓ') es una cola, tenemos que en el nudo p únicamente había dos componentes conexas S y $\bar{S}$, y de forma que $\ell' = (i', j') \in \varepsilon(S, \bar{S})$, con $\bar{c}_{\ell'} = 0$. Por lo tanto $V_p(SST) = 0$ y por definición de $\lambda_p = \min \{\bar{c}_{ij} \mid i \in S, j \in \bar{S}\} = 0$.

Por otra parte, en el nudo $p+2$, se cumple:

$$L_{p+2} = L_p \quad \text{y} \quad N^o_{p+2} = N^o_p$$

ya que no hemos fijado ninguna arista a estar en la solución, sino que hemos obligado a la arista ℓ' a no aparecer en la solución.

Así, tenemos:

$$CINF_{p+2} = \sum_{\ell \in A_R \cup L_p} c_\ell + \sum_{i \in N_p^o} u_{p+2}(i) + V_{p+2}(SST) + 2\lambda_{p+2} \geq$$

$$\geq CINF_p + \sum_{i \in N_p^o} u_{p+2}(i) - \sum_{i \in N_p^o} u_p(i)$$

y puesto que $u_{p+2}(i) = u_p(i) \quad \forall i \neq i', j'$, tenemos

$$CINF_{p+2} \geq CINF_p + D(i') \left[u_{p+2}(i') - u_p(i')\right] + D(j') \left[u_{p+2}(j') - u_p(j')\right]$$

Comentario 5.3

Las proposiciones 5.4 y 5.5 nos ofrecen la posibilidad de no calcular una cota inferior cuando fijamos una arista, o bien la certeza de que en algunos nudos vamos a saturar sin necesidad de calcular la cota inferior.

En el caso de la Proposición 5.4, cuando, al fijar un arista y pasar de un nudo p a un sucesor $p+1$, calculamos el valor de $\mathcal{D}$ y añadido a la cota inferior en el nudo p no supera, o iguala, el valor de la cota superior, tenemos la certeza de que el cálculo de una cota inferior en el nudo $p+1$ es totalmente improductivo (recordemos que $\mathcal{D}$ es el incremento máximo que podía alcanzar).

La proposición 5.5 proporciona el incremento mínimo que va a tener la cota inferior en el nudo $p+2$ respecto a la obtenida en el nudo p. Si el valor obtenido para la cota inferior en el nudo p junto a dicho incremento superan, o igualan, el valor de la cota superior, el nudo $p+2$ es saturado sin necesidad de calcular la cota inferior en dicho nudo.

S E C C I O N 6

RESULTADOS COMPUTACIONALES

A partir de los resultados obtenidos en las Secciones 2,3,4 y 5, hemos construido un algoritmo de Branch and Bound que obtiene la solución exacta al Problema del Cartero Rural en un grafo no dirigido.

En esta Sección, en primer lugar, veremos un ejemplo que ilustrará el desarrollo completo del funcionamiento de nuestro algoritmo.

En segundo lugar mostraremos los datos que hemos obtenido como aplicación de nuestro algoritmo a una colección de 24 problemas de diversos tamaños y características. Compararemos dichos resultados a los obtenidos utilizando el Método del Subgradiente en algunos nudos del árbol de Branch and Bound, para intentar mejorar el valor de la Cota Inferior,y así acortar el proceso de enumeración.

Por último, comentaremos los resultados obtenidos y haremos una breve valoración de los mismos.

6.1 APLICACION DEL ALGORITMO DE BRANCH AND BOUND A UN PROBLEMA CONCRETO. EJEMPLO

Vamos a aplicar el algoritmo de solución del RPP al problema número 9 de la tabla 6.1. En la figura 6.a está representado el grafo original G, donde las aristas requeridas están dibujadas con trazo contínuo.

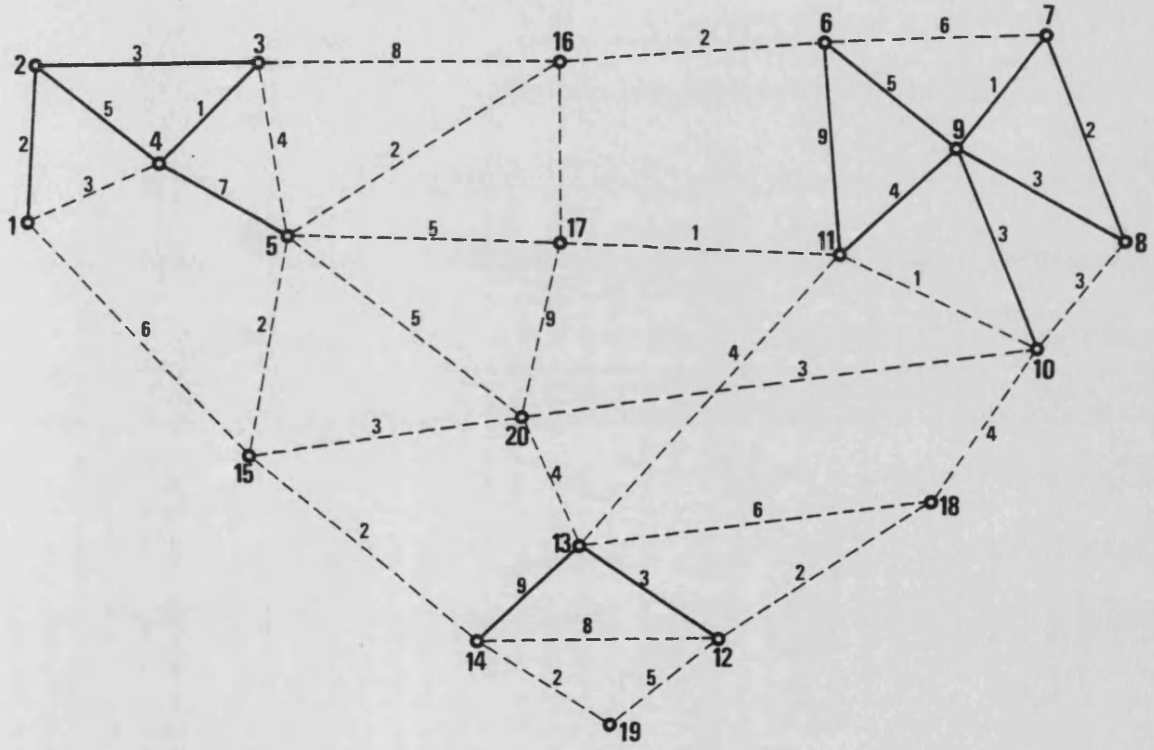


Figura 6.a: El grafo original

Los números que aparecen junto a las aristas representan el coste de atravesar dicha arista (distancia,tiempo, coste monetario,etc.), mientras que los otros representan a los vértices del grafo.

En la figura 6.b representamos el árbol completo de Branch and Bound necesario para resolver el RPP en este caso. A la derecha de cada nudo, tenemos la cota inferior corres

pondiente a ese nudo si fué calculada. El par de números encima de cada rama indica la arista fijada o excluida en la ramificación. Así, por ejemplo, (5,6) indica que la arista (5,6) debe aparecer al menos una vez en la solución al RPP, y $\overline{(5,6)}$ que la arista no puede aparecer en el circuito solución.

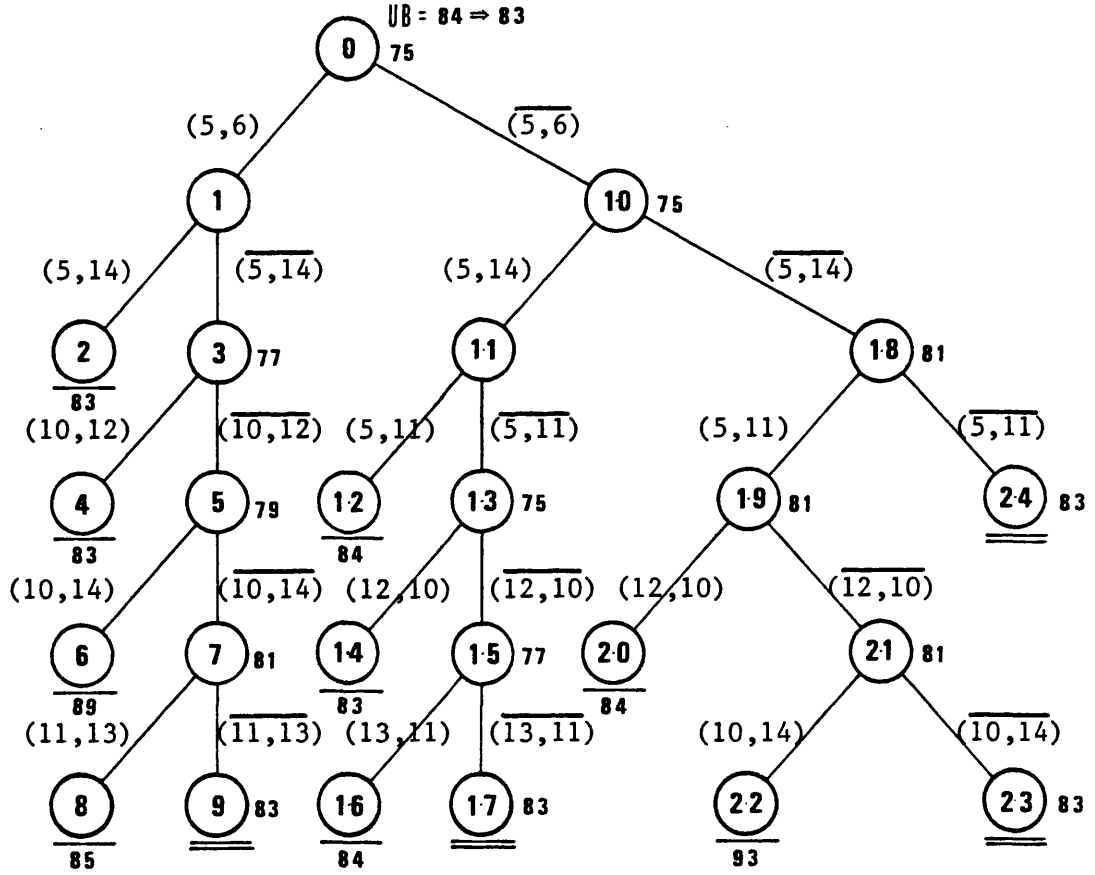


Figura 6.b: Arbol de B&B del ejemplo

En el nudo 0, correspondiente al problema original, aparece otro número que indicará el valor de la cota superior obtenida por el algoritmo heurístico (y mejorado), tal como fué descrito en la Sección 4. Esta cota irá actualizándose a medida que vamos encontrando soluciones posibles al RPP

con coste menor. Así, cuando el proceso de enumeración finalice, éste valor corresponderá al de la solución óptima del problema.

Los nudos que aparecen subrayados una sola vez, indican que han sido saturados por la obtención en ellos de una solución posible al RPP , tal como se establece en la Proposición 5.3. El valor que aparece bajo dichos nudos corresponde al valor de la solución posible hallada. Aquellos nudos que aparecen subrayados dos veces indican que han sido saturados por la cota inferior.

Podemos observar que no hay una única solución óptima al problema, por ejemplo, las soluciones obtenidas para los subproblemas definidos en los nudos 2,4 y 14 son todas ellas óptimas. La figura 6.c representa una solución óptima al RPP en este problema.

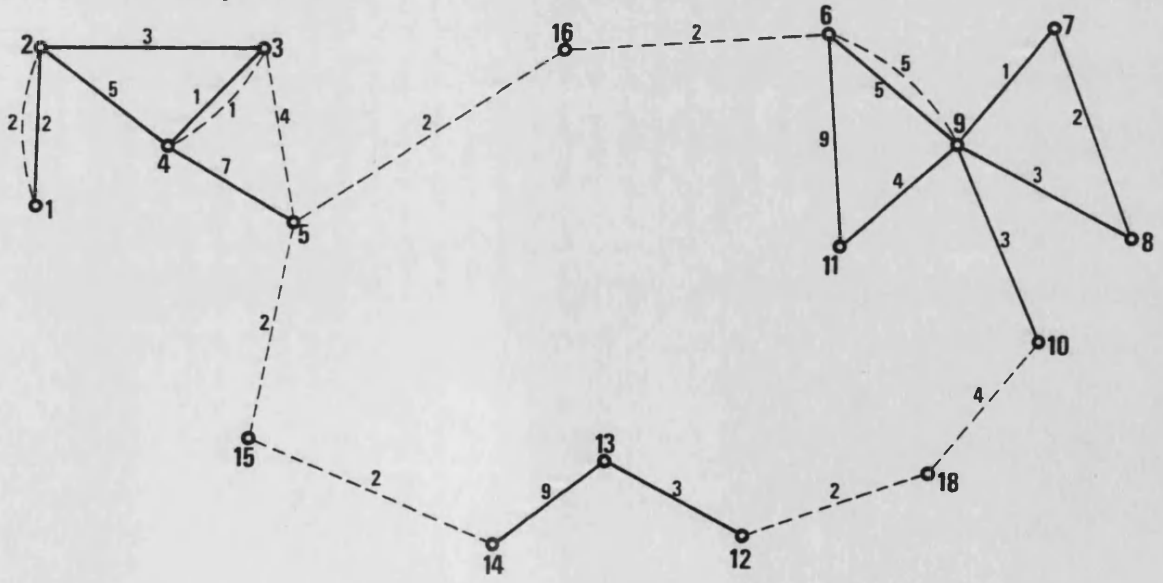


Figura 6.c: Una solución óptima

6.2 RESULTADOS COMPUTACIONALES

El algoritmo descrito para el Problema del Cartero Rural ha sido probado en 24 problemas de diversos tamaños, variando hasta 84 vértices, 180 aristas, 74 aristas requeridas y 8 componentes conexas. Los costes de las aristas son números enteros comprendidos entre 0 y 20.

El algoritmo fué programado en FORTRAN ASCII, se utilizó el compilador FTN, ocupaba aproximadamente 45 K's de memoria y se probó en un ordenador UNIVAC 1100/60. Todos los tiempos de computación dados en las tablas son segundos del UNIVAC 1100/60.

En la tabla 6.1 se dan las características de todos los problemas estudiados: n°de componentes, n°de vértices y aristas en el grafo original $G=(N,A)$, y n°de vértices, aristas requeridas y aristas no requeridas en el grafo simplificado $G_C=(N_R, A_R \cup A_S)$.

En la tabla 6.2 figuran los resultados obtenidos para los problemas de la tabla 6.1: valor de la cota superior, óptimo, valor de la cota inferior y tiempo de CPU total (el utilizado para realizar las transformaciones sobre los grafos y el utilizado por el algoritmo de Branch and Bound). Por último se dan el número total de nudos del árbol de enumeración y el número de nudos saturados por la cota inferior únicamente (no están contabilizados los nudos saturados al encontrar una solución posible ni los nudos cuyo subproblema resulta ser imposible).

1	2	3	4	5	6	7
1	4	17	11	21	7	6
2	4	22	14	35	12	24
3	4	31	28	58	26	31
4	3	20	17	43	22	13
5	5	36	20	56	16	20
6	7	24	24	49	20	26
7	3	28	23	51	24	23
8	2	25	17	49	24	16
9	3	20	14	40	14	15
10	4	15	12	24	10	11
11	3	11	9	18	7	7
12	3	9	7	15	5	15
13	3	10	7	13	4	6
14	6	40	28	75	31	52
15	8	40	26	51	19	18
16	7	42	31	87	34	57
17	5	25	19	47	17	27
18	8	28	23	40	16	15
19	7	50	33	73	29	27
20	7	72	50	125	63	35
21	6	71	49	139	67	43
22	6	84	50	180	74	108
23	6	78	50	184	78	81
24	7	53	41	110	55	70

1- Número de Problema

2- Número de Componentes Conexas

3- Número total de vértices en el grafo original G

4- Número de vértices del grafo simplificado G_C

5- Número total de aristas en el grafo original G

6- Número de aristas requeridas

7- Número de aristas no requeridas en el grafo G_C

Tabla 6.1: Características de los Problemas

1	2	3	4	5	6	7
1	76	76	68	0.219	13	4
2	164	163	153	0.615	57	14
3	102	102	101	1.817	17	5
4	84	84	78	0.612	31	4
5	135	129	121	1.58	79	19
6	107	102	92	5.14	837	162
7	130	130	125	1.357	33	6
8	122	122	118	0.689	9	1
9	84	83	75	0.463	25	4
10	80	80	71	0.474	37	8
11	23	23	23	0.159	1	1
12	22	21	21	0.157	17	3
13	38	38	37	0.125	5	2
14	212	209	193	8.12	857	150
15	445	445	399	4.65	299	48
16	203	203	193	44.65	4293	1053
17	116	112	99	2.61	401	52
18	148	148	139	2.58	227	37
19	280	263	246	4.88	179	45
20	400	398	373	28.24	1209	134
21	372	366	341	207.61	8923	1135
22	632	621	592	300.71	14791	1120
23	480	480	465	174.50	8537	870
24	411	405	388	82.09	5751	850

1- Número de Problema
2- Cota Superior
3- Valor óptimo del RPP
4- Cota Inferior
5- Tiempo total de resolución
6- Nudos del árbol de Branch and Bound
7- Nudos saturados por la cota inferior

Tabla 6.2: Resultados Computacionales

Finalmente, en la tabla 6.3, comparamos los resultados anteriores con los obtenidos utilizando el Método del Subgradiente para incrementar el valor de la cota inferior, proporcionada por la solución de $(P_R(u))$, y así acortar el proceso de enumeración del Branch and Bound. Realizamos 20 iteraciones del Método del Subgradiente en el nudo 0 del árbol y en todos aquellos nudos donde el valor de la cota inferior superaba el 95% del valor de la mejor solución posible obtenida. El número de iteraciones permitido aumentaba hasta un máximo de 50 si, durante las 20 primeras iteraciones, el valor obtenido para la cota inferior se encontraba a menos de un 2% del óptimo local.

Como puede observarse en los resultados de la Tabla 6.3, el proceso de enumeración se acortó en algunos casos pero a costa de un mayor tiempo de computación. Esta diferencia de tiempos de resolución crece con el tamaño de los problemas. Así, por ejemplo, en un problema relativamente pequeño como es el problema 17, el tiempo empleado para resolverlo utilizando el Método del Subgradiente fué superior a los 60 segundos frente a los 2.61 segundos empleados para resolverlo sin su utilización. Estos resultados demuestran, experimentalmente, la inefectividad de la utilización del Método del Subgradiente para la solución del RPP.

	UTILIZANDO M.SUBGRAD.			SIN UTILIZAR M.SUBGRA.		
	1	2	3	1	2	3
1	1.862	13	4	0.219	13	4
2	18.960	53	12	0.615	57	14
3	13.667	13	4	1.817	17	5
4	5.630	31	8	0.612	31	4
7	9.597	19	4	1.357	33	6
9	7.074	25	4	0.463	25	4
10	9.181	29	8	0.474	37	8
12	0.826	15	3	0.157	17	3
13	0.682	1	1	0.125	5	2
17	> 60.			2.610	401	52

1- Tiempo total de resolución (segundos)

2- Nudos del árbol de Branch and Bound

3- Nudos saturados por la Cota Inferior

Tabla 6.3: Comparación de resultados

6.3 COMENTARIOS Y VALORACION

El problema del Cartero Rural (RPP) es un problema de Optimización Combinatorial, que, como ya señalamos, pertenece a la clase *NPC* (problemas NP-completos). El hecho de que el RPP sea un problema NP-completo nos proporciona una idea acerca de la dificultad que entraña su resolución, puesto que, a menos que *NP=P*, no existe ningún algoritmo acotado polinomialmente que lo resuelva (si existiera, todos los problemas de *NP-P* podrían ser resueltos también por algoritmos acotados polinomialmente). Para ilustrar lo que se suele denominar la "explosión combinatorial", veamos el siguiente ejemplo de V.Klee (1980):

"Si un computador necesita únicamente 10^{-10} segundos para investigar un determinado árbol generador de un grafo completo, el tiempo necesario para investigar todos (n^{n-2}) los árboles generadores sería: 0.01 segundos para n=10 (n es el número de vértices del grafo), 54 horas para n=15, $8.3*10^{5}$ años para n=20 y $4.5*10^{14}$ años para n=25".

Esto explica por qué un algoritmo de enumeración es útil únicamente cuando trabajamos con problemas muy pequeños y que el objetivo de la Optimización Combinatorial sea el encontrar algoritmos que sean útiles para problemas grandes.

Intentando participar de éste objetivo de la Optimización Combinatorial, nosotros hemos realizado el estudio, objeto de esta memoria, de un problema concreto, el de encontrar un circuito que atraviese cada arista de un subconjunto

dado incurriendo en el mínimo coste, y hemos diseñado un algoritmo específico para su resolución basándonos en dicho estudio. Los resultados son, creemos, positivos en dos aspectos fundamentalmente: en primer lugar es la primera vez que que éste problema aparece formulado como un PLE y es objeto de estudio particular, al mismo tiempo que se ofrece un método de solución exacta; en segundo lugar, los tiempos de resolución son buenos teniendo en cuenta los tamaños de los problemas resueltos, tiempos que permiten pensar en una posible aplicación práctica a problemas tales como el de recogida de basuras, limpieza de calles,etc. Además, sería posible convertir éste algoritmo exacto en un algoritmo ε-aproximado que nos proporcionaría, en un tiempo de computación sensiblemente inferior,una solución ε-óptima y por lo tanto sería capaz de resolver con la precisión deseada problemas de tamaño superior a los aquí resueltos de forma óptima.

BIBLIOGRAFIA

BALAS,E. (1980)

Cutting planes from conditional bounds: a new approach to Set Covering.

Math. Programming Study 12 (1980) 279-312

BALAS,E. and N.CHRISTOFIDES (1981)

A restricted Lagrangean approach to the Travelling Salesman Problem.

Math. Programming 21 (1981) 19-46

BALINSKI,M.L. Ed. (1974)

Approaches to Integer Programming

Math. Programming Study 2 (1974)

BELLMAN,R. and K.L.COOKE (1969)

The Konigsberg bridges problem generalized.

Jl. of Math.Anal.and Appl. 25 (1969) 1

BELTRAMI,E.J. and L.D.BODIN (1974)

Networks and vehicle routing for municipal waste collection

Networks 4 (1974) 65-94

BERGE,C. (1962)

The Theory of Graphs and its Applications

John Wiley, New York (1962)

BONDY,J. and V.MURTY (1976)

Graph Theory with Applications

Mac Millan Press, London (1976)

BUSACKER,R.G. and T.L.SAATY (1965)

Finite Graphs and Networks

McGraw Hill, New York (1965)

CHRISTOFIDES,N. (1972)

Bounds for the Travelling Salesman Problem.

Operations Research 20 (1972) 1044-1055

CHRISTOFIDES,N. (1973)

The optimal traversal of a graph.

Omega 1 (1973) p.1

CHRISTOFIDES,N. (1975)

Graph Theory. An Algorithmic Approach

Academic Press, New York (1975)

CHRISTOFIDES,N. (1976)

Worst case analysis of a new heuristic for the Travelling Salesman Problem.

Tech. Report, Grad. School Industrial Admin., Carnegie Mellon University

CHRISTOFIDES,N. (1979)

The Travelling Salesman Problem,in *Combinatorial Optimization* (N.Christofides et al. Eds.)

John Wiley (1979)

CHRISTOFIDES,N.,A.MINGOZZI and P. TOTH (1979)

The Vehicle Routing Problem, in *Combinatorial Optimization* (N.Christofides et al. Eds.)

John Wiley (1979)

CHRISTOFIDES,N.,A.MINGOZZI and P.TOTH (1981)

Exact Algorithms for the Vehicle Routing Problem, based on Spanning Tree and Shortest Path Relaxations.

Math. Programming 20 (1981) 255-282

COBHAM,A. (1964)

The intrinsic computational difficulty of functions, in *Proc. 1964 International Congress for Logic Methodology and Philosophy of Science*

North Holland (1964)

COOK,S. (1971)

The Complexity of Theorem-Proving Procedures.

Proc.3rd Ann. A.C.M. Symp. on Theory of Computing Assoc. for Computing Machinery, New York 151-158

CORNUEJOLS,G. and G.L.NEMHAUSSER (1978)

Tight Bounds for Christofides' Travelling Salesman Heuristic.

Math. Programming 14 (1978) 116-121

DEMYANOV,V.F. (1966)

Algorithms for some Minimax Problems.

J.Comp.Syst.Sci. 2 (1966) 342-380

DREYFUS,S.E. and A.LAW (1978)

The Art and Theory of Dynamic Programming

Academic Press, New York (1978)

EDMONDS,J. (1965)

The Chinese Postman's Problem

Bulletin of the O.R.Soc.of America 13 (1965) Supplement 1 ,pB-73

EDMONDS,J. (1965b)

Maximum Matching and a Polyhedron with (0,1) Vertices.

J.Res.Nat.Bur.Stand. 69B (1965) 125-130

EDMONDS,J. and E.JOHNSON (1973)

Matching, Euler Tours and the Chinese Postman's Problem.

Math. Programming 5 (1973) 88-124

FISHER,M (1980)

Worst Case Analysis of Heuristic Algorithms.

Management Science 26 (1980) 1-17

FISHER,M. (1981)

Lagrangian Relaxation Methods for Solving Integer Programming.

Management Science 27 (1981) 1-18

FORD,L.R. and D.R.FULKERSON (1962)

Flows in Networks

Princeton University Press (1962)

FREDERICKSON,G.N. (1979)

Aproximation Algorithms for some Postman Problems.

Journal A.C.M. 26 (1979) 538-554

GAREY,M.R. and D.S.JOHNSON (1979)

Computers and Intractability: A guide to the Theory of NP-completeness

Freeman and Co., San Francisco (1979)

GARFINKEL,R. and G.NEMHAUSER (1972)

Integer Programming

John Wiley, New York (1972)

GARFINKEL,R. (1979)

Branch and Bound Methods for Integer Programming in *Combinatorial Optimization* (N.Christofides et al. Eds.)

John Wiley (1979)

GEOFFRION,A. (1974)

Lagrangean Relaxation for Integer Programming.

Math. Programming Study 2 (1974) 82-114

GOFFIN,J.L. (1977)

On the Convergence Rates of Subgradient Optimization Methods.

Math. Programming 13 (1977) 329-347

GOLDEN,B. (1976)

Shortest Path Algorithms: a comparison.

Operations Research 24 (1976) 1164-1168

GOLDEN,B.L. and R.T.WONG (1981)

Capacited Arc Routing Problems.

Networks 11 (1981) 305-315

HAMMER,P.L.,E.L.JOHNSON,B.H.KORTE and G.L.NEMHAUSER (1977)

Studies in Integer Programming

North Holland (1977)

HAMMER,P.L., E.L.JOHNSON and B.H.KORTE (1977)

Discrete Optimization I

North Holland (1977)

HAMMER,P.L., E.L.JOHNSON and B.H.KORTE (1977)

Discrete Optimization II

North Holland (1977)

HARARY,F. (1969)

Graph Theory

Addison-Wesley, Reading,Mass. (1969)

HELD,M. and R.M.KARP (1970)

The Travelling Salesman Problem and Minimum Spanning Trees.

Operations Research 18 (1970) 1138-1182

HELD,M. and R.M.KARP (1971)

The Travelling Salesman Problem and Minimum Spaning Trees :Part II.

Math. Programming 1 (1971) 6-26

HELD,M.,P.WOLFE and H.P.CROWDER (1974)

Validation of Subgradient Optimization.

Math. Programming 6 (1974) 62-88

HU,T.C. (1969)

Integer Programming and Network Flow

Addison-Wesley, Reading, Mass. (1969)

JEROSLOW,R. (1979)

The Theory of Cutting Planes , in *Combinatorial Optimization* (N.Christofides et al. Eds.)

John Wiley (1979)

KAPPAUF,Ch.H. and G.J.KOEHLER (1979)

The Mixed Postman Problem.

Discrete Applied Mathematics 1 (1979) 89-103

KARP,R.M. (1975)

On the Computational Complexity of Combinatorial Problems.

Networks 5 (1975) 45-68

KLEE,V. (1980)

Combinatorial Optimization: What is the State of the Art.

Math. of Operations Research 5 (1980) 1-26

KWAN,M.K. (1962)

Graphic Programming using even or odd points.

Chinese Mathematics 1 (1962) p273

LAND,A.H. and A.DOIG (1960)

An Automatic Method for Solving Discrete Programming.

Econometrica 28 (1960) 497-520

LAND,A.H. and S.POWELL (1973)

Fortran Codes for Mathematical Programming

John Wiley, New York (1973)

LAND,A.H. and S.POWELL (1979)

Computer Codes for problems in Integer Programming ,in *Discrete Optimization II* (P.L.Hammer et al. Eds.)

North Holland (1979)

LAWLER,E. (1976)

Combinatorial Optimization : Networks and Matroids

Holt,Rinehart & Winston (1976)

LENSTRA,J.K. and A.H.G.RINOOY KAN (1976)

On General Routing Problems.

Networks 6 (1976) 593-597

LENSTRA,J.K. and A.H.G.RINOOY KAN (1981)

Complexity of Vehicle Routing and Scheduling Problems.

Networks 11 (1981) 221-227

LITTLE,J.D.,K.G.MURTY,D.W.SWEENEY and C.KAREL (1963)

An algorithm for the Travelling Salesman Problem.

Operations Research 11 (1963) 979-989

MALE,J.W. and J.C.LIEBMAN (1978)

Districting and Routing for Solid Waste Collection.

Journal of the Environmental Engineering Div.(1978)

MILIOTIS,P. (1976)

Integer Programming Approach to the Travelling Salesman Problem.

Math. Programming 10 (1976) 367-378

MILIOTIS,P. (1978)

Using Cutting Planes to Solve the Symmetric Travelling Salesman Problem.

Math. Programming 15 (1978) 177-188

MINIEKA,E. (1978)

Optimization Algorithms for Networks and Graphs

Marcel Dekker, New York (1978)

MINIEKA,E. (1979)

The Chinese Postman Problem for Mixed Networks.

Management Science 25 (1979) 643-648

MINOUX,M. et M.GONDRAN (1979)

Graphes et Algorithmes

Ed.Eyrolles, Paris (1979)

MURTY,K.G. (1976)

Linear and Combinatorial Programming

John Wiley, New York (1976)

ORLOFF,C.S. (1974)

A fundamental problem in Vehicle Routing.

Networks 4 (1974) 35-64

PADBERG,M.W. and S.HONG (1980)

On the symmetric Travelling Salesman Problem:

A Computational Study.

Math.Programming Study 12 (1980) 78-107

PAPADIMITRIOU,C. (1976)

On the Complexity of Edge Traversing.

J.Assoc.Comput.Mach. 23 (1976) 544-554

POLJAK,B. (1967)

A general Method for solving Extremum Problems.

Soviet Math. Dokl. 8 (1967) 593-597

SANDI,C. (1979)

Subgradient Optimization , in *Combinatorial Optimization* (N.Christofides et al. Eds.) John Wiley (1979)

SHAPHIRO,J.F. (1971)

Generalized Lagrange Multipliers in Integer Programming.

Operations Research 19 (1971) 1070-1075

SHAPHIRO,J.F. (1979)

A Survey of Lagrangean Techniques for Discrete Optimization.

Annals of Discrete Math. 5 (1979) 113-138

SHAPHIRO,J.F (1979)

Mathematical Programming.Structures and Algorithms John Wiley, New York (1979)

STERN,H. and M.DROR (1979)

Routing Electric Meter Readers.

Comput. Oper. Res. 6 (1979) 209-223

UNIVERSIDAD DE VALENCIA FACULTAD DE CIENCIAS MATEMATICAS

Reunido el Tribunal que suscribe en el día de la fecha, acordó otorgar, por unanimidad, a esta Tesis doctoral de, D. Angel Corberán Salvador

la calificación de Sobresaliente cum laude

Valencia, a 14 de Julio de 1982

El Secretario,

El Presidente