Universidad de Valladolid
Facultad de Ciencias

Trabajo Fin De Grado

Grado en Estadística

Problemas de diseño de redes con restricciones de saltos (hop constraints)

Alumno: Andrés Bravo Núñez

Tutor: Jesús Sáez Aguado

Problemas de diseño de redes con restricciones de saltos (hop constraints)

Andrés Bravo Núñez

Índice general

Índice de figuras

Índice de cuadros

Dedicado a mi familia

Agradecimientos

A mis profesores por haberme dado el conocimiento del que hoy hago uso y con los que he aprendido a ser mejor persona.

A mis compañeros, a los que deseo un gran futuro y con los que espero no perder el contacto.

A mi familia, por que sin su apoyo no podria haber llegado hasta aquí.

A mi tutor, Jesús Saez que siempre estuvo para resolver cualquier problema haciendo este trabajo posible.

Resumen

Resumen: En el presente documento se expondrán diferentes métodos para dar solución al problema del mínimo árbol generador con restricciones de salto (HMST, *Hop-constrained minimun spanning tree problem*). Se afrontará su resolución tanto de manera exacta como aproximada búscando siempre la obtención de buenas soluciones en tiempos razonables

Palabras claves: HMST, Restricciones de salto, Heurísticas, Metaheurísticas

Abstract: In this document, we will make a review of the existing methods to solve the Hop-constrained minimun spanning tree problem. We will solve the problem exactly and also aapproximately always trying to get good solutions in reasonable times.

Key Words: HMST, Hop Constrains, Heuristics, Metaheuristics

Introducción

El objetivo de este proyecto es dar una visión general sobre el problema del mínimo árbol generador con restricciones de salto (HMST, *Hop-constrained minimun spanning tree problem*). Para ello, se explorarán tanto soluciones exáctas como aproximaciones mediante heurísticas, comparandose tanto la exactitud de los procedimientos como su coste temporal.

El problema HMST trata de encontrar un mínimo árbol generador con mínimo coste en un grafo tal que los caminos desde la raíz a cualquier otro nodo no superen un número de saltos predetermidado H. Se puede observar, que el problema no es más que una extensión sobre la búsqueda de un mínimo árbol generador para el que se conocen soluciones con tiempos de ejecución $\mathcal{O}(m \log n)$ (m es el número de arcos y n el de vértices)[23][29]. Sin embargo HMST es un problema NP-Hard,es decir para el que no se conoce un método de resolución exacta de orden polinómico, ya que contiene como caso particular una versión NP-Hard del problema de localización de servicios sin capacidades (simple plant location problem(SPL))[7].

El primer estudio sobre el problema HMST data de 1995[16] por lo que es relativamente reciente. Su principal campo de aplicación es el diseño de redes de telecomunicación centralizadas con restricciones sobre la calidad del servicio entendida como disponibilidad y confiabilidad. La disponibilidad se define como la probabilidad de que todas las lineas de transmisión de un cliente a un servidor funcionen. La confiabilidad es la probabilidad de que una sesión no sea interrumpida por el fallo de un enlace. En general tanto la disponibilidad como la confiabilidad aumentan al reducir el número de conexiones entre servidor y cliente.

Estructura del documento

El documento consta de cinco capítulos:

- Resolución Exacta: Se mostrarán diferentes modelos para la resolución exacta del problema así como algunas de las ideas detrás de los mismos.

- Heurísticas: Desarrollaremos deferentes métodos que buscan encontrar una solución próxima al óptimo ya sea construyendo una solución desde cero o realizando una búsqueda en el entorno de una solución conocida.
- Metaheurísticas: Se desarrollaran un conjunto de métodos que tratan de mejorar los métodos heurísticos introduciendo cierta aleatoriedad en los mismos.
- Resultados: Se mostrarán los resultados obtenidos con los diferentes métodos propuestos y se comentarán los mismos con el fin de facilitar la interpretación.
- Conclusiones: se expondrá el conjunto de conclusiones obtenidas a lo largo del trabajo y se realizará una recomendación sobre como resolver este problema en un caso real.

Herramientas utilizadas

Se han utilizado las siguientes programas:

- El entorno de modelado y resolución de problemas de optimización: Xpress-Mosel (rellenar la información de la versión)
- El lenguaje de programación R

Xpress-Mosel ha sido utilizado para la resolución exacta y R para las heurístecas y metaheurística. La elección de este conjunto de programas, se debe principalmente a que su uso está muy extendido y a la facilidad de acceso, ya sea por su gratuidad o debido a que ofertan una versión para estudiantes.

Los cáculos han sido realizados cuando ha sido posible en un ordenador con las siguientes especificaciones:Intel® Core™ i5-4200U 1.65GHz con 8 GB de RAM. En caso de imposibilidad debido a límites en la licencia estudiantil de Xpress-Mosel se ha usado un servidor cedido por la Universidad de Valladolid para este propósito.

Capítulo 1

Resolución Exacta

La resolución exacta nos ofrece la mejor solución que es, la que prefeririamos tener siempre que podamos. Para obtener esta solución en el problema HMST necesitamos un modelo matemático con restricciones. Geir Dahl, Luis Gouveia y Cristina Requejo definen el problema de la siguiente manera[9]:

Dado $G = (V, E)$ un grafo con un conjunto de nodos $V = \{0, 1, \ldots, n\}$ y un conjunto de aristas E, c_e costes asociados a cada arista $e \in E$ y un número natural H, deseamos encontrar en el grafo un árbol generador T con coste mínimo tal que el camino único desde el nodo raiz (nodo 0 a partir de ahora) a cualquier otro no incluya más de H saltos (aristas).

A continuación, se mostrarán una selección de modelos exactos para este problema. La razón por la que se exponen más de un modelo exácto es que las relajaciones de los mismos, pueden aportar cotas más o menos ajustadas según los parámetros o características del problema. Que un modelado del problema tenga una relajación con una cota más ajustada implica por lo general, un menor tiempo para encontrar la solución óptima (esto se debe a que los programas de optimización usan algoritmos de tipo ramificación y poda).

Cabe decir, que para la evaluación de los modelos exactos se han usado las relajaciones que usa Xpress-Mosel por defecto y no los recomendados en la literatura [7] [17] [3].

1.1. Formulación de flujos multi-producto

En este apartado, se describirá una formulación de flujos multi-producto para HMST que utiliza arcos dirigidos (La arista $\{i, j\}$ es remplazada por los arcos (i, j) y (j, i)) y algunas variantes surgidas de la misma. Las variables de decisión son $X_{ij}((i, j) \in A)$ que indican si el arco (i, j) del conjunto de arcos A se incluye en la solución e $Y_{ijk}((i, j) \in A; k \in V \setminus \{0\}; k \neq i)$ que muestran si el camino único del nodo raíz hasta el nodo k atraviesa el arco (i, j).

Modelo MCF *Multicommodity Flow*[15]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.1a}$$

sujeto a

$$\sum_{i\in V\setminus\{j\}} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.1b}$$

$$\sum_{i\in V\setminus\{k\}} Y_{ijk} - \sum_{i\in V\setminus\{0\}} Y_{jik} = \begin{cases} -1, & j = 0 \\ 0, & j \neq 0, k \quad j \in V \setminus \{0, k\} \\ 1, & j = k \end{cases} \tag{1.1c}$$

$$\sum_{(i,j)\in A} \leq H \tag{1.1d}$$

$$Y_{ijk} \in \{0, 1\}, \quad (i, j) \in A; k \in V \tag{1.1e}$$

$$Y_{ijk} \leq X_{i,j}, \quad (i, j) \in A; k \in V \setminus \{0\} \tag{1.1f}$$

$$X_{ij} \in \{0, 1\}, \quad (i, j) \in A \tag{1.1g}$$

Las restricciones (1.1f) relacionan los dos conjuntos de variables. Obligan a que si el arco (i, j) esta en un camino de tamaño H al nodo k entonces esta en el conjunto solución. Las restricciones (1.1c) indican que si un arco entra (o sale) en un nodo debe haber un arco saliente (o entrante) excepto para el nodo k, en el que hay un arco entrante más, y el nodo raíz, en el que hay un arco saliente más. Las restricciones (1.1d) marcan que no puede haber más de H arcos del nodo raíz al nodo k.

En conjunto las restricciones (1.1c)-(1.1e) hacen que los conjuntos de arcos definidos por $Y_{.k}$ incluyan los caminos de tamaño H (aunque también se incluyen paseos de tamaño y otros conjuntos de arcos) pero gracias a las otras restriccionnes solo los H-caminos pueden formar parte de la solución. Estas restricciones se pueden sustituir por otras que obliguen a que solo los H-caminos esten en la solución. De una manera más general:

Modelo MCF General[15]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.2a}$$

sujeto a

$$\sum_{i\in V\setminus\{j\}} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.2b}$$

$$Y_{.k} \in F_k, \quad (i, j) \in A; k \in V \tag{1.2c}$$

$$Y_{ijk} \leq X_{i,j}, \quad k \in V \setminus \{0\} \tag{1.2d}$$

$$X_{ij} \in \{0, 1\}, \quad (i, j) \in A \tag{1.2e}$$

Donde $Y_{.k}$ es el vector de componentes $Y_{ijk}((i,j) \in A; k \in V \setminus \{0\}; k \neq i)$ y F_k es el conjunto de todos los H-caminos de 0 a k. Idealmente, se buscan restricciones que definan el conjunto más pequeño que contenga los caminos de tamaño H y esto es, la envolvente convexa para cada F_k, $conv(F_k)$. Actualmente se conocen formulaciones de este tipo para $H = 2$ y $H = 3$:

Modelo MCF* $H = 2$ [7]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij} X_{ij} \tag{1.3a}$$

sujeto a

$$\sum_{i \in V\setminus\{j\}} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.3b}$$

$$X_{0j} \geq X_{jk}, \quad (j,k) \in A \tag{1.3c}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.3d}$$

Modelo MCF* $H = 3$ [15]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij} X_{ij} \tag{1.4a}$$

sujeto a

$$\sum_{i \in V\setminus\{j\}} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.4b}$$

$$\sum_{i \in V\setminus\{k\}} Y_{ijk} - \sum_{i \in V\setminus\{0\}} Y_{jik} = \begin{cases} -1, & j = 0 \\ 0, & j \neq 0, k \quad j \in V \setminus \{0,k\} \\ 1, & j = k \end{cases} \tag{1.4c}$$

$$Y_{jkk} \geq \sum_{i \in V\setminus 0} Y_{ijk}, \quad j \in V \setminus 0, k; k \in V \setminus 0 \tag{1.4d}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.4e}$$

$$Y_{ijk} \in \{0,1\}, \quad (i,j) \in A; k \in V \tag{1.4f}$$

Para $H > 4$ encontrar la envolvente convexa se vuelve complicado ya que encontrar la envolvente convexa es en si mismo un problema NP-Hard.

Sea W_k el conjunto de H-paseos de 0 a k. Está demostrado que sustituir en estos problemas $conv(F_k)$ por $conv(W_k)$ lleva a la misma solución óptima[9]. Encontrar una descripción de $conv(w_k)$ es más sencillo pero genera un conjunto exponencial de restricciones[8] por lo cual no es práctico. Sin embargo encontrar el camino más corto en W_k se puede modelar en un grafo acíclico expandido con facilidad. El grafo expandido $G_E = (V_E, A_E)$ es el grafo original replicado H veces. V_E se define como:

$$V_E = (0,0) \cup \{(i,h) : 1 \leq h \leq H-1 \text{ and } V \setminus \{0\}\} \cup (k,H) \tag{1.5}$$

Es decir (i, h), implica que en el grafo original el nodo i es visitado en el salto h. Los arcos A_E son definidos por:

$$\begin{aligned} A_E = &\{((0,0),(j,1)) : j \in V \setminus \{0\}\} \cup \\ &\{((i,h),(j,h+1)) : (i,j) \in A, i \neq 0, i \neq k \text{ and } 1 < h < H-2\} \cup \\ &\{((i,H-1),(k,H)) : i \in V \setminus \{0\} \text{ and } 1 < h < H-2\} \cup \\ &\{((k,h),(k,h+1)) : 1 < h < H-1\}. \end{aligned} \tag{1.6}$$

Los arcos de la forma $((k,h),(k,h+1))$ equivaldrían a bucles en el nodo k en el grafo original y serían descartados a efectos de la solución.

Si asociamos una variable Z_{ijk} a cada arco $((i,h-1),(j,h))$ de G_E podemos escribir un modelo para encontrar el H-camino de menor coste de 0 a H. Además las variables Z_{ijk} pueden reinterpretarse en G como indicadores de si el arco (i,j) es el h-ésimo arco en el camino del nodo raíz al nodo k. El modelo resultante es el siguiente:

Modelo Hop-Path(k),$k \in V \setminus \{0\}$ [17]

$$\text{minimizar} \sum_{(i,j)\in A} \sum_{q=1}^{H} c_{ij} Z_{ijq} + \sum_{i=1}^{n} \sum_{q=2}^{H} 0 Z_{iiq} \tag{1.7a}$$

sujeto a

$$\sum_{j\in V} Z_{jkH} = 1 \tag{1.7b}$$

$$\sum_{j\in V} Z_{jih} - \sum_{j\in V\setminus\{0\}} Z_{ij,h+1} = 0 \quad i \in V \setminus \{0\}; h = 2, \ldots, H-1 \tag{1.7c}$$

$$Z_{0i1} - \sum_{j\in V} Z_{ij2} = 0, \quad (0,i) \in A \tag{1.7d}$$

$$Z_{0j1} \in \{0,1\}, \quad (0,j) \in A \tag{1.7e}$$

$$Z_{ijh} \in \{0,1\}, \quad (i,j) \in A; i \neq 0; h = 2, \ldots, H \text{ or } i = j = k. \tag{1.7f}$$

Las restricciones (1.7d) muestran que el arco $(0,i)$ es el primer arco del camino solamente si existe un arco saliendo de i en $h = 2$. Las restricciones (1.7c) obligan a que si un arco entra en i en la posición h otro debe salir de i en $h+1$. Por último (1.7b) hace que solamente un arco pueda entrar en k en la posición H.

Una vez tenemos esto podemos combinar esta idea en el modelo 1.2. Para ello, debemos añadir a las Z un indíce más por cada camino que buscamos, esto es un nuevo indice w en V y además algunas condiciones $w \in V \setminus \{0\}$ ya que no tiene sentido buscar un camino de la raíz a sí misma:

Modelo HopMCF [17]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.8a}$$

sujeto a

$$\sum_{i\in V\setminus\{j\}} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.8b}$$

$$\sum_{j\in V} Z_{jkHw} = 1 \tag{1.8c}$$

$$\sum_{j\in V} Z_{jihw} - \sum_{j\in V\setminus\{0\}} Z_{ij,h+1,w} = 0 \quad i \in V \setminus \{0\}; h = 2, \ldots, H-1 \tag{1.8d}$$

$$Z_{0i1w} - \sum_{j\in V} Z_{ij2w} = 0, \quad (0,i) \in A \tag{1.8e}$$

$$Z_{0j1w} \in \{0,1\}, \quad (0,j) \in A, w \in V \setminus \{0\} \tag{1.8f}$$

$$Z_{ijhw} \in \{0,1\}, \quad (i,j) \in A; w \in V \setminus \{0\}; i \neq 0; h = 2, \ldots, H \text{ or } i = j = k. \tag{1.8g}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.8h}$$

$$Z_{0j1w} \leq X_{0j}, \quad (0,j) \in A; w \in V \setminus \{0\} \tag{1.8i}$$

$$\sum_{h=2}^{H} Z_{ijhw} \leq X_{ij}, \quad (i,j) \in A; i \neq 0; w \in V \setminus \{0\} \tag{1.8j}$$

Como conclusión a este apartado, lo esperable es que para $H = 2, 3$ los modelos 1.3 y 1.4 sean los más rápidos en encontrar la solución óptima y para $H \geq 4$ lo sea el modelo 1.8 o el modelo 1.1. En principio el modelo 1.1 dará mejores resultados que el modelo 1.8 cuanto mayor sea H [7].

1.2. Formulaciones basadas en restricciones de Miller-Tucker-Zemlin

Consideremos un conjunto de variables $u_i (i \in V)$ que especifican la posición del nodo i en la solución (considerando como origen el nodo raíz). Las restricciones de Miller-Tucker-Zemlin[25] son unas conocidas restricciones con forma:

$$u_i - u_j + n * x_{ij} <= n - 1 \tag{1.9}$$

Su principal función es la eliminación de subtours, es decir, evitan que aparezcan ciclos secundarios en la solución. Esto se entiende mejor en el problema al que se aplican estas restricciones originalmente: el TSP (*Traveling Salesman Problem* o Problema del Viajante). En el TSP se trata de minimizar el coste de la ruta que pasa por cada nodo a lo

sumo una vez. Sin estas restricciones, un optimizador podría encontrar una solución con dos rutas inconexas que en conjunto cumplan las restricciones de pasar por cada nodo una vez. En cuanto a nuestro problema, el HMST, podemos adaptar estás restricciones para evitar la existencia de ciclos. Para entender esto mejor podemos ejemplificarlo con la siguiente formulación:

Modelo MTZ(*Miller-Tucker-Zemlin*) [16]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.10a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.10b}$$

$$nX_{ij} + u_i \leq u_j + (n-1), \quad i,j \in V \setminus \{0\} \tag{1.10c}$$

$$1 \leq u_i \leq H, \quad i \in V \setminus \{0\} \tag{1.10d}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.10e}$$

Las restricciones 1.10b hacen que solo un arco entre en cada nodo, exceptuando la raiz. Las restricciones 1.10c evitan la existencia de ciclos. Para observar esto basta con suponer que hay un ciclo y aplicando las restricciones en dicho ciclo llegaremos a una contradicción del tipo $u_i + 1 \leq u_i$. Por último las restricciones 1.10d garantizan que los caminos de la raíz al resto de nodos contienen como máximo H arcos.

Si nos fijamos en las restricciones 1.10c las variables u_i no tienen porque marcar de manera precisa la posición del nodo i en el camino. Imaginemos un problema con $H = 3$ y un camino con 2 nodos. No hace falta que los valores del las u sean 1 para el primer nodo y 2 para el segundo, también podrian ser 2 y 3 o 1 y 3 respectivamente. Lo importante es que las u_i cumplan lo siguiente:

$$u_i - u_j \leq (H-1) \quad i,j \in V \setminus \{0\} \tag{1.11}$$

Esta desigualdad es redundante con las restricciones 1.10d y puede integrarse en 1.10c para dar una nueva formulación:

Modelo EMTZ(*Extended Miller-Tucker-Zemlin*) [16]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.12a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.12b}$$

$$HX_{ij} + u_i \leq u_j + (H-1), \quad i,j \in V \setminus \{0\} \tag{1.12c}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.12d}$$

Esta formulación 1.12, puede tener unas cotas más ajustadas con su relajación lineal que la formulación 1.10 ya que toda solución que cumpla 1.12c cumplirá 1.10c. Para demostrarlo basta con añadir $(n-H)X_{ij}$ a cada lado de 1.12c:

$$(n-H)X_{ij} + HX_{ij} + u_i \leq (n-H)X_{ij} + u_j + (H-1) \quad i,j \in V \setminus \{0\} \tag{1.13}$$

que es equivalente a:

$$nX_{ij} + u_i \leq (n-H)X_{ij} + u_j + (H-1) \quad i,j \in V \setminus \{0\} \tag{1.14}$$

como X_{ij} puede tomar únicamente los valores $\{0,1\}$ el lado derecho de la anterior desigualdad cumple:

$$(n-H)X_{ij} + u_j + (H-1) \leq (n-H) + u_j + (H-1) = u_j + (n-1) \quad i,j \in V \setminus \{0\} \tag{1.15}$$

y sustitutendo de vuelta en 1.14:

$$nX_{ij} + u_i \leq u_j + (n-1) \quad i,j \in V \setminus \{0\} \tag{1.16}$$

que equivale a las restricciones 1.10c del modelo MTZ 1.10.

1.2.1. Elevando las restricciones

Para reforzar el modelo 1.12, es decir, que las cotas de sus relajaciones sean más ajustadas, necesitamos hacer uso de técnicas de elevamiento de restricciones [27][34]. El elevamiento, es una técnica por la cual, una restricción válida para un subconjunto de la región de soluciones, nos referiremos a ella como *restricción semilla*, es modificada para que su validez sea global. Habitualmente, la restricción semilla es derivada bajo la asunción de que ciertas variables están fijas. Posteriormente esta restricción es relajada hasta que sea globalmente válida.

Intentaremos realizar una descripción más matemática aunque informal. Tenemos un conjunto de soluciones factibles a nuestro problema de programación entera $S \subseteq \{0,1\}^n$ (n son el número de variables que componen la solución) y además sabemos que existen soluciones con la variable $x_a = 1$ (si no sospecharamos esto no intentaríamos elevar las restricción) o en otras palabras $S \cap \{x : x_a = 1\} \neq \emptyset$. Ahora supongamos que tenemos una restricción de la forma:

$$\sum_{i=1;i\neq a}^{n} \alpha_i x_i \leq \beta \tag{1.17}$$

Que es válida para $S \cap \{x : x_a = 0\}$ aunque tambien es válida para $S \cap \{x : x_a = 1\}$ y para S puesto que simplemente no incluye x_a. Pero como sabemos que $S \cap \{x : x_a = 1\} \neq \emptyset$ podemos pensar que existe un cierto *hueco* $\beta - \sum_{i=1;i\neq a}^{n} \alpha_i x_i$ que podemos ocupar con un nuevo término $\alpha_a x_a$. Lo que planteamos es *elevar* la restricción 1.17 para que sea

válida en un espacio con una dimensión más (puesto que hemos añadido una variable). La restricción elevada tendrá la forma:

$$\sum_{i=1;i\neq a}^{n} \alpha_i x_i + \alpha_a x_a \leq \beta \tag{1.18}$$

Por último debemos hallar el valor del nuevo coeficiente α_a. Siguiendo con la idea de aprovechar el *hueco* α_a valdrá:

$$\alpha_a = \beta - \max\{ \sum_{i=1;i\neq a}^{n} \alpha_i x_i : x \in S, x_a = 1\} \tag{1.19}$$

Por lo tanto todo se reduce a resolver otro problema de programación entera lo cual es difícil. Sin embargo, estos problemas pueden ser fáciles de resolver (por su tamaño o por otros motivos) y aportar una ventaja al ajustar más la definición del problema al conjunto S.

De vuelta al problema HMST y el modelo EMTZ 1.12 podemos aplicar esta idea para mejorar la formulación. Existen elevaciones de las restricciones de Miller-Tucker-Zemlin [10] que se han adapatado a nuestro problema [16].Las restricciones 1.12c puede elevarse a:

$$(H-2)X_{ji} + HX_{ij} + u_i \leq u_j + (H-1), \quad i,j \in V \setminus \{0\}; H \geq 2 \tag{1.20}$$

Donde $H-2$ se sostiene de:

$$\alpha_a = (H-1) - \max\{HX_{ij} + u_i - u_j, \quad i,j \in V \setminus \{0\}; x_{ji} = 1\} \tag{1.21}$$

Puesto que si $x_{ji} = 1$ entonces $x_{ij} = 0$ y además $u_j + 1 = u_i$ (H debe ser mayor o igual a 2) la solución a ese problema de maximización es 1 y por tanto $\alpha_a = (H-2)$. Sustituyendo 1.12c por 1.20 obtenemos la nueva formulación:

Modelo L1EMTZ(*Lift-1 Extended Miller-Tucker-Zemlin*) [16]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij} X_{ij} \tag{1.22a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.22b}$$

$$(H-2)X_{ji} + HX_{ij} + u_i \leq u_j + (H-1), \quad i,j \in V \setminus \{0\} \tag{1.22c}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.22d}$$

Otra restricción para la que se ha propuesto un elevamiento es la siguiente:

$$\sum_{k=1;k\neq i}^{n} X_{kj} + HX_{ij} + u_i \leq u_j + (H-1) \quad i,j \in V \setminus \{0\} \tag{1.23}$$

Por 1.22b el sumando puede ser como máximo 1 y por la misma razón el sumando y X_{ij} no pueden ser 1 a la vez. Si el sumando vale 0, entonces 1.23 se reduce a 1.12c y por tanto es válida en este caso. Si el sumando vale 1 entonces algún $X_{kj} = 1$ para algún $k \neq 0, 1; j \neq 0$. Por tanto, la diferencia entre las posiciones de los nodos i e j en la solucions (las u) no pueden diferir más de $(H-2)$ lo cual es consistente con 1.23. Dado esto concluimos que 1.23 es una restricción válida para el problema HMST y por ello, la versión elevada de 1.23 al añadir la variable X_{ji} es:

$$\sum_{k=1;k\neq i}^{n} X_{kj} + (H-3)X_{ji} + HX_{ij} + u_i \leq u_j + (H-1) \quad i,j \in V \setminus \{0\}; H \geq 3 \tag{1.24}$$

donde el escalar $(H-3)$ seo obtiene al resolver:

$$\alpha_a = (H-1) - \max\{\sum_{k=1;k\neq i}^{n} X_{kj} + HX_{ij} + u_i - u_j, \quad i,j \in V \setminus \{0\}; x_{ji} = 1\} \tag{1.25}$$

La solución se obtiene igual que en 1.21 pero ahora tenemos un sumatorio $\sum_{k=1;k\neq i}^{n} X_{kj}$ que puede valer 1 y por tanto resulta en $(H-3)$. El modelo usando la nueva restricción es el siguiente:

Modelo L2EMTZ(*Lift-2 Extended Miller-Tucker-Zemlin*) [16]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.26a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.26b}$$

$$\sum_{k=1;k\neq i}^{n} X_{kj} + (H-3)X_{ji} + HX_{ij} + u_i \leq u_j + (H-1), \quad i,j \in V \setminus \{0\} \tag{1.26c}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.26d}$$

En principio los dos modelos elevados 1.22 y 1.26 deberían dar mejor resultados que el modelo EMTZ 1.12.

1.2.2. Restricciones topológicas mejoradas

En una solución del problema HMST los nodos son o bien nodos hoja o nodos centrales del árbol. Los nodos hoja tienen valencia (número de aristas incidentes sobre un nodo) uno y los nodos centrales tienen mayor valencia. Se puede utilizar esta distinción entre dos tipos de nodo para restringir el modelo. Por ejemplo, una restricción que podríamos implementar usando esta distinción es que dos nodos hoja nunca pueden estar conectados.

Sea w_{ic} y w_{il} un par de variables binarias que indican si el nodo i es central u hoja respectivamente. Utilizando esa idea junto con la restricciones de MTZ obtenemos el siguiente modelo:

Modelo MTZ/ITEF(*MTZ/Improved Topology-Enforcing constraints*) [3]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.27a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.27b}$$

$$nX_{ij} + u_i \leq u_j + (n-1), \quad i,j \in V \setminus \{0\} \tag{1.27c}$$

$$1 \leq u_i \leq H, \quad i \in V \setminus \{0\} \tag{1.27d}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.27e}$$

$$w_{ic} + w_{il} = 1, \quad i \in V \tag{1.27f}$$

$$\sum_{j\in V} X_{0j} \geq 1, \tag{1.27g}$$

$$\sum_{j\in V\setminus\{0\}} X_{0j} \geq 1 + w_{0c}, \tag{1.27h}$$

$$\sum_{j\in V\setminus\{0\}} X_{0j} \leq (n-1) + (n-2)w_{0l}, \tag{1.27i}$$

$$\sum_{j\in V} X_{ji} + \sum_{j\in V\setminus\{0\}} X_{ij} \geq 1 + w_{ic}, \quad i \in V \setminus \{0\} \tag{1.27j}$$

$$\sum_{j\in V} X_{ji} + \sum_{j\in V\setminus\{0\}} X_{ij} \leq (n-1) + (n-2)w_{0l}, \quad i \in V \setminus \{0\} \tag{1.27k}$$

$$\sum_{j\in V\setminus\{0\}} X_{ij} \geq 1 - w_{il}, \tag{1.27l}$$

$$X_{ij} \leq w_{ic}, \quad i,j \in V \setminus \{0\} \tag{1.27m}$$

$$X_{ij} + w_{il} + w_{jl} \leq 2, \quad j \in V \setminus \{0\} \tag{1.27n}$$

$$X_{i0} = 0, \tag{1.27o}$$

$$X_{ij} + X_{ji} \leq 1, \quad i < j \tag{1.27p}$$

$$\sum_{j\in V\setminus\{i\}} X_{ij} = n - 1, \tag{1.27q}$$

$$w_{ic}, w_{il} \in \{0,1\}, \quad i \in V \tag{1.27r}$$

Las restricciones 1.27f obligan a cada nodo a ser una hoja o un nodo central. Las restricciones 1.27g-1.27i marcan limites superiores e inforiores a el número de arcos salientes del nodo raiz. Las restricciones 1.27j-1.27l ponen limites a las valencias de los nodos no

raiz. Las restricciones 1.27m hacen que los nodos no raíz con arcos salientes sean centrales. Las restricciones 1.27n impiden arcos entre nodos hoja. Las restricciones 1.27o no permiten que haya arcos entrantes en la raiz. Las restricciones 1.27p evitan que haya dos arcos con sentidos opuestos conectando dos nodos. Por último las restricciones 1.27q hacen que haya n-1 arcos lo que es un hecho para los árboles.

Además de añadir estas restricciones topológicas podemos usar las nuevas variables w para mejorar las restricciones de Miller-Tucker-Zemlin. Esto se consigue añadiendo las restricciones:

$$u_i \geq H w_{il} \quad i \in V \setminus \{0\} \tag{1.28}$$

$$u_i \leq H - w_{ic} \quad i \in V \setminus \{0\} \tag{1.29}$$

La primera restricción 1.28 combinada con 1.27d o 1.12c hace que los nodos hoja no raíz tengan $u = H$(recordemos que las variables u indicaban la posición en el camino). La segunda restricción restringe la u de los nodos centrales a H-1 como máximo. Si incluimos las restricciones en el modelo anterior obtenemos (Se ha cambiado 1.27d por 1.12c para contemplar todas las variaciones):

Modelo IMTZ/ITEF(*Improved MTZ/Improved Topology-Enforcing constraints*) [3]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij} X_{ij} \tag{1.30a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.30b}$$

$$H X_{ij} + u_i \leq u_j + (H-1), \quad i,j \in V \setminus \{0\} \tag{1.30c}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.30d}$$

$$w_{ic} + w_{il} = 1, \quad i \in V \tag{1.30e}$$

$$\sum_{j\in V} X_{0j} \geq 1, \tag{1.30f}$$

$$\sum_{j\in V\setminus\{0\}} X_{0j} \geq 1 + w_{0c}, \tag{1.30g}$$

$$\sum_{j\in V\setminus\{0\}} X_{0j} \leq (n-1) + (n-2) w_{0l}, \tag{1.30h}$$

$$\sum_{j\in V} X_{ji} + \sum_{j\in V\setminus\{0\}} X_{ij} \geq 1 + w_{ic}, \quad i \in V \setminus \{0\} \tag{1.30i}$$

$$\sum_{j\in V} X_{ji} + \sum_{j\in V\setminus\{0\}} X_{ij} \leq (n-1) + (n-2)w_{0l}, \quad i \in V\setminus\{0\} \tag{1.30j}$$

$$\sum_{j\in V\setminus\{0\}} X_{ij} \geq 1 - w_{il}, \tag{1.30k}$$

$$X_{ij} \leq w_{ic}, \quad i,j \in V\setminus\{0\} \tag{1.30l}$$

$$X_{ij} + w_{il} + w_{jl} \leq 2, \quad j \in V\setminus\{0\} \tag{1.30m}$$

$$X_{i0} = 0, \tag{1.30n}$$

$$X_{ij} + X_{ji} \leq 1, \quad i < j \tag{1.30o}$$

$$\sum_{j\in V\setminus\{i\}} X_{ij} = n-1, \tag{1.30p}$$

$$w_{ic}, w_{il} \in \{0,1\}, \quad i \in V \tag{1.30q}$$

$$u_i \geq H w_{il}, \quad i \in V\setminus\{0\} \tag{1.30r}$$

$$u_i \leq H - w_{ic}, \quad i \in V\setminus\{0\} \tag{1.30s}$$

Por último se incluye una variante de 1.30 en la que se eliman algunas restricciones y que según la fuente[3] tiene buenos resultados computacionales:

Modelo REL-M(*Relaxed Model*) [3]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.31a}$$

sujeto a

$$\sum_{i\in V} X_{ij} = 1, \quad j \in V\setminus\{0\} \tag{1.31b}$$

$$nX_{ij} + u_i \leq u_j + (n-1), \quad i,j \in V\setminus\{0\} \tag{1.31c}$$

$$1 \leq u_i \leq H, \quad i \in V\setminus\{0\} \tag{1.31d}$$

$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.31e}$$

$$w_{ic} + w_{il} = 1, \quad i \in V \tag{1.31f}$$

$$\sum_{j\in V} X_{0j} \geq 1, \tag{1.31g}$$

$$\sum_{j\in V\setminus\{0\}} X_{0j} \geq 1 + w_{0c}, \tag{1.31h}$$

$$\sum_{j\in V\setminus\{0\}} X_{0j} \leq (n-1) + (n-2)w_{0l}, \tag{1.31i}$$

$$\sum_{j\in V} X_{ji} + \sum_{j\in V\setminus\{0\}} X_{ij} \leq (n-1) + (n-2)w_{0l}, \quad i \in V \setminus \{0\} \tag{1.31j}$$
$$X_{ij} \leq w_{ic}, \quad i,j \in V \setminus \{0\} \tag{1.31k}$$
$$X_{ij} + w_{il} + w_{jl} \leq 2, \quad j \in V \setminus \{0\} \tag{1.31l}$$
$$X_{i0} = 0, \tag{1.31m}$$
$$X_{ij} + X_{ji} \leq 1, \quad i < j \tag{1.31n}$$
$$\sum_{j\in V\setminus\{i\}} X_{ij} = n-1, \tag{1.31o}$$
$$w_{ic}, w_{il} \in \{0,1\}, \quad i \in V \tag{1.31p}$$
$$u_i \geq H w_{il}, \quad i \in V \setminus \{0\} \tag{1.31q}$$
$$u_i \leq H - w_{ic}, \quad i \in V \setminus \{0\} \tag{1.31r}$$
$$u_j \geq H - (H-1)X_{0j} + (H-1)W_{jl}, \quad j \in V \setminus \{0\} \tag{1.31s}$$
$$\sum_{j\neq 0} X_{ij} \geq w_{il} - 1, \quad i \in V \setminus \{0\} \tag{1.31t}$$

1.2.3. Modelos de Sherali y Driscoll

Sherali y Driscoll desarrollaron un modelo para el problema ATSP (*Asymmetric traveling salesperson problem*) aplicando una versión de RLT (*Reformulation-Linearization Technique*)[1][31][32] a una formulacion con restricciones de Miller-Tucker-Zemlin[33]. Posteriormente este modelo fue adaptado al problema HMST[2].

Modelo HMST-SD(*HMST-Sherali and Driscoll*) [2]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij} X_{ij} \tag{1.32a}$$
sujeto a
$$u_0 = 0, \tag{1.32b}$$
$$u_i \geq 1, \quad i \in V \setminus \{0\} \tag{1.32c}$$
$$u_i \geq 0, \quad i \in V \tag{1.32d}$$
$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.32e}$$
$$\sum_{(i,j)\in A, j\neq 0} y_{ij} + 1 = u_j, \quad i \in V \setminus \{0\} \tag{1.32f}$$
$$X_{ij} \leq y_{ij}, \quad i,j \in V \setminus \{0\} \tag{1.32g}$$
$$y_{ij} \geq 0, \tag{1.32h}$$
$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.32i}$$
$$u_i - u_j + nX_{ij} \leq n-1, \quad j \in V \setminus \{0\} \tag{1.32j}$$
$$u_i \leq H, \quad i \in V \setminus \{0\} \tag{1.32k}$$

$$\sum (i,j) \in A, j \neq 0 y_{i,j} + H \geq u_i, \tag{1.32l}$$
$$y_{i,j} \leq (H-1)X_{ij}, \quad j \in V \setminus \{0\} \tag{1.32m}$$
$$u_j + (H-1)X_{ij} - H(1-X_{ji}) \leq y_{ij}y_{ji}, \quad j \in V \setminus \{0\} \tag{1.32n}$$
$$y_{ij} - y_{ji}) \leq u_j - (1-X_{ji}), \quad j \in V \setminus \{0\} \tag{1.32o}$$
$$2 - X_{0j}) \leq u_j, \quad j \in V \setminus \{0\} \tag{1.32p}$$
$$u_j) \leq H - (H-1)X_{0j}, \quad j \in V \setminus \{0\} \tag{1.32q}$$

Combinando este modelo con las restricciones topológicas del apartado anterior obtenemos una formulación con una relajación lineal más ajustada:

Modelo HMST-SD/ITEF [2]

$$\text{minimizar} \sum_{(i,j)\in A} c_{ij}X_{ij} \tag{1.33a}$$

sujeto a

$$u_0 = 0, \tag{1.33b}$$
$$u_i \geq 1, \quad i \in V \setminus \{0\} \tag{1.33c}$$
$$u_i \geq 0, \quad i \in V \tag{1.33d}$$
$$X_{ij} \in \{0,1\}, \quad (i,j) \in A \tag{1.33e}$$
$$\sum_{(i,j)\in A, j\neq 0} y_{ij} + 1 = u_j, \quad i \in V \setminus \{0\} \tag{1.33f}$$
$$X_{ij} \leq y_{ij}, \quad i,j \in V \setminus \{0\} \tag{1.33g}$$
$$y_{ij} \geq 0, \tag{1.33h}$$
$$\sum_{i\in V} X_{ij} = 1, \quad j \in V \setminus \{0\} \tag{1.33i}$$
$$u_i - u_j + nX_{ij} \leq n-1, \quad j \in V \setminus \{0\} \tag{1.33j}$$
$$u_i \leq H, \quad i \in V \setminus \{0\} \tag{1.33k}$$
$$\sum (i,j) \in A, j \neq 0 y_{i,j} + H \geq u_i, \tag{1.33l}$$
$$y_{ij} \leq (H-1)X_{ij}, \quad j \in V \setminus \{0\} \tag{1.33m}$$
$$u_j + (H-1)X_{ij} - H(1-X_{ji}) \leq y_{ij}y_{ji}, \quad j \in V \setminus \{0\} \tag{1.33n}$$
$$y_{ij} - y_{ji} \leq u_j - (1-X_{ji}), \quad j \in V \setminus \{0\} \tag{1.33o}$$
$$2 - X_{0j} \leq u_j, \quad j \in V \setminus \{0\} \tag{1.33p}$$
$$u_j \leq H - (H-1)X_{0j}, \quad j \in V \setminus \{0\} \tag{1.33q}$$
$$w_{ic} + w_{il} = 1, \tag{1.33r}$$
$$\sum jX_{0j} \geq 1, \tag{1.33s}$$
$$\sum j \in V \setminus \{0\} X_{0j} \geq 1 + w_{0c}, s \tag{1.33t}$$

$$\sum j \in V \setminus \{0\} X_{0j} \leq (H-1) - (H-2) w_{0l}, \tag{1.33u}$$

$$\sum j X_{ji} + \sum j \in V \setminus \{0\} X_{ij} \geq 1 + w_{ic}, \quad i \in V \setminus \{0\} \tag{1.33v}$$

$$\sum j X_{ji} + \sum j \in V \setminus \{0\} X_{ij} \leq (H-1) - (H-2) w_{0l}, \quad i \in V \setminus \{0\} \tag{1.33w}$$

$$\sum j \in V \setminus \{0\} X_{0j} \geq 1 - w_{il}, \quad i \in V \setminus \{0\} \tag{1.33x}$$

$$X_{ij} \leq w_{ic}, \quad i, j \in V \setminus \{0\} \tag{1.33y}$$

$$X_{ij} + w_{il} + wjl \leq 2, \quad j \in V \setminus \{0\} \tag{1.33z}$$

$$X_{i0} = 0, \tag{1.33aa}$$

$$X_{ij} + X_{ji} \leq 1, \quad i < j \tag{1.33ab}$$

$$\sum_{j \neq i} X_{ij} \leq H - 1, \tag{1.33ac}$$

$$w_{ic} \in \{0, 1\}, \tag{1.33ad}$$

$$w_{il} \in \{0, 1\}, \tag{1.33ae}$$

Capítulo 2

Heurísticas

En el capítulo anterior se vieron diferentes formulaciones que nos permiten alcanzar la solución óptima del problema HMST. Sin embargo, ya que estamos ante un problema NP-Hard el tiempo que puede llevar solucionar problemas de una embergadura moderada (entendida como número de arcos) puede hacer esta opción impracticable. Además cabe la posibilidad de que por motivos económicos u de otra índole no se tenga acceso a un software de optimización y por tanto haya que recurrir a métodos heurísticos.

La manera de enfrentar esta problemática es mediante el uso de métodos herísticos. Es decir utilizando algorítmos que nos permitan encontrar soluciones factibles que sean relativamente buenas. Se suelen diferenciar dos tipos de heuríticas:

- Heurísticas de Construcción: Construir paso a paso una solución factible siguiendo unas reglas.
- Herísticas de Mejora: Dada una solución factible inicial realizar pequeñas mejoras sucesivas hasta que no sea posible ninguna mejora.

Habitualmente estas heurísticas se combinan de manera que existe una fase de construcción y una de mejora.

Encontrar una solución factible para el problema HMST es trivial, basta con conectar todos los nodos directamente a la raíz. Por tanto las heurísticas desarrolladas pertenecen al grupo de las heurísticas de mejora. Mostraremos tres aproximaciones diferentes:(i) una heurística derivada e la heurística de Essau-Williams para el problema del mínimo arbol generador con capacidades (en el que se limíta el número máximo de nodos en un subárbol) [14],(ii) diferentes versiones de un algorítmo de segundo orden [14] y (iii) algorítmos de busqueda local en vecindarios derivados de un modelo de programación dinámica para el problema HMST [18].

2.1. Heurística de ahorros

Vamos a utilizar una modificación de la heurística de ahorros de Esau-Williams [11]. Se comienza con una configuración de estrella (todos los nodos conectados directamente a la raíz). Se calcula el ahorro S_{ijw} que se consigue al intercambiar un arco $(w, 0)$ por otro (i, j) siendo $S_{ijw} = c_{w0} - c_{ij}$ si lleva a una solución factible y $S_{ijw} = -\infty$ si no. Se realiza el intercambio de ahorro máximo y se vuelven a calcular los ahorros nuevamente. El algoritmo finaliza cuando no hay ahorros mayores a 0. Se puede ver el pseudocódigo del algorítmo en 1.

Algorithm 1 HMST Esau-Williams [14]

```
1: arcos = set_configuracion_estrella(nodos,raiz)
2: loop
3:    for w/arcos(w,0) = 1  do
4:       nodo1 = w
5:       for i,j/arcos(i,j) = 0  do
6:          nodo2 = i
7:          nodo3 = j
8:          arcos(i,j) = 1
9:          arcos(w,0) = 0
10:         {H se refiere al número de saltos máximo permitido}
11:         if esFactible(arcos,raiz,H) then
12:            ahorros(i,j,w) = c(w,0) - c(i,j)
13:         end if
14:         arcos(i,j) = 0
15:         arcos(w,0) = 1
16:      end for
17:   end for
18:   if max(ahorro) <= 0 then
19:      break
20:   end if
21:   cambioi,cambioj,cambiow = indices_de(ahorro,max(ahorro))
22:   arcos(cambioi,cambioj) = 1
23:   arcos(cambiow,0) = 0
24:   resetear_ahorros
25: end loop
```

2.2. Algoritmos de segundo orden

Los algorítmos de segundo orden son una clase de algorítmos que aplican una heurística base (e.g HMST Esau-Williams 1) iterativamente a versiones modificadas del problema.

Estas modificaciones implican fijar parte de la solución, en el contexto del problema HMST se obliga la inclusión o exclusión ciertos arcos. En cada paso se prueban todas las modificaciones generadas por una regla y se fijan las modificaciones que llevan a la mejor solución. Este proceso se repite hasta que ninguna modificación mejora la mejor solución anterior. La idea es aumentar las posibilidades de econtrar una buena solución al aumentar el número de condiciones iniciales sobre las que se aplica la heurística base.

Los algorítmos de segundo orden fueron propuestos inicialmente para el problema del arbol generador mínimo con capacidades(minimizar el coste del árbol generador con una restricción sobre la cantidad de nodos en los subárboles)[21] y posteriormente se adapto al HMST [14].

Procedemos a describir una versión general para este tipo de algorítmos. Sean S1 el conjunto de arcos a fijar y S2 el conjunto de arcos a inhibir en una modificación concreta. Tras una pasada del algorítmo de segundo orden, se fijaran las modificaciones que hayan dado la mejor solución, es decir, los elementos de S1 se incluyen en SP1 (con P de permante) y los elementos de S2 en SP2. HEUR(S1,S2,SP1,SP2) implica la ejecución de la heurística base sobre el problema modificado según los conjuntos S1, S2, SP1 y SP2 (fijandose los arcos en $S1 \cup SP1$ e inhibiendose los incluidos en $S2 \cup SP2$). Podemos ver una versión en pseudocódigo del algoritmo en 2.

En el contexto del problema HMST la manera de inhibir un arco es establecer su coste en infinito (o un coste lo suficientemente alto para que resulte prohibitivo incluirlo) y la manera de forzar su inclusión es darle un coste lo suficientemente bajo (si todos los costes son positivos 0 resultaría suficiente).

Algunas definiciones concretas para los conjuntos S2 y SP2 son:

- Inhibición simple [14]. S2 contendrá en cada pasada cada uno de los arcos que pertenezcan a la solución actual y que no contengan la raíz. La inhibición que de mejor resultado pasará a formar parte de SP2. Con este esquema el bucle interior del algoritmo 2 realizará n (número de nodos del problema) pasadas.
- Inhibición pareada [14]. Igual que la inhibición simple pero considerando en S2 pares de arcos en vez de arcos individuales. Esto lleva a que el bucle interior 2 realize n^2 pasadas.

y en cuanto a S1 y SP1 tenemos:

- Inclusión limitada [14]. S1 contendrá en cada pasada el arco de menor coste incidente a cada nodo o el arco de menor coste incidente a cada tal que el otro extremo este más cerca de la raiz. La inclusión que lleve a la mejor solución se incluirá en SP1. El bucle interior realizará 2n pasadas como máximo.

- Inclusión general [14]. Se puede pensar que cualquier arco que no esté en la solución es candidato para estar en S1. Como la cantidad de arcos de este tipo es muy elevada solo se evaluan los z arcos de menor coste incidentes en cada nodo. El arco que lleve a la mejor solución es fijado en S1.

Cabe destacar que si se fuerza la inclusión de arcos se necesita comprobar que estos puedan llevar a una solución factible. Es decir, que no existan ni ciclos ni caminos excesivamente largos.

Algorithm 2 Algoritmo de segundo orden

```
1: S1 = ∅
2: S2 = ∅
3: SP1 = ∅
4: SP2 = ∅
5: solucion_actual = HEUR(S1,S2,SP1,SP2)
6: coste = COSTE(solucion_actual)
7: loop
8:    flag = 0
9:    {El número de iteraciones del bucle depende del número posible de modificaciones
      que podamos realizar con la regla que elijamos}
10:   for i = 0, ... do
11:      modificar S1 y/o S2
12:      solucion_prueba = HEUR(S1,S2,SP1,SP2)
13:      if COSTE(solucion_prueba) < coste then
14:         flag = 1
15:         solucion_actual = solucion_prueba
16:         coste = COSTE(solucion_actual)
17:         A1 = S1
18:         A2 = S2
19:      end if
20:   end for
21:   if flag=0 then
22:      break
23:   end if
24:   SP1 = SP1 ∪ A1
25:   SP2 = SP2 ∪ A2
26: end loop
```

2.3. Vecindarios basados en un modelo de programación dinámica

2.3.1. Representación de un árbol mediante el nivel de los vértices

Una representación de algunas soluciones factibles (incluyendo la óptima) se puede realizar utilizando la definición del nivel de un vertice propuesta por Gruber et al. [20]. El nivel de un nodo se define como la máxima distancia, con respecto al número de arcos, que puede tener dicho nodo de la raiz. Si tenemos esta asignación podemos reconstruir el árbol generador al que pertenece. Sea $n(i)$ el nivel del nodo i:

$$\text{Para cada nodo } i \in V \setminus \{0\}, \quad \begin{array}{l}(1) \text{ Determinar } j^* \text{tal que} \\ \qquad min\{c_{ij} : n(j) = 0, \ldots, n(i)-1\} = c_{ij^*} \\ (2) \text{ Insertar el arco } (i, j^*)\end{array} \tag{2.1}$$

Esta manera de representar un árbol ya nos da una idea rudimentaria de un vecindario basado en cambiar el nivel de diferentes nodos. Para que tengamos una manera sistemática de definir un vecindario y recorrelo, necesitamos el modelo de programación dinámica.

2.3.2. Un modelo de programación dinámica para el problema HMST

La programación dinámica es una técnica para resolver problemas que tengan subestructuras óptimas y subproblemas que se solapen [6]. Un problema tiene subestructuras óptimas si se puede obtener su solución óptima mediante soluciones óptimas de sus subproblemas. Un problema tiene supbroblemas que se solapan si es divisible en subproblemas que se repiten, es decir, si dividimos el problema en los subproblemas más pequeños posibles existirán muchas repeticiones. Un ejemplo clásico de este tipo de problemas es generar el n-ésimo término de la serie de Fibonacci. Lo que necesitamos para un modelo de programación dinámica es la solución óptima de los problemas más simples y alguna regla para generar la solución de problemas más grandes. En el caso de la secuencia de Fibonacci:

$$fibonacci(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ fibonacci(n-1) + fibonacci(n-2) & \text{if } n > 1 \end{cases} \tag{2.2}$$

Para el problema que nos ocupa es fácil comprobar que existe una subestructura óptima. Supongamos que conocemos la solución óptima de un problema HMST con máximo

número de saltos H. Si eliminamos los vértices con nivel H obtendremos la solución óptima del problema sin estos vértices y máximo número de saltos H-1. Necesitamos ser capaces de conocer el coste de añadir los vértices de nivel k a una solución con vértices con nivel máximo k-1. Sea S_k los vértices que tienen nivel k. El coste de añadir los vértices en S_k a otro conjunto de vértices S que no los contenga es:

$$\begin{aligned} &coste(S_k, S) = \sum_{i \in S_k} c_{i(S),i}, \quad S \neq \emptyset \\ &coste(S_k, \emptyset) = \sum_{i \in S_k} c_{0,i}, \\ &coste(\emptyset, S) = 0 \end{aligned} \tag{2.3}$$

Donde $i(S)$ indica el nodo en S más cercano al nodo i en S_k. Sea $z(S, k)$ el coste del árbol óptimo construido con los vértices en S con un número máximo de saltos k. Un estado queda definido por el conjunto de vértices S y el número máximo de saltos k. Si estamos en un estado (S, k) queremos determinar S_k tal que la suma de los costes de $coste(S_k, S/setminusS_k$ y $z(S/setminusS_k, k-1)$ sea minimo. Escrito de otra manera:

$$\begin{aligned} &z(S, \emptyset) = 0, \quad k = 1, \ldots, n \\ &z(S, 1) = coste(S, \emptyset) = \sum_{i \in S} c_{0,i}, \quad S \subseteq V \setminus \{0\} \\ &z(S, k) = min\{coste(S_k, S/setminusS_k + z(S/setminusS_k, k-1)\}, \\ &\qquad k = 2, \ldots, H, \quad S \subseteq V \setminus \{0\}, \quad S_k \subseteq S \end{aligned} \tag{2.4}$$

Este problema tiene una gran cantidad de estados y requiere un tiempo $\mathcal{O}(H3^n)$ para resolver una instancia del problema HMST lo cual lo hace no apto para ser utilizado. Sin embargo podemos añadir restricciones a las transiciones entre estados que reduzcan los estados vecinos lo suficiente para que su exploración sea computacionalmente abordable. En concreto dos restricciones:

1. Restricción sobre el espacio de estados. Dado un entero $d > 0$ y una solución inicial $S_{inicial}$ que podemos particionar como $\{S_1, \ldots, S_H\}$ (cada elemento de la partición contiene los nodos en cada nivel) solo se consideraran los estados (S, k) tal que $|S \setminus \{S_1 \cup \cdots \cup S_k\}| + |\{S_1 \cup \cdots \cup S_k\} \setminus S| <= d$, para $k = 1, \ldots, H-1$. Dicho en una manera más intuitiva: si existiera una pared entre cada nivel de los vértices solo podriamos pasar d vértices por cada pared.

2. Restriccion sobre la transición de estados. Dada una solución inicial $S_{inicial}$ que podemos particionar como $\{S_1, \ldots, S_H\}$ solo se restringen las transiciones desde un estado $(S^*, k-1)$ a un estado (S, k) tal que $Sneq\{S_1, \ldots, S_k\}$. Sea $N1$ y $N1^*$ los vértices (el conjunto de los vértices no el número de los mismos) nuevos en S y S^* respectivamente en relación a los que existen en $\{S_1, \ldots, S_k\}$ y $\{S_1, \ldots, S_{k-1}\}$. Sea $N2$ y $N2^*$ los vértices que desaparecen en S y S^* frente a los que había en $\{S_1, \ldots, S_k\}$ y $\{S_1, \ldots, S_{k-1}\}$. Solo se permiten las transiciones que cumplen al menos una de las condiciones siguientes:

$$\begin{aligned} &1. \quad S^* = \{S_1, \ldots, S_{k-1}\} \\ &2. \quad N1 = N1^* \text{ y } N2 = N2^* \end{aligned} \tag{2.5}$$

Se intentará mostrar la explicación de estas restricciones utilizando para demostrar la validez/invalidez de la transición ilustrada en la figura 2.1.

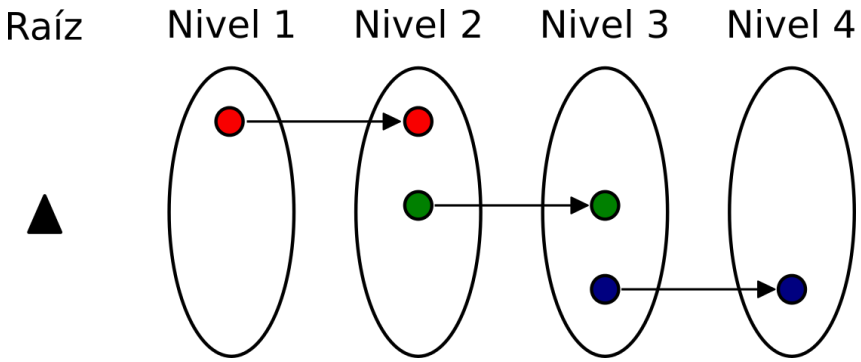

Figura 2.1: Ejemplo de una transición (adaptado de [18])

Si suponemos $d = 1$ las transiciones de la figura 2.1 cumplen la primera de las restricciones 1. Para $k = 1$ tenemos que el vértice rojo abandona el primer nivel es decir $|S \setminus S_1| = 1$ (aqui S se refiere solo a los vértices originalmente en el nivel 1) y nada entre por tanto $|S_1 \setminus S| = 0$. En el caso de $k = 2$ solo la bola verde abandona el conjunto de los niveles 1 y 2, es decir, $|S \setminus S_1 \cup S_2| = 1$ los intercambios internos como el vértice rojo no afectan a la comprobación. Es fácil comprobar que para $k = 3$ también se cumple.

Para comprobar la restricción sobre las transiciones 2, nos referiremos por S_i a los elementos en el nivel i originalmente y por S_i^* a los elementos en el nivel i tras los cambios. Empezamos por el estado $(S_1^*, 1)$ que transiciona a $(\{S_1^* \cup S_2^*\}, 2)$. Lo primero que comprobamos es si se aplican las restricciones, como $\{S_1^* \cup S_2^*\} \neq \{S_1 \cup S_2\}$ (ya que el vértice verde no está en $\{S_1^* \cup S_2^*\}$) si que se aplican. La primera parte de la ecuación 2.5 no se cumple ya que $\{S_1^*\} \neq \{S_1^*\}$ (por el vértice rojo). La segunda parte 2.5 no se cumple tampoca ya que en el primer estado $(S_1^*, 1)$ está saliendo el vértice rojo y en cambio en $(\{S_1^* \cup S_2^*\}, 2)$ es el verde, es decir no es el mismo vértice como obliga la restricción. Por tanto este intercambio no es posible con en estas restricciones lo cual implica que no existe una relación de vecindad entre la solución previa a los intercambios y la posterior a los mismos.

2.3.3. Vecindarios

Si definimos una clase de intercambios tambien tendríamos un vecindario de estados que podemos explorar en busca de mejores soluciones. En este trabajo vamos a abordar los desarrollados por Gouveia et al. [18] aunque no son los únicos que existen.

El primer vecindario se conforma por los estados que distan del estado actual (recordemos que un estado es un reparto de los vértices en los diferentes niveles) por un *movimiento simple*. Definimos un movimiento simple por el cambio de nivel de un vértice. Este vecindario tiene tamaño $\mathcal{O}(nH)$. Se puede ver un ejemplo de este movimiento en la figura 2.2.

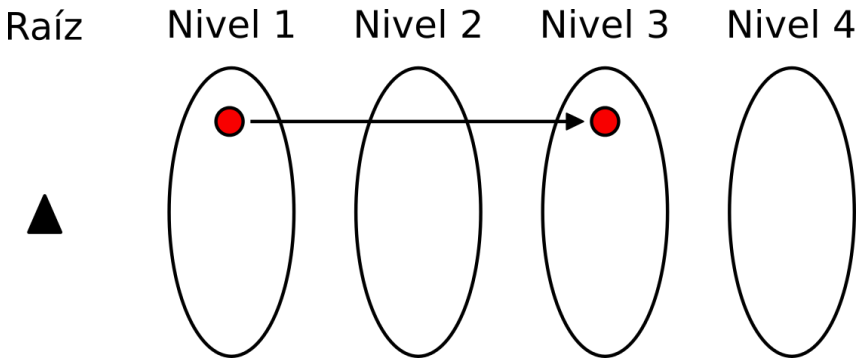

Figura 2.2: Movimiento simple (adaptado de [18])

El segundo vecindario se conforma por los estados que difieren del estado actual por un *intercambio*. Un intercambio implica que dos vértices intercambian sus niveles. Este vecindario tiene tamaño $\mathcal{O}(n^2)$. Se puede ver un ejemplo de este movimiento en la figura 2.3.

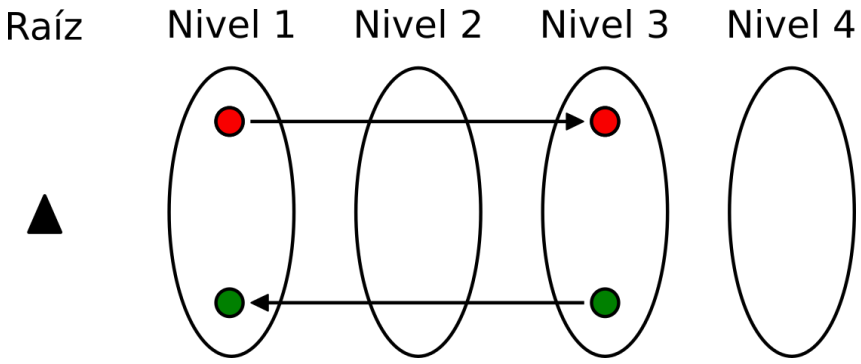

Figura 2.3: Intercambio (adaptado de [18])

Por último podemos pensar en un vecindario en la que un estado es vecino si se puede llegar mediante un movimiento simple o un intercambio. Este vecindario tiene tamaño $\mathcal{O}(n^2)$ aunque como es obvio es ligeramente más grande que el vecindario de intercambios.

Capítulo 3

Metaheurísticas

Las heurísticas utilizadas en problemas de programación entera suelen ser algorítmos vorazes, es decir, toman en cada momento la mejor decisión con respecto a un criterio dado sin tener en cuenta que otra opción peor puede, trás más pasos, llegar a mejores soluciones. Otra característica de las heurísticas es que son dependientes del problema. Las metaheurísticas por su parte son procedimientos de alto nivel (con respecto a las heurísticas), agnosticas frente al problema y cuyo objetivo es intentar aumentar la capacidad de una heurística para escapar los óptimos locales del espacio de búsqueda.

Las metaheurísticas consiguen ampliar el espacio de búsqueda eliminando de distintas maneras (según las metaheurísticas) el determinismo propio de las heurísticas. En otras palabras, las heurísticas obtienen siempre el mismo resultado dada una entrada en cambio, las metaheurísticas pueden dar diferentes resultados en cada ejecución aunque la entrada sea la misma.

Las metaheurísticas se suelen dividir en tres grande grupos:

1. Basadas en trayectoria. Son metaheurísticas que tratan de extender las habilidades de heurísticas de búsqueda. Algunos ejemplos pueden ser recocido simulado (*Simulated Annealing* (SA)), búsqueda tabu (*Tabu search* (TS)) o búsqueda local guiada (*Guided local search (GLS)*).

2. Basadas en poblaciones. Son metaheurísitcas que mantienen una población de soluciones en la que sus elementos son seleccionados y combinados para encontrar mejores soluciones. Ejemplos de esta clas de metaheurísticas son algorítmos genéticos (GA), busqueda dispersa (*Scatter search*) o algorítmos meméticos.

3. Basadas en construcción. En estas metaheurísticas se trata de construir (no mejorar) una solución de gran calidad. Podemos destacar GRASP (*Greedy randomized adaptive search procedure*) o los algorítmos de colonias de hormigas (ACO) como miembros de esta clase.

Para este trabajo se han utilizado dos metaheurísticas: Simulated annealing y Grasp.

3.1. Recocido Simulado (*Simulated annealing*)

Este algorítmo metaheurístico, debe su nombre al proceso de enfriamiento lento de metales con el fin de que estos cristalizen obteniendo en el proceso mejores propiedades. Su creación se debe al trabajo de Kirkpatric et al. [22] y surge de una adaptación del algorítmo de Metropolis et al. [24].

La idea es que la transición entre diferentes soluciones al problema este gobernada por una probabilidad de transición. Si en el vecindario de la solución actual se encuentra una mejor solución esta probabilidad será 1. En cambio si esta nueva solución es peor se realizará la transición con probabilidad:

$$e^{-\frac{|f_{act}-f_{cand}|}{T}} \tag{3.1}$$

Donde f_{act} es el valor de la función objetivo con la solución actual, f_{cand} el de la solución candidata y T es la *temperatura*.

La temperatura varía la facilidad con la que se aceptan transiciones a soluciones con peor valor para la función objetivo. Esta temperatura se reduce progresigamente consiguiendo que el algorítmo converja. Por un lado se debe escoger la temperatura inicial y por otro el esquema de enfriamiento y la tasa de su reducción. Una temperatura inicial demasiado alta provoca que todas las transiciones se acepten complicando que el algorítmo converga hacia un óptimo (aunque sea local) en cambio, si es demasiado baja se parecerá demasiado a una heurística voraz y no será capaz de escapar de los óptimos locales. Si se conoce la variacióm máxima posible para la función objetivo se puede fijar la temperatura inicial para que resulte en una determinada probabilidad p_0:

$$T_{inicial} = -\frac{max(|f_{act} - f_{cand}|)}{ln(p_0)} \tag{3.2}$$

Si no se conoce se debe buscar una temperatura inicial mediante pruebas.En cuanto a los esquemas de enfriamiento se puede distinguir entre enfriamiento lineal:

$$T = T_{inicial} - \alpha \text{número\_de\_transiciones} \tag{3.3}$$

o geométrico:

$$T = T_{inicial}\alpha^{\text{número\_de\_transiciones}} \tag{3.4}$$

Habitualmente se utiliza el enfriamiento geométrico ya que no requiere fijar un número de máximo de iteraciones ya que la temperatura tiende a 0 y el algorítmo finaliza cuando se desciende de una temperatura prefijada. La tasa de reducción α suelte tomar valores entre 0.7 y 0.999 para que el enfriamiento sea lo suficientemente lento.

Por último, es conveniente que se realizen varias transiciones en cada temperatura con el fin de que el algoritmo pueda estabilizarse. El número de transiciones puede ser fijo para cada temperatura o aumentar cuando la temperatura sea menor con el fin de explorar los óptimos locales. El pseudocódigo del algoritmo usando enfriamiento geométrico y un número de transiciones por temperatura fijo se puede ver en 3.

Algorithm 3 Recocido Simulado (*simulated annealing*)

```
1: mejorSolución = soluciónInicial
2: {valor() devuelve el valor objetivo de la solución}
3: mejorValor = valor(mejorSolución)
4: T_inicial = T_0
5: T_final = T_f
6: T = T_inicial
7: α = α_0
8: transiciones = n {número de transiciones por temperatura}
9: while T > T_final do
10:    t = 0
11:    while t < transiciones do
12:       Generar r, un número aleatorio en [0,1]
13:       soluciónCandidata = vecino(mejorSolución) {vecino(x) devuelve un vecino al azar
          de x}
14:       variación = mejorValor - valor(soluciónCandidata)
15:       if variación < 0 then
16:          mejorSolución = soluciónCandidata
17:          mejorValor = valor(soluciónCandidata)
18:       else if e^(-|variación|/T) > r then
19:          mejorSolución = soluciónCandidata
20:          mejorValor = valor(soluciónCandidata)
21:       end if
22:       t=t+1
23:    end while
24:    T = Tα
25: end while
```

3.2. GRASP (*Greedy randomized adaptive search procedure*)

GRASP (*Greedy randomized adaptive search procedure*) es una metaheurística multi-arranque, es decir, utiliza soluciones iniciales variadas con el fin de explorar el espacio de soluciones de manera más eficiente. Es introducida por Feo T y Resende M en 1995 [13][12] y desde entonces es una de las metaheurísticas más usadas.

GRASP es un procedimiento iterativo con dos etapas: construcción y búsqueda local. En la fase de construcción es habitual utilizar una heurística voraz aleatorizada. Una heurística voraz aleatorizada se caracteríza por introducir en cada paso un elemento al azar de una lista que contiene los n elementos menos costosos (si no fuera aleatorizad se eligiria la opción menos costosa). A la lista de elementos candidatos se la conoce como lista de candidatos restringida (RCL). Al tener un componente estocástico este tipo de heurística nos permite conseguir varias soluciones iniciales y a cada una de ellas se le aplica un método de búsqueda local.

Como criterio de parada normalmente se fija un número máximo de iteraciones MAX_{it} y se escoge la mejor de las soluciones obtenidas. Por tanto se tienen dos parámetros: el tamaño de la lista RCL y el número de iteraciones. Se puede ver una versión en pseudo-código en 4.

Algorithm 4 GRASP (*Greedy randomized adaptive search procedure*)

```
1: Solución_mejor = ∅
2: Coste_mejor = ∞
3: for k=1,...,MAX_it do
4:    Solución = GreedyAleatorizado(Tamaño_RCL)
5:    Solución = Solución)
6:    Coste = Valor(Solución)
7:    if Coste < Coste_mejor then
8:       Coste_mejor = Coste
9:       Solución_mejor = Solución
10:   end if
11: end for
```

Uno de los problemas que tiene GRASP es que con un tamaño de lista RCL fijo puede que no sea posible obtener la solución óptima [26]. En [30] se analizaron los resultados obtenidos con diferentes tamaños de lista RCL. Se demostró, que GRASP no es capaz de encontrar buenas soluciones si el valor medio de las soluciones iniciales es muy bajo aunque tengan mucha varianza como en el caso de listas RCL muy grandes. Además tampoco encuentra buenas soluciones si la varianza en las soluciones es muy pequeña aunque el valor medio sea alto como cuando el tamaño de la lista RCL es 1. Es necesario

realizar pruebas para escoger un tamaño que garantize una varianza y un valor medio de las soluciones iniciales relativamente altos. Otra opción es utilizar algun método que seleccione el tamaño de la lista automáticamente como *Reactive GRASP* [28].

Capítulo 4

Resultados Computacionales

En este capítulo, se mostrarán los resultados obtenidos mediante el uso del ordenador tanto para los modelos exáctos como para los algoritmos heurísticos.

El conjunto de datos que se ha utilizado pertenece a la OR-Library [5][4]. En concreto se han utilizado los datos pertenecientes al problema del mínimo árbol de Steiner. Estos datos contienen coordenadas de puntos en dos dimensiones. Se ha decidido que en principio toda conexión es posible y que los costes de las aristas sean las distancias euclideas entre los puntos. Los datos estan agrupados en archivos con problemas numerados con la misma cantidad de puntos como se indica en la tabla 4.1.

archivo	número de puntos	archivo	número de puntos
estein10	10	estein60	60
estein20	20	estein70	70
estein30	30	estein80	80
estein40	40	estein90	90
estein50	50	estein100	100

Cuadro 4.1: Archivos en OR-Library para el problema del mínimo árbol de Steiner

Debido a la gran cantidad de problemas posibles que se pueden generar con los datos (combinando el número del problema, el nodo raíz y la cantidad máxima de saltos H), se ha decidido escoger un subconjunto de problemas para probar las heurísticas. Este subconjunto esta indicado en la tabla 4.2.

archivo	problema	raíz	H	archivo	problema	raíz	H
estein10.txt	3	3	2	estein50.txt	3	3	2
estein10.txt	3	3	3	estein50.txt	3	3	3
estein10.txt	3	3	4	estein50.txt	3	3	4
estein10.txt	3	3	5	estein50.txt	3	3	5
estein10.txt	3	8	2	estein50.txt	3	8	2
estein10.txt	3	8	3	estein50.txt	3	8	3
estein10.txt	3	8	4	estein50.txt	3	8	4
estein10.txt	3	8	5	estein50.txt	3	8	5
estein10.txt	4	3	2	estein50.txt	4	3	2
estein10.txt	4	3	3	estein50.txt	4	3	3
estein10.txt	4	3	4	estein50.txt	4	3	4
estein10.txt	4	3	5	estein50.txt	4	3	5
estein10.txt	4	8	2	estein50.txt	4	8	2
estein10.txt	4	8	3	estein50.txt	4	8	3
estein10.txt	4	8	4	estein50.txt	4	8	4
estein10.txt	4	8	5	estein50.txt	4	8	5
estein20.txt	3	3	2	estein60.txt	3	3	2
estein20.txt	3	3	3	estein60.txt	3	3	2
estein20.txt	3	3	4	estein60.txt	3	3	2
estein20.txt	3	3	5	estein60.txt	3	3	2
estein20.txt	3	8	2	estein60.txt	3	3	2
estein20.txt	3	8	3	estein60.txt	3	3	2
estein20.txt	3	8	4	estein60.txt	3	3	2
estein20.txt	3	8	5	estein60.txt	3	3	2
estein20.txt	4	3	2	estein60.txt	3	3	2
estein20.txt	4	3	3	estein60.txt	3	3	2
estein20.txt	4	3	4	estein60.txt	3	3	2
estein20.txt	4	3	5	estein60.txt	3	3	2
estein20.txt	4	8	2	estein60.txt	3	3	2
estein20.txt	4	8	3	estein60.txt	3	3	2
estein20.txt	4	8	4	estein60.txt	3	3	2
estein20.txt	4	8	5	estein60.txt	3	3	2
estein30.txt	3	3	2	estein70.txt	3	3	2
estein30.txt	3	3	3	estein70.txt	3	3	2
estein30.txt	3	3	4	estein70.txt	3	3	2
estein30.txt	3	3	5	estein70.txt	3	3	2
estein30.txt	3	8	2	estein70.txt	3	3	2
estein30.txt	3	8	3	estein70.txt	3	3	2
estein30.txt	3	8	4	estein70.txt	3	3	2
estein30.txt	3	8	5	estein70.txt	3	3	2
estein30.txt	4	3	2	estein70.txt	3	3	2

estein30.txt	4	3	3	estein70.txt	3	3	2
estein30.txt	4	3	4	estein70.txt	3	3	2
estein30.txt	4	3	5	estein70.txt	3	3	2
estein30.txt	4	8	2	estein70.txt	3	3	2
estein30.txt	4	8	3	estein70.txt	3	3	2
estein30.txt	4	8	4	estein70.txt	3	3	2
estein30.txt	4	8	5	estein70.txt	3	3	2
estein40.txt	3	3	2	estein80.txt	3	3	2
estein40.txt	3	3	3	estein80.txt	3	3	2
estein40.txt	3	3	4	estein80.txt	3	3	2
estein40.txt	3	3	5	estein80.txt	3	3	2
estein40.txt	3	8	2	estein80.txt	3	3	2
estein40.txt	3	8	3	estein80.txt	3	3	2
estein40.txt	3	8	4	estein80.txt	3	3	2
estein40.txt	3	8	5	estein80.txt	3	3	2
estein40.txt	4	3	2	estein80.txt	3	3	2
estein40.txt	4	3	3	estein80.txt	3	3	2
estein40.txt	4	3	4	estein80.txt	3	3	2
estein40.txt	4	3	5	estein80.txt	3	3	2
estein40.txt	4	8	2	estein80.txt	3	3	2
estein40.txt	4	8	3	estein80.txt	3	3	2
estein40.txt	4	8	4	estein80.txt	3	3	2
estein40.txt	4	8	5	estein80.txt	3	3	2

Cuadro 4.2: Problemas seleccionados para probar los modelos exactos y las heurísticas

Por la gran cantidad de problemas que se han realizado se mostrarán en el texto principal solo el primer cuarto de problemas corresponidiente a cada archivo. Estos problemas siempre son el tercer problema del archivo indicado y su raíz es el vértice 3. El resto se podrán encontrar en el apéndice.

Cuando los tiempos de ejecución han sido demasiado largos se ha decidido no realizar las pruebas en los archivos con más grandes. Normalmete con los mayores a 40 puntos aunque hay excepciones.

Con el fin de reducir los tamaños de los problemas, y por tantos los tiempos de los diferentes algoritmos, se ha aplicado un procedimiento de eliminación de arcos [15]. Si $c_{ij} > c_{0j}$ entonces la solución óptima no utiliza el arco (i, j) y si $c_{ij} = c_{0j} (i \neq 0)$, entonces hay una solución óptima sin el arco (i, j). Es decir se pueden eliminar las aristas que cumplan $c_{ij} >= c_{0j}$. Esta eliminación provoca que para una misma cantidad de puntos si la raíz es un punto más centrado se eliminen más arcos que si fuera un punto más periférico [9]. Para tener en cuenta este hecho en la tabla A.6 del apéndice se mostrará

una medida de centralidad del nodo raíz de cada problema. La cercanía de un nodo se define como la inversa de la suma de los caminos más cortos al resto de nodos 4.1.

$$C(x) = (\frac{1}{\sum_{v \in V} d(x,v)}) \\ \text{donde } d(x,v) \text{ es la distancia directa entre x y v} \tag{4.1}$$

En nuestro caso hemos utilizado la cercanía del nodo raiz pero usando los caminos directos entre la misma medida del nodo que de mayor resultado 4.2.

$$C_{modificada}(x) = \frac{1}{\sum_{v \in V} d(x,v)} \\ \text{donde } d(x,v) \text{ es la distancia directa entre x y v} \tag{4.2}$$

Por último se ha restado el mínimo y se ha divido por el rango para que la medida tome valores entre 0 y 1. De esta manera si el valor es 1 entonces el nodo tiene la máxima cercanía modificada de la red y si es 0 la menor.

4.1. Resultados de los modelos exactos

Para probar los modelos exactos se ha limitado el tiempo de resolución a 300 segundos, es decir que si no se encuentra la solución óptima en ese tiempo el software devuelve la mejor solución que ha encontrado. Además para acortar el tiempo de espera solo se han resuelto los problemas con un tamaño menor a 40 puntos.

Los resultados de los modelos basados en flujos multi-producto se pueden ver en la tabla 4.3 y A.1.

Podemos observar que todos los problemas se resuelven de manera óptima salvo algún caso de los problemas con 40 puntos. Entre estos cuatro modelos el MCF parace ser el que da mejores resultados aunque el MCF* tiene mejores tiempos cuando H es 2 o 3. Esto está acorde a lo que se predijo en el capítulo 1.1.

			MCF		HopMCF		MCF* H=2		MCF* H=3	
p	raíz	H	sol	t(s)	sol	t(s)	sol	t(s)	sol	t(s)
estein10.txt										
3	3	2	2.703	0.007	2.703	0.007	2.703	0.004	—	—
3	3	3	2.417	0.01	2.417	0.053	—	—	2.417	0.019
3	3	4	2.33	0.009	2.33	0.098	—	—	—	—
3	3	5	2.33	0.009	2.33	0.129	—	—	—	—
estein20.txt										
3	3	2	3.482	0.121	3.482	0.031	3.482	0.006	—	—
3	3	3	2.938	0.486	2.938	1.6	—	—	2.938	0.483
3	3	4	2.715	0.153	2.715	2.36	—	—	—	—
3	3	5	2.631	0.196	2.631	3.149	—	—	—	—
estein30.txt										
3	3	2	6.498	2.566	6.498	0.103	6.498	0.011	—	—
3	3	3	4.815	13.017	4.815	5.081	—	—	4.815	2.992
3	3	4	4.344	5.65	4.344	9.139	—	—	—	—
3	3	5	4.133	89.024	4.133	27.064	—	—	—	—
estein40.txt										
3	3	2	6.923	3.335	6.923	0.575	6.923	0.015	—	—
3	3	3	5.801	39.977	5.801	126.602	—	—	5.801	15.863
3	3	4	5.252	45.947	5.696	299.815	—	—	—	—
3	3	5	4.888	41.892	4.888	86.834	—	—	—	—

Cuadro 4.3: Soluciones obtenidas con los modelos MCF 1.1, HopMCF 1.8 y MCF* 1.3 1.4

En cuanto a los modelos que incluyen restricciones de Miller-Tucker-Zemlin los más básicos tienen sus resultados en las tablas 4.4 y A.2.

			Modelo MTZ		Modelo EMTZ	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.007	2.703	0.037
3	3	3	2.417	0.01	2.417	0.035
3	3	4	2.33	0.007	2.33	0.004
3	3	5	2.33	0.008	2.33	0.005
estein20.txt						
3	3	2	3.482	0.025	3.482	1.51
3	3	3	2.938	0.333	2.938	7.1
3	3	4	2.715	1.599	2.715	3.765
3	3	5	2.631	0.619	2.631	4.037

estein30.txt						
3	3	2	6.498	0.08	6.654	299.828
3	3	3	4.815	299.525	4.895	299.629
3	3	4	4.344	299.855	4.434	299.59
3	3	5	4.177	299.755	4.264	299.547
estein40.txt						
3	3	2	6.923	0.05	7.053	299.702
3	3	3	5.801	95.639	6.008	299.686
3	3	4	5.254	299.135	5.404	299.686
3	3	5	4.888	27.303	4.888	299.67

Cuadro 4.4: Soluciones obtenidas con los modelos MTZ 1.10 y EMTZ 1.12

Parece que el modelo MTZ tiene un mejor comportamiento que el modelo EMTZ, aún así, no superan el comportamiento del modelo MCF.

Los resultados para los modelos con elevaciones se encuentran en las tablas 4.5 y A.3.

			Modelo L1EMTZ		Modelo L2EMTZ	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.087	2.341	0.007
3	3	3	2.417	0.032	2.417	0.016
3	3	4	2.33	0.006	2.33	0.006
3	3	5	2.33	0.005	2.33	0.006
estein20.txt						
3	3	2	3.482	6.089	2.304	0.038
3	3	3	2.938	23.829	2.938	1.229
3	3	4	2.715	8.964	2.715	1.611
3	3	5	2.631	5.619	2.631	2.381
estein30.txt						
3	3	2	6.654	299.965	6.498	3.651
3	3	3	4.857	299.49	4.815	44.088
3	3	4	4.47	300.005	4.409	299.125
3	3	5	4.271	299.44	4.139	299.425
estein40.txt						
3	3	2	7.053	299.665	6.923	4.845
3	3	3	5.999	299.63	5.84	300.145
3	3	4	5.352	299.661	5.32	299.46
3	3	5	4.888	110.717	4.888	136.226

Cuadro 4.5: Soluciones obtenidas con los modelos L1EMTZ 1.22 y L2EMTZ 1.26

Ambos modelos superan al modelo del que derivan (el EMTZ) pero no al modelo de

flujos MCF.

Los resultados de los modelos con mejoras topológicas están en las tablas 4.6 y A.4.

			MTZ/ITEF		IMTZ/ITEF		REL-M	
p	raíz	H	solución	tiempo	solución	tiempo	solución	tiempo
estein10.txt								
3	3	2	2.703	0.006	2.703	0.01	2.703	0.004
3	3	3	2.417	0.005	2.417	0.007	2.417	0.004
3	3	4	2.33	0.004	2.33	0.004	2.33	0.004
3	3	5	2.33	0.004	2.33	0.005	2.33	0.004
estein20.txt								
3	3	2	3.482	0.044	3.482	0.014	3.482	0.013
3	3	3	2.938	0.761	2.938	0.365	2.938	0.549
3	3	4	2.715	0.91	2.715	2.25	2.715	0.442
3	3	5	2.631	0.838	2.631	0.932	2.631	0.916
estein30.txt								
3	3	2	6.498	0.115	6.498	0.062	6.498	0.029
3	3	3	4.815	299.946	4.815	133.949	4.815	1.357
3	3	4	4.344	299.764	4.344	299.331	4.344	14.669
3	3	5	4.211	299.42	4.133	300.041	4.133	53.253
estein40.txt								
3	3	2	6.923	0.141	6.923	0.1	6.923	0.025
3	3	3	5.801	162.265	5.801	43.198	5.801	13.337
3	3	4	5.27	299.983	5.306	299.666	5.252	38.877
3	3	5	4.888	78.516	4.892	299.181	4.888	38.421

Cuadro 4.6: Soluciones obtenidas con los modelos MTZ/ITEF 1.27, IMTZ/ITEF 1.30 y REL-M 1.31

Los modelos se comportan de manera similar entre ellos aunque el que da mejores resultados es el REL-M. Esto coincide con los resultados de la fuente [3]. A pesar de ello el modelo MCF parece ser el mejor hasta el momento.

Finalmente los resultados para los modelos que usan las ideas de Sherali y Driscoll se encuentran en las tablas 4.7 y A.5.Ambos modelos dan buenos resultados pero a pesar de ello no son mejores que el modelo de flujos MCF.

De todos los modelos propuestos aquellos que parecen resolver los problemas de una manera más rápida son el MCF y el MCF* para H=2 o H=3. A pesar de que en este caso los modelos exactos den buenos resultados las cosas pueden cambiar para tamaños de problema más grandes para lo que se requeriria la aplicación de algun método heurístico o metaheurístico.

			HMST-SD		HMST-SD/ITEF	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.006	2.703	0.005
3	3	3	2.417	0.012	2.417	0.008
3	3	4	2.33	0.007	2.33	0.005
3	3	5	2.33	0.007	2.33	0.005
estein20.txt						
3	3	2	3.482	0.013	3.482	0.02
3	3	3	2.938	1.801	2.938	1.471
3	3	4	2.715	2.747	2.715	1.293
3	3	5	2.631	1.833	2.631	1.79
estein30.txt						
3	3	2	6.498	0.038	6.498	0.072
3	3	3	4.815	4.378	4.815	4.87
3	3	4	4.344	78.109	4.344	104.717
3	3	5	4.133	148.076	4.152	299.28
estein40.txt						
3	3	2	6.923	0.015	6.923	0.063
3	3	3	5.801	4.516	5.801	10.828
3	3	4	5.252	61.218	5.252	63.672
3	3	5	4.888	41.453	4.888	99.718

Cuadro 4.7: Soluciones obtenidas con los modelos HMST-SD 1.32 y HMST-SD/ITEF 1.33

4.2. Resultados de las Heurísticas

En primer lugar comprobamos los resultados que se obtienen con la heurística de ahorros 1.

archivo	problema	raiz	H	solución	tiempo
estein10.txt	3	3	2	2.794	0.149
estein10.txt	3	3	3	2.561	0.010
estein10.txt	3	3	4	2.561	0.010
estein10.txt	3	3	5	2.561	0.009
estein20.txt	3	3	2	4.232	0.264
estein20.txt	3	3	3	3.260	0.323
estein20.txt	3	3	4	2.844	0.324
estein20.txt	3	3	5	2.750	0.330
estein20.txt	3	8	2	4.594	0.193
estein30.txt	3	3	2	8.723	1.540
estein30.txt	3	3	3	5.555	1.744

estein30.txt	3	3	4	4.696	1.790
estein30.txt	3	3	5	4.323	1.866
estein40.txt	3	3	2	7.785	3.702
estein40.txt	3	3	3	6.144	4.410
estein40.txt	3	3	4	5.478	4.673
estein40.txt	3	3	5	5.289	4.508
estein50.txt	3	3	2	12.380	11.141
estein50.txt	3	3	3	7.909	13.915
estein50.txt	3	3	4	7.096	14.931
estein50.txt	3	3	5	6.316	15.424
estein60.txt	3	3	2	15.564	24.450
estein60.txt	3	3	3	9.257	31.494
estein60.txt	3	3	4	7.913	34.068
estein60.txt	3	3	5	7.028	36.266
estein70.txt	3	3	2	17.388	39.710
estein70.txt	3	3	3	10.522	51.737
estein70.txt	3	3	4	7.668	57.647
estein70.txt	3	3	5	7.226	58.183
estein80.txt	3	3	2	17.185	73.249
estein80.txt	3	3	3	9.898	101.079
estein80.txt	3	3	4	8.755	96.262
estein80.txt	3	3	5	8.147	94.073

Cuadro 4.8: Soluciones obtenidas con la heurística de ahorros EW

En las tablas 4.8 y A.6 podemos comprobar que la heurística de ahorros consigue soluciones iniciales de una calidad alta (en los casos que hemos comprobado relativamente cercanas al óptimo) aunque invierte una buena cantidad de tiempo. Esto puede resultar un problema para las heurísticas de búsqueda que se basan en aplicar heurísticas de manera repetida como los algorítmos de segundo orden 2 descritos en el apartado 2.2. Por este motivo se ha decidido aplicar las diferentes versiones de algoritmos de segundo orden únicamente a los problemas con 40 vértices o menos. Los resultados se pueden ver en las tablas 4.9 y A.7.

			Inhib. simple		Inhib. pareada		Incl. limitada		Incl. general	
p	raiz	H	sol	t	sol	t	sol	t	sol	t
estein10.txt										
3	3	2	2.794	0.077	2.794	0.435	2.794	0.151	2.794	0.175
3	3	3	2.561	0.068	2.561	0.775	2.561	0.137	2.561	0.115
3	3	4	2.561	0.054	2.561	0.754	2.561	0.13	2.561	0.118
3	3	5	2.561	0.054	2.561	0.804	2.561	0.166	2.561	0.151

estein20.txt										
3	3	2	4.196	5.24	4.196	292.91	4.128	32.796	3.942	46.809
3	3	3	3.185	11.916	3.185	710.892	3.185	27.136	3.185	35.257
3	3	4	2.834	8.695	2.834	658.621	2.842	19.369	2.844	18.555
3	3	5	2.75	5.095	2.75	436.188	2.75	9.421	2.75	19.027
estein30.txt										
3	3	2	7.06	179.743	7.06	8888.381	7.362	313.846	7.333	617.896
3	3	3	4.908	128.159	4.815	11902.568	4.908	357.712	4.908	585.789
3	3	4	4.462	176.668	4.424	13224.491	4.604	210.2	4.58	583.558
3	3	5	4.214	200.428	4.214	14970.705	4.323	134.978	4.251	612.124
estein40.txt										
3	3	2	7.27	465.09	—	—	7.39	1177.95	7.249	2680.158
3	3	3	5.944	590.438	—	—	5.937	2151.278	5.967	2531.725
3	3	4	5.373	371.862	—	—	5.373	1155.617	5.444	1241.656
3	3	5	5.052	643.573	—	—	5.134	1426.866	5.024	3103.015

Cuadro 4.9: Soluciones obtenidas con los algoritmos de segundo orden

Se puede comprobar que los tiempos son bastante elevados y en algunos casos se ha desistido de finalizar la búsqueda (se ha señalado con —). Si que existe una cierta mejora aunque no merece la pena usar estas heurísticas por la cantidad de tiempo que necesitan.

			Mov. simple		Intercambio		Ambos	
p	raíz	H	solución	tiempo	solución	tiempo	solución	tiempo
estein10.txt								
3	3	2	2.703	0.088 (0.193)	2.794	0.099 (0.15)	2.703	0.079 (0.138)
3	3	3	2.465	0.004 (0.011)	2.561	0.003 (0.015)	2.465	0.008 (0.009)
3	3	4	2.465	0.01 (0.019)	2.561	0.003 (0.01)	2.465	0.009 (0.009)
3	3	5	2.465	0.014 (0.025)	2.561	0.004 (0.01)	2.465	0.011 (0.01)
estein20.txt								
3	3	2	3.557	0.015 (0.262)	3.833	0.244 (0.285)	3.482	0.078 (0.256)
3	3	3	3.143	0.039 (0.314)	3.241	0.159 (0.458)	3.073	0.178 (0.283)
3	3	4	2.725	0.059 (0.329)	2.844	0.069 (0.298)	2.725	0.15 (0.48)
3	3	5	2.631	0.055 (0.321)	2.75	0.077 (0.363)	2.631	0.157 (0.341)

estein30.txt								
3	3	2	6.678	0.054 (1.416)	7.579	0.925 (1.42)	6.59	0.603 (1.339)
3	3	3	5.07	0.109 (1.863)	5.206	1.62 (1.943)	4.908	0.932 (1.828)
3	3	4	4.696	0.056 (1.885)	4.604	0.669 (1.904)	4.604	0.723 (1.851)
3	3	5	4.25	0.152 (1.911)	4.323	0.404 (1.9)	4.232	0.778 (1.893)
estein40.txt								
3	3	2	6.923	0.116 (3.898)	7.218	2.173 (4.013)	6.923	0.447 (3.918)
3	3	3	6.048	0.175 (4.836)	6.094	2.912 (4.472)	6.045	1.647 (4.53)
3	3	4	5.431	0.238 (4.677)	5.455	1.831 (4.533)	5.407	3.45 (4.774)
3	3	5	5.243	0.327 (4.897)	5.287	2.028 (4.603)	5.239	3.496 (4.582)
estein50.txt								
3	3	2	9.424	0.644 (11.166)	10.475	9.382 (11.253)	9.31	3.259 (11.755)
3	3	3	7.228	0.894 (14.03)	7.442	21.487 (14.11)	7.179	9.266 (17.547)
3	3	4	6.697	1.765 (14.915)	6.75	14.422 (14.852)	6.456	19.909 (15.666)
3	3	5	6.27	0.88 (15.825)	6.249	6.916 (15.443)	6.239	5.851 (16.004)
estein60.txt								
3	3	2	11.072	1.014 (24.45)	13.331	28.937 (23.484)	10.744	11.478 (23.412)
3	3	3	8.263	1.643 (32.189)	8.408	44.596 (30.188)	7.995	13.794 (30.428)
3	3	4	7.332	2.538 (35.298)	7.372	73.022 (33.519)	6.991	29.82 (33.626)
3	3	5	6.905	0.891 (34.823)	6.892	14.212 (32.809)	6.62	17.829 (32.744)
estein70.txt								
3	3	2	11.7	2.231 (47.152)	13.997	56.347 (39.74)	11.658	10.318 (40.003)
3	3	3	8.48	4.693 (54.768)	9.143	118.809 (52.098)	8.289	45.014 (52.61)
3	3	4	7.579	2.351 (60.326)	7.532	37.05 (57.516)	7.532	41.867 (57.917)

3	3	5	7.108	1.437 (61.602)	7.077	51.653 (58.029)	7.077	45.262 (58.238)
estein80.txt								
3	3	2	12.309	3.46 (67.567)	13.593	111.768 (69.195)	11.986	29.032 (65.798)
3	3	3	9.418	3.09 (91.039)	9.281	149.724 (86.785)	9.207	105.776 (87.477)
3	3	4	8.216	10.305 (103.238)	8.014	216.074 (91.318)	7.893	126.901 (92.313)
3	3	5	8.103	2.842 (94.591)	8.014	148.326 (93.138)	8.008	98.941 (93.875)

Cuadro 4.10: Soluciones obtenidas con los vecindarios basados en un modelo de programación dinámica

En cuanto a los vecindarios basados en un nodelo de programación dinámica podemos observar sus resultados en las tablas 4.10 y A.8.

Entre paréntesis se encuentran el tiempo que tarda la heurística de ahorros. Se puede comprobar que son métodos de búsqueda local bastante rápidos y que mejoran los resultados de la heurística base de una manera percibible. Podemos apreciar que el mejor método es el que combina movimientos simples e intercambios.

4.3. Resultados de las Metaheurísticas

4.3.1. SA

Los resultados obtenidos con simulated annealing se pueden encontrar en la tabla 4.11 y en la tabla A.9. El método de búsqueda usado a sido el movimiento simple de los vecidarios basados en un modelo de programación dinámica. La temperatura inicial ha tomado valor 1 y la final 0.001. El enfriamiento ha sido de tipo geométrico con una tasa de enfriamiento de valor 0.999. Se han realizado 10 transiciones por valor de la temperatura.

problema	raiz	H	solución	tiempo
estein10.txt				
3	3	2	2.703	28.226 (0.281)
3	3	3	2.485	27.805 (0.02)
3	3	4	2.485	30.03 (0.017)
3	3	5	2.457	31.005 (0.019)

estein20.txt				
3	3	2	3.578	53.665 (0.495)
3	3	3	3.122	59.621 (0.595)
3	3	4	2.725	63.263 (0.6)
3	3	5	2.715	64.045 (0.649)
estein30.txt				
3	3	2	6.498	86.941 (2.584)
3	3	3	4.859	98.821 (3.401)
3	3	4	4.773	104.682 (3.267)
3	3	5	4.639	106.552 (3.565)
estein40.txt				
3	3	2	7.001	135.727 (6.118)
3	3	3	6.068	156.691 (7.841)
3	3	4	5.899	162.917 (8.084)
3	3	5	5.661	167.221 (9.807)
estein50.txt				
3	3	2	9.528	172.343 (18.642)
3	3	3	7.323	223.888 (24.891)
3	3	4	6.91	238.668 (24.118)
3	3	5	6.334	250.797 (20)
estein60.txt				
3	3	2	11	247.355 (35.941)
3	3	3	8.319	306.732 (54.83)

3	3	4	7.384	332.339 (38.451)
3	3	5	7.129	300.781 (55.297)
estein70.txt				
3	3	2	11.5	307.219 (61.362)
3	3	3	8.287	387.531 (93.086)
3	3	4	7.368	417.671 (103.577)
3	3	5	7.44	426.961 (78.121)
estein80.txt				
3	3	2	11.974	394.462 (89.423)
3	3	3	9.314	487.895 (108.479)
3	3	4	8.277	536.659 (148.657)
3	3	5	8.32	558.536 (154.459)

Cuadro 4.11: Soluciones obtenidas con Simulated Annealing

Entre paréntesis encontramos el tiempo que ha tardado la heurística base. Los resultados son muy buenos ya que se alcanza el óptimo en varios problemas en un tiempo razonablemente corto.

4.3.2. GRASP

Para la metaheurística grasp se han utilizado dos tamaños de lista de candidatos restringida (RCL). El número de iteraciones ha sido de 20 en todas las pruebas (esto significa 20 soluciones iniciales). El método de búsqueda local usado ha sido el de movimiento simple e itercambios de los vecindarios basados en un modelo de programación dinámica. Se ha escogido este método por ser el que mejor resultados había dado. Los resultados se encuentran en la tabla 4.12 y en la tabla A.10.

			Tamaño RCL = 3		Tamaño RCL = 5	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.574	2.703	1.043
3	3	3	2.465	0.464	2.465	0.909
3	3	4	2.465	0.519	2.383	0.956
3	3	5	2.437	0.566	2.411	1.009
estein20.txt						
3	3	2	3.482	6.797	3.482	12.881
3	3	3	2.938	9.659	2.938	19.102
3	3	4	2.715	9.092	2.715	18.565
3	3	5	2.631	10.592	2.631	20.637
estein30.txt						
3	3	2	6.498	40.51	6.498	70.303
3	3	3	4.815	63.213	4.815	124.17
3	3	4	4.507	64.972	4.344	126.992
3	3	5	4.232	68.917	4.232	129.738
estein40.txt						
3	3	2	6.923	95.624	6.923	189.595
3	3	3	5.832	173.803	5.815	202.096
3	3	4	5.365	188.782	5.356	186.392
3	3	5	5.146	169.919	5.018	192.047
estein50.txt						
3	3	2	9.181	313.094	9.181	334.978
3	3	3	7.021	553.313	7.023	495.187
3	3	4	6.374	628.686	6.342	525.594
3	3	5	6.036	620.198	5.984	527.441
estein60.txt						
3	3	2	10.744	727.454	10.744	619.865
3	3	3	7.963	1045.591	7.871	1158.104
3	3	4	6.921	1368.802	6.826	1350.078
3	3	5	6.41	1314.271	6.41	1480.899
estein70.txt						
3	3	2	11.456	1118.269	11.456	1129.148
3	3	3	8.16	2441.529	8.089	2266.131
3	3	4	7.236	2281.714	7.178	2839.532
3	3	5	6.882	2656.255	6.881	2735.298

Cuadro 4.12: Soluciones obtenidas con GRASP

Los resultados son los mejores entre los métodos heurísticos usados aunque el tiempo puede resultar excesivo y es por ello que no se ha aplicado a las instancias de 80 puntos.

Capítulo 5

Conclusiones

En este documento se ha realizado un repaso a los diferentes métodos que existen para resolver el problema del mínimo árbol generador con restricciones de salto (HMST, *Hop-constrained minimun spanning tree problem*).

Se ha mostrado que los métodos de resolución exacta pueden resolver problemas de un tamaño moderado (con un número de nodos $<= 40$) en un tiempo razonable (inferior a 5 minutos). Con las pruebas realizadas los modelos más rápidos parecen ser los modelos de flujo multi-producto aunque esto no asegura que con instancias reales vayan a serlo.

Sin embargo los métodos de resolución exacta tienen su límite. No solo porque para instancias grandes sus tiempos son prohibitivos si no que ademas el software que los aplica también puede tener un coste prohibitivo. Por ello es necesario realizar una evaluación de los métodos heurísticos y metaheurísticos disponibles.

En el caso de las heurísticas nos econtramos con una heurística de construcción que da unos resultados bastante buenos. En cuanto a los métodos de búsqueda local aquellos basados en la repetición de heurísticas de construcción tardan tiempos muy elevados y en cambio los que aprovechan una solución ya existente proporcionan mejores resultados.

Los mejores métodos para nuestro problema han resultado ser los métodos metaheurísticos que hemos probado: Simulated Annealing y GRASP. Siendo GRASP el mejor entre ambos aunque a cambio de invertir más tiempo.

A pesar de que se ha realizado un esfuerzo para crea un documento completo existen muchas heurísticas y modelos exactos que no han sido evaluados y podrían dar buenos resultados. Otro camino que se puede seguir resulta de hibridar diferentes metaheurísticas.

Si alguien se enfrentara a este problema en la vida real las recomendaciones variarian según el tamaño del problema y el tiempo disponible para resolverlo. Si el tamaño del problema es pequeño los métodos exactos pueden ser una buena opción. En otro caso si hay suficiente tiempo recomendaría aplicar GRASP y si no Simulated Annealing.

Apéndices

Apéndice A

Tablas de resultados extendidas

A.1. Modelos Exactos

A.2. Resultados de los modelos exactos

A.2.1. Modelos de flujo

			MCF		HopMCF		MCF*$H=2$		MCF*$H=3$	
p	raíz	H	sol	t	sol	t	sol	t	sol	t
estein10.txt										
3	3	2	2.703	0.007	2.703	0.007	2.703	0.004	—	—
3	3	3	2.417	0.01	2.417	0.053	—	—	2.417	0.019
3	3	4	2.33	0.009	2.33	0.098	—	—	—	—
3	3	5	2.33	0.009	2.33	0.129	—	—	—	—
3	8	2	2.687	0.007	2.687	0.007	2.687	0.004	—	—
3	8	3	2.42	0.01	2.42	0.054	—	—	2.42	0.02
3	8	4	2.384	0.009	2.384	0.109	—	—	—	—
3	8	5	2.365	0.015	2.365	0.128	—	—	—	—
4	3	2	2.152	0.008	2.152	0.007	2.152	0.004	—	—
4	3	3	1.944	0.021	1.944	0.054	—	—	1.944	0.02
4	3	4	1.834	0.01	1.834	0.101	—	—	—	—
4	3	5	1.826	0.01	1.826	0.126	—	—	—	—
4	8	2	2.726	0.012	2.726	0.006	2.726	0.004	—	—
4	8	3	2.216	0.069	2.216	0.074	—	—	2.216	0.034
4	8	4	2.034	0.04	2.034	0.12	—	—	—	—
4	8	5	1.929	0.021	1.929	0.133	—	—	—	—
estein20.txt										

3	3	2	3.482	0.121	3.482	0.031	3.482	0.006	—	—
3	3	3	2.938	0.486	2.938	1.6	—	—	2.938	0.483
3	3	4	2.715	0.153	2.715	2.36	—	—	—	—
3	3	5	2.631	0.196	2.631	3.149	—	—	—	—
3	8	2	3.558	0.103	3.558	0.032	3.558	0.006	—	—
3	8	3	2.923	0.371	2.923	1.742	—	—	2.923	0.531
3	8	4	2.659	0.101	2.659	2.347	—	—	—	—
3	8	5	2.586	0.101	2.586	2.997	—	—	—	—
4	3	2	3.741	0.272	3.741	0.033	3.741	0.006	—	—
4	3	3	3.029	2.068	3.029	1.867	—	—	3.029	2.559
4	3	4	2.787	1.669	2.787	2.82	—	—	—	—
4	3	5	2.651	0.404	2.651	3.501	—	—	—	—
4	8	2	4.088	0.533	4.088	0.031	4.088	0.006	—	—
4	8	3	3.112	1.378	3.112	1.933	—	—	3.112	0.961
4	8	4	2.815	3.255	2.815	2.633	—	—	—	—
4	8	5	2.682	1.156	2.682	5.366	—	—	—	—
estein30.txt										
3	3	2	6.498	2.566	6.498	0.103	6.498	0.011	—	—
3	3	3	4.815	13.017	4.815	5.081	—	—	4.815	2.992
3	3	4	4.344	5.65	4.344	9.139	—	—	—	—
3	3	5	4.133	89.024	4.133	27.064	—	—	—	—
3	8	2	5.601	2.15	5.601	0.105	5.601	0.01	—	—
3	8	3	4.485	2.22	4.485	4.835	—	—	4.485	2.383
3	8	4	4.108	2.465	4.108	7.862	—	—	—	—
3	8	5	3.958	3.565	3.958	23.516	—	—	—	—
4	3	2	6.587	5.86	6.587	0.116	6.587	0.012	—	—
4	3	3	5.059	35.313	5.059	7.471	—	—	5.059	3.938
4	3	4	4.547	26.148	4.547	13.991	—	—	—	—
4	3	5	4.301	41.172	4.301	54.158	—	—	—	—
4	8	2	6.193	3.935	6.193	0.25	6.193	0.011	—	—
4	8	3	4.814	18.259	4.814	11.264	—	—	4.814	3.272
4	8	4	4.394	15.269	4.394	19.899	—	—	—	—
4	8	5	4.214	21.878	4.214	55.801	—	—	—	—
estein40.txt										
3	3	2	6.923	3.335	6.923	0.575	6.923	0.015	—	—
3	3	3	5.801	39.977	5.801	126.602	—	—	5.801	15.863
3	3	4	5.252	45.947	5.696	299.815	—	—	—	—
3	3	5	4.888	41.892	4.888	86.834	—	—	—	—
3	8	2	7.671	7.555	7.671	0.485	7.671	0.021	—	—
3	8	3	6.24	192.172	6.831	299.4	—	—	6.24	128.091
3	8	4	5.421	116.287	5.421	69.855	—	—	—	—
3	8	5	5.112	18.819	5.112	84.62	—	—	—	—

4	3	2	7.058	37.047	7.058	0.565	7.058	0.022	—	—
4	3	3	5.37	130.947	5.37	57.601	—	—	5.37	17.784
4	3	4	4.855	251.053	4.855	115.872	—	—	—	—
4	3	5	4.748	300.93	5.009	300.025	—	—	—	—
4	8	2	6.572	15.844	6.572	0.545	6.572	0.017	—	—
4	8	3	5.159	67.156	5.159	39.293	—	—	5.159	11.629
4	8	4	4.73	86.734	4.73	62.221	—	—	—	—
4	8	5	4.536	299.785	4.814	299.435	—	—	—	—

Cuadro A.1: Soluciones obtenidas con los modelos con los modelos MCF, HopMCF y MCF*

A.2.2. Modelos con restricciones MTZ

			Modelo MTZ		Modelo EMTZ	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.007	2.703	0.037
3	3	3	2.417	0.01	2.417	0.035
3	3	4	2.33	0.007	2.33	0.004
3	3	5	2.33	0.008	2.33	0.005
3	8	2	2.687	0.005	2.687	0.056
3	8	3	2.42	0.007	2.42	0.033
3	8	4	2.384	0.007	2.384	0.013
3	8	5	2.365	0.009	2.365	0.009
4	3	2	2.152	0.015	2.152	0.035
4	3	3	1.944	0.017	1.944	0.033
4	3	4	1.834	0.025	1.834	0.029
4	3	5	1.826	0.038	1.826	0.076
4	8	2	2.726	0.01	2.726	0.072
4	8	3	2.216	0.041	2.216	0.082
4	8	4	2.034	0.059	2.034	0.053
4	8	5	1.929	0.052	1.929	0.053
estein20.txt						
3	3	2	3.482	0.025	3.482	1.51
3	3	3	2.938	0.333	2.938	7.1
3	3	4	2.715	1.599	2.715	3.765
3	3	5	2.631	0.619	2.631	4.037
3	8	2	3.558	0.017	3.558	1.665
3	8	3	2.923	0.475	2.923	5.785
3	8	4	2.659	0.42	2.659	0.715
3	8	5	2.586	0.517	2.586	1.398
4	3	2	3.741	0.035	3.741	4.515
4	3	3	3.029	0.533	3.029	8.143

4	3	4	2.787	0.957	2.787	6.554
4	3	5	2.651	0.653	2.651	1.547
4	8	2	4.088	0.034	4.088	37.183
4	8	3	3.112	1.456	3.112	40.364
4	8	4	2.815	4.494	2.815	36.422
4	8	5	2.682	4.24	2.682	15.738
estein30.txt						
3	3	2	6.498	0.08	6.654	299.828
3	3	3	4.815	299.525	4.895	299.629
3	3	4	4.344	299.855	4.434	299.59
3	3	5	4.177	299.755	4.264	299.547
3	8	2	5.601	0.035	5.601	112.964
3	8	3	4.485	2.98	4.485	299.652
3	8	4	4.108	6.63	4.108	193.171
3	8	5	3.958	7.559	3.958	106.593
4	3	2	6.587	0.085	6.753	299.936
4	3	3	5.073	299.105	5.163	299.717
4	3	4	4.547	299.78	4.551	299.701
4	3	5	4.301	299.771	4.394	299.452
4	8	2	6.193	0.045	6.293	299.639
4	8	3	4.814	120.717	4.848	299.702
4	8	4	4.394	195.092	4.475	299.702
4	8	5	4.214	108.447	4.214	299.593
estein40.txt						
3	3	2	6.923	0.05	7.053	299.702
3	3	3	5.801	95.639	6.008	299.686
3	3	4	5.254	299.135	5.404	299.686
3	3	5	4.888	27.303	4.888	299.67
3	8	2	7.671	0.07	8.238	299.811
3	8	3	6.29	300.01	6.56	299.702
3	8	4	5.569	299.691	5.612	299.639
3	8	5	5.241	299.46	5.259	299.608
4	3	2	7.058	0.105	7.809	299.686
4	3	3	5.373	299.265	5.719	299.639
4	3	4	4.931	299.665	5.084	299.577
4	3	5	4.818	299.615	4.678	299.577
4	8	2	6.572	0.07	7.117	299.577
4	8	3	5.191	299.225	5.319	299.733
4	8	4	4.845	299.615	4.994	299.561
4	8	5	4.562	299.53	4.618	299.498

Cuadro A.2: Soluciones obtenidas con los modelos MTZ y EMTZ

A.2.3. Modelos con restricciones elevadas

			Modelo L1EMTZ		Modelo L2EMTZ	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.087	2.341	0.007
3	3	3	2.417	0.032	2.417	0.016
3	3	4	2.33	0.006	2.33	0.006
3	3	5	2.33	0.005	2.33	0.006
3	8	2	2.687	0.197	2.362	0.016
3	8	3	2.42	0.17	2.42	0.042
3	8	4	2.384	0.027	2.384	0.023
3	8	5	2.365	0.012	2.365	0.024
4	3	2	2.152	0.082	1.752	0.043
4	3	3	1.944	0.304	1.944	0.053
4	3	4	1.834	0.043	1.834	0.052
4	3	5	1.826	0.078	1.826	0.048
4	8	2	2.726	0.54	1.566	0.005
4	8	3	2.216	1.414	2.216	0.238
4	8	4	2.034	0.825	2.034	0.232
4	8	5	1.929	1.191	1.929	0.107
estein20.txt						
3	3	2	3.482	6.089	2.304	0.038
3	3	3	2.938	23.829	2.938	1.229
3	3	4	2.715	8.964	2.715	1.611
3	3	5	2.631	5.619	2.631	2.381
3	8	2	3.558	6.225	2.304	0.106
3	8	3	2.923	14.269	2.923	0.702
3	8	4	2.659	5.069	2.659	1.582
3	8	5	2.586	1.94	2.586	1.41
4	3	2	3.741	12.474	2.364	0.082
4	3	3	3.029	25.958	3.029	1.98
4	3	4	2.787	7.855	2.787	1.721
4	3	5	2.651	3.175	2.651	0.857
4	8	2	4.088	114.048	2.492	0.133
4	8	3	3.112	174.938	3.112	5.744
4	8	4	2.815	38.942	2.815	4.558
4	8	5	2.682	13.399	2.682	6.202
estein30.txt						
3	3	2	6.654	299.965	3.842	3.651
3	3	3	4.857	299.49	4.815	44.088
3	3	4	4.47	300.005	4.409	299.125
3	3	5	4.271	299.44	4.139	299.425

3	8	2	5.601	299.33	3.647	3.485
3	8	3	4.493	299.496	4.485	9.004
3	8	4	4.108	207.022	4.108	20.684
3	8	5	3.958	108.732	3.958	32.628
4	3	2	6.753	299.29	3.911	1.035
4	3	3	5.238	299.83	5.059	25.358
4	3	4	4.575	299.585	4.584	299.885
4	3	5	4.41	299.445	4.31	299.276
4	8	2	6.326	299.315	3.901	2.805
4	8	3	4.909	299.867	4.814	75.925
4	8	4	4.432	299.762	4.406	300
4	8	5	4.225	299.397	4.214	226.18
estein40.txt						
3	3	2	7.053	299.665	4.401	4.845
3	3	3	5.999	299.63	5.84	300.145
3	3	4	5.352	299.661	5.32	299.46
3	3	5	4.888	110.717	4.888	136.226
3	8	2	8.238	299.91	4.322	0.13
3	8	3	6.639	299.685	6.384	299.76
3	8	4	5.708	299.601	5.576	299.61
3	8	5	5.227	299.778	5.151	299.48
4	3	2	7.809	300.155	4.11	0.415
4	3	3	5.793	299.495	5.423	300.125
4	3	4	5.114	299.55	4.895	299.34
4	3	5	4.671	299.89	4.811	299.415
4	8	2	7.386	299.371	4.285	3.275
4	8	3	5.415	299.48	5.473	299.075
4	8	4	4.99	299.741	4.869	299.576
4	8	5	4.744	299.496	4.741	299.576

Cuadro A.3: Soluciones obtenidas con los modelos L1EMTZ y L2EMTZ

A.2.4. Modelos con restricciones topológicas mejoradas

			MTZ/ITEF		IMTZ/ITEF		REL-M	
p	raíz	H	solución	tiempo	solución	tiempo	solución	tiempo
estein10.txt								
3	3	2	2.703	0.006	2.703	0.01	2.703	0.004
3	3	3	2.417	0.005	2.417	0.007	2.417	0.004
3	3	4	2.33	0.004	2.33	0.004	2.33	0.004
3	3	5	2.33	0.004	2.33	0.005	2.33	0.004
3	8	2	2.687	0.007	2.687	0.004	2.687	0.005

3	8	3	2.42	0.008	2.42	0.013	2.42	0.006
3	8	4	2.384	0.009	2.384	0.006	2.384	0.009
3	8	5	2.365	0.007	2.365	0.008	2.365	0.007
4	3	2	2.152	0.009	2.152	0.004	2.152	0.004
4	3	3	1.944	0.035	1.944	0.037	1.944	0.01
4	3	4	1.834	0.036	1.834	0.067	1.834	0.037
4	3	5	1.826	0.047	1.826	0.046	1.826	0.062
4	8	2	2.726	0.01	2.726	0.009	2.726	0.006
4	8	3	2.216	0.088	2.216	0.105	2.216	0.069
4	8	4	2.034	0.099	2.034	0.118	2.034	0.1
4	8	5	1.929	0.099	1.929	0.084	1.929	0.094
estein20.txt								
3	3	2	3.482	0.044	3.482	0.014	3.482	0.013
3	3	3	2.938	0.761	2.938	0.365	2.938	0.549
3	3	4	2.715	0.91	2.715	2.25	2.715	0.442
3	3	5	2.631	0.838	2.631	0.932	2.631	0.916
3	8	2	3.558	0.029	3.558	0.011	3.558	0.009
3	8	3	2.923	0.823	2.923	0.681	2.923	0.349
3	8	4	2.659	0.599	2.659	0.669	2.659	0.264
3	8	5	2.586	0.745	2.586	0.665	2.586	0.412
4	3	2	3.741	0.035	3.741	0.013	3.741	0.014
4	3	3	3.029	1.008	3.029	0.554	3.029	0.542
4	3	4	2.787	1.052	2.787	0.893	2.787	0.393
4	3	5	2.651	0.89	2.651	0.641	2.651	0.69
4	8	2	4.088	0.053	4.088	0.017	4.088	0.018
4	8	3	3.112	1.936	3.112	1.685	3.112	0.557
4	8	4	2.815	2.003	2.815	2.089	2.815	0.865
4	8	5	2.682	0.88	2.682	2.275	2.682	1.16
estein30.txt								
3	3	2	6.498	0.115	6.498	0.062	6.498	0.029
3	3	3	4.815	299.946	4.815	133.949	4.815	1.357
3	3	4	4.344	299.764	4.344	299.331	4.344	14.669
3	3	5	4.211	299.42	4.133	300.041	4.133	53.253
3	8	2	5.601	0.109	5.601	0.075	5.601	0.019
3	8	3	4.485	7.469	4.485	5.25	4.485	0.857
3	8	4	4.108	30.968	4.108	17.869	4.108	3.323
3	8	5	3.958	25.25	3.958	81.624	3.958	1.892
4	3	2	6.587	0.218	6.587	0.055	6.587	0.028
4	3	3	5.059	299.889	5.059	215.281	5.059	6.493
4	3	4	4.591	299.749	4.598	300.031	4.547	45.218
4	3	5	4.301	299.592	4.341	299.881	4.301	78.414
4	8	2	6.193	0.203	6.193	0.09	6.193	0.022
4	8	3	4.814	38.859	4.814	9.51	4.814	1.764

4	8	4	4.428	299.577	4.394	221.65	4.394	9.002
4	8	5	4.214	128.905	4.223	299.82	4.214	12.954
estein40.txt								
3	3	2	6.923	0.141	6.923	0.1	6.923	0.025
3	3	3	5.801	162.265	5.801	43.198	5.801	13.337
3	3	4	5.27	299.983	5.306	299.666	5.252	38.877
3	3	5	4.888	78.516	4.892	299.181	4.888	38.421
3	8	2	7.671	0.172	7.671	0.179	7.671	0.1
3	8	3	6.323	299.983	6.338	299.785	6.24	216.705
3	8	4	5.517	299.733	5.591	299.98	5.43	299.36
3	8	5	5.173	299.624	5.245	299.28	5.136	299.89
4	3	2	7.058	0.234	7.058	0.21	7.058	0.17
4	3	3	5.375	299.342	5.375	299.46	5.37	70.22
4	3	4	4.914	299.655	4.88	299.93	5	299.545
4	3	5	4.869	299.623	4.71	300.083	4.792	299.605
4	8	2	6.572	0.14	6.572	0.159	6.572	0.1
4	8	3	5.222	299.311	5.199	299.674	5.179	299.295
4	8	4	4.904	299.717	4.854	299.455	4.846	299.335
4	8	5	4.575	299.545	4.673	299.955	4.598	299.7

Cuadro A.4: Soluciones obtenidas con los modelos MTZ/ITEF, IMTZ/ITEF y REL-M

A.2.5. Modelos con restricciones de Sherali y Driscoll

			HMST-SD		HMST-SD/ITEF	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.006	2.703	0.005
3	3	3	2.417	0.012	2.417	0.008
3	3	4	2.33	0.007	2.33	0.005
3	3	5	2.33	0.007	2.33	0.005
3	8	2	2.687	0.005	2.687	0.006
3	8	3	2.42	0.013	2.42	0.009
3	8	4	2.384	0.013	2.384	0.01
3	8	5	2.365	0.008	2.365	0.008
4	3	2	2.152	0.007	2.152	0.006
4	3	3	1.944	0.059	1.944	0.027
4	3	4	1.834	0.069	1.834	0.021
4	3	5	1.826	0.043	1.826	0.024
4	8	2	2.726	0.007	2.726	0.007
4	8	3	2.216	0.109	2.216	0.091
4	8	4	2.034	0.443	2.034	0.269
4	8	5	1.929	0.351	1.929	0.191

estein20.txt						
3	3	2	3.482	0.013	3.482	0.02
3	3	3	2.938	1.801	2.938	1.471
3	3	4	2.715	2.747	2.715	1.293
3	3	5	2.631	1.833	2.631	1.79
3	8	2	3.558	0.012	3.558	0.015
3	8	3	2.923	0.665	2.923	1.059
3	8	4	2.659	1.512	2.659	1.307
3	8	5	2.586	1.753	2.586	1.564
4	3	2	3.741	0.014	3.741	0.021
4	3	3	3.029	1.25	3.029	1.926
4	3	4	2.787	2.221	2.787	2.577
4	3	5	2.651	3.625	2.651	1.998
4	8	2	4.088	0.015	4.088	0.028
4	8	3	3.112	2.661	3.112	2.467
4	8	4	2.815	4.74	2.815	3.647
4	8	5	2.682	4.948	2.682	1.565
estein30.txt						
3	3	2	6.498	0.038	6.498	0.072
3	3	3	4.815	4.378	4.815	4.87
3	3	4	4.344	78.109	4.344	104.717
3	3	5	4.133	148.076	4.152	299.28
3	8	2	5.601	0.015	5.601	0.047
3	8	3	4.485	1.609	4.485	3.484
3	8	4	4.108	6.532	4.108	8.468
3	8	5	3.958	8.593	3.958	13.703
4	3	2	6.587	0.031	6.587	0.14
4	3	3	5.059	4	5.059	10.563
4	3	4	4.547	27.344	4.547	153.312
4	3	5	4.301	217.593	4.301	277.327
4	8	2	6.193	0.015	6.193	0.062
4	8	3	4.814	3.125	4.814	7.766
4	8	4	4.394	12.219	4.394	19.329
4	8	5	4.214	25.375	4.214	46.235
estein40.txt						
3	3	2	6.923	0.015	6.923	0.063
3	3	3	5.801	4.516	5.801	10.828
3	3	4	5.252	61.218	5.252	63.672
3	3	5	4.888	41.453	4.888	99.718
3	8	2	7.671	0.046	7.671	0.125
3	8	3	6.24	13.422	6.24	40.828
3	8	4	5.421	92.625	5.421	161.671

3	8	5	5.116	299.389	5.114	299.093
4	3	2	7.058	0.047	7.058	0.125
4	3	3	5.37	15.843	5.37	57.297
4	3	4	4.862	299.686	4.862	299.139
4	3	5	4.703	299.795	4.765	299.608
4	8	2	6.572	0.046	6.572	0.093
4	8	3	5.159	14.047	5.159	46.421
4	8	4	4.762	299.342	4.833	299.155
4	8	5	4.527	299.858	4.563	300.092

Cuadro A.5: Soluciones obtenidas con los modelos HMST-SD y HMST-SD/ITEF

A.3. Heurísticas

A.3.1. Heurística de ahorros

archivo	problema	raiz	H	solución	tiempo	centralidad_modificada
estein10.txt	3	3	2	2.794	0.149	0.934
estein10.txt	3	3	3	2.561	0.010	0.934
estein10.txt	3	3	4	2.561	0.010	0.934
estein10.txt	3	3	5	2.561	0.009	0.934
estein10.txt	3	8	2	2.753	0.010	0.946
estein10.txt	3	8	3	2.516	0.012	0.946
estein10.txt	3	8	4	2.481	0.017	0.946
estein10.txt	3	8	5	2.481	0.012	0.946
estein10.txt	4	3	2	2.290	0.012	0.965
estein10.txt	4	3	3	2.208	0.011	0.965
estein10.txt	4	3	4	2.095	0.013	0.965
estein10.txt	4	3	5	1.927	0.013	0.965
estein10.txt	4	8	2	2.726	0.020	0
estein10.txt	4	8	3	2.451	0.039	0
estein10.txt	4	8	4	2.438	0.018	0
estein10.txt	4	8	5	2.078	0.022	0
estein20.txt	3	3	2	4.232	0.264	0.530
estein20.txt	3	3	3	3.260	0.323	0.530
estein20.txt	3	3	4	2.844	0.324	0.530
estein20.txt	3	3	5	2.750	0.330	0.530
estein20.txt	3	8	2	4.594	0.193	0.597
estein20.txt	3	8	3	3.154	0.297	0.597
estein20.txt	3	8	4	2.686	0.305	0.597
estein20.txt	3	8	5	2.674	0.296	0.597
estein20.txt	4	3	2	4.368	0.253	0.365

estein20.txt	4	3	3	3.389	0.308	0.365
estein20.txt	4	3	4	2.957	0.348	0.365
estein20.txt	4	3	5	2.680	0.387	0.365
estein20.txt	4	8	2	4.857	0.297	0.080
estein20.txt	4	8	3	3.520	0.369	0.080
estein20.txt	4	8	4	3.137	0.422	0.080
estein20.txt	4	8	5	2.875	0.435	0.080
estein30.txt	3	3	2	8.723	1.540	0.202
estein30.txt	3	3	3	5.555	1.744	0.202
estein30.txt	3	3	4	4.696	1.790	0.202
estein30.txt	3	3	5	4.323	1.866	0.202
estein30.txt	3	8	2	6.345	1.181	0.880
estein30.txt	3	8	3	4.687	1.518	0.880
estein30.txt	3	8	4	4.242	1.646	0.880
estein30.txt	3	8	5	4.066	1.622	0.880
estein30.txt	4	3	2	8.243	1.288	0.304
estein30.txt	4	3	3	5.606	1.635	0.304
estein30.txt	4	3	4	4.808	1.753	0.304
estein30.txt	4	3	5	4.509	1.882	0.304
estein30.txt	4	8	2	7.028	1.194	0.505
estein30.txt	4	8	3	5.086	1.531	0.505
estein30.txt	4	8	4	4.672	1.596	0.505
estein30.txt	4	8	5	4.373	1.761	0.505
estein40.txt	3	3	2	7.785	3.702	0.942
estein40.txt	3	3	3	6.144	4.410	0.942
estein40.txt	3	3	4	5.478	4.673	0.942
estein40.txt	3	3	5	5.289	4.508	0.942
estein40.txt	3	8	2	9.532	3.926	0.445
estein40.txt	3	8	3	6.776	5.061	0.445
estein40.txt	3	8	4	6.039	5.405	0.445
estein40.txt	3	8	5	5.549	5.596	0.445
estein40.txt	4	3	2	8.415	4.297	0.464
estein40.txt	4	3	3	6.042	5.299	0.464
estein40.txt	4	3	4	5.662	5.398	0.464
estein40.txt	4	3	5	5.584	5.362	0.464
estein40.txt	4	8	2	7.421	4.373	0.759
estein40.txt	4	8	3	5.466	5.216	0.759
estein40.txt	4	8	4	4.916	5.314	0.759
estein40.txt	4	8	5	4.607	5.327	0.759
estein50.txt	3	3	2	12.380	11.141	0.287
estein50.txt	3	3	3	7.909	13.915	0.287
estein50.txt	3	3	4	7.096	14.931	0.287
estein50.txt	3	3	5	6.316	15.424	0.287

estein50.txt	3	8	2	12.122	10.203	0.411
estein50.txt	3	8	3	7.614	13.768	0.411
estein50.txt	3	8	4	6.895	14.402	0.411
estein50.txt	3	8	5	6.353	14.863	0.411
estein50.txt	4	3	2	12.336	10.474	0.379
estein50.txt	4	3	3	7.533	14.464	0.379
estein50.txt	4	3	4	6.645	15.666	0.379
estein50.txt	4	3	5	6.216	16.432	0.379
estein50.txt	4	8	2	10.765	8.997	0.650
estein50.txt	4	8	3	7.368	12.484	0.650
estein50.txt	4	8	4	6.239	12.634	0.650
estein50.txt	4	8	5	5.815	13.472	0.650
estein60.txt	3	3	2	15.564	24.450	0.109
estein60.txt	3	3	3	9.257	31.494	0.109
estein60.txt	3	3	4	7.913	34.068	0.109
estein60.txt	3	3	5	7.028	36.266	0.109
estein60.txt	3	8	2	11.442	18.570	0.876
estein60.txt	3	8	3	7.744	23.919	0.876
estein60.txt	3	8	4	6.797	24.648	0.876
estein60.txt	3	8	5	6.116	25.731	0.876
estein60.txt	4	3	2	15.428	24.451	0.230
estein60.txt	4	3	3	8.675	30.625	0.230
estein60.txt	4	3	4	7.862	31.191	0.230
estein60.txt	4	3	5	6.708	32.048	0.230
estein60.txt	4	8	2	13.503	21.595	0.398
estein60.txt	4	8	3	8.430	29.211	0.398
estein60.txt	4	8	4	7.386	29.183	0.398
estein60.txt	4	8	5	6.660	29.559	0.398
estein70.txt	3	3	2	17.388	39.710	0.335
estein70.txt	3	3	3	10.522	51.737	0.335
estein70.txt	3	3	4	7.668	57.647	0.335
estein70.txt	3	3	5	7.226	58.183	0.335
estein70.txt	3	8	2	13.400	47.549	0.687
estein70.txt	3	8	3	8.612	68.676	0.687
estein70.txt	3	8	4	7.543	50.990	0.687
estein70.txt	3	8	5	6.889	61.019	0.687
estein70.txt	4	3	2	16.969	41.251	0.390
estein70.txt	4	3	3	10.109	54.265	0.390
estein70.txt	4	3	4	8.486	56.725	0.390
estein70.txt	4	3	5	7.575	58.233	0.390
estein70.txt	4	8	2	16.198	38.468	0.418
estein70.txt	4	8	3	9.779	50.716	0.418
estein70.txt	4	8	4	9.148	51.418	0.418

estein70.txt	4	8	5	7.598	55.111	0.418
estein80.txt	3	3	2	17.185	73.249	0.636
estein80.txt	3	3	3	9.898	101.079	0.636
estein80.txt	3	3	4	8.755	96.262	0.636
estein80.txt	3	3	5	8.147	94.073	0.636
estein80.txt	3	8	2	17.846	66.973	0.630
estein80.txt	3	8	3	10.293	93.106	0.630
estein80.txt	3	8	4	8.962	99.351	0.630
estein80.txt	3	8	5	8.117	92.665	0.630
estein80.txt	4	3	2	15.840	65.126	0.678
estein80.txt	4	3	3	9.454	90.533	0.680
estein80.txt	4	3	4	8.122	91.802	0.680
estein80.txt	4	3	5	7.575	93.931	0.680
estein80.txt	4	8	2	17.590	77.820	0.372
estein80.txt	4	8	3	11.828	93.903	0.372
estein80.txt	4	8	4	8.730	101.754	0.372
estein80.txt	4	8	5	8.056	104.619	0.372

Cuadro A.6: Soluciones obtenidas con la heurística de ahorros de EW

A.3.2. Algoritmos de segundo orden

			Inhib. simple		Inhib. pareada		Incl. limitada		Incl. general	
p	raiz	H	sol	t	sol	t	sol	t	sol	t
estein10.txt										
3	3	2	2.794	0.077	2.794	0.435	2.794	0.151	2.794	0.175
3	3	3	2.561	0.068	2.561	0.775	2.561	0.137	2.561	0.115
3	3	4	2.561	0.054	2.561	0.754	2.561	0.13	2.561	0.118
3	3	5	2.561	0.054	2.561	0.804	2.561	0.166	2.561	0.151
3	8	2	2.748	0.178	2.748	1.439	2.748	0.341	2.748	0.395
3	8	3	2.516	0.093	2.516	1.599	2.516	0.165	2.516	0.15
3	8	4	2.481	0.074	2.481	1.576	2.481	0.185	2.481	0.22
3	8	5	2.481	0.135	2.481	1.569	2.481	0.176	2.481	0.164
4	3	2	2.29	0.19	2.29	1.919	2.222	0.489	2.222	0.661
4	3	3	2.113	0.304	2.113	3.027	2.207	0.145	2.038	0.707
4	3	4	2.095	0.093	1.997	3.989	2.03	0.335	1.929	0.854
4	3	5	1.922	0.227	1.922	4.172	1.927	0.186	1.92	0.556
4	8	2	2.726	0.164	2.726	3.656	2.726	0.297	2.726	0.584
4	8	3	2.406	0.396	2.406	12.595	2.359	0.789	2.359	1.466
4	8	4	2.267	0.372	2.106	11.168	2.334	0.669	2.202	1.312
4	8	5	2.037	0.367	2.037	13.579	2.078	0.4	2.037	1.287
estein20.txt										

3	3	2	4.196	5.24	4.196	292.91	4.128	32.796	3.942	46.809
3	3	3	3.185	11.916	3.185	710.892	3.185	27.136	3.185	35.257
3	3	4	2.834	8.695	2.834	658.621	2.842	19.369	2.844	18.555
3	3	5	2.75	5.095	2.75	436.188	2.75	9.421	2.75	19.027
3	8	2	3.706	12.798	3.653	575.362	4.08	22.841	3.927	41.841
3	8	3	3.074	13.066	3.01	888.366	3.108	21.652	3.074	50.04
3	8	4	2.659	9.253	2.659	786.274	2.66	22.793	2.659	36.649
3	8	5	2.586	9.466	2.586	717.321	2.6	23.051	2.586	36.247
4	3	2	4.105	9.307	3.943	667.356	4.178	12.971	4.079	50.099
4	3	3	3.078	22.925	3.078	1071.65	3.102	37.515	3.043	67.338
4	3	4	2.834	11.015	2.812	1326.059	2.835	18.992	2.835	46.073
4	3	5	2.663	12.208	2.663	854.877	2.68	10.253	2.663	48.104
4	8	2	4.298	23.701	4.293	1420.233	4.39	53.023	4.537	42.483
4	8	3	3.417	17.27	3.209	2796.237	3.269	57.387	3.383	112.072
4	8	4	2.844	27.414	2.844	1341.345	3.137	11.438	3.041	59.04
4	8	5	2.682	13.635	2.682	915.486	2.682	22.954	2.682	60.39
estein30.txt										
3	3	2	7.06	179.743	7.06	8888.381	7.362	313.846	7.333	617.896
3	3	3	4.908	128.159	4.815	11902.568	4.908	357.712	4.908	585.789
3	3	4	4.462	176.668	4.424	13224.491	4.604	210.2	4.58	583.558
3	3	5	4.214	200.428	4.214	14970.705	4.323	134.978	4.251	612.124
3	8	2	5.885	121.526	5.83	8263.928	5.885	463.989	5.902	399.382
3	8	3	4.485	139.785	—	—	4.485	636.093	4.485	842.206
3	8	4	4.135	142.732	—	—	4.168	172.367	4.168	347.631
3	8	5	3.958	148.905	—	—	4.014	255.723	4.007	557.931
4	3	2	7.118	116.931	—	—	7.735	334.606	7.289	780.021
4	3	3	5.293	147.356	—	—	5.321	477.618	5.31	583.693
4	3	4	4.808	40.504	—	—	4.808	128.694	4.776	406.251
4	3	5	4.473	122.089	—	—	4.41	278.249	4.476	399.801
4	8	2	6.444	122.83	—	—	6.426	601.184	6.322	947.214
4	8	3	4.865	62.131	—	—	4.82	130.482	4.82	319.836
4	8	4	4.537	71.041	—	—	4.561	269.264	4.493	334.915
4	8	5	4.223	75.552	—	—	4.253	197.249	4.251	546.678
estein40.txt										
3	3	2	7.27	465.09	—	—	7.39	1177.95	7.249	2680.158
3	3	3	5.944	590.438	—	—	5.937	2151.278	5.967	2531.725
3	3	4	5.373	371.862	—	—	5.373	1155.617	5.444	1241.656
3	3	5	5.052	643.573	—	—	5.134	1426.866	5.024	3103.015
3	8	2	8.406	763.42	—	—	8.574	1809.719	8.736	3600.817
3	8	3	6.368	1150.813	—	—	6.371	3009.947	6.581	2204.802
3	8	4	5.681	592.834	—	—	5.692	1669.016	5.801	2386.245
3	8	5	5.218	638.753	—	—	5.305	2196.148	5.355	2418.307

4	3	2	7.594	526.519	—	—	8.215	1832.205	8.286	1054.125
4	3	3	5.949	559.558	—	—	5.967	2733.495	5.969	1296.702
4	3	4	5.648	284.685	—	—	5.511	1678.076	5.417	3228.537
4	3	5	5.039	629.654	—	—	5.267	1726.569	5.052	1984.876
4	8	2	6.706	1161.175	—	—	6.876	2925.949	6.695	4242.579
4	8	3	5.201	954.45	—	—	5.432	1130.477	5.262	5833.209
4	8	4	4.789	1170.29	—	—	4.852	1616.106	4.856	4425.28
4	8	5	4.607	168.077	—	—	4.55	989.558	4.55	4074.715

Cuadro A.7: Soluciones obtenidas con los algoritmos de segundo orden

A.3.3. Vecindarios basados en un modelo de programación dinámica

			Mov. simple		Intercambio		Ambos	
p	raíz	H	solución	tiempo	solución	tiempo	solución	tiempo
estein10.txt								
3	3	2	2.703	0.088 (0.193)	2.794	0.099 (0.15)	2.703	0.079 (0.138)
3	3	3	2.465	0.004 (0.011)	2.561	0.003 (0.015)	2.465	0.008 (0.009)
3	3	4	2.465	0.01 (0.019)	2.561	0.003 (0.01)	2.465	0.009 (0.009)
3	3	5	2.465	0.014 (0.025)	2.561	0.004 (0.01)	2.465	0.011 (0.01)
3	8	2	2.691	0.002 (0.017)	2.717	0.007 (0.01)	2.687	0.008 (0.011)
3	8	3	2.42	0.005 (0.021)	2.485	0.008 (0.014)	2.42	0.006 (0.018)
3	8	4	2.384	0.005 (0.02)	2.449	0.009 (0.013)	2.384	0.009 (0.011)
3	8	5	2.384	0.01 (0.021)	2.449	0.008 (0.068)	2.384	0.01 (0.012)
4	3	2	2.221	0.003 (0.021)	2.222	0.008 (0.011)	2.152	0.021 (0.022)
4	3	3	2.084	0.005 (0.014)	2.207	0.003 (0.012)	1.969	0.033 (0.011)
4	3	4	2.026	0.004 (0.023)	2.03	0.022 (0.018)	1.961	0.027 (0.024)
4	3	5	1.857	0.006 (0.025)	1.922	0.01 (0.056)	1.853	0.03 (0.013)
4	8	2	2.726	0.001 (0.029)	2.726	0.003 (0.026)	2.726	0.004 (0.018)
4	8	3	2.451	0.001 (0.022)	2.406	0.007 (0.022)	2.406	0.011 (0.024)

4	8	4	2.294	0.004 (0.021)	2.438	0.004 (0.02)	2.147	0.029 (0.02)
4	8	5	2.057	0.006 (0.038)	2.037	0.011 (0.022)	2.011	0.024 (0.022)
estein20.txt								
3	3	2	3.557	0.015 (0.262)	3.833	0.244 (0.285)	3.482	0.078 (0.256)
3	3	3	3.143	0.039 (0.314)	3.241	0.159 (0.458)	3.073	0.178 (0.283)
3	3	4	2.725	0.059 (0.329)	2.844	0.069 (0.298)	2.725	0.15 (0.48)
3	3	5	2.631	0.055 (0.321)	2.75	0.077 (0.363)	2.631	0.157 (0.341)
3	8	2	3.617	0.019 (0.241)	4.07	0.171 (0.209)	3.558	0.099 (0.201)
3	8	3	3.079	0.017 (0.299)	3.079	0.218 (0.314)	3.079	0.067 (0.276)
3	8	4	2.686	0.014 (0.339)	2.686	0.078 (0.323)	2.686	0.073 (0.306)
3	8	5	2.674	0.031 (0.329)	2.674	0.139 (0.344)	2.674	0.139 (0.347)
4	3	2	3.832	0.021 (0.284)	4.259	0.159 (0.264)	3.741	0.088 (0.262)
4	3	3	3.167	0.03 (0.37)	3.288	0.209 (0.359)	3.029	0.297 (0.353)
4	3	4	2.831	0.074 (0.381)	2.835	0.242 (0.391)	2.831	0.164 (0.365)
4	3	5	2.676	0.062 (0.424)	2.68	0.092 (0.377)	2.676	0.154 (0.391)
4	8	2	4.242	0.014 (0.311)	4.652	0.358 (0.311)	4.088	0.208 (0.29)
4	8	3	3.373	0.026 (0.41)	3.375	0.423 (0.403)	3.296	0.295 (0.432)
4	8	4	3.063	0.045 (0.421)	3.036	0.349 (0.436)	3.063	0.105 (0.412)
4	8	5	2.873	0.041 (0.428)	2.786	0.182 (0.406)	2.786	0.204 (0.406)
estein30.txt								
3	3	2	6.678	0.054 (1.416)	7.579	0.925 (1.42)	6.59	0.603 (1.339)
3	3	3	5.07	0.109 (1.863)	5.206	1.62 (1.943)	4.908	0.932 (1.828)
3	3	4	4.696	0.056 (1.885)	4.604	0.669 (1.904)	4.604	0.723 (1.851)
3	3	5	4.25	0.152 (1.911)	4.323	0.404 (1.9)	4.232	0.778 (1.893)

3	8	2	5.643	0.041 (1.215)	5.966	0.69 (1.283)	5.635	0.334 (1.214)
3	8	3	4.687	0.042 (2.075)	4.643	0.812 (1.578)	4.643	0.833 (1.79)
3	8	4	4.242	0.083 (1.851)	4.21	0.615 (1.785)	4.21	0.625 (1.827)
3	8	5	4.053	0.121 (1.639)	4.053	0.725 (1.623)	4.053	0.45 (1.621)
4	3	2	6.746	0.076 (1.315)	7.259	1.375 (1.329)	6.699	0.307 (1.304)
4	3	3	5.387	0.078 (1.708)	5.392	1.334 (1.695)	5.286	0.966 (1.644)
4	3	4	4.794	0.118 (1.818)	4.808	0.33 (1.831)	4.779	0.673 (1.84)
4	3	5	4.509	0.082 (1.838)	4.509	0.461 (1.768)	4.509	0.425 (1.808)
4	8	2	6.345	0.081 (1.213)	6.613	0.745 (1.178)	6.193	0.557 (1.177)
4	8	3	4.82	0.152 (1.563)	5.036	0.587 (1.594)	4.82	0.344 (1.796)
4	8	4	4.628	0.078 (1.651)	4.628	0.621 (1.712)	4.628	0.381 (1.622)
4	8	5	4.373	0.085 (1.873)	4.373	0.374 (1.7)	4.373	0.371 (1.76)
estein40.txt								
3	3	2	6.923	0.116 (3.898)	7.218	2.173 (4.013)	6.923	0.447 (3.918)
3	3	3	6.048	0.175 (4.836)	6.094	2.912 (4.472)	6.045	1.647 (4.53)
3	3	4	5.431	0.238 (4.677)	5.455	1.831 (4.533)	5.407	3.45 (4.774)
3	3	5	5.243	0.327 (4.897)	5.287	2.028 (4.603)	5.239	3.496 (4.582)
3	8	2	7.936	0.208 (4.065)	8.595	1.911 (3.899)	7.671	1.614 (4.295)
3	8	3	6.585	0.17 (5.217)	6.451	5.201 (5.105)	6.542	4.157 (5.268)
3	8	4	5.769	0.743 (7.928)	5.806	2.721 (5.256)	5.716	2.57 (5.546)
3	8	5	5.468	0.64 (5.844)	5.378	5.261 (5.569)	5.398	4.168 (5.927)
4	3	2	7.133	0.343 (4.559)	7.284	3.249 (4.24)	7.078	1.264 (4.429)
4	3	3	5.591	0.564 (5.158)	5.776	2.805 (5.127)	5.591	1.255 (5.523)

4	3	4	5.408	0.319 (5.319)	5.288	9.825 (5.335)	5.009	7.949 (5.82)
4	3	5	5.365	0.477 (5.403)	5.008	10.292 (5.547)	4.903	4.588 (5.648)
4	8	2	6.64	0.145 (4.394)	6.793	2.793 (4.41)	6.574	1.331 (4.504)
4	8	3	5.257	0.415 (5.292)	5.407	2.149 (5.271)	5.205	3.333 (5.354)
4	8	4	4.916	0.123 (5.333)	4.871	2.523 (5.373)	4.862	2.077 (5.689)
4	8	5	4.553	0.313 (5.321)	4.499	6.131 (5.52)	4.486	2.727 (5.602)
estein50.txt								
3	3	2	9.424	0.644 (11.166)	10.475	9.382 (11.253)	9.31	3.259 (11.755)
3	3	3	7.228	0.894 (14.03)	7.442	21.487 (14.11)	7.179	9.266 (17.547)
3	3	4	6.697	1.765 (14.915)	6.75	14.422 (14.852)	6.456	19.909 (15.666)
3	3	5	6.27	0.88 (15.825)	6.249	6.916 (15.443)	6.239	5.851 (16.004)
3	8	2	9.553	0.548 (12.617)	10.841	10.178 (10.321)	9.415	3.528 (10.363)
3	8	3	7.184	0.931 (14.473)	7.253	16.482 (13.734)	7.039	10.684 (13.698)
3	8	4	6.726	1.106 (16.077)	6.784	6.26 (14.405)	6.716	5.403 (14.607)
3	8	5	6.234	1.807 (15.76)	6.075	13.804 (14.98)	5.937	25.55 (15.767)
4	3	2	8.735	0.722 (11.021)	10.412	10.934 (10.155)	8.681	1.786 (10.182)
4	3	3	7.129	1.059 (14.128)	7.127	13.039 (13.646)	6.783	10.843 (13.844)
4	3	4	6.389	0.992 (15.393)	6.231	16.457 (14.66)	6.197	17.743 (15.47)
4	3	5	5.909	1.035 (14.65)	5.973	11.751 (14.474)	5.896	6.391 (14.777)
4	8	2	8.368	0.613 (9.005)	9.514	14.241 (8.824)	8.125	2.913 (9.139)
4	8	3	6.588	1.474 (12.256)	6.759	17.661 (11.842)	6.43	9.676 (11.85)
4	8	4	5.822	1.987 (12.366)	6.001	10.709 (12.297)	5.778	8.453 (12.199)
4	8	5	5.569	1.161 (13.534)	5.493	26.167 (12.797)	5.293	16.319 (12.843)

estein60.txt								
3	3	2	11.072	1.014 (24.45)	13.331	28.937 (23.484)	10.744	11.478 (23.412)
3	3	3	8.263	1.643 (32.189)	8.408	44.596 (30.188)	7.995	13.794 (30.428)
3	3	4	7.332	2.538 (35.298)	7.372	73.022 (33.519)	6.991	29.82 (33.626)
3	3	5	6.905	0.891 (34.823)	6.892	14.212 (32.809)	6.62	17.829 (32.744)
3	8	2	9.441	0.986 (19.09)	10.075	22.082 (18.553)	9.198	12.308 (19.075)
3	8	3	7.53	0.892 (24.499)	7.441	30.28 (23.298)	7.371	28.539 (24.18)
3	8	4	6.768	0.625 (24.911)	6.773	8.464 (24.585)	6.722	9.302 (24.664)
3	8	5	6.109	0.859 (26.123)	6.083	18.664 (26.028)	6.083	15.706 (26.165)
4	3	2	10.853	1.456 (24.193)	12.107	27.522 (24.593)	10.623	5.082 (24.448)
4	3	3	7.769	1.381 (31.229)	7.812	21.006 (30.935)	7.667	15.961 (30.941)
4	3	4	7.318	1.011 (36.174)	7.132	58.579 (31.338)	7.073	46.138 (31.451)
4	3	5	6.582	1.677 (34.988)	6.546	47.252 (32.106)	6.566	16.598 (32.387)
4	8	2	10.555	0.915 (30.763)	12.006	20.354 (21.758)	10.376	12.36 (21.688)
4	8	3	7.52	2.395 (29.764)	7.73	44.044 (28.453)	7.423	28.03 (28.297)
4	8	4	6.781	3.074 (31.521)	6.858	47.103 (29.297)	6.641	21.256 (29.441)
4	8	5	6.163	3.128 (30.504)	6.208	47.364 (29.555)	6.126	30.202 (29.717)
estein70.txt								
3	3	2	11.7	2.231 (47.152)	13.997	56.347 (39.74)	11.658	10.318 (40.003)
3	3	3	8.48	4.693 (54.768)	9.143	118.809 (52.098)	8.289	45.014 (52.61)
3	3	4	7.579	2.351 (60.326)	7.532	37.05 (57.516)	7.532	41.867 (57.917)
3	3	5	7.108	1.437 (61.602)	7.077	51.653 (58.029)	7.077	45.262 (58.238)
3	8	2	10.471	2.049 (39.112)	11.649	42.376 (38.35)	10.164	22.064 (38.276)

3	8	3	8.217	1.955 (49.67)	8.001	79.94 (49.715)	7.807	58.716 (50.203)
3	8	4	7.294	3.405 (63.285)	7.291	83.315 (50.764)	6.923	84.504 (51.082)
3	8	5	6.751	2.725 (55.894)	6.671	81.35 (53.297)	6.671	49.122 (54.951)
4	3	2	12.302	2.542 (41.716)	13.744	53.056 (41.866)	11.842	19.478 (42.27)
4	3	3	8.912	2.979 (54.48)	9.147	69.769 (54.983)	8.824	44.937 (54.773)
4	3	4	7.827	3.732 (58.811)	7.867	94.952 (57.48)	7.573	95.095 (58.72)
4	3	5	7.249	4.28 (60.627)	7.356	47.077 (62.219)	7.239	22.506 (68.496)
4	8	2	12.027	1.874 (39.231)	13.837	62.797 (39.16)	11.819	13.594 (39.085)
4	8	3	9.027	2.005 (51.245)	8.996	61.169 (52.802)	8.865	32.272 (51.24)
4	8	4	7.935	4.718 (51.671)	8.203	87.003 (51.822)	7.823	28.129 (55.898)
4	8	5	7.376	4.553 (54.831)	7.359	61.087 (54.309)	7.32	40.832 (54.646)
estein80.txt								
3	3	2	12.309	3.46 (67.567)	13.593	111.768 (69.195)	11.986	29.032 (65.798)
3	3	3	9.418	3.09 (91.039)	9.281	149.724 (86.785)	9.207	105.776 (87.477)
3	3	4	8.216	10.305 (103.238)	8.014	216.074 (91.318)	7.893	126.901 (92.313)
3	3	5	8.103	2.842 (94.591)	8.014	148.326 (93.138)	8.008	98.941 (93.875)
3	8	2	12.161	3.56 (64.775)	14.705	89.06 (64.011)	12.027	24.141 (64.497)
3	8	3	9.151	3.72 (88.768)	9.255	133.154 (87.826)	8.988	53.763 (88.119)
3	8	4	8.63	4.127 (91.114)	8.508	249.467 (90.968)	8.408	126.023 (90.954)
3	8	5	7.732	6.206 (93.516)	7.796	111.09 (94.433)	7.719	67.338 (93.448)
4	3	2	11.382	3.365 (63.863)	13.371	107.332 (77.912)	11.11	36.188 (64.509)
4	3	3	8.575	5.464 (91.6)	8.741	108.814 (91.315)	8.41	52.87 (90.735)
4	3	4	7.674	4.769 (91.76)	7.736	76.168 (92.44)	7.65	81.44 (101.658)

4	3	5	7.184	6.455 (93.482)	7.269	57.064 (94.254)	7.043	66.42 (94.577)
4	8	2	12.433	2.553 (84.381)	14.722	114.545 (76.657)	12.101	33.809 (77.444)
4	8	3	9.34	6.194 (90.348)	10.078	267.286 (94.18)	8.913	110.927 (91.267)
4	8	4	8.419	3.64 (101.176)	8.257	157.318 (106.832)	8.145	127 (101.307)
4	8	5	7.6	6.013 (104.378)	7.457	152.043 (109.584)	7.443	82.128 (117.054)

Cuadro A.8: Soluciones obtenidas con los vecindarios basados en un modelo de programación dinámica

A.4. Metaheurísticas

A.4.1. Simulated Annealing

problema	raiz	H	solución	tiempo
estein10.txt				
3	3	2	2.703	28.226 (0.281)
3	3	3	2.485	27.805 (0.02)
3	3	4	2.485	30.03 (0.017)
3	3	5	2.457	31.005 (0.019)
3	8	2	2.691	25.548 (0.012)
3	8	3	2.49	29.058 (0.023)
3	8	4	2.384	29.786 (0.022)
3	8	5	2.384	30.039 (0.022)
4	3	2	2.153	26.28 (0.023)
4	3	3	2.153	30.031 (0.021)
4	3	4	2.153	31.207 (0.025)
4	3	5	1.864	30.303 (0.025)

4	8	2	2.835	20.887 (0.034)
4	8	3	2.289	25.78 (0.043)
4	8	4	2.22	28.368 (0.037)
4	8	5	2.261	29.841 (0.04)
estein20.txt				
3	3	2	3.578	53.665 (0.495)
3	3	3	3.122	59.621 (0.595)
3	3	4	2.725	63.263 (0.6)
3	3	5	2.715	64.045 (0.649)
3	8	2	3.558	55.801 (0.368)
3	8	3	3.075	61.468 (0.538)
3	8	4	2.688	61.825 (0.576)
3	8	5	2.713	65.166 (0.579)
4	3	2	3.741	52.712 (0.425)
4	3	3	3.159	58.067 (0.577)
4	3	4	2.937	60.726 (0.617)
4	3	5	2.753	62.129 (0.617)
4	8	2	4.098	51.972 (0.539)
4	8	3	3.308	57.805 (0.808)
4	8	4	2.861	59.714 (0.835)
4	8	5	2.872	64.989 (0.863)
estein30.txt				

3	3	2	6.498	86.941 (2.584)
3	3	3	4.859	98.821 (3.401)
3	3	4	4.773	104.682 (3.267)
3	3	5	4.639	106.552 (3.565)
3	8	2	5.647	78.762 (2.181)
3	8	3	4.754	98.392 (2.776)
3	8	4	4.734	103.795 (3.356)
3	8	5	4.636	104.514 (2.858)
4	3	2	6.698	79.292 (2.428)
4	3	3	5.353	99.051 (3.172)
4	3	4	5.034	104.536 (3.392)
4	3	5	4.803	107.779 (3.257)
4	8	2	6.238	84.453 (2.12)
4	8	3	4.936	97.278 (2.7)
4	8	4	4.881	107.409 (3.121)
4	8	5	4.406	108.576 (3.288)
estein40.txt				
3	3	2	7.001	135.727 (6.118)
3	3	3	6.068	156.691 (7.841)
3	3	4	5.899	162.917 (8.084)
3	3	5	5.661	167.221 (9.807)

3	8	2	7.771	133.162 (7.325)
3	8	3	6.537	157.785 (9.023)
3	8	4	5.956	165.235 (9.429)
3	8	5	5.771	170.859 (9.135)
4	3	2	7.127	130.418 (7.997)
4	3	3	5.392	155.321 (9.468)
4	3	4	5.206	166.134 (9.135)
4	3	5	5.026	173.449 (9.917)
4	8	2	6.644	115.906 (8.154)
4	8	3	5.395	152.646 (9.411)
4	8	4	5.043	163.922 (9.596)
4	8	5	4.906	167.224 (9.543)
estein50.txt				
3	3	2	9.528	172.343 (18.642)
3	3	3	7.323	223.888 (24.891)
3	3	4	6.91	238.668 (24.118)
3	3	5	6.334	250.797 (20)
3	8	2	9.542	181.826 (17.832)
3	8	3	7.037	220.969 (24.969)
3	8	4	6.669	235.384 (26.23)
3	8	5	6.191	244.204 (22.356)
4	3	2	8.807	185.065 (19.144)

4	3	3	6.849	220.08 (21.16)
4	3	4	6.176	239.125 (18.261)
4	3	5	5.933	249.979 (25.611)
4	8	2	8.224	187.801 (15.098)
4	8	3	6.503	222.606 (20.874)
4	8	4	5.994	235.995 (19.962)
4	8	5	5.964	246.724 (23.006)
estein60.txt				
3	3	2	11	247.355 (35.941)
3	3	3	8.319	306.732 (54.83)
3	3	4	7.384	332.339 (38.451)
3	3	5	7.129	300.781 (55.297)
3	8	2	9.265	262.902 (22.627)
3	8	3	7.401	304.342 (29.775)
3	8	4	6.507	327.146 (42.461)
3	8	5	6.22	334.233 (34.656)
4	3	2	10.77	245.952 (30.971)
4	3	3	8.03	296.732 (46.141)
4	3	4	7.223	271.552 (36.702)
4	3	5	7.165	322.987 (53.878)
4	8	2	10.62	243.139 (34.393)
4	8	3	7.318	291.62 (50.427)

4	8	4	6.674	315.778 (52.958)
4	8	5	6.538	323.691 (50.633)
estein70.txt				
3	3	2	11.5	307.219 (61.362)
3	3	3	8.287	387.531 (93.086)
3	3	4	7.368	417.671 (103.577)
3	3	5	7.44	426.961 (78.121)
3	8	2	10.291	326.41 (61.505)
3	8	3	7.867	390.108 (61.474)
3	8	4	7.119	415.837 (75.818)
3	8	5	7.026	426.614 (82.049)
4	3	2	12.087	312.38 (66.672)
4	3	3	8.95	395.104 (91.16)
4	3	4	8.032	416.264 (99.773)
4	3	5	7.355	437.257 (84.096)
4	8	2	11.891	278.264 (57.12)
4	8	3	8.986	390.994 (84.594)
4	8	4	7.865	415.252 (80.769)
4	8	5	7.709	424.276 (76.589)
estein80.txt				
3	3	2	11.974	394.462 (89.423)
3	3	3	9.314	487.895 (108.479)

3	3	4	8.277	536.659 (148.657)
3	3	5	8.32	558.536 (154.459)
3	8	2	12.048	402.625 (95.589)
3	8	3	9.089	505.054 (113.632)
3	8	4	8.502	484.054 (159.54)
3	8	5	7.786	388.181 (105.767)
4	3	2	11.182	239.988 (67.706)
4	3	3	8.841	357.111 (93.022)
4	3	4	7.779	369.232 (95.072)
4	3	5	7.357	407.854 (103.511)
4	8	2	12.18	284.603 (83.342)
4	8	3	9.023	375.963 (95.073)
4	8	4	8.258	414.071 (92.215)
4	8	5	7.906	416.358 (95.043)

Cuadro A.9: Soluciones obtenidas con Simulated Annealing

A.4.2. GRASP

			Tamaño RCL = 3		Tamaño RCL = 5	
problema	raiz	H	solución	tiempo	solución	tiempo
estein10.txt						
3	3	2	2.703	0.574	2.703	1.043
3	3	3	2.465	0.464	2.465	0.909
3	3	4	2.465	0.519	2.383	0.956
3	3	5	2.437	0.566	2.411	1.009
3	8	2	2.687	0.486	2.687	0.711
3	8	3	2.42	0.55	2.42	0.962
3	8	4	2.384	0.773	2.384	1.128

3	8	5	2.384	0.897	2.384	1.24
4	3	2	2.152	0.43	2.152	0.801
4	3	3	1.969	0.999	1.969	1.422
4	3	4	1.859	0.828	1.859	1.559
4	3	5	1.851	1.044	1.851	1.655
4	8	2	2.726	0.658	2.726	1.185
4	8	3	2.216	0.866	2.216	1.5
4	8	4	2.034	0.918	2.034	1.84
4	8	5	1.969	1.021	1.969	1.854
estein20.txt						
3	3	2	3.482	6.797	3.482	12.881
3	3	3	2.938	9.659	2.938	19.102
3	3	4	2.715	9.092	2.715	18.565
3	3	5	2.631	10.592	2.631	20.637
3	8	2	3.558	7.123	3.558	11.653
3	8	3	3.049	9.844	2.923	16.478
3	8	4	2.659	9.576	2.659	19.428
3	8	5	2.594	10.955	2.6	19.651
4	3	2	3.741	6.188	3.741	12.723
4	3	3	3.029	11.755	3.029	22.392
4	3	4	2.791	10.263	2.787	23.597
4	3	5	2.651	11.032	2.651	22.601
4	8	2	4.088	8.988	4.088	14.718
4	8	3	3.112	13.094	3.112	24.926
4	8	4	2.834	13.787	2.821	25.721
4	8	5	2.682	13.338	2.682	27.913
estein30.txt						
3	3	2	6.498	40.51	6.498	70.303
3	3	3	4.815	63.213	4.815	124.17
3	3	4	4.507	64.972	4.344	126.992
3	3	5	4.232	68.917	4.232	129.738
3	8	2	5.635	34.228	5.635	64.163
3	8	3	4.485	68.429	4.485	103.183
3	8	4	4.135	54.422	4.186	102.655
3	8	5	3.98	51.101	3.967	122.439
4	3	2	6.587	34.353	6.587	65.288
4	3	3	5.059	58.154	5.073	111.65
4	3	4	4.562	68.879	4.601	122.189
4	3	5	4.472	72.048	4.405	132.408
4	8	2	6.193	33.587	6.193	45.865
4	8	3	4.82	48.501	4.82	99.065
4	8	4	4.489	53.537	4.436	111.615

4	8	5	4.223	64.326	4.22	114.691
estein40.txt						
3	3	2	6.923	95.624	6.923	189.595
3	3	3	5.832	173.803	5.815	202.096
3	3	4	5.365	188.782	5.356	186.392
3	3	5	5.146	169.919	5.018	192.047
3	8	2	7.671	113.516	7.671	113.248
3	8	3	6.284	181.188	6.286	173.984
3	8	4	5.608	207.232	5.517	206.953
3	8	5	5.182	225.347	5.254	223.886
4	3	2	7.058	96.574	7.058	103.099
4	3	3	5.37	175.799	5.374	173.709
4	3	4	4.872	223.054	4.874	218.524
4	3	5	4.657	215.838	4.656	239.461
4	8	2	6.572	115.242	6.572	111.673
4	8	3	5.179	177.705	5.159	184.368
4	8	4	4.789	188.855	4.791	212.113
4	8	5	4.486	183.312	4.486	204.774
estein50.txt						
3	3	2	9.181	313.094	9.181	334.978
3	3	3	7.021	553.313	7.023	495.187
3	3	4	6.374	628.686	6.342	525.594
3	3	5	6.036	620.198	5.984	527.441
3	8	2	9.343	295.25	9.343	253.349
3	8	3	6.954	534.053	6.954	483.705
3	8	4	6.243	557.458	6.226	481.755
3	8	5	5.912	565.708	5.908	494.036
4	3	2	8.681	273.877	8.681	244.041
4	3	3	6.759	462.782	6.673	434.43
4	3	4	5.973	584.17	6.015	531.58
4	3	5	5.687	526.519	5.607	513.091
4	8	2	8.114	241.47	8.098	219.416
4	8	3	6.271	436.928	6.271	407.962
4	8	4	5.758	528.721	5.719	456.348
4	8	5	5.279	506.807	5.293	454.898
estein60.txt						
3	3	2	10.744	727.454	10.744	619.865
3	3	3	7.963	1045.591	7.871	1158.104
3	3	4	6.921	1368.802	6.826	1350.078
3	3	5	6.41	1314.271	6.41	1480.899
3	8	2	9.163	606.112	9.163	620.577
3	8	3	7.161	1265.547	7.15	1348.005
3	8	4	6.419	1100.761	6.366	1147.966

3	8	5	5.984	1120.965	6.004	1145.276
4	3	2	10.623	640.185	10.623	615.615
4	3	3	7.664	1120.826	7.661	1251.491
4	3	4	6.796	1408.968	6.624	1473.604
4	3	5	6.284	1352.41	6.3	1563.407
4	8	2	10.376	675.998	10.376	620.784
4	8	3	7.212	1201.393	7.212	1335.666
4	8	4	6.332	1351.29	6.438	1353.985
4	8	5	6.017	1556.585	6.005	1493.665
estein70.txt						
3	3	2	11.456	1118.269	11.456	1129.148
3	3	3	8.16	2441.529	8.089	2266.131
3	3	4	7.236	2281.714	7.178	2839.532
3	3	5	6.882	2656.255	6.881	2735.298
3	8	2	10.164	1366.837	10.164	1344.093
3	8	3	7.771	2166.116	7.715	2379.553
3	8	4	6.857	2506.184	6.82	2535.715
3	8	5	6.508	3148.967	6.412	2447.359
4	3	2	11.842	1785.087	11.842	1235.727
4	3	3	8.616	3266.683	8.624	2377.69
4	3	4	7.517	3738.849	7.548	2644.483
4	3	5	7.054	2707.656	7.124	3917.296
4	8	2	11.819	1459.597	11.819	1990.376
4	8	3	8.623	2881.471	8.613	3484.131
4	8	4	7.599	3932.883	7.718	4334.386
4	8	5	7.179	4212.583	7.213	3227.992

Cuadro A.10: Soluciones obtenidas con GRASP

Bibliografía

[1] Warren P. Adams y Hanif D. Sherali. «A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems». En: *SIAM Journal on Discrete Math* (1990).

[2] Ibrahim Akgün. «New formulations for the hop-constrained minimum spanning tree problem via Sherali and Driscoll's tightened Miller-Tucker-Zemlin constraints». En: *Computers and Operations Research*. 2011. DOI: 10.1016/j.cor.2010.05.003.

[3] Ibrahim Akgün y Barbaros Ç Tansel. «New formulations of the Hop-Constrained Minimum Spanning Tree problem via Miller-Tucker-Zemlin constraints». En: *European Journal of Operational Research* (2011). ISSN: 03772217. DOI: 10.1016/j.ejor.2011.01.051.

[4] J. E. B Easley. «Or-library: Distributing test problems by electronic mail». En: *Journal of the Operational Research Society* (1990). ISSN: 14769360. DOI: 10.1057/jors.1990.166.

[5] J.E. Beasly. *OR-Library*. 2005.

[6] Thomas H. Cormen y col. *Introduction to Algorithms, Third Edition*. 2009. ISBN: 9780262033848. DOI: 10.1163/9789004256064_hao_introduction. arXiv: 2010(ret.29.4.2010).

[7] Geir Dahl. «The 2-hop spanning tree problem». En: *Operations Research Letters* (1998). ISSN: 01676377. DOI: 10.1016/S0167-6377(98)00029-7.

[8] Geir Dahl, Njål Foldnes y Luis Gouveia. «A note on hop-constrained walk polytopes». En: *Operations Research Letters* (2004). ISSN: 01676377. DOI: 10.1016/j.orl.2003.10.008.

[9] Geir Dahl, Luis Gouveia y C Requejo. «On Formulations and Methods for the Hop-Constrained Minimum Spanning Tree Problem». En: 2006, págs. 493-515. DOI: 10.1007/978-0-387-30165-5_19.

[10] Martin Desrochers y Gilbert Laporte. «Improvements and extensions to the Miller-Tucker-Zemlin subtour elimination constraints». En: *Operations Research Letters* (1991). ISSN: 01676377. DOI: 10.1016/0167-6377(91)90083-2.

[11] L. R. Esau y K. C. Williams. «On teleprocessing system design, Part II: A method for approximating the optimal network». En: *IBM Systems Journal* (1966). ISSN: 0018-8670. DOI: 10.1147/sj.53.0142.

[12] Thomas A. Feo y Mauricio G C Resende. «A probabilistic heuristic for a computationally difficult set covering problem». En: *Operations Research Letters* (1989). ISSN: 01676377. DOI: 10.1016/0167-6377(89)90002-3.

[13] Thomas A. Feo y Mauricio G.C. Resende. «Greedy Randomized Adaptive Search Procedures». En: *Journal of Global Optimization* (1995). ISSN: 09255001. DOI: 10.1007/BF01096763.

[14] Manuela Fernandes, Luis Gouveia y Stefan Voss. «Determining Hop-Constrained Spanning Trees with Repetitive Heuristics». En: *Journal of Telecommunications and Information Technology* (2007).

[15] Luis Gouveia. «Multicommodity flow models for spanning trees with hop constraints». En: *European Journal of Operational Research* (1996). ISSN: 03772217. DOI: 10.1016/0377-2217(95)00090-9.

[16] Luis Gouveia. «Using the Miller-Tucker-Zemlin constraints to formulate a minimal spanning tree problem with hop constraints». En: *Computers and Operations Research* (1995). ISSN: 03050548. DOI: 10.1016/0305-0548(94)00074-I.

[17] Luis Gouveia. «Using variable redefinition for computing lower bounds for minimum spanning and Steiner trees with hop constraints». En: *INFORMS Journal on Computing* (1998). ISSN: 10919856. DOI: 10.1287/ijoc.10.2.180.

[18] Luis Gouveia, Ana Paias y Dushyant Sharma. «Restricted dynamic programming based neighborhoods for the hop-constrained minimum spanning tree problem». En: *Journal of Heuristics* (2011). ISSN: 13811231. DOI: 10.1007/s10732-009-9123-5.

[19] Luís Gouveia y Dushyant Sharma. «Local Search Heuristics for the Hop-Constrained Minimum Spanning Tree Problem». En: (2007).

[20] Martin Gruber, Jano van Hemert y Günther Raidl. «Neighborhood Searches for the Bounded Diameter Minimum Spanning Tree Problem Embedded in a VNS, EA, and ACO». En: *GECCO 2006 - Genetic and Evolutionary Computation Conference*. Vol. 2. 2006. DOI: 10.1145/1143997.1144185.

[21] Maurice Karnaugh. «A New Class of Algorithms for Multipoint Network Optimization». En: *IEEE Transactions on Communications* (1976). ISSN: 00906778. DOI: 10.1109/TCOM.1976.1093334.

[22] S. Kirkpatrick, C. D. Gelatt y M. P. Vecchi. «Optimization by simulated annealing». En: *Science* (1983). ISSN: 00368075. DOI: 10.1126/science.220.4598.671.

[23] Joseph B. Kruskal. «On the Shortest Spanning Subtree of a Graph and the Traveling Salesman Problem». En: *Proceedings of the American Mathematical Society* (1956). ISSN: 00029939. DOI: 10.2307/2033241.

[24] Nicholas Metropolis y col. «Equation of state calculations by fast computing machines». En: *The Journal of Chemical Physics* (1953). ISSN: 00219606. DOI: 10.1063/1.1699114.

[25] C. E. Miller, A. W. Tucker y R. A. Zemlin. «Integer Programming Formulation of Traveling Salesman Problems». En: *Journal of the ACM* (1960). ISSN: 00045411. DOI: 10.1145/321043.321046.

[26] J. Mockus y col. «Bayesian discrete and global optimization». En: *Kluwer Academic Publisher* (1997). ISSN: 00401706. DOI: 10.1007/978-1-4757-2627-5.

[27] Manfred W. Padberg. «On the facial structure of set packing polyhedra». En: *Mathematical Programming* (1973). ISSN: 00255610. DOI: 10.1007/BF01580121.

[28] Marcelo Paris y Celso C. Ribeiro. «Reactive GRASP: An Application to a Matrix Decomposition Problem in TDMA Traffic Assignment». En: *INFORMS Journal on Computing* (2000). ISSN: 10919856.

[29] R. C. Prim. «Shortest Connection Networks And Some Generalizations». En: *Bell System Technical Journal* (1957). ISSN: 15387305. DOI: 10.1002/j.1538-7305.1957.tb01515.x.

[30] Mauricio G C Resende y José Luis Gonzáles Velarde. «Greedy randomized adaptive search procedures (GRASP)». En: *Encyclopedia of optimization* (2003). DOI: doi:10.1088/1475-7516/2007/08/005.

[31] Hanif D. Sherali y Warren P. Adams. «A hierarchy of relaxations and convex hull characterizations for mixed-integer zero-one programming problems». En: *Discrete Applied Mathematics* (1994). ISSN: 0166218X. DOI: 10.1016/0166-218X(92)00190-W.

[32] Hanif D. Sherali, Warren P. Adams y Patrick J. Driscoll. «Exploiting Special Structures in Constructing a Hierarchy of Relaxations for 0-1 Mixed Integer Problems». En: *Operations Research* (1998). ISSN: 0030-364X. DOI: 10.1287/opre.46.3.396.

[33] Hanif D. Sherali y Patrick J. Driscoll. «On Tightening the Relaxations of Miller-Tucker-Zemlin Formulations for Asymmetric Traveling Salesman Problems». En: *Operations Research* (2003). ISSN: 0030-364X. DOI: 10.1287/opre.50.4.656.2865.

[34] Laurence Wolsey. «Technical Note–Facets and Strong Valid Inequalities for Integer Programs». En: *Operations Research* 24 (abr. de 1976), págs. 367-372. DOI: 10.1287/opre.24.2.367.

[35] Kathleen A. Woolston y Susan L. Albin. «The design of centralized networks with reliability and availability constraints». En: *Computers and Operations Research* 15.3 (ene. de 1988), págs. 207-217. ISSN: 03050548. DOI: 10.1016/0305-0548(88)90033-0. URL: http://linkinghub.elsevier.com/retrieve/pii/0305054888900330.