Trabajo Fin de Máster

Optimización de rutas de camión eléctrico: comparación de métodos exactos y heurísticos

Natalia Álvarez Tejero

Máster en Técnicas Estadísticas

Curso 2025-2026

Propuesta de Trabajo Fin de Máster

Título en galego: Optimización de rutas de camión eléctrico: comparación de métodos exactos e heurísticos
Título en español: Optimización de rutas de camión eléctrico: comparación de métodos exactos y heurísticos
English title: Electric truck routing optimization: comparison of exact and heuristic methods
Modalidad: Modalidad A
Autora: Natalia Álvarez Tejero, Universidad de A Coruña
Directores: Alejandro Saavedra Nieves, Universidad de Santiago de Compostela, Laura Davila Pena, Universidad de Santiago de Compostela
Tutora: Lucía Tajuelo López, Trucksters
Breve resumen del trabajo: Este Trabajo Fin de Máster se centrará en el estudio y comparación de diferentes enfoques para resolver el problema de rutas de vehículos (VRP) en el contexto de una empresa de reparto de última milla. La estudiante analizará y aplicará técnicas de resolución exactas (como programación lineal con *solvers* tipo CBC o Gurobi), heurísticas y metaheurísticas (como algoritmos genéticos, búsqueda tabú o recocido simulado). El objetivo es comparar el rendimiento, calidad de soluciones y tiempo de cómputo en distintos escenarios logísticos reales proporcionados por la empresa, generando además recomendaciones para su aplicación práctica.

Don Alejandro Saavedra Nieves, de la Universidad de Santiago de Compostela, doña Laura Davila Pena, de la Universidad de Santiago de Compostelay doña Lucía Tajuelo López, de Trucksters, informan que el Trabajo Fin de Máster titulado

Optimización de rutas de camión eléctrico: comparación de métodos exactos y heurísticos

fue realizado bajo su dirección por doña Natalia Álvarez Tejero para el Máster en Técnicas Estadísticas. Estimando que el trabajo está terminado, dan su conformidad para su presentación y defensa ante un tribunal. Además, Don Alejandro Saavedra Nieves, doña Laura Davila Pena y doña Natalia Álvarez Tejero

√ sí □ no

autorizan a la publicación de la memoria en el repositorio de acceso público asociado al Máster en Técnicas Estadísticas.

En Santiago de Compostela, a 22 de enero de 2026.

El director:
Don Alejandro Saavedra Nieves

SAAVEDRA NIEVES ALEJANDRO - 45873390M
Firmado digitalmente por SAAVEDRA NIEVES ALEJANDRO - 45873390M
Fecha: 2026.01.21 23:49:43 +01'00'

La directora:
Doña Laura Davila Pena

Firmado digitalmente por DAVILA PENA LAURA - 78811589L
Fecha: 2026.01.21 23:45:32 +01'00'

La tutora:
Doña Lucía Tajuelo López

Signed by:
03F2DA7D09284E9...

La autora:
Doña Natalia Álvarez Tejero

Declaración responsable. Para dar cumplimiento a la Ley 3/2022, de 24 de febrero, de convivencia universitaria, referente al plagio en el Trabajo Fin de Máster (Artículo 11, Disposición 2978 del BOE núm. 48 de 2022), **la autora declara** que el Trabajo Fin de Máster presentado es un documento original en el que se han tenido en cuenta las siguientes consideraciones relativas al uso de material de apoyo desarrollado por otros/as autores/as:

- Todas las fuentes usadas para la elaboración de este trabajo han sido citadas convenientemente (libros, artículos, apuntes de profesorado, páginas web, programas,...)
- Cualquier contenido copiado o traducido textualmente se ha puesto entre comillas, citando su procedencia.
- Se ha hecho constar explícitamente cuando un capítulo, sección, demostración,... sea una adaptación casi literal de alguna fuente existente.

Y, acepta que, si se demostrara lo contrario, se le apliquen las medidas disciplinarias que correspondan.

Índice general

Resumen

Resumen en español

Los problemas de rutas de vehículos son un campo ampliamente estudiado en el ámbito de la optimización matemática, los cuales cuentan con diversas formulaciones según las necesidades de los clientes. Además, son muchos los métodos de resolución existentes, puesto que no resulta sencillo resolver el problema de forma directa.

En este Trabajo Fin de Máster se estudiará el caso particular de los problemas de rutas de vehículos eléctricos que plantea la *startup* española *Trucksters*. En este contexto, además de las restricciones usuales de capacidad, tiempo, distancia, etc., se añaden nuevas restricciones relativas a la carga de la batería, lo que implica una revisión de los métodos usuales de resolución. Se abordarán para ello métodos exactos y diferentes heurísticas incorporando estas nuevas restricciones.

Por último, se hará uso del lenguaje Python para obtener la solución del problema formulado, implementando un modelo exacto con el solver SCIP y desarrollando las heurísticas del vecino más próximo y *2-opt*. Estas soluciones se calcularán para diferentes instancias, permitiendo así comparar el rendimiento y la calidad de los distintos métodos de resolución.

English abstract

Vehicle routing problems constitute a widely studied field within mathematical optimization and include a variety of formulations depending on client requirements. Moreover, there exists a broad range of solution methods, as solving these problems directly is often challenging.

This document focuses on the specific case of electric vehicle routing problems proposed by the Spanish startup *Trucksters*. In this context, in addition to the usual constraints related to capacity, time, distance, etc., new battery-charging constraints are introduced, requiring an adaptation of traditional solution methods. Both exact methods and different heuristics will be explored while incorporating these additional restrictions.

Finally, the Python programming language will be used to obtain solutions to the formulated problem, implementing an exact model with the SCIP *solver* and developing the nearest-neighbour and *2-opt* heuristics. These solutions will be computed for multiple instances, thereby enabling a comparison of the performance and solution quality of the various solution methods.

Capítulo 1

Introducción

Trucksters es una *startup* española de logística y tecnología que trabaja en el sector del transporte de mercancías por carretera.

El transporte constituye uno de los pilares de la actividad económica y, al mismo tiempo, una de las principales fuentes de emisiones contaminantes. En España, el sector del transporte representa más del 30 % de las emisiones de gases contaminantes y, en concreto, el transporte por carretera conforma el 28.4 % del total [19]. Por ello, surge la necesidad de avanzar hacia modelos de movilidad más eficientes y sostenibles.

En este contexto, el uso de vehículos que empleen baterías eléctricas en el transporte constituye una alternativa clave. Numerosos estudios han demostrado la eficacia de los camiones eléctricos para reducir las emisiones de CO_2 y otros gases de efecto invernadero [3, 23]. Asimismo, los vehículos eléctricos contribuyen a disminuir la contaminación acústica, ya que el ruido del tráfico representa cerca del 70 % de las emisiones sonoras en las grandes ciudades [8]. Más allá del avance tecnológico, es importante destacar que en el sector del transporte por carretera la optimización juega un papel fundamental. Una planificación eficiente de rutas permite reducir distancias recorridas, tiempos operativos y, en consecuencia, emisiones. Esto es especialmente relevante en el caso de vehículos eléctricos, cuyos requerimientos (autonomía limitada, necesidad de recarga y disponibilidad de puntos de carga) añaden complejidad al proceso de planificación.

Trucksters basa su modelo operativo en un innovador sistema de relevos que permite recorrer mayores distancias en menos tiempo. En lugar de que un único conductor complete un trayecto largo, se establecen puntos de intercambio de remolques, que permiten mantener la mercancía en movimiento sin necesidad de paradas prolongadas. Gracias a este sistema, la compañía consigue mejorar la calidad de vida de los conductores, permitiendo que permanezcan a una menor distancia de sus hogares, y logra disminuir los kilómetros en vacío recorridos por sus camiones, reduciendo a su vez las emisiones de gases responsables del efecto invernadero.

Trucksters busca reducir su huella de carbono y avanzar hacia un transporte más sostenible, por lo que ha incorporado un camión eléctrico a su flota. Inicialmente, este vehículo se empleará en operaciones para tareas de carga y descarga en Cataluña. La implementación de la tractora eléctrica ha supuesto que el equipo operativo asuma manualmente la planificación de sus rutas, teniendo en cuenta las particularidades propias de los vehículos eléctricos, como la autonomía y los tiempos de recarga. Esta gestión adicional representa una carga significativa para el equipo, por lo que la empresa está enfocada en desarrollar soluciones que permitan automatizar la planificación diaria de su camión eléctrico, optimizando tanto la eficiencia operativa como la sostenibilidad del transporte. El problema que plantea

Trucksters consiste, por tanto, en desarrollar un algoritmo que permita integrar este camión eléctrico en sus operaciones diarias, reduciendo los tiempos de planificación, que actualmente ascienden a 30 minutos. La planificación actual tiene como objetivo minimizar los costes operativos; por ello, los modelos analizados deberán optimizar tanto los costes de recarga eléctrica como el consumo de batería. Además, se analizará y determinará qué algoritmo de optimización resulta más adecuado para la casuística concreta de la empresa, evaluando su rendimiento tanto sobre los datos actuales, como su capacidad de escalado ante futuros proyectos de mayor complejidad.

A lo largo de este Trabajo Fin de Máster se abordará la automatización de la planificación comparando distintos métodos de resolución. Para ello, se seleccionarán tres técnicas: un método exacto, la heurística del vecino más próximo y la heurística *2-opt*, que serán aplicados al caso real de *Trucksters* y evaluados mediante criterios de rendimiento y calidad de solución. Este objetivo guía todo el desarrollo del trabajo, desde la definición de un caso real hasta el análisis comparativo final.

El caso de estudio describe la situación real de los problemas de rutas determinado por un transportista, que debe hacer las entregas que son demandadas cada día en la provincia de Barcelona. Para ello, el camión eléctrico debe salir de la base operativa, recorrer los puntos de recogida de mercancía, visitar un almacén donde se efectuará la descarga y volver de nuevo a la base operativa. Además, debemos tener en cuenta que se dispone de un vehículo eléctrico, por lo que a lo largo de la ruta es necesario parar en un punto de carga. Se puede ver una representación del problema planteado en la Figura 1.1.

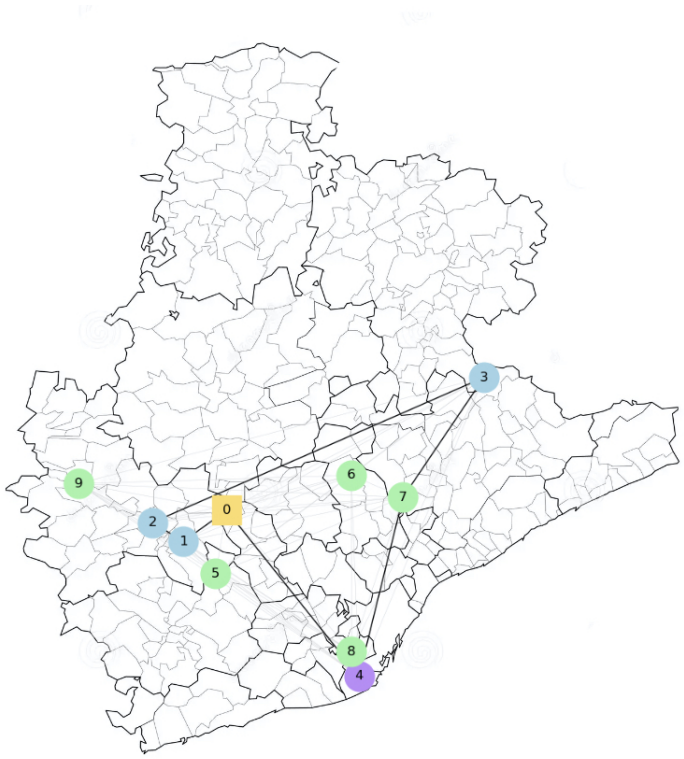


Figura 1.1: Representación del problema real. El nodo amarillo (0) se corresponde con la base operativa, los azules (1-3) son los puntos de carga, el nodo morado (4) el punto de descarga y los verdes (5-9) los cargadores. La solución óptima se correspondería con la ruta (0, 1, 2, 3, 7, 4, 0).

Además de abordar la casuística descrita, el trabajo incluye un análisis de escalabilidad de los

algoritmos, evaluando su desempeño en conjuntos de datos más grandes que simulan un mayor número de nodos de recogida y descarga, así como escenarios con distancias máximas de ruta variables, acordes a las capacidades de distintos vehículos eléctricos. Esta evaluación permite determinar la robustez y eficiencia de los métodos seleccionados frente a situaciones más complejas y potenciales ampliaciones de la operativa de *Trucksters*.

Para resolver las cuestiones planteadas, el Trabajo Fin de Máster se dividirá en dos partes: una metodología y una parte práctica. La metodológica estará compuesta por una revisión teórica y conceptual que servirá de base para la aplicación práctica posterior. La segunda parte recogerá el desarrollo de los casos de estudio y el análisis de los resultados obtenidos.

En el Capítulo 2 se presentarán los conceptos necesarios para abordar los problemas de rutas: los problemas de optimización (Sección 2.1) y la optimización en redes (Sección 2.2). En el Capítulo 3 se hará una revisión de los problemas del viajante de comercio, los problemas de rutas de vehículos y el caso concreto de los problemas de rutas de vehículos eléctricos, en las secciones 3.1, 3.2 y 3.3, respectivamente. Asimismo, en la Sección 3.4, se estudiarán los tipos de métodos de resolución de los problemas planteados, abordando los métodos exactos y las heurísticas y metaheurísticas más comunes.

En el Capítulo 4 se estudiará el problema concreto propuesto por *Trucksters* (Sección 4.1) y se planteará una formulación a partir del caso general, incorporando las nuevas restricciones relativas a la batería del vehículo. Se continuará en la Sección 4.2 con la selección de los métodos de resolución y una explicación de su desarrollo en los programas empleados. Además, se incluirá el pseudocódigo empleado para mayor comprensión y se fijarán los criterios de comparación. En la Sección 4.3 se implementarán los algoritmos previamente escogidos y se procederá con un estudio comparativo observando las diferencias de la calidad de las soluciones y rendimiento de cada uno. En el Capítulo 5 se extenderá el estudio comparativo a situaciones más complejas, generando instancias con un mayor número de nodos y añadiendo una limitación de distancia. En la Sección 5.1 se proporcionará una nueva formulación que se adapte a los nuevos datos empleados y en la Sección 5.2 se explicará la implementación de los algoritmos para estas nuevas instancias. Finalmente, en la Sección 5.3 se realizará un nuevo estudio comparativo de los métodos, con el apoyo de gráficos que permitan visualizar los resultados obtenidos.

Por último, en el Capítulo 6, se recogerán las conclusiones obtenidas a lo largo del trabajo y se verán las posibles líneas de trabajo futuro.

Capítulo 2

Estado del arte

El problema que se plantea en este Trabajo Fin de Máster es el de encontrar la mejor solución para una situación concreta con unas limitaciones iniciales. Este tipo de problema se enmarca dentro de la programación matemática, y más concretamente, dentro de los problemas de optimización.

En la Sección 2.1 se presentan los conceptos más básicos de los problemas de optimización, necesarios para describir el modelo que se abordará. Además, en la Sección 2.2 se hace una revisión de los problemas de optimización en redes, que sientan las bases de los problemas de rutas de vehículos, y se estudian los casos concretos del *problema de flujo en redes a coste mínimo*, y del *problema de transporte*.

2.1. Problemas de optimización

Los **problemas de optimización** constituyen una de las áreas fundamentales de la programación matemática. Están presentes en sectores como la ingeniería, la economía o la logística, ya que permiten encontrar soluciones a casuísticas muy variadas. En esta sección se tomará como referencia principal [1], que ofrece un recorrido por los fundamentos de la programación matemática.

Un problema de programación matemática se puede escribir del siguiente modo:

$$\begin{array}{lll} \text{minimizar} & f(x) & \\ \text{sujeto a} & g_i(x) \leq 0 & i = 1, \ldots, m \\ & h_j(x) = 0 & j = 1, \ldots, l \\ & x \in X, & \end{array} \tag{2.1}$$

donde f, g_i, h_j son funciones reales definidas en $\mathbb{R}^n$, X es un conjunto de $\mathbb{R}^n$ y $x = (x_1, \ldots, x_n)$ es un vector de n componentes. La función $f(x)$ se llama **función objetivo**, y las funciones g_i, h_j conforman **la región factible**. El problema consiste en encontrar los valores de las variables $x_1, \ldots, x_n$ que satisfagan las restricciones dadas en la región factible. A estos valores de x se les denomina **soluciones factibles**. El objetivo es determinar de entre todas las soluciones factibles aquellas que minimicen la función objetivo. En este sentido, si los puntos $x_1, \ldots, x_n$ encontrados satisfacen

$$f(x) \leq f(y), \quad \forall y \in X$$

se dice que x es un **óptimo global** del problema (2.1). Por el contrario, si la desigualdad se cumple únicamente en un entorno de x, se dice que x es un **óptimo local** del problema (2.1).

Un ejemplo sencillo de un problema de optimización es el siguiente:

$$\begin{aligned} \text{minimizar} \quad & f(x) = \sin(x_1) + x_2 \\ \text{sujeto a} \quad & x_1^2 + x_2^2 \leq 5 \\ & x_1 \geq 0 \\ & x_2 \geq 0. \end{aligned} \tag{2.2}$$

En este caso, la región factible corresponde al interior de la circunferencia de radio $\sqrt{5}$ centrada en el eje de coordenadas, únicamente en el primer cuadrante. Se aprecia que el punto donde la función objetivo alcanza su mínimo es en el $(x_1, x_2) = (0, 0)$, el cual constituye el óptimo global del problema (2.2). En la Figura 2.1 se muestra una representación de la región factible, junto con el punto óptimo (color azul oscuro) y las curvas de nivel de la función, que son algunas de las posibles soluciones del problema.

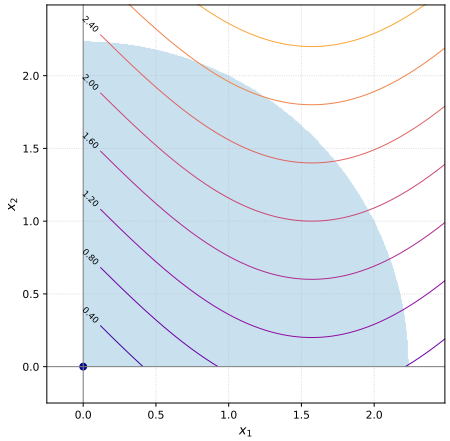


Figura 2.1: Representación de la región de soluciones factibles para el problema (2.2).

Según la naturaleza de las funciones que determinan la región factible de los problemas de optimización, estos se pueden clasificar en diferentes grupos.

2.1.1. Programación convexa

En esta sección se presentan los problemas de programación convexa, que, por las propiedades que cumplen, son interesantes desde el punto de vista metodológico. Para ello, se introducen en primer lugar las siguientes definiciones:

Definición 2.1. Un conjunto $S \subseteq \mathbb{R}^n$ se dice **convexo** si cualquier combinación convexa de puntos de S también pertenece a S. Es decir, si dados $x, y \in S$, entonces $\lambda x + (1 - \lambda)y \in S \quad \forall \lambda \in [0, 1]$.

Definición 2.2. Dado un conjunto no vacío y convexo $S \subseteq \mathbb{R}^n$, y una función $f : S \to \mathbb{R}$, la **función f es convexa** en S si, para todo $x \in S$ e $y \in S$ y para todo $\lambda \in (0, 1)$, se tiene

$$f(\lambda x + (1 - \lambda)y) \leq \lambda f(x) + (1 - \lambda) f(y).$$

Definición 2.3. Una función $f : S \subset \mathbb{R}^n \to \mathbb{R}$ se dice **afín** si existe $a \in \mathbb{R}^n$ y $b \in \mathbb{R}$ tales que

$$f(x) = a^T x + b, \quad \forall x \in S.$$

Los problemas de **programación convexa** son aquellos en los que tanto la función objetivo f como la región factible son convexas En concreto, las funciones g_i tienen que ser convexas y las funciones h_j afines.

Este tipo de problemas son muy interesantes por sus propiedades, que hacen su resolución más sencilla. A continuación se presentan algunos resultados destacados sobre su resolución:

Teorema 2.4. *Sea $S \subseteq \mathbb{R}^n$ un conjunto no vacío y convexo y sea $f : S \to \mathbb{R}$ una función convexa en S. Si el vector x es un óptimo local del problema (2.1), entonces*

(i) El vector x es un óptimo global.

(ii) Si x es un óptimo local estricto o f es estrictamente convexa, entonces x es el único óptimo global.

Lema 2.5. *Dado un problema de programación convexa, toda combinación convexa de dos puntos x e y que sean óptimos locales (globales) también será un óptimo local (global).*

Se presenta a continuación un ejemplo de un problema de programación convexa. El problema (2.3) es un ejemplo de esta formulación, cuya representación se puede ver en la Figura 2.2.

$$\begin{aligned} \text{minimizar} \quad & f(x) = x_1^2 + x_2^2 \\ \text{sujeto a} \quad & 0 \leq x_1 \leq 4 \\ & 0 \leq x_2 \leq 4. \end{aligned} \tag{2.3}$$

En este caso, la región factible son los x_1 y x_2 no negativos y no mayores que 4, lo que forma un cuadrado, y que se corresponde con un conjunto convexo. La función objetivo son circunferencias concéntricas de centro $(0, 0)$. Por la definición vista, se puede observar que se trata de una función convexa.

Cada radio de las circunferencias posibles forma una curva de nivel. El óptimo global vuelve a ser el punto $(0, 0)$, representado en azul oscuro (Figura 2.2).

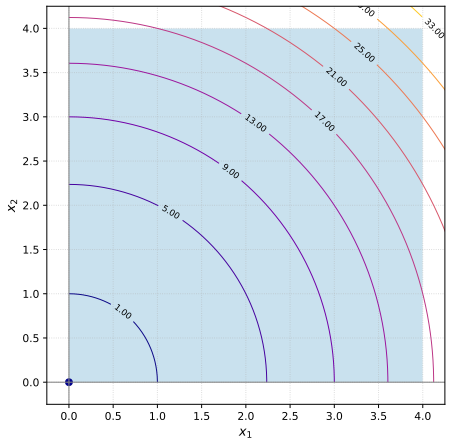


Figura 2.2: Representación del problema (2.3).

Dentro de la programación convexa, la **programación lineal** constituye un caso particular de especial interés. En un problema de programación lineal, tanto la función objetivo como las funciones

que definen las restricciones son afines, por lo que son convexas, y la región factible resulta ser también convexa. En consecuencia, todos los resultados anteriores para problemas convexos son aplicables también a los problemas de programación lineal.

2.1.2. Programación entera

Hasta ahora, se han considerado problemas en los que las variables x tomaban valores reales. Sin embargo, en ocasiones se justifica que las variables tomen valores en $\mathbb{Z}^n$. Los problemas de **programación entera** son aquellos en los que las variables x deben tomar valores enteros. Ahora, el problema que se quiere resolver, de forma general, es el siguiente:

$$\begin{array}{rll} \text{minimizar} & f(x) & \\ \text{sujeto a} & g_i(x) \leq 0 & i = 1, \dots, m \\ & h_j(x) = 0 & j = 1, \dots, l \\ & x \in \mathbb{Z}^n. & \end{array} \tag{2.4}$$

Estos problemas ya no cuentan con las buenas propiedades que caracterizaban a los problemas de optimización convexa, ya que su región factible no es un conjunto convexo. Es por ello que para resolverlos se necesitan métodos más sofisticados que en el caso anterior.

Además, cuando las variables de decisión pueden tomar únicamente dos valores, $x_i \in \{0, 1\}$, el problema de optimización asociado se dice un problema de **programación entera binaria**.

Por ejemplo, se quiere minimizar el coste de manufactura de tres productos, seleccionando cuáles producir. El primero, x_1, tiene un precio de 2 unidades; el segundo, x_2, de 3 unidades; y el tercero, x_3, de 5. Se debe tener en cuenta que como mínimo es necesario producir dos de ellos. Este problema se puede formular de la siguiente manera:

$$\begin{array}{rl} \text{minimizar} & 2x_1 + 3x_2 + 5x_3 \\ \text{sujeto a} & x_1 + x_2 + x_3 \geq 2 \\ & x_1, x_2, x_3 \in \{0, 1\}. \end{array} \tag{2.5}$$

En este caso, las variables tomarán el valor 1 si se producen y el valor 0 en caso contrario. Se ve fácilmente cómo la solución óptima es $(x_1, x_2, x_3) = (1, 1, 0)$, que se corresponde con los productos que menos gastos requieren.

Si el problema engloba variables enteras y continuas, este será un problema de **programación entera mixta** (MIP)[1]. Se verá más adelante que esta es la formulación que basará la metodología a emplear para la resolución del problema propuesto.

Retomando el ejemplo anterior, se quiere minimizar el coste de los dos productos seleccionados. En este caso se requiere determinar la cantidad producida de cada uno. El producto x_1 se mide en metros, pero solo permite ser cortado por unidades (es decir, un metro, dos metros,...) y tiene un coste 5 unidades monetarias cada uno. Por su parte, el producto x_2 también se mide en metros, pero en este caso se permite la producción por fracciones (es decir, medio metro, tres cuartos,...), con un coste de 4 unidades cada metro. Nótese que x_1 solo pueda tomar valores enteros, mientras que x_2 toma valores

[1] Del inglés *Mixed-Integer Programming*.

continuos. La empresa quiere producir, como mínimo, 10 productos.

$$\begin{aligned} \text{minimizar} \quad & 5x_1 + 4x_2 \\ \text{sujeto a} \quad & x_1 + x_2 \geq 10 \\ & x_1 \in \mathbb{Z} \\ & x_2 \in \mathbb{R}^+. \end{aligned} \tag{2.6}$$

El problema planteado se recoge en la formulación (2.6). Dado que tenemos variables enteras, x_1, y variables continuas, x_2, cumple las hipótesis de un problema de programación mixta. La solución óptima es el punto $(x_1, x_2) = (0, 10)$.

Los modelos de programación entera y de programación entera mixta resultan especialmente útiles en problemas reales donde ciertas decisiones solo pueden tomarse en valores discretos, como seleccionar qué instalaciones abrir, qué rutas activar o qué vehículos utilizar. En particular, muchos problemas de optimización en redes, como los modelos de flujo en redes o transporte, pueden formularse como problemas de programación lineal o entera, en los que las variables de decisión representan flujos o conexión entre nodos. Esta relación entre programación entera y optimización en redes motiva el estudio de la optimización en redes que se presenta en la siguiente sección.

2.2. Optimización en redes

Este apartado presenta los **problemas de optimización en redes**, los cuales constituyen un caso particular de los problemas de programación lineal (aquellos en los que las funciones f, g_i y h_j son funciones lineales). Esta clase de problemas aparecen en infinidad de campos, como el diseño de carreteras entre localidades, el cálculo de la ruta más corta entre varios puntos o problemas de asignación de tareas. La principal referencia que se sigue son los apuntes de la asignatura *Programación lineal y entera* [12], así como el Trabajo Fin de Máster [17], que desarrolla una situación real como un problema de optimización en redes.

Asimismo, como se mostrará en el Capítulo 3, los problemas clásicos de optimización combinatoria, como el *problema del viajante de comercio*, el *problema de rutas de vehículos* o el *problema de rutas de vehículos eléctricos*, pueden formularse como problemas de redes, al modelarse mediante nodos, arcos y flujos. De este modo, el análisis de los problemas de optimización en redes servirá como base conceptual y metodológica para abordar la casuística desarrollada en el Capítulo 4.

2.2.1. Conceptos básicos

La definición de esta clase de problemas se basa en el concepto de grafo. Un **grafo** G es un par (V, E) consistente en un conjunto V de elementos llamados **nodos** o vértices y un conjunto E cuyos elementos representan **arcos** o aristas. Los grafos permiten representar relaciones entre distintos elementos y constituyen la estructura fundamental sobre la que se construyen los modelos de redes.

Según cómo sean los elementos de E, se puede distinguir dos tipos principales de grafos:

- *Grafos dirigidos*: son aquellos en los que $V \subset E \times E$, es decir, los arcos son pares ordenados. El arco (i, j) con $i, j \in V$, empieza en el nodo i y termina en el nodo j. Se representan con flechas para indicar el sentido.
- *Grafos no dirigidos*: en ellos G está compuesto por subconjuntos de E de dos elementos. En este caso, los arcos no tienen dirección, de modo que el arco (i, j) es equivalente al arco (j, i).

En la Figura 2.3 se puede ver un ejemplo de un grafo dirigido (izquierda) y de otro no dirigido (derecha). Nótese que ambos grafos contienen el mismo número de nodos y el mismo número de arcos, la diferencia fundamental radica en la orientación de los arcos. Así, en el primer grafo se permite ir de 1 a 2 y de 2 a 3, pero no el recorrido inverso. Sin embargo, en el grafo de la derecha, dado que no es dirigido, ambos caminos estarían permitidos.

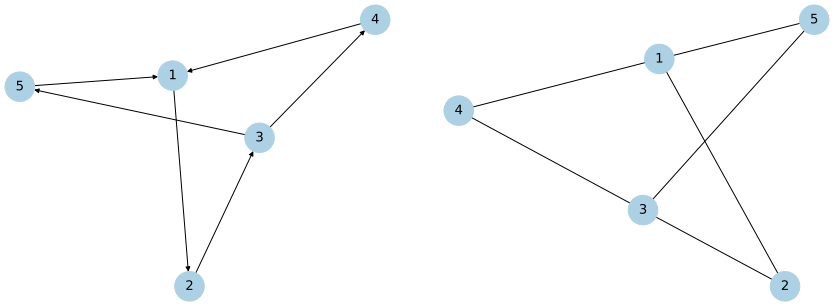


Figura 2.3: Representación de un grafo dirigido (izquierda) y un grafo no dirigido (derecha).

A continuación, se presentan otros conceptos necesarios. En un grafo no dirigido G, se denomina **cadena** a la secuencia de aristas distintas $(a_1, a_2, \dots a_r)$, tales que $a_l = (v_{l-1}, v_l)$, para $l \in \{1, 2, \dots, r\}$, siendo $(v_0, v_1, \dots, v_r)$ vértices del grafo. Si la cadena empieza y termina en el mismo nodo $(v_0 = v_r)$, se trata de una **cadena cerrada**. Cuando todos los vértices de la cadena son distintos, se define como un **camino**. Se hablará de **circuito** o ciclo cuando se corresponda con una cadena cerrada en la que los únicos nodos coincidentes son el inicial y el final. Definiremos un **circuito hamiltoniano** como un circuito que contiene a todos los vértices de la red exactamente una vez.

Se verá a continuación, con ayuda del grafo no dirigido de la Figura 2.3, un ejemplo de estos conceptos. Si se toma la sucesión de nodos $(1, 2, 3, 5, 1)$, esta será una cadena cerrada, pues se empieza y se termina en el mismo nodo, y será además un circuito, ya que no se repite ningún nodo, excluyendo el inicial y el final. Si tomamos la cadena $(1, 2, 3, 4)$, esta se corresponderá con un camino, pero no con un circuito o cadena cerrada. Podemos formar un circuito hamiltoniano recorriendo todos los nodos en el orden (2, 3, 4, 1, 5, 3).

Un grafo, entendido como una estructura formada por nodos y aristas, no es suficiente para plantear un problema de optimización. Para ello, se introduce el concepto de red, que consiste en un grafo con uno o más números asociados con cada arco o nodo. Estos valores pueden representar costes, distancias, fiabilidades u otros parámetros de interés.

Llamaremos **flujo** al envío de elementos u objetos de un lugar a otro dentro de una red. Estos objetos serán las unidades de flujo, que pueden ser personas, bienes, agua,... Denotaremos por f_k al flujo correspondiente al arco k. Estos modelos se llaman **modelos de redes con flujo**.

A cada arco k le asignaremos tres parámetros: la **cota inferior** (l_k), que es la cantidad mínima de flujo que debe pasar por el arco k, la **capacidad** (u_k), que es la cantidad máxima de flujo que el arco k puede transportar, y el **coste o beneficio** (c_k), que si es positivo denota el coste por unidad de flujo que pasa por el arco k y si es negativo representa los beneficios.

2.2.2. El problema de flujo en redes a coste mínimo

Introducida la notación necesaria, en este punto se presentan los **problemas de flujo en redes a coste mínimo** (PFCM). Dada una red con capacidades, estos problemas consisten en determinar el flujo que ha de pasar por cada arco de tal manera que el coste asociado sea mínimo y se cumplan las restricciones de conservación de flujo y las impuestas por las capacidades.

Formalmente, las restricciones de conservación de flujo establecen que, en cada nodo, el flujo que entra debe ser igual al flujo que sale (excepto en los nodos origen y destino del sistema). Además, cada arco debe satisfacer las limitaciones impuestas por sus cotas inferior y superior, es decir:

$$l_k \leq f_k \leq u_k.$$

El objetivo es, por tanto, encontrar la asignación óptima de flujos f_k que minimice el coste total de la red cumpliendo dichas condiciones.

El PFCM se puede formular como un problema de optimización matemática del siguiente modo:

$$\text{minimizar} \quad \sum_{i,j \in V} c_{ij} f_{ij} \tag{2.7}$$

$$\text{sujeto a} \quad \sum_{i,j \in V} f_{ij} - \sum_{j,h \in V} f_{jh} = 0 \tag{2.8}$$

$$f_{ij} \leq u_{ij} \qquad i,j \in V \tag{2.9}$$

$$f_{ij} \geq l_{ij} \qquad i,j \in V. \tag{2.10}$$

Nótese que se ha modificado la notación para mayor simplicidad. En lugar de etiquetar cada flujo, coste y cota por el índice del arco correpondiente k, se ha denotado en función de los nodos que se conectan, de forma que $k = (i, j)$.

La función objetivo (2.7) representa el coste total asociado al flujo que circula por la red y busca minimizar el coste de transportar unidades de flujo desde el nodo i hasta el nodo j. La restricción (2.8) hace referencia a la conservación de flujo. Las ecuaciones (2.9) y (2.10) son las restricciones relativas a las limitaciones de la red, estableciendo el límite superior e inferior de flujo en cada arco.

Podríamos ejemplificar un problema de este tipo con una red, donde los nodos son ciudades y los arcos representan carreteras entre ellas. Se quiere abastecer de un determinado producto a las ciudades, enviándolo desde una ciudad origen hasta una ciudad destino. Cada arco tiene un coste asociado c_{ij} por unidad de flujo que circula por él, así como una capacidad máxima u_{ij} y mínima l_{ij}. El objetivo es determinar la cantidad de producto que debe enviarse por cada carretera para minimizar el coste total, asegurando que se respeten las capacidades y la conservación de flujo en cada ciudad.

2.2.3. El problema de transporte

El **problema de transporte** es un caso particular del problema de flujo en redes a coste mínimo, en el cual se tienen que minimizar los costes asociados a los flujos de la red. Sin embargo, ahora se trata de un grafo bipartito, donde los nodos se dividen en dos conjuntos diferenciados: N_1 y N_2.

Los nodos i de N_1 se denominan *nodos de suministro* y tienen una capacidad asociada $s_i > 0$. Necesariamente todos los arcos del problema partirán de un nodo de este conjunto. Por otro lado, los nodos j de N_2 se definen como *nodos de demanda* y tienen asociada una demanda $d_j > 0$. Los arcos de la red incidirán en los nodos de este conjunto. Cada arco $k = (i, j)$ representa un canal de distribución con un coste asociado c_k.

El problema de transporte consiste en usar los nodos de suministro para satisfacer todas las demandas a coste mínimo. Para que el problema tenga solución es necesario que $\sum_{i \in N_1} s_i \geq \sum_{j \in N_2} d_j$. Se puede formular como un problema de optimización del siguiente modo:

$$\text{minimizar} \quad \sum_{i,j \in V} c_{ij} f_{ij} \tag{2.11}$$

$$\text{sujeto a} \quad \sum_{i,j \in V} f_{ij} \leq s_i \qquad i \in N_1 \tag{2.12}$$

$$\sum_{i,j \in V} f_{ij} = d_j \qquad j \in N_2 \tag{2.13}$$

$$f_{ij} \geq 0 \tag{2.14}$$

La función objetivo (2.11) representa el coste total asociado al flujo que circula por la red, que es lo que se quiere minimizar. La restricción (2.12) es necesaria para asegurar que se respeta la capacidad de los orígenes. Por otro lado, la restricción (2.13), indica que a cada elemento de N_2 han de llegar exactamente d_j unidades, para satisfacer la demanda de los destinos.

En este problema, es muy común considerar los nodos de N_1 como almacenes o fábricas y los nodos de N_2 como clientes o tiendas.

Se puede observar una representación de un ejemplo del problema de transporte en la Figura 2.4, donde los nodos de suministro $1, 2$ y 3 son los nodos representados en la parte izquierda y los nodos de demanda $4, 5$ y 6 los situados en la parte derecha. Cada uno de ellos está asociado a una capacidad (s_i) o demanda (d_j) respectivamente. Además, los arcos tienen un coste por unidad de flujo asociado (c_{ij}) y un flujo (f_{ij}) que circula por ellos y que es lo que queremos determinar para minimizar el coste total.

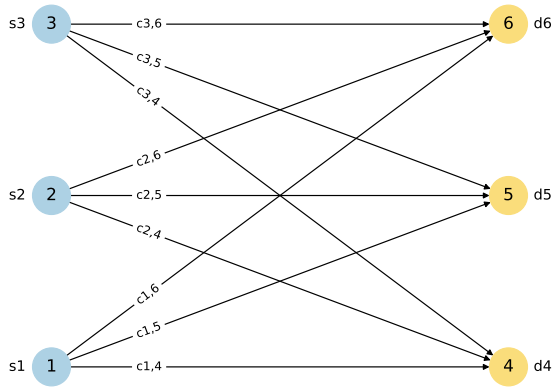


Figura 2.4: Representación del problema de transporte.

En el problema formulado anteriormente, se ha considerado un caso que puede no ser equilibrado, es decir, que la suma de las capacidades de los nodos de suministro no sea igual a la suma de las demandas de los nodos de demanda. La condición necesaria y suficiente para que un problema de transporte tenga una solución factible es que el problema esté equilibrado. Si no es así, se puede transformar en uno equilibrado añadiendo un nodo ficticio de suministro o de demanda, según sea necesario.

Capítulo 3

Problemas de rutas

En el capítulo anterior se han introducido los fundamentos teóricos de la optimización, desde los modelos convexos y enteros hasta los problemas clásicos de optimización en redes. Estos contenidos proporcionan el marco teórico necesario para abordar uno de los campos más relevantes de la optimización combinatoria: los problemas de rutas.

En este capítulo se presentan los principales problemas de optimización asociados a la planificación y diseño de rutas. Se parte del problema más general, el *problema del viajante de comercio* (Sección 3.1). A continuación, se introducen los *problemas de rutas de vehículos* (Sección 3.2), los cuales son una extensión del problema anterior, incorporando múltiples vehículos y restricciones operativas propias. Posteriormente, se aborda el caso particular de los *problemas de rutas de vehículos eléctricos* (Sección 3.3), que constituye el objeto de estudio de este Trabajo Fin de Máster. Además de presentar estas formulaciones, se describen las principales metodologías empleadas para su resolución (Sección 3.4). Dado que estos problemas suelen ser difíciles de resolver, se introducen tanto métodos exactos, diseñados para encontrar soluciones óptimas, como métodos heurísticos, que ofrecen soluciones de buena calidad con un coste computacional reducido.

3.1. El problema del viajante de comercio

En esta sección se hará una revisión del **problema del viajante de comercio** (TSP)[1], que constituye uno de los problemas más estudiados en optimización combinatoria y que, además, sirve de base para el desarrollo del problema de rutas de vehículos. Se usará [18] como referencia principal para esta sección.

Este problema fue estudiado por primera vez en el siglo XVIII por los matemáticos Hamilton y Kirkman, pero no fue hasta el siglo XX cuando se empezó a analizar en profundidad. Es un problema NP-duro, lo que implica que no se conocen algoritmos que lo resuelvan en tiempo polinomial ni para verificar la optimalidad de una solución dada. En el caso de una casuística con n ciudades, el número posible de rutas es $\frac{(n-1)!}{2}$, lo que hace inviable la búsqueda exhaustiva de la solución óptima para valores grandes de n. Es por ello que se han desarrollado numerosos métodos heurísticos y metaheurísticos para encontrar soluciones aproximadas en tiempos razonables.

En el problema del viajante de comercio se busca encontrar la ruta más corta que permite a un viajante visitar un conjunto de ciudades exactamente una vez y regresar a la ciudad de origen. Este

[1] Del inglés *Traveling Salesman Problem*.

problema se puede modelar mediante un grafo G dirigido con n nodos, donde cada nodo representa una ciudad y cada arco (i, j) tiene un coste asociado c_{ij}, que representa la distancia o el coste de viajar de la ciudad i a la ciudad j.

Si se plantea como un problema de flujo en redes, se tratará de un problema de transporte donde cada nodo tiene una demanda y una capacidad igual a 1. En este modelo se asignan flujos a los arcos, que denotaremos por $x_{ij} \in \{0, 1\}$, indicando si el viajante se desplaza de la ciudad i a la ciudad j o no. El objetivo es encontrar un circuito hamiltoniano, que se define como un circuito que contiene a todos los vértices de la red exactamente una vez.

La formulación matemática del TSP es la siguiente:

$$\text{minimizar} \quad \sum_{i,j \in V} c_{ij} x_{ij} \tag{3.1}$$

$$\text{sujeto a} \quad \sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V \tag{3.2}$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V \tag{3.3}$$

$$\sum_{i \in S, j \in V \setminus S} x_{ij} \geq 1 \qquad \forall S \subset V \tag{3.4}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in V \tag{3.5}$$

La función objetivo (3.1) busca minimizar el coste total del recorrido, que es la suma de los costes asociados a cada arco que forme parte de la ruta. Las restricciones (3.2) y (3.3) aseguran que a cada ciudad se llega y se sale exactamente una vez, cumpliendo las condiciones del problema por conservación de flujo. La restricción (3.4) es necesaria para evitar la formación de subciclos en la solución, asegurando que la ruta visitará todas las ciudades en un único recorrido. Para cada subconjunto de nodos S, es necesario que el vehículo tenga, al menos, una arista de salida. Finalmente, la restricción (3.5) define las variables de decisión como binarias, indicando si un arco forma parte de la ruta o no.

Un ejemplo de problema TSP se puede observar en la Figura 3.1, donde se representa una red con 5 ciudades, nodos de color azul (1-5), junto con la ciudad de origen (0), que aparece representada de color naranja. La ruta que minimiza el camino que tiene que realizar el viajante se representa con los arcos dirigidos de color negro.

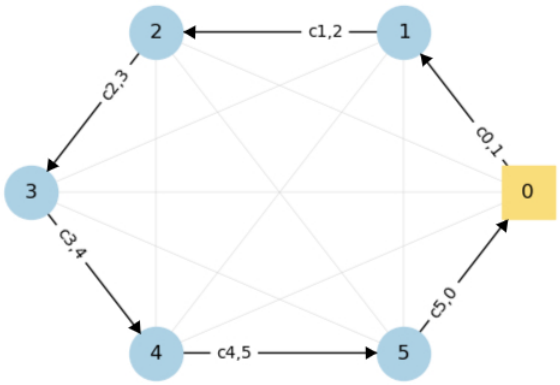


Figura 3.1: Representación del TSP.

Los problemas TSP tienen múltiples aplicaciones prácticas, como la planificación de rutas, la logística de distribución y la optimización de itinerarios de viaje. En concreto, el TSP sirve como base para el desarrollo del problema de rutas de vehículos, que se verá a continuación.

3.2. El problema de rutas de vehículos

Los **problemas de rutas de vehículos** (VRP)[1] son una adaptación del problema teórico anterior a la industria del transporte y la logística. Fue planteado por primera vez en 1959 por Dantzig y Ramser [7] y desde entonces ha sido objeto de numerosos estudios e investigaciones debido a su relevancia práctica y su complejidad computacional. Su objetivo principal consiste en determinar las rutas óptimas que deben seguir un conjunto de vehículos, partiendo de un almacén o base operativa y satisfaciendo las demandas de un conjunto de clientes, minimizando los costes asociados al transporte.

El VRP tiene diversas aplicaciones en diferentes sectores, como la logística y distribución, servicios de mensajería y paquetería, transporte público o sanitario, entre otros. En un mundo cada vez más globalizado, estas aplicaciones son fundamentales para garantizar la eficiencia y sostenibilidad de las operaciones. Además, hace del VRP un problema de gran impacto económico y social. Como el TSP, se trata de un problema NP-duro. Por todo ello, se vuelve crucial el estudio y desarrollo de métodos eficientes para su resolución.

Se han desarrollado múltiples variantes del VRP para adaptarse a diferentes contextos y restricciones. Entre las más conocidas están el problema VRP con capacidades (CVRP)[2], el VRP con flota heterogénea (FSMVRP)[3] o el VRP con ventanas de tiempo (VRPTW)[4]. Estas variantes aparecen recogidas en [22]. Dado que se trata de un ámbito de investigación en continua evolución, siguen desarrollándose nuevos modelos para dar respuesta a las necesidades logísticas actuales. Entre ellos destaca el problema de rutas de vehículos con colaboración en clientes compartidos (SSC-VRP)[5], descrito en [9].

Además, para abordar la resolución de estos problemas y todas sus variantes, existen diferentes

[1] Del inglés *Vehicle Routing Problem.*
[2] Del inglés *Capacitated Vehicle Routing Problem.*
[3] Del inglés *Fleet Size and Mix Vehicle Routing Problem.*
[4] Del inglés *Vehicle Routing Problem with Time Windows.*
[5] Del inglés *Shared Customer Collaboration Vehicle Routing Problem.*

enfoques y técnicas de resolución. En la Sección 3.4 se abordarán métodos heurísticos, como el algoritmo del vecino más próximo o el algoritmo *2-opt*, útiles para encontrar soluciones aproximadas en tiempos razonables.

Se presentará a continuación una formulación matemática del CVRP, que es una de las variantes más estudiadas y servirá de apoyo para poder presentar el problema de rutas de vehículos eléctricos en la siguiente sección. La notación empleada, obtenida de [22], será:

- V=$\{0, 1, \ldots, n\}$ es el conjunto de nodos, donde el nodo 0 representa el almacén y los nodos $1, \ldots, n$ representan los clientes.
- S se corresponde con un conjunto de clientes arbitrario tal que $S \subseteq V \backslash \{0\}$.
- $K = \{1, \ldots, m\}$, con $m \in \mathbb{N}, m \geq 1$ es el conjunto de vehículos disponibles, cada uno con capacidad C.
- δ_i hace referencia a la demanda del cliente $i \in V \backslash \{0\}$.
- $r(S)$ indica la cantidad mínima de vehículos necesarios para satisfacer la demanda del conjunto S.

La formulación matemática del CVRP es la siguiente:

$$\text{minimizar} \quad \sum_{i,j \in V} c_{ij} x_{ij} \tag{3.6}$$

$$\text{sujeto a} \quad \sum_{j \in V} x_{0j} = m \tag{3.7}$$

$$\sum_{i \in V} x_{i0} = m \tag{3.8}$$

$$\sum_{j \in V} x_{ij} = 1 \qquad \forall i \in V \backslash \{0\} \tag{3.9}$$

$$\sum_{i \in V} x_{ij} = 1 \qquad \forall j \in V \backslash \{0\} \tag{3.10}$$

$$\sum_{i \in S, j \in V \backslash S} x_{ij} \geq r(S) \qquad \forall S \subset V \backslash \{0\} \tag{3.11}$$

$$x_{ij} \in \{0, 1\} \qquad \forall i, j \in V \tag{3.12}$$

Vemos que, como en el TSP, la función objetivo (3.6) busca minimizar el coste total del recorrido, que es la suma de los costes asociados a cada arco que conforma la ruta. La restricción (3.7) impone que los m vehículos empleados deben salir del almacén, mientras que la restricción (3.8) asegura que los m vehículos vuelven al mismo punto al finalizar la ruta. Las restricciones (3.9) y (3.10) garantizan que cada cliente es visitado exactamente una vez. La restricción (3.11) actúa como eliminación de subtours y asegura que la demanda total de los clientes no sea superior a C, como veremos a continuación. Por último, la restricción (3.12) indica que las variables x_{ij} son binarias y tomarán el valor 1 si el arco (i, j) forma parte de la ruta y 0 en caso contrario.

Para determinar el número mínimo de vehículos necesarios $r(S)$ es necesario resolver el siguiente

problema:

$$\text{minimizar} \quad \sum_{k \in K} y_k \tag{3.13}$$

$$\text{sujeto a} \quad \sum_{i \in S} \delta_i z_{ik} \leq y_k C \qquad \forall k \in K \tag{3.14}$$

$$\sum_{k \in K} z_{ik} = 1 \qquad \forall i \in S \tag{3.15}$$

$$y_k \in \{0, 1\} \qquad \forall k \in K, \tag{3.16}$$

$$z_{ik} \in \{0, 1\} \qquad \forall i \in S, k \in K, \tag{3.17}$$

donde K es el conjunto de vehículos disponibles, z_{ik} es la variable binaria que indica si el cliente i es atendido por el vehículo k e y_k se corresponde con la variable binaria que indica si el vehículo k es utilizado.

El valor de $r(S)$ será la suma de las variables y_k que cumplan las restricciones planteadas, es decir, el valor de la función objetivo (3.13). La primera, ecuación (3.14), asegura que la demanda total de los clientes que son atendidos por el vehículo k no supere la capacidad del mismo. La restricción (3.15) garantiza que cada cliente es atendido por un único vehículo. Finalmente, las restricciones (3.16) y (3.17) define el dominio de las variables binarias y_k y z_{ik}.

La formulación 3.6 recoge un modelo general para el CVRP que se puede adaptar fácilmente a la casuística del problema. Por ejemplo, el conjunto K es un conjunto no acotado en el que suponemos que no existe una limitación de los vehículos de los que podemos hacer uso, sin embargo, esto se puede adaptar a un problema donde dispongamos de un número ilimitado de ellos.

Nótese además que la cota inferior para $r(S)$ es $\lceil \frac{\sum_{i \in S} \delta_i}{C} \rceil$, por lo que podríamos sustituir este valor en la restricción (3.11) para acotar la región fatible y reducir los tiempos de computación.

Se puede ver un ejemplo del problema presentado en la Figura 3.2. Esta es una modificación del ejemplo propuesto en la Figura 3.1, donde de nuevo, el nodo origen (0) aparece representado de color naranja y los 5 nodos relativos a los clientes (1-5) de color azul. En este caso, se hace uso de dos vehículos para poder satisfacer las demandas de los clientes cumpliendo la restricción de capacidad.

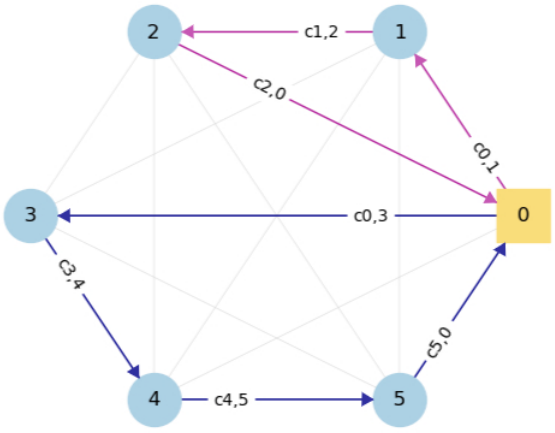


Figura 3.2: Representación CVRP.

3.3. El problema de rutas de vehículos eléctricos

Los **problemas de rutas de vehículos eléctricos** (EVRP)[1] son una extensión de los problemas VRP, en los que se incorporan las particularidades operativas de los vehículos eléctricos. En este contexto, los medios de transporte empleados, generalmente camiones o furgonetas de reparto, presentan limitaciones en la distancia que pueden recorrer debido a la capacidad finita de sus baterías. Por ello, es necesario considerar la planificación de las paradas de recarga a lo largo de las rutas.

La necesidad de recargar las baterías introduce restricciones adicionales en comparación con los VRP, donde se deben tener en cuenta situaciones temporales y económicas a mayores. Por un lado, los vehículos requieren un tiempo específico para realizar la recarga, lo que aumenta el tiempo necesario para completar la ruta. Por otro lado, el proceso de carga conlleva un coste asociado al consumo energético, que debe tenerse en cuenta en la función objetivo.

Actualmente, numerosas empresas están incorporando estos novedosos medios de transporte, por lo que el dilema de obtener soluciones óptimas que minimicen los costes de este problema se vuelve cada vez más latente. Gracias a este aumento de vehículos eléctricos, cada vez se cuenta con más puntos de recarga, lo que hace que no haya que desviarse en exceso de las rutas fijadas y se logren mejores soluciones.

El objetivo principal de los EVRP se basa en encontrar las rutas óptimas para un conjunto de vehículos que parten de un almacén o base operativa, recorren los nodos relativos a los clientes que tienen unas demandas asociadas y vuelven al punto inicial de la ruta. En este trayecto se debe tener en cuenta que el camión tiene un tiempo de autonomía limitado, por lo que antes de que se agote la batería el vehículo deberá parar en una estación o punto de carga. Esto añade decisiones adicionales respecto del VRP sobre dónde se debe de recargar la batería, cuándo se debe parar y cuánta energía se debe de recargar en cada parada.

En este trabajo se presenta una formulación matemática del EVRP con ventanas temporales y flota homogénea, en la que cada cliente debe ser atendido dentro de un intervalo de tiempo predefinido. En primer lugar, se introducirán los conjuntos, parámetros y variables empleados, que se obtuvieron de [16]:

- $N = \{1, \ldots, n\}$ es el conjunto de clientes que se deben visitar.
- $F = \{n+1, \ldots, n+p\}$ se corresponde con las estaciones de carga disponibles.
- $V = \{0\} \cup N \cup F = \{0, 1, \ldots, n, n+1, \ldots, n+p\}$ representa la totalidad de los nodos, donde el nodo 0 es el punto de inicio y fin de la ruta.
- $K = \{1, \ldots, m\}$, con $m \in \mathbb{N}, m \geq 1$, es el conjunto de vehículos disponibles, cada uno con capacidad C.
- $[a_i, b_i]$ es el intervalo temporal en el que debe ser visitado el nodo i.
- s_i es el tiempo que un vehículo tarda en completar la operación una vez se encuentra en el nodo i.
- δ_i se corresponde con la demanda del nodo i.
- w_i es el precio de cargar el camión en el punto de carga i.
- d_{ij} representa el tiempo que se tarda en ir del nodo i al nodo j.

[1] Del inglés *Electric Vehicle Routing Problem*.

- c_{ij} hace referencia al coste de ir del nodo i al nodo j.
- τ indica el tiempo de carga que se requiere por unidad de energía.
- q es la energía consumida por unidad de distancia.
- Q se corresponde con la capacidad energética de los vehículos. Suponemos que es igual para todos.
- t_i^k es una variable de decisión continua que define el tiempo de llegada del vehículo k al punto i.
- y_i^k es una variable de decisión continua que indica el nivel de energía del vehículo k en el nodo i.
- $x_{ij}^k \in \{0,1\}$ es una variable de decisión binaria que vale 1 si el arco (i,j) está en la ruta del vehículo k y 0 en otro caso.
- $z_i \in \{0,1\}$ es una variable de decisión binaria que vale 1 si la estación i es usada y 0 en otro caso.

El problema de optimización que se ha de resolver es el siguiente:

$$\text{minimizar} \quad \sum_{k\in K}\sum_{i,j\in V} c_{ij}x_{ij}^k + \sum_{i\in F} w_i z_i \tag{3.18}$$

$$\text{sujeto a} \quad \sum_{k\in K}\sum_{j\in V} x_{ij}^k = 1, \qquad \forall i \in N \tag{3.19}$$

$$\sum_{k\in K}\sum_{j\in V} x_{ij}^k \leq 1, \qquad \forall i \in F \tag{3.20}$$

$$\sum_{j\in V} x_{0j}^k = 1, \qquad \forall k \in K \tag{3.21}$$

$$\sum_{i\in V} x_{ij}^k - \sum_{h\in V} x_{jh}^k = 0, \qquad \forall j \in V \setminus \{0\}, k \in K \tag{3.22}$$

$$\sum_{i\in V} x_{i0}^k = 1, \qquad \forall k \in K \tag{3.23}$$

$$\sum_{i\in N} \delta_i \sum_{j\in V} x_{ij}^k \leq C, \qquad \forall k \in K \tag{3.24}$$

$$t_i^k + (s_i + d_{ij})x_{ij}^k - t_j^k \leq (1 - x_{ij}^k)L, \qquad \forall i,j \in N, k \in K, L \ggg 0 \tag{3.25}$$

$$t_i^k + d_{ij}x_{ij}^k + \tau(Q - y_i^k) - t_j^k \leq (L + \tau Q)(1 - x_{ij}^k), \qquad \forall i \in F, j \in N, k \in K, L \ggg 0 \tag{3.26}$$

$$(qd_{ij})x_{ij}^k - Q(1 - x_{ij}^k) \leq y_i^k - y_j^k, \qquad \forall i,j \in N, k \in K \tag{3.27}$$

$$y_i^k - y_j^k \leq (qd_{ij})x_{ij}^k + Q(1 - x_{ij}^k), \qquad \forall i,j \in N, k \in K \tag{3.28}$$

$$\sum_{i\in N, j\in F} x_{ij}^k = 1, \qquad \forall k \in K \tag{3.29}$$

$$x_{ij}^k \leq z_i, \qquad \forall i \in F, j \in N, k \in K \tag{3.30}$$

$$x_{ij}^k \leq z_j, \qquad \forall i \in N, j \in F, k \in K \tag{3.31}$$

$$a_i \leq t_i^k \leq b_i, \qquad \forall i \in N, k \in K \tag{3.32}$$

$$x_{ij}^k, z_i \in \{0,1\}, \qquad \forall i,j \in V, k \in K \tag{3.33}$$

$$y_i^k \in \mathbb{R}, \qquad \forall i \in V, k \in K \tag{3.34}$$

$$t_i^k \in \mathbb{R}^+, \qquad \forall i \in V, k \in K. \tag{3.35}$$

Se formuló un problema de minimización, donde la función objetivo (3.18) pretende minimizar el coste total, que consiste en la suma de los costes asociado a la distancia recorrida por el vehículo y los costes

asociados a la recarga de la batería. Se puede ver la explicación de las restricciones en [16], que se incluirán a continuación. La restricción (3.19) asegura que todos los clientes son visitados por algún vehículo. En la restricción (3.20) se obliga a que cada estación de carga sea visitada como máximo una vez. Las restricciones (3.21) y (3.23) garantizan que los camiones inician y finalizan el recorrido en el almacén. La restricción (3.22) establece la conservación de flujo por vehículo en cada nodo (si un vehículo llega a un nodo debe existir la correspondiente salida). La ecuación (3.24) impone la restricción de capacidad del vehículo, evitando que la suma de las demandas atendidas supere C. Las restricciones (3.25) y (3.26) establecen una secuencia temporal para las visitas a los nodos y las cargas de las baterías, usando una constante suficientemente grande, L, para desactivar las desigualdades cuando el arco no es seleccionado. Las restricciones (3.27) y (3.28) relacionan el consumo de energía por distancia con el nivel de carga de la batería antes y después de cada desplazamiento. La restricción (3.29) obliga a que cada vehículo incluya en su recorrido una parada en una estación de carga, mientras que las restricciones (3.30) y (3.31) aseguran que si no se usa una estación, no puede haber arcos incidentes. Finalmente, la restricción (3.32) asegura que las llegadas a clientes respeten sus ventanas temporales, y las restricciones (3.33), (3.34) y (3.35) definen el dominio de las variables empleadas.

Se incluye un ejemplo de un problema EVRP con vehículos eléctricos en la Figura 3.3. Esto representa la solución del problema que se planteará en la Sección 4, que encaja con el problema general descrito con un solo camión. Se observan los elementos que ya se introdujeron en los ejemplos anteriores. Sin embargo, ahora se cuenta con nodos para las estaciones de carga eléctrica (12-16) (de color verde) donde los camiones podrían parar a recargar las baterías, y un nodo correspondiente a la descarga (11) (de color morado), que es el último visitado antes de volver a la base operativa.

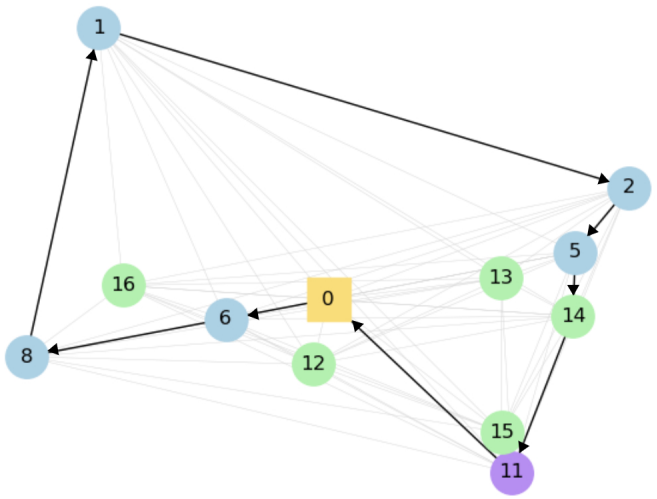


Figura 3.3: Representación EVRP.

3.4. Métodos de resolución

A la hora de abordar un problema de optimización, un aspecto fundamental es la selección de los métodos de resolución, ya que cada uno de ellos tiene diferentes propiedades en relación con la calidad de las soluciones obtenidas, que pueden ser el óptimo o aproximaciones de este, y los diferentes tiempos de ejecución obtenidos.

En el caso de los problemas de vehículos eléctricos definidos en la Sección 3.3, se trata de problemas NP-duros, por lo que esta elección se vuelve aún más importante. Existen dos grandes métodos para encontrar la solución a los problemas planteados: los métodos exactos y las heurísticas.

Los **métodos exactos** permiten encontrar una solución con la garantía de que esta es la óptima. Sin embargo, su principal inconveniente es un coste computacional muy elevado, que suele incrementarse con la dificultad del problema. Por otro lado, las **heurísticas** son procedimientos específicos diseñados para cada tipo de problema con el objetivo de encontrar soluciones buenas, pese a que no aseguran que sean óptimas. Estas técnicas se basan en la estructura del problema y, en general, son más veloces que los métodos exactos.

Además de los métodos presentados en esta sección, es posible encontrar un análisis más detallado de otras técnicas en [22], así como una descripción más profunda de cada algoritmo en las referencias específicas correspondientes.

3.4.1. Métodos exactos

Los métodos exactos se basan en formulaciones matemáticas rigurosas y en algoritmos capaces de explorar de forma exhaustiva el espacio de soluciones, lo que asegura encontrar una solución óptima. No obstante, su aplicabilidad puede verse limitada en problemas excesivamente complejos o de gran tamaño, ya que para resolverlos se emplearían tiempos excesivos. Por este motivo, los métodos exactos se emplean principalmente en problemas de pequeño tamaño o como herramienta para comparar la eficiencia de otros métodos.

La técnica más empleada para resolver problemas MIP, como el que abordaremos más adelante, es el método de ***Branch and Bound***[1] (B&B) [4]. De manera general, este algoritmo divide el espacio en regiones, proceso que se llama *ramificación*, y en cada región busca una solución factible que compara con la mejor solución obtenida hasta el momento, que constituye la *acotación*.

En la fase de ramificación, el problema original se divide en subproblemas más pequeños al imponer restricciones adicionales sobre ciertas variables. De esta manera, se genera un árbol cuyos nodos representan versiones restringidas del problema inicial. Cada subproblema se resuelve relajando las restricciones de integralidad, lo que produce un límite inferior (o superior, según el tipo de optimización) para el valor óptimo del subproblema.

La fase de acotación consiste en comparar el valor de la relajación de cada nodo con la mejor solución entera conocida hasta ese momento. Si el valor de la relajación indica que el subproblema no puede conducir a una solución mejor, dicho nodo se descarta. Este proceso, conocido como poda, reduce de forma significativa el número de subproblemas que deben explorarse.

Al inicio del proceso se definen los conjuntos de soluciones factibles y de subespacios aún no explorados y se actualizan en cada punto del algoritmo. Se parte de una solución inicial igual a ∞, y un subespacio que es el conjunto total.

[1] En castellano *ramificar y acotar*.

Existen además otros métodos exactos como el ***Branch and Cut***[1] (B&C) [5], que es una extensión del método *Branch and Bound*. En este caso, además del proceso de ramificación, se incorporan de forma iterativa *cortes* que eliminan las soluciones no enteras que aparecen al resolver la relajación lineal del problema, sin descartar ninguna solución entera factible.. Estos cortes permiten ir reduciendo progresivamente el tamaño del árbol de búsqueda y mejorando los tiempos de cómputo del algoritmo.

Otra técnica de resolución exacta ampliamente conocida es el **algoritmo de Benders** [2], que se basa en la descomposición del problema original en un problema maestro y uno o varios subproblemas. Este método resulta especialmente adecuado para problemas que presentan variables complicantes, que son problemas en los que parte importante de la complejidad de su resolución se debe a un subconjunto, típicamente pequeño, del conjunto de variables. El algoritmo resuelve de forma iterativa el problema maestro, que contiene las variables complicantes, incorporando información procedente de los subproblemas hasta converger a una solución óptima del problema original.

3.4.2. Heurísticas

Las técnicas heurísticas nacen de la necesidad de reducir los altos tiempos computacionales de los métodos exactos para la resolución de los problemas de optimización. Estas técnicas se diseñan para obtener soluciones de buena calidad y son especialmente útiles en problemas de gran dimensión o excesivamente complicados, donde el cálculo de la solución óptima resulta imposible.

Distinguiremos dos tipos de heurísticas: las constructivas y las de mejora.

- **Heurísticas constructivas**

Las heurísticas constructivas generan una solución paso a paso, construyéndola desde cero mediante la incorporación progresiva de elementos o decisiones del problema. Su objetivo es obtener rápidamente una solución inicial factible, que posteriormente puede ser utilizada como punto de partida para otros métodos de mejora.

Existen una infinidad de heurísticas constructivas. Una ampliamente conocida y empleada para resolver los VRP es **el algoritmo del vecino más próximo** [14], que se basa en construir una solución inicial seleccionando a cada paso el nodo más cercano al último nodo visitado. Este proceso se repite hasta que todos los nodos han sido visitados, construyendo una solución localmente óptima en cada iteración.

Para implementar este algoritmo se sigue el siguiente procedimiento:

1. Se elige un nodo inicial, normalmente el depósito o base operativa.
2. A partir del punto selecionado, se busca entre los nodos no visitados aquel más cercano.
3. Se añade el nodo nuevo a la ruta y se marca como visitado.
4. Se actualiza la posición del vehículo.
5. Se repiten los pasos previos hasta que ya no quedan nodos sin visitar y se finaliza la ruta.

Se puede ver un ejemplo de construcción de la ruta en la Figura 3.4, donde se observa cómo en la figura de la izaquierda, que se correspondería con la primera iteración del algoritmo, se unieron los nodos 0 y 1, por ser este uno de los más cercano, junto con 5. A continuación, en la figura de la

[1] En castellano *ramificar y cortar*.

derecha, se añadió el nodo 2, que era el más cercano al nodo 1. El algoritmo continuaría hasta haber recorrido todos los nodos o hasta satisfacer algún criterio de parada.

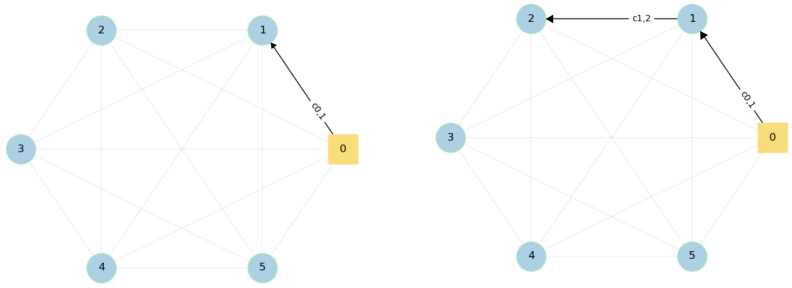


Figura 3.4: Representación de las dos primeras iteraciones del algoritmo del vecino más próximo.

- **Heurísticas de mejora**

Las heurísticas de mejora parten de una solución inicial factible, como podría ser la obtenida por el algoritmo del vecino más próximo, y buscan mejorarla mediante la aplicación de pequeñas modificaciones. Su objetivo principal es evitar los óptimos locales a favor de una solución óptima globalmente.

Entre las heurísticas de mejora más conocidas se encuentra el algoritmo ***2-opt*** [6], muy empleado para la resolución de los problemas TSP y VRP. Partiendo de una solución inical, este método consiste en seleccionar dos nodos de la ruta que formen un segmento e invertir el sentido de los arcos que conforman dicho segmento. El objetivo de este procedimiento es reducir el coste de la función objetivo de forma que se eliminen rutas ineficientes. Si este cambio mejora la solución, entonces se acepta y se incorpora a la nueva ruta. Este proceso se repite iterativamente hasta que no se pueden encontrar más mejoras.

A continuación se recogen los pasos a seguir de forma detallada:

1. Se toma una solución inicial, por ejemplo la dada por el algoritmo del vecino más próximo.
2. Se seleccionan dos nodos i y j de la ruta.
3. Se eliminan todos los arcos que unen estos dos puntos, que pueden ser uno o varios si hay nodos intermedios.
4. Se vuelven a conectar los nodos i y j de forma que se invierte su orden y todos los intermedios.
5. Se verifica si esta ruta es factible y se calcula su función objetivo.
6. Si la solución es mejor que la anterior, se acepta el cambio.
7. Se repite el proceso hasta haber comprobado todos los intercambios posibles.

Se presenta un ejemplo gráfico en la Figura 3.5, donde se observa cómo se hace un intercambio con el algoritmo *2-opt* para intentar reducir el coste total de la solución. En él se seleccionan los nodos 2

y 3 y se intercambia el arco $(2, 3)$ por el arco $(3, 2)$. Es decir, partimos de la ruta $(0, 1, \mathbf{2}, \mathbf{3}, 4, 5, 0)$ y obtenemos la ruta $(0, 1, \mathbf{3}, \mathbf{2}, 4, 5, 0)$. Se puede ver fácilmente que esta solución no mejora el coste de la ruta, por lo tanto no se aceptaría este intercambio y se seguiría con la siguiente iteración.

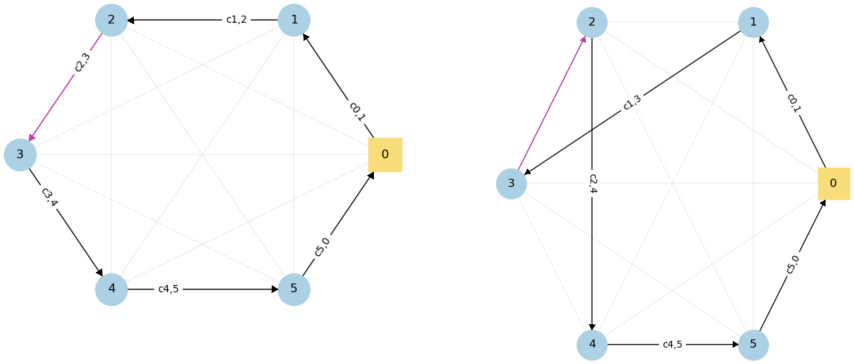


Figura 3.5: Representación de un intercambio *2-opt*.

Capítulo 4

Implementación y evaluación de resultados

Una vez establecidas las bases teóricas de los problemas de rutas de vehículos y analizada en detalle la variante con vehículos eléctricos, se está en disposición de abordar un caso real que permita poner en práctica los conceptos estudiados. En este capítulo se formula la casuística propuesta por *Trucksters*, Sección 4.1 y se hace una selección de los métodos empleados para su resolución, Sección 4.2. Asimismo, se incluye el pseudocódigo utilizado en el desarrollo de los programas implementados. Finalmente, en la Sección 4.3 se muestran y evalúan los resultados obtenidos, comparando el comportamiento del modelo exacto con los distintos algoritmos aplicados.

4.1. Formulación del caso de estudio

Para ilustrar los conceptos teóricos introducidos en los capítulos anteriores, la empresa *Trucksters* plantea un problema de rutas con vehículos eléctricos que se ajusta a las características del EVRP descrito en la Sección 3.3. El objetivo de esta sección es detallar la casuística de la empresa, desde la definición del problema de negocio hasta la formulación del mismo.

En este caso, se cuenta con un único camión que debe partir de su base operativa y regresar a ella en un tiempo máximo establecido. El cliente ha proporcionado un conjunto de puntos de recogida en los que el vehículo debe detenerse para cargar la mercancía, así como una ubicación final donde debe realizarse la descarga tras completar todas las recogidas. Como se trabaja con un vehículo eléctrico, el camión tiene una autonomía limitada, por lo que resulta necesario incluir una parada en una estación de carga a lo largo de la ruta. Existen dos empresas eléctricas cuyos cargadores tienen diferentes precios asociados. Por consiguiente, se requiere decidir el orden en el que el vehículo visitará los puntos de recogida y, además, determinar en qué estación de carga y en qué punto de la ruta se debe hacer la parada.

Bajo este contexto, el problema abordado en este Trabajo Din de Máster puede interpretarse como una instancia concreta del EVRP sin ventanas temporales, con flota homogénea de un único vehículo, múltiples puntos de recogida, un único punto de descarga final y estaciones de carga heterogéneas en coste y ubicación.

A continuación, se introducen los conjuntos, parámetros y variables necesarios para llevar a cabo la formulación del problema:

- $N = N^C \cup N^D = \{1, \ldots, n\}$ es el conjunto de nodos asociados al cliente.

  - $N^C = \{1, \ldots, n-1\}$ hace referencia a los nodos de recogida.

  - $N^D = \{n\}$ hace referencia al nodo de descarga.

- $F = \{n+1, \ldots, n+p\}$ se corresponde con las estaciones de carga disponibles.

- $V = \{0\} \cup \{0^*\} \cup N \cup F = \{0, 0^*, 1, \ldots, n-1, n, n+1, \ldots, n+p\}$ representa el conjunto de nodos, donde el nodo 0 es la base operativa y 0^*[1] representa una base operativa ficticia.

- c_{ij} hace referencia al coste de ir del nodo i al nodo j. Este coste se calculará como el producto de los kilómetros recorridos y el precio del kilómetro.

- w_i es el precio de cargar el camión en el punto de carga i.

- d_{ij} representa el tiempo que se tarda en ir del nodo i al nodo j.

- t_i es una variable de decisión continua que define el tiempo de llegada del vehículo al punto i.

- $x_{ij} \in \{0, 1\}$ es una variable de decisión binaria que vale 1 si el arco (i, j) está en la ruta y 0 en otro caso.

- $z_i \in \{0, 1\}$ es una variable de decisión binaria que vale 1 si la estación i es usada y 0 en otro caso.

Desde la perspectiva de negocio, el objetivo general conlleva tanto la reducción de costes fijos, tales como el alquiler del camión, el salario del conductor o los seguros, así como de costes variables, asociados directamente a la ruta y a las operaciones de recarga. Dado que los costes fijos no dependen de la planificación concreta, el objetivo del problema consiste en minimizar los costes variables, definidos como la suma del coste derivado de los kilómetros recorridos y el coste asociado a la recarga del vehículo eléctrico.

La formulación del EVRP adaptada a las necesidades de *Trucksters* es:

[1] Este nodo es necesario para la programación del problema, ya que evitamos la formación de ciclos en la solución.

$$\text{minimizar} \quad \sum_{i,j\in V} c_{ij}x_{ij} + \sum_{i\in F} w_i z_i \tag{4.1}$$

$$\text{sujeto a} \quad \sum_{j\in V} x_{ij} = 1, \qquad \forall i \in V\backslash\{F \cup \{0^*\}\} \tag{4.2}$$

$$\sum_{i\in V} x_{ij} - \sum_{h\in V} x_{jh} = 0, \qquad \forall j \in V \setminus \{\{0\} \cup \{0^*\}\} \tag{4.3}$$

$$\sum_{j\in V\backslash\{\{0\}\cup\{0^*\}\}} x_{0j} = 1 \tag{4.4}$$

$$\sum_{i\in V\backslash\{\{0\}\cup\{0^*\}\}} x_{i0^*} = 1 \tag{4.5}$$

$$x_{0^*j} = 0, \qquad \forall j \in V \tag{4.6}$$

$$t_j \geq t_i - (1 - \sum_{h\in V} x_{hi})L \qquad \forall i \in N^C, j \in N^D, L \ggg 0 \tag{4.7}$$

$$t_i + d_{ij}x_{ij} - t_j \leq (1 - x_{ij})L, \qquad \forall i, j \in V,\ L \ggg 0 \tag{4.8}$$

$$\sum_{i\in N\cup\{0\}, j\in F} x_{ij} = 1 \tag{4.9}$$

$$x_{ij} \leq z_i, \qquad \forall i \in F, j \in N \tag{4.10}$$

$$x_{ij} \leq z_j, \qquad \forall i \in N, j \in F \tag{4.11}$$

$$x_{ij}, z_i \in \{0, 1\}, \qquad \forall i, j \in V \tag{4.12}$$

$$t_i \in \mathbb{R}^+, \qquad \forall i \in V. \tag{4.13}$$

Se ha planteado un problema de minimización, donde la función objetivo, (4.1), recoge las variables asociadas a los costes. Como se dijo anteriormente, el propósito del problema es minimizar los costes que conlleva la distancia de la ruta y los costes asociados a la batería eléctrica. Nótese que ambos sumandos de esta función se miden en euros, pudiendo contabilizar su valor de manera sencilla.

Se verá a continuación qué representan las restricciones planteadas. La primera, (4.2), nos asegura que el camión saldrá de todos los puntos de recogida, el punto de descarga y la base operativa, exactamente una vez. En la restricción (4.3) se impone que si existe un arco de llegada a un nodo, también tiene que haber un arco de salida, con excepción del punto de partida y de llegada, que es la base operativa ficticia. La restricción (4.4) obliga al vehículo a partir de la base operativa y la (4.5) obliga a volver a la base operativa ficticia, que en el problema real serían el mismo punto. La siguiente restricción, la ec. (4.6), evita que el vehículo, una vez llegue a la base operativa ficticia, siga con la ruta. Por otro lado, la restricción (4.7) impone que el nodo de descarga solo se visite una vez que el camión haya pasado por todos los puntos de recogida. La restricción (4.8) hace referencia a la ordenación temporal de las visitas a los distintos puntos de recogida. La necesidad de parar en un punto de carga aparece recogida en la restricción (4.9). Las restricciones (4.10) y ec. (4.11) nos dicen que si un cargador no es usado, no existirá un arco que vaya hasta él. Por último, la restricción (4.12) nos indica que dichas variables son binarias.

Se puede ver cómo esta formulación es una adaptación del modelo general de la Sección 3.3 a este caso concreto. Se ha eliminado la restricción relativa a la capacidad del vehículo (restricción (3.24) del modelo general), ya que se asume que el camión tiene capacidad suficiente para atender a todas las demandas. De forma similar, al tratarse de instancias pequeñas, se ha prescindido de las variables relativas a la energía consumida (q) y la capacidad energética (Q), así como de sus restricciones asociadas (restricciones (3.26), (3.27) y (3.28)). La ruta se realiza con un solo camión, por lo que se han eliminado los índices correspondientes a los vehículos en las variables y restricciones, así como las

referencias a las ventanas temporales (restricción (3.32)), ya que no es relevante para la resolución. Por último, se han añadido nuevas restricciones, que son las ecuaciones (4.6) y (4.7), que no estaban presentes en el modelo general y el nodo ficticio 0^*.

4.2. Técnicas seleccionadas

Una vez planteado el problema que se va a abordar y explicitado su formulación, es necesario seleccionar qué métodos se emplearán para su resolución. Esta selección viene motivada tanto por el objetivo de negocio de *Trucksters*, que consiste en automatizar el proceso manual de planificación del camión eléctrico, reduciendo los tiempos de planificación actuales y proporcionando rutas factibles a un coste operativo competitivo, como por el objetivo técnico de este Trabajo Fin de Máster, que es analizar y determinar qué algoritmo de optimización resulta más adecuado para la casuística concreta de la empresa.

Con el fin de dar respuesta a ambos objetivos, y apoyándose en las técnicas descritas en la Sección 3.4, se seleccionan tres métodos de resolución con características complementarias. En primer lugar, se considera un método exacto, implementado mediante el *solver* SCIP, que permite obtener soluciones óptimas o de referencia para el problema planteado. En segundo lugar, se emplea una heurística constructiva, concretamente el algoritmo del vecino más próximo, caracterizada por su simplicidad y bajo tiempo de ejecución. Por último, se aplica una heurística de mejora, el algoritmo *2-opt*, cuyo objetivo es refinar soluciones iniciales y mejorar su calidad sin incurrir en elevados costes computacionales.

El objetivo de esta sección es comparar estos tres métodos para seleccionar el más adecuado para el caso particular que plantea *Trucksters*. Esta comparación se realizará atendiendo a dos criterios fundamentales: la calidad de la solución, en términos de costes de solución, y el tiempo de ejecución, directamente relacionado con la viabilidad de sustituir el proceso manual actual por un procedimiento automático.

4.2.1. Método exacto

En primer lugar, se va a resolver el problema utilizando la formulación matemática planteada. Este enfoque permite obtener soluciones óptimas, lo que lo convierte en una referencia de calidad frente a los resultados obtenidos mediante métodos aproximados. Aunque su coste computacional puede ser elevado para instancias de gran tamaño, resulta adecuado para el problema abordado, proporcionando así una referencia de optimalidad.

Existen diversos *solvers* comerciales y de código abierto capaces de resolver variantes del EVRP mediante métodos exactos, entre los que destacan Gurobi, CPLEX y SCIP. En este trabajo se emplea el *solver* SCIP, un optimizador de código abierto ampliamente utilizado en programación lineal entera mixta.

La implementación del modelo se realiza utilizando la librería OR-Tools [24], desarrollada por Google, que proporciona un entorno de modelado de alto nivel y permite la integración de distintos solvers externos. En concreto, OR-Tools se utiliza como herramienta de formulación del modelo, mientras que la resolución se delega en el *solver* SCIP.

El modelo se codifica siguiendo las variables de decisión, parámetros, restricciones y función objetivo definidas en la Sección 4.1. Adicionalmente, el modelo incorpora dos hiperparámetros: un valor grande L y un tiempo máximo de ejecución. En este trabajo se fija $L = 10^5$, considerado suficientemente grande para garantizar la validez de las restricciones de tipo big-M (en el problmema, las restricciones 4.7 y

4.8). No obstante, este valor podría ajustarse de forma más precisa a partir de los límites superiores de las variables del modelo, lo que se plantea como una línea futura de trabajo.

En cuanto al tiempo máximo de ejecución, este se adapta al tamaño de las instancias consideradas y su impacto en la calidad de las soluciones se analiza en la Sección 4.3. Esta parametrización permite equilibrar la obtención de soluciones de alta calidad con la viabilidad computacional del enfoque exacto. En este caso se tomará como tiempo máximo un valor de 5 minutos.

4.2.2. Algoritmo del vecino más próximo

Se ha seleccionado como heurística constructiva el algoritmo del vecino más próximo, por ser un algoritmo sencillo de implementar y que da lugar a soluciones localmente óptimas en cada iteración. Además, su coste computacional es bajo en comparación con otras heurísticas, lo que lo convierte en un método interesante para comparar tanto las soluciones obtenidas como los tiempos de ejecución.

Este método se define originalmente para el TSP y cuenta con múltiples adaptaciones estándar para el VRP. En la literatura existen también extensiones de esta heurística orientadas a resolver variantes del EVRP.

En este Trabajo Fin de Máster se propone una adaptación específica de la heurística del vecino más próximo para resolver el problema real planteado. El objetivo principal de esta heurística es la obtención de soluciones factibles en tiempos de ejecución reducidos, por lo que, durante su adaptación, se prioriza explícitamente la eficiencia computacional frente a la calidad óptima de la solución.

Una de las particularidades del caso de uso considerado radica en la existencia de una estructura de precedencia en las visitas, ya que es obligatorio visitar previamente todos los puntos de carga antes de acceder al punto de descarga. Además, el problema presenta un único punto de descarga y un conjunto significativamente mayor de puntos de carga.

Dado que el mayor margen de optimización se encuentra en el orden de visita de los puntos de carga, la heurística propuesta se estructura en dos fases. En una primera fase se construye una ruta que parte de la base operativa y conecta todos los nodos de carga siguiendo un criterio optimalidad local. En una segunda fase, una vez completada la secuencia de cargas, se añade el punto de descarga.

Adicionalmente, en este caso de uso es suficiente con visitar un único punto de recarga en cualquier punto de la ruta. Por este motivo, la heurística finaliza insertando un único cargador en la posición que minimiza el incremento de distancia total de la ruta construida.

Cabe destacar que el algoritmo propuesto no garantiza, en términos generales, la obtención de una solución factible, ya que, una vez finalizado el proceso de enrutamiento, la ruta resultante podría exceder el límite máximo de distancia permitido.

Para evitar esta situación, sería necesario incorporar un test de factibilidad durante la construcción de la ruta, evaluando las restricciones en cada inserción de nodo, lo que conllevaría un incremento de los tiempos de ejecución. No obstante, tras la validación experimental del algoritmo sobre el conjunto de datos utilizado, se ha observado que las soluciones generadas cumplen las restricciones del problema.

En el contexto del estudio comparativo realizado, este algoritmo se plantea explícitamente como una solución de referencia orientada a la rapidez de ejecución, asumiendo que la calidad de las soluciones obtenidas puede ser inferior a la de métodos más complejos. Por este motivo, y con el objetivo de no incrementar la complejidad del algoritmo ni penalizar los tiempos de ejecución, no se ha incorporado dicho mecanismo de comprobación.

Para implementar el algoritmo, se definen previamente las listas de puntos de recogida no visitados

(*loadings*), puntos de descarga no visitados (*unloadings*) y cargadores (*chargers*). A mayores, a la base operativa la llamaremos *parking* y denotaremos como `current_loc` la posición del vehículo en cada momento. Se puede ver el pseudocódigo empleado para la resolución en el Algoritmo 1.

Algoritmo 1 Algoritmo del vecino más próximo para el EVRP

```
Inicializar solución vacía
Marcar todos los loadings y unloadings como no visitados
while existan nodos por visitar do
    Iniciar nueva ruta desde parking
    current_loc ← parking
                                                          ▷ Loadings
    while existan loadings no visitados do
        Elegir el loading más cercano a current_loc
        Añadirlo a la ruta
        Marcarlo como visitado
        current_loc ← loading elegido
    end while
                                                          ▷ Unloadings
    while existan unloadings no visitados do
        Elegir el unloading más cercano a current_loc
        Añadirlo a la ruta
        Marcarlo como visitado
        current_loc ← unloading elegido
    end while
                                                          ▷ Insertar charger
    if hay chargers disponibles then
        Elegir el más cercano al último nodo
        Buscar posición que minimice aumento de distancia
        Insertar charger
    end if
    Terminar ruta en base operativa
    Añadir ruta a la solución
end while
Verificar restricciones
return solución
```

4.2.3. Algoritmo *2-opt*

Como heurística de mejora se ha seleccionado el algoritmo *2-opt*, partiendo de la solución inicial obtenida con el algoritmo del vecino más próximo. Este algoritmo resulta apropiado ya que da lugar a mejoras en todas las soluciones, gracias al intercambio de aristas que reducen los costes de la ruta. Veremos que tiene un coste computacional mayor que el algoritmo de construcción, pero da mejores resultados también. Esta heurística combina tiempos bajos de ejecución y soluciones competitivas.

El funcionamiento del algoritmo se basa en analizar, dentro de cada ruta, todos los posibles pares de posiciones (i, j) y evaluar si invertir el subtrayecto comprendido entre ellas mejora la distancia total. Este procedimiento se repite de manera iterativa hasta que no es posible encontrar ninguna solución que mejore la inicial.

Previamente a realizar la inversión del segmento, se comprueba si se respetan las restricciones, es decir:

- Los *loadings* deben aparecer antes que los *unloadings* en la ruta.
- En cada ruta solo puede haber un *charger*.
- El camión debe salir y retornar al *parking*.
- La distancia total recorrida es inferior a la distancia máxima fijada.

Durante la ejecución, el algoritmo examina el segmento de la ruta entre los nodos i y j. Antes de realizar la inversión, se comprueba que dicha operación no viola ninguna de las restricciones anteriores. Si el segmento es válido, se invierte y se calcula la nueva distancia total de la ruta. Si la inversión representa una mejora, pasa a considerarse la mejor ruta encontrada hasta ese momento y se reinicia el proceso de búsqueda para seguir explorando posibles mejoras.

Este procedimiento continúa hasta que se recorre la ruta completa sin encontrar ninguna inversión que reduzca la distancia. Tras la comprobación de que el nuevo recorrido cumple todas las restricciones impuestas, se añade a la nueva solución optimizada. El proceso se repite de manera independiente para cada nueva ruta de la solución inicial, hasta que ya no se consiguen mejoras y por tanto no hay nuevas rutas.

El pseudocódigo del algoritmo se muestra en el Algoritmo 2.

Algoritmo 2 Algoritmo *2-opt* para el EVRP

```
Inicializar nueva solución vacía
for each ruta en la solución inicial do
    Copiar la ruta como best_route
    mejorar ← True
    while mejorar do
        mejorar ← False
        for i ← 1 until penúltimo nodo antes del final do
            for j ← i + 1 until penúltimo nodo do
                Seleccionar segmento entre i y j                ▷ Respetar restricciones
                if mezcla loadings y unloadings then
                    continue
                end if
                if más de un charger then
                    continue
                end if
                Invertir segmento para obtener new_route
                Calcular distancia de new_route
                if new_route mejora then
                    best_route ← new_route
                    mejorar ← True
                    break                                       ▷ Reiniciar búsqueda
                end if
            end for
        end for
    end while
    Agregar best_route a nueva solución
end for
Verificar factibilidad
return solución optimizada
```

4.2.4. Lenguaje de programación y herramientas

La implementación de los modelos y algoritmos propuestos se ha realizado en Python 3.11, elegido por su claridad, facilidad de prototipado y amplia disponibilidad de librerías especializadas. Además, se trata del *software* de referencia en *Trucksters*. Se ha seguido una arquitectura orientada a objetos para representar los elementos del problema de optimización, como nodos de clientes, cargas y descargas, así como los algoritmos empleados, lo que facilita la extensión y mantenimiento del código.

Se han utilizado diversas librerías según su función: OR-Tools [13] para la interacción con el *solver* SCIP; NumPy [21] para el cálculo y la implementación de heurísticas; pandas [15] para la generación del reporte comparativo; y Folium [10] para la visualización interactiva de rutas. Python permite así integrar de manera eficiente la resolución del modelo exacto, la implementación de heurísticas, el análisis de datos y la generación de visualizaciones o reportes, garantizando resultados reproducibles y un desarrollo flexible del proyecto. El código completo, de elaboración propia, se puede encontrar en [20], un repositorio de *Github* creado con la finalidad de poder reproducir los resultados presentados en este Trabajo Fin de Máster.

4.2.5. Criterios de comparación

Para determinar cuál de los algoritmos seleccionados resulta más adecuado para el caso de *Trucksters*, se establecen criterios de comparación que reflejan tanto el desempeño operativo como la viabilidad técnica de su implementación.

En primer lugar, se evaluará la calidad de la solución, medida a través de la función objetivo del problema, que combina la distancia recorrida, los costes de recarga y cualquier otro coste operativo relevante. Minimizar esta función objetivo permitirá identificar la solución que resulta más eficiente desde el punto de vista de la planificación de rutas y de los costes asociados.

En segundo lugar, se considerarán los tiempos de computación, que deben ser inferiores a 5 minutos por planificación. Superar este umbral incrementaría de manera significativa la complejidad de la infraestructura técnica necesaria para ejecutar el algoritmo de manera diaria, reduciendo su viabilidad operativa.

Por último, se evaluará la escalabilidad de los algoritmos, analizando su rendimiento frente a conjuntos de datos aumentados que incluyen un mayor número de nodos. Este criterio permitirá determinar la capacidad de los métodos para mantener su eficiencia y calidad de solución en situaciones más complejas, anticipando cómo se comportarían en posibles ampliaciones de la flota o incrementos en los puntos de recogida.

La combinación de estos criterios garantiza que el algoritmo seleccionado no solo proporcione soluciones de calidad, sino que también sea robusto, práctico y sostenible, cumpliendo con los objetivos de negocio y técnicos definidos en este trabajo.

4.2.6. Datos empleados

Para poder implementar los códigos y comparar los distintos métodos, *Trucksters* ha facilitado un *dataset* extraído del proyecto que mantiene con su cliente. Estos datos constan de una serie de puntos, que intervienen en la planificación diaria del camión eléctrico y que serán los nodos del problema, cada uno de ellos con su nombre, dirección, coordenadas y tipología operativa. En total disponemos de 17 nodos:

- 1 nodo correspondiente a la base operativa.
- 1 nodo de descarga.
- 5 nodos que representan estaciones de recarga.
- 10 nodos que actúan como posibles puntos de recogida.

Se puede ver cómo se distribuyen los 17 nodos en la Figura 4.1. Aparece representado en naranja el relativo a la base operativa (0), en morado el nodo de descarga (11), en verde las estaciones de recarga (11-16) y en azul los puntos de recogida (1-10). Se podría pensar que la elección del cargador es trivial, dado que el nodo 15 se sitúa muy cerca del nodo de descarga. Sin embargo, se corresponde con la empresa cuyos cargadores son más costosos, por lo que esta decisión no es inmediata.

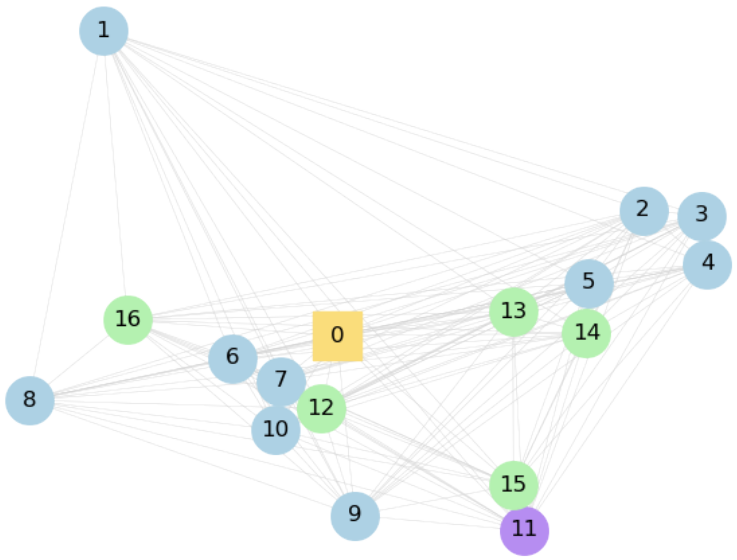


Figura 4.1: Representación del problema propuesto por *Trucksters*.

En la operativa diaria, el equipo de planificación de *Trucksters* recibe el día previo a la entrega los puntos de carga que debe visitar. Normalmente, el cliente facilita entre 3 y 6 de los 10 posibles puntos de recogida, aunque en algunas ocasiones podría requerirse la visita a todos los puntos de carga. Para evaluar los algoritmos frente a distintos escenarios, se generaron aleatoriamente instancias de prueba con tamaños comprendidos entre 3 y 9 puntos de reocogida, es decir, instancias con 3, 4, 5, 6, 7, 8 y 9 nodos relativos a las recogidas. Para cada tamaño se crearon 10 configuraciones distintas seleccionando los puntos de carga de manera aleatoria, dando lugar a un total de 70 instancias representativas de posibles planificaciones. Así, se puede analizar cómo se comportan los algoritmos bajo la variabilidad típica del proyecto real.

Para el cálculo de las distancias entre nodos se utilizó la distancia geodésica, adecuada para trabajar con coordenadas y estimar con precisión la distancia recorrida por el vehículo. Estas distancias se obtuvieron mediante la librería geopy [11], concretamente empleando la función `geodesic`.

Para los puntos de recarga de la batería se consideraron cargadores pertenecientes a dos empresas distintas: Ionity y Atlante. Ambas disponen de una tarifa fija por recarga completa y, dado que el camión siempre repone batería hasta alcanzar el 100 % de su capacidad, se incorporó al modelo un

coste de carga constante para los puntos de carga de cada empresa. En concreto, los cargadores de Ionity tienen un precio de 0.61 € por kWh, mientras que los de Atlante presentan una tarifa de 0.64 € por kWh. Se asumió una capacidad total de batería de 300 kWh, por lo que los costes asociados a una recarga completa son de 183 € y 192 €, respectivamente.

También fue necesario estimar el tiempo de desplazamiento del camión entre los distintos puntos, con el objetivo de construir una secuencia temporal a lo largo de la ruta. Para ello, se utilizaron las distancias geodésicas previamente calculadas y se asumió que el vehículo circula a una velocidad media de 70 km/h.

Asimismo, para poder expresar la solución en una única unidad de medida, se asumió un coste por kilómetro de 0.623 €, valor que permite traducir la distancia recorrida directamente en un coste económico. Así, la función objetivo recogerá el coste de traslado del camión y el coste de la carga de la batería, todo ello medido en €.

4.3. Comparativa

En esta sección se presenta una comparativa entre los distintos métodos implementados, evaluando tanto sus tiempos de ejecución como la calidad de las soluciones obtenidas. El objetivo es analizar el comportamiento de los algoritmos frente a escenarios de distinta complejidad y establecer conclusiones sobre su rendimiento relativo en condiciones controladas. Para ello, se han empleado las 70 instancias de prueba descritas en el apartado anterior.

Es importante destacar que el equipo utilizado para la evaluación es un MacBook Air (13-inch, 2017) equipado con un procesador Intel Core i5 de doble núcleo a 1.8 GHz (4 hilos lógicos), 8 GB de memoria RAM DDR3 a 1600 MHz y arquitectura x64. Los mismos programas ejecutados en un ordenador diferente podría cambiar los tiempos obtenidos.

En la Tabla 4.1 aparecen recogidos, para cada tamaño de $|N|$, el valor medio de la función objetivo y del tiempo medio que el programa tardó en generar cada solución, todo esto para cada uno de los tres métodos. Esta tabla recoge un resumen de los datos, además, se pueden ver la totalidad de los resultados en el Apéndice A. En un primer momento se observa cómo, a medida que aumentamos el tamaño del conjunto de nodos explorados, el valor de la función objetivo también aumenta. Esto tiene sentido pues a mayor número de puntos que visitar, mayor es también el coste total de la operación. Sin embargo, esto no ocurre siempre, ya que al escoger los nodos de recogida de manera aleatoria, pueden ser seleccionados nodos más alejados y por tanto aumentar la distancia entre ellos.

A mayores de la tabla resumen, se incluyen boxplots relativos al tiempo de ejecución, coste de las soluciones y diferencias porcentuales de las soluciones respecto al método exacto, que permiten visualizar el comportamiento para cada tamaño de instancia. Estos gráficos los podemos encontrar en las Figuras 4.3, 4.4 y 4.5.

	Modelo exacto		**Vecino más próximo**		***2-opt***	
$\|N\|$	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)
3	246.182	$2.9693 \cdot 10^2$	262.030	$8.4 \cdot 10^{-5}$	255.026	$1.87 \cdot 10^{-4}$
4	251.099	$3.0087 \cdot 10^2$	268.585	$9.8 \cdot 10^{-5}$	259.914	$2.82 \cdot 10^{-4}$
5	256.379	$3.0087 \cdot 10^2$	272.948	$1.05 \cdot 10^{-4}$	265.305	$4.58 \cdot 10^{-4}$
6	266.427	$3.0143 \cdot 10^2$	283.021	$1.91 \cdot 10^{-4}$	275.313	$5.49 \cdot 10^{-4}$
7	268.065	$3.0946 \cdot 10^2$	287.399	$1.74 \cdot 10^{-4}$	276.814	$1.054 \cdot 10^{-3}$
8	274.940	$3.0149 \cdot 10^2$	299.886	$3.19 \cdot 10^{-4}$	283.514	$1.269 \cdot 10^{-3}$
9	280.855	$3.0204 \cdot 10^2$	306.094	$1.44 \cdot 10^{-4}$	288.430	$1.453 \cdot 10^{-3}$

Tabla 4.1: Comparación de los tres métodos. Se recogen los valores medios de las funciones objetivo y los tiempos medios por cada tamaño de instancia a lo largo de las 10 ejecuciones.

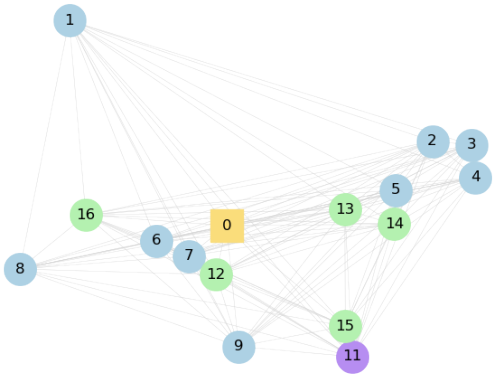
El modelo exacto se ejecutó fijando un tiempo máximo de cinco minutos, considerado razonable para instancias pequeñas. Cabe destacar que, para garantizar la obtención de la solución óptima, el modelo debería ejecutarse sin límite de tiempo. Se puede observar cómo este método emplea el tiempo máximo establecido (Figura 4.3a).

En este experimento, el modelo exacto logra las mejores soluciones en la totalidad de los casos. Para los tamaños de problema más pequeños, la heurística *2-opt* presenta desviaciones en torno al 3.5 % respecto al *solver*, mientras que la del vecino más próximo se sitúa aproximadamente en el 7 %. Al incrementar el número de nodos, el comportamiento de ambas heurísticas varía: el vecino más próximo incrementa progresivamente su diferencia hasta valores cercanos al 10 %, mientras que *2-opt* tiende a reducirla, alcanzando en algunos casos desviaciones próximas al 1 %. Se pueden observar estas variaciones en la Figura 4.5.

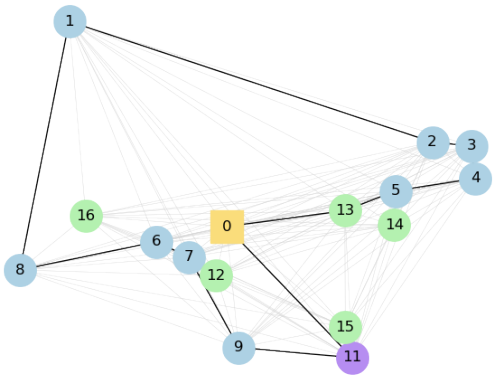
El algoritmo del vecino más próximo es una heurística que busca conseguir soluciones de forma veloz, de modo que reduce significativamente el tiempo de cómputo frente a otros métodos, aunque no garantiza una buena calidad de la solución obtenida. Se puede observar en la Tabla 4.1 y en la Figura 4.3b cómo el algoritmo del vecino más próximo tarda en encontrar una solución tiempos extremadamente bajos, de entre 0.000084 y 0.000319 segundos de media. Sin embargo, analizando la calidad de las soluciones, estas presentan diferencias notables respecto a las obtenidas con el método exacto. En la Figura 4.5a se aprecia que la solución obtenida es de entre un 4 % y un 10 % superior a la obtenida con el *solver* SCIP, siendo esta diferencia mayor para las instancias grandes.

Por último, el algoritmo *2-opt* mejora las soluciones obtenidas por la heurística del vecino más próximo, ya que toma sus costes como solución inicial, aunque sin llegar a alcanzar las soluciones óptimas. Aún así, se puede observar que, en las instancias de 9 nodos, existen casos donde esta técnica se queda muy cerca del óptimo calculado, como se comentaba anteriormente. Esto se puede ver gráficamente en la Figura 4.5b, donde se observan diferencias porcentuales muy pequeñas en las mayores instancias. En terminos de tiempo de ejecución sorprende que *2-opt* presenta tiempos muy pequeños, superiores solo por milisegundos a los obtenidos por la heurística del vecino más próximo.

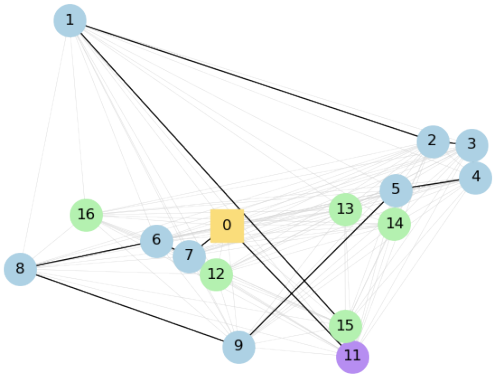
En la Figura 4.2 se muestra un ejemplo de las soluciones obtenidas mediante los tres métodos analizados, utilizando como referencia una instancia que contiene 9 puntos de recogida. La Figura 4.2a representa el conjunto completo de nodos considerados en el problema. En amarillo aparece plasmada la base operativa (0), en azul los nodos relativos a los puntos de recogida (1-9), en verde las estaciones de carga (12-16) y en morado el nodo de descarga (11). En la Figura 4.2b se presenta la solución óptima obtenida por SCIP, cuya estructura es coherente y se corresponde con la solución óptima de la ruta. Dicha solución tiene un coste de 294.04 € y se incluye la visita a un cargador de Ionity, con un precio de carga de 183 €. La Figura 4.2c recoge la solución generada por el algoritmo del vecino más próximo, se observa como el recorrido marcado obliga al camión a recorrer una mayor distancia, haciendo rutas entre nodos absurdas. Esta solución tiene un coste de 312 € y se visita un cargador de la empresa Atlante, con coste 192 €. Finalmente, la Figura 4.2d muestra la solución mejorada mediante el algoritmo *2-opt*, que difiere de la solución óptima solo en el punto de carga, que sigue siendo de Ionity, y obteniendo así una ruta más razonable que la proporcionada por el vecino más próximo, con un coste de 292.73 €.



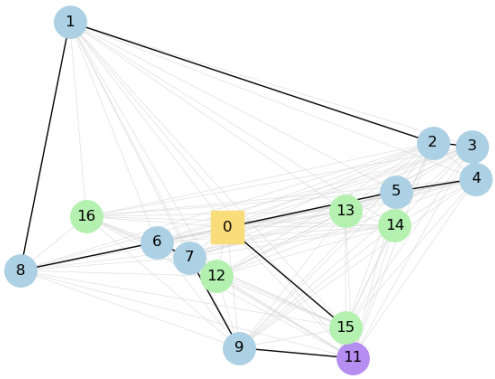
(a) Nodos del conjunto.



(b) Solución exacta:
(0, 13, 5, 4, 3, 2, 1, 8, 6, 7, 9, 11, 0)



(c) Solución del vecino más próximo:
(0, 7, 6, 8, 9, 5, 4, 3, 2, 1, 15, 11, 0)

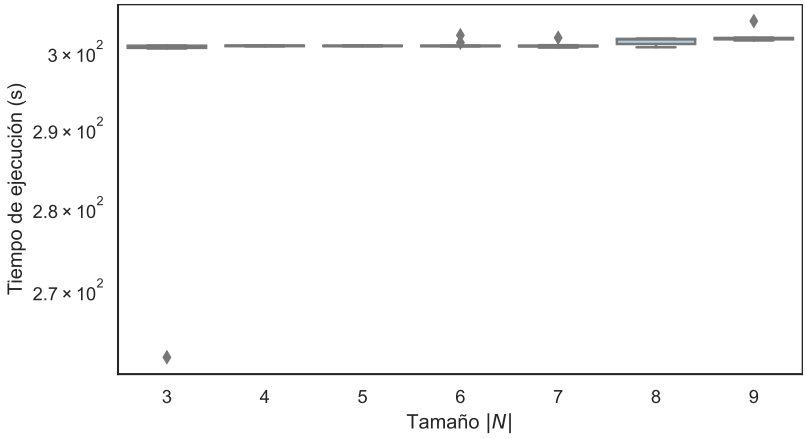


(d) Solución de *2-opt*:
(0, 5, 4, 3, 2, 1, 8, 6, 7, 9, 11, 15, 0)

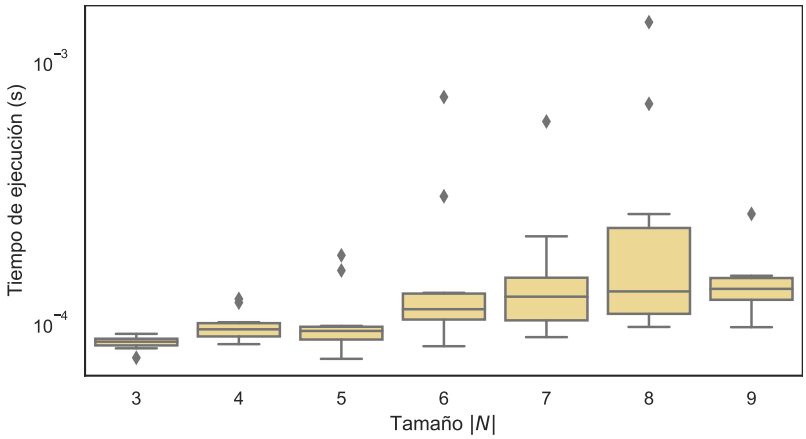
Figura 4.2: Representación de las soluciones de cada método para una instancia de 9 puntos de recogida.

En conclusión, considerando los criterios establecidos para la selección del algoritmo más adecuado, la heurística *2-opt* representa la mejor alternativa para la operativa de *Trucksters*. En términos de

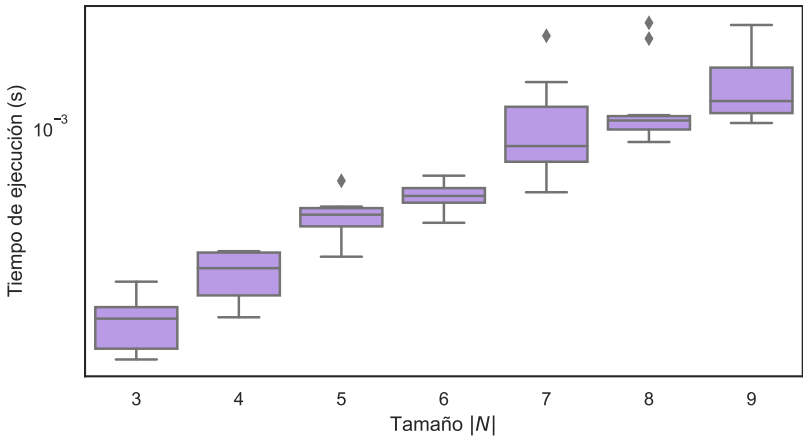
calidad de la solución, *2-opt* mejora significativamente la solución inicial generada por el vecino más próximo, alcanzando resultados cercanos al óptimo. Respecto a los tiempos de ejecución, este algoritmo mantiene valores muy bajos, prácticamente despreciables, lo que garantiza que la planificación automática no exceda los 5 minutos, cumpliendo con el límite operativo fijado por la empresa y asegurando la viabilidad diaria del proceso. Por lo tanto, el algoritmo *2-opt* ofrece un equilibrio óptimo que satisface tanto el objetivo de negocio de automatización eficiente como el objetivo técnico de determinar el método más adecuado para la casuística concreta de *Trucksters*.



(a) Modelo exacto.

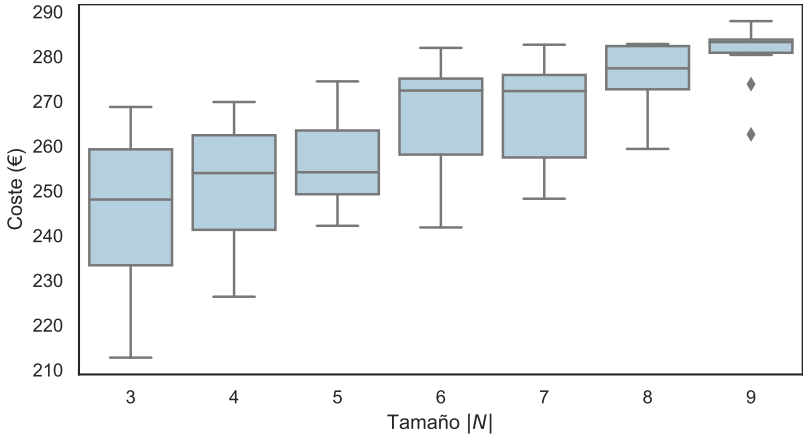


(b) Heurística del vecino más próximo.

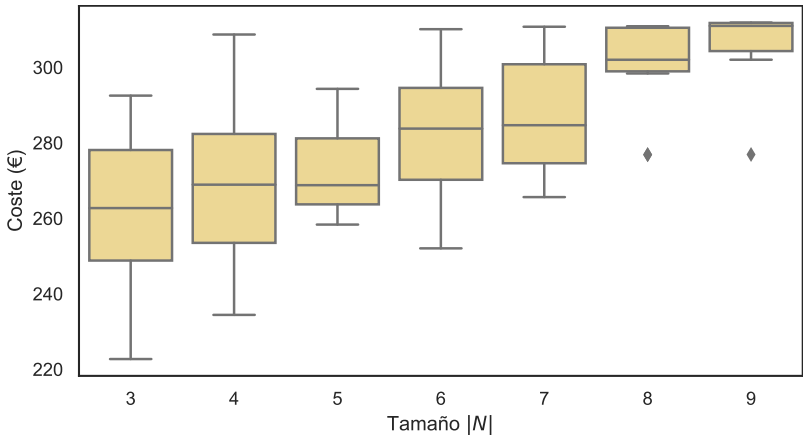


(c) Heurística *2-opt*.

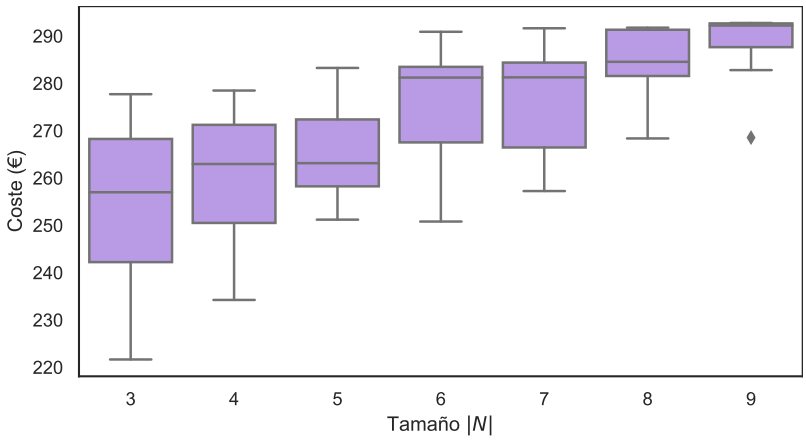
Figura 4.3: Boxplots de los tiempos de ejecución para cada método.



(a) Modelo exacto.

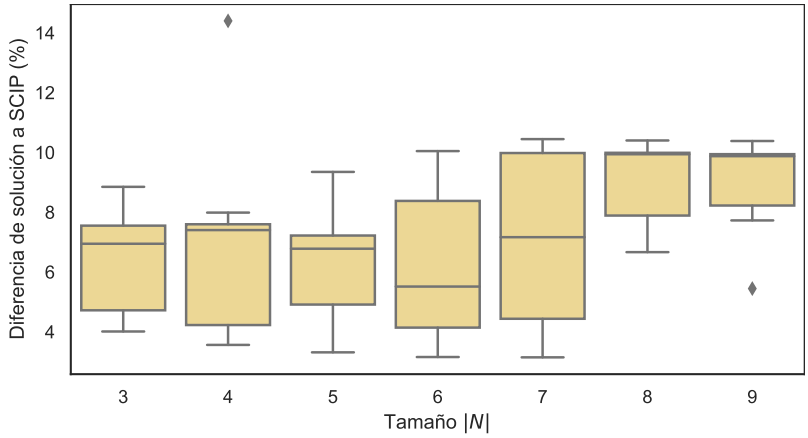


(b) Heurística del vecino más próximo.

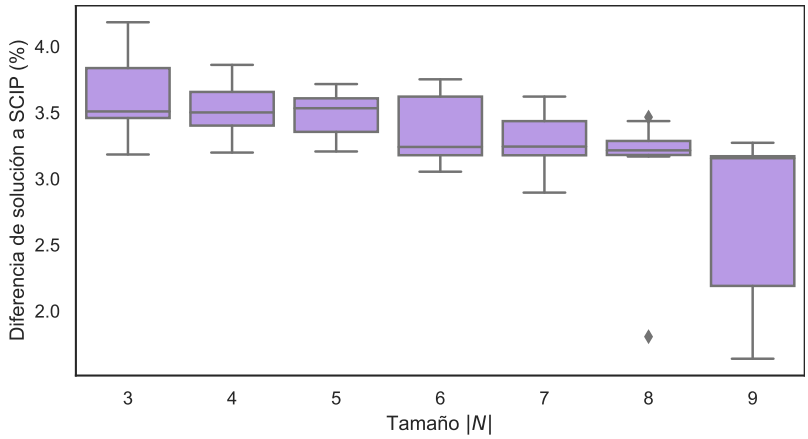


(c) Heurística *2-opt*.

Figura 4.4: Boxplots de los costes asociados a la solución.



(a) Heurística del vecino más próximo.



(b) Heurística *2-opt*.

Figura 4.5: Boxplots de las diferencias porcentuales de cada heurística respecto al modelo exacto.

Capítulo 5

Estudio con datasets aumentados

En el Capítulo 4 se ha presentado una comparativa entre los distintos métodos implementados, utilizando como base de datos la proporcionada por *Trucksters*, que se corresponde con una casuística real. Sin embargo, el conjunto de datos es limitado, con solo 10 puntos de recogida disponibles, lo que supone una restricción para evaluar el comportamiento de los algoritmos en problemas de gran escala.

En este capítulo se presenta una generalización del problema real, considerando un escenario con múltiples vehículos eléctricos y un mayor número de nodos de recogida y descarga. Esta extensión surge de la planificación estratégica de *Trucksters*, que está evaluando la adquisición de un segundo vehículo eléctrico para poder atender la posibilidad de que el cliente amplíe los puntos de recogida en futuras operaciones. De este modo, el trabajo no solo aborda la casuística real actual, sino que también proporciona información relevante para la planificación y escalabilidad futura de la operativa eléctrica de la empresa.

Además, dado que la distancia máxima que puede recorrer un vehículo depende de su tipo y autonomía, el estudio también contempla evaluar el rendimiento de los algoritmos frente a rutas de distintas longitudes, garantizando que los modelos sean robustos y aplicables a escenarios con vehículos de diferentes capacidades.

Se generarán instancias más grandes a partir de datos reales mantienendo las principales características del caso real planteado anteriormente. Así, se garatiza el análisis de escenarios plausibles y útiles para la planificación de rutas.

Se presenta en la Sección 5.1 una nueva formulación del problema, que se adapta a las nuevas características del mismo. Además, en la Sección 5.2 se describen las técnicas empleadas para ajustar los métodos de resolución empleados con anterioridad y se presentan los resultados obtenidos con esta nueva casuística en la Sección 5.3.

5.1. Formulación del nuevo caso

En la Sección 4.1 se presentó una formulación matemática para el problema de rutas de vehículos eléctricos planteado por la empresa. Dicha formulación estaba diseñada para un solo camión, que debía partir de la base operativa y volver a ella, visitando una serie de puntos de recogida y un punto de descarga. Además, se imponía la restricción de que el vehículo tenía que hacer una parada en una estación de carga.

Para el nuevo caso, donde el camión tiene que recorrer un mayor número de nodos, es necesario incluir un límite en la distancia recorrida, ya que la autonomía del vehículo es finita. Los camiones eléctricos pueden recorrer un máximo de 400 km al día, por lo que se precisa añadir esta restricción al modelo. Anteriormente, el número de nodos era lo suficientemente pequeño para que no existiese un problema debido a la distancia recorrida y no fuese necesaria esta restricción.

Asimismo, la inclusión del límite de distancia recorrida permite que haya nodos en la ruta que no sean visitados por el vehículo. Para solventar este problema, se incorpora al modelo la posibilidad de emplear más camiones eléctricos, de manera que, si un camión no puede visitar todos los nodos debido a la restricción de distancia, otro camión puede encargarse de los nodos restantes.

Para formular el problema, se van a utilizar algunos conjuntos, parámetros y variables ya empleados anteriormente y se van a incluir nuevas definiciones:

- $N = N^C \cup N^D = \{1, \ldots, n\}$ es el conjunto de puntos a visitar del cliente.
  - $N^C = \{1, \ldots, n-1\}$ hace referencia a los puntos de recogida.
  - $N^D = \{n\}$ hace referencia al punto de descarga.
- $F = \{n+1, \ldots, n+p\}$ se corresponde con las estaciones de carga disponibles.
- $V = \{0\} \cup \{0^*\} \cup N \cup F = \{0, 0^*, 1, \ldots, n-1, n, n+1, \ldots, n+p\}$ representa el conjunto de nodos, donde el nodo 0 es la base operativa y 0^*[1] representa una base operativa ficticia.
- $K = \{1, \ldots, m\}$, con $m \in \mathbb{N}, m \geq 1$ es el conjunto de camiones eléctricos disponibles.
- c_{ij} hace referencia al coste de ir del nodo i al nodo j. Este coste se calculará como el producto de los kilómetros recorridos y el precio del kilómetro.
- w_i es el precio de cargar un camión en el punto de carga i.
- d_{ij} representa el tiempo que se tarda en ir del nodo i al nodo j.
- d_{MAX} se corresponde con la distancia máxima que puede recorrer un camión.
- t_i^k es una variable de decisión continua que define el tiempo de llegada del vehículo al punto i.
- $x_{ij}^k \in \{0, 1\}$ es una variable de decisión binaria que vale 1 si el arco (i, j) está en la ruta del vehículo k y 0 en otro caso.
- $z_i^k \in \{0, 1\}$ es una variable de decisión binaria que vale 1 si la estación de carga i es usada por el vehículo k y 0 en otro caso. Como cada camión solamente usa una estación de carga, indica a mayores si el camión k es usado ($z_i^k = 1$) o no ($z_i^k = 0$).

Como novedad, se ha integrado d_{MAX} como parámetro del problema y se han incluido superíndices k a las variables t_i^k, x_{ij}^k y z_i^k para indicar el camión al que se refieren.

El uso de múltiples vehículos implica que la función objetivo y las restricciones deben adaptarse para incluir las nuevas variables. Integrar más de un camión en la ruta conlleva un aumento en el coste total: cada camión tiene un gasto fijo de 150 € por día de uso, por lo que el modelo buscará hacer rutas con el menor número de camiones posible. Así, el modelo se puede formular de la siguiente manera:

[1] Este nodo es necesario para la programación del problema, ya que evitamos la formación de ciclos en la solución.

$$\text{minimizar} \quad \sum_{i,j\in V,k\in K} c_{ij}x_{ij}^k + \sum_{i\in F,k\in K} w_i z_i^k + \sum_{i\in F,k\in K} 150 z_i^k \tag{5.1}$$

$$\text{sujeto a} \quad \sum_{j\in V} x_{ij}^k = 1, \qquad \forall i \in V\backslash\{F\cup\{0^*\}\}, k\in K \tag{5.2}$$

$$\sum_{i\in V} x_{ij}^k - \sum_{h\in V} x_{jh}^k = 0, \qquad \forall j \in V\setminus\{\{0\}\cup\{0^*\}\}, k\in K \tag{5.3}$$

$$\sum_{j\in V\backslash\{\{0\}\cup\{0^*\}\}} x_{0j}^k = 1, \qquad \forall k\in K \tag{5.4}$$

$$\sum_{i\in V\backslash\{\{0\}\cup\{0^*\}\}} x_{i0^*}^k = 1, \qquad \forall k\in K \tag{5.5}$$

$$x_{0^*j} = 0, \qquad \forall j\in V \tag{5.6}$$

$$t_j^k \geq t_i^k - (1-\sum_{h\in V} x_{hi}^k)L \qquad \forall i\in N^C, j\in N^D, k\in K, L\ggg 0 \tag{5.7}$$

$$t_i^k + d_{ij}x_{ij}^k - t_j^k \leq (1-x_{ij}^k)L, \qquad \forall i,j\in V, k\in K, L\ggg 0 \tag{5.8}$$

$$\sum_{i\in N\cup\{0\}, j\in F} x_{ij}^k = 1, \qquad \forall k\in K \tag{5.9}$$

$$\sum_{i,j\in V} d_{ij}x_{ij}^k \leq d_{MAX}, \qquad \forall k\in K \tag{5.10}$$

$$x_{ij} \leq z_i^k, \qquad \forall i\in F, j\in N, k\in K \tag{5.11}$$

$$x_{ij} \leq z_j^k, \qquad \forall i\in N, j\in F, k\in K \tag{5.12}$$

$$x_{ij}, z_i^k \in \{0,1\}, \qquad \forall i,j\in V, k\in K \tag{5.13}$$

$$t_i^k \in \mathbb{R}^+, \qquad \forall i\in V, k\in K \tag{5.14}$$

La función objetivo (5.1) minimiza el coste variable de la operación, que incluye el coste derivado de los kilómetros recorridos, el coste asociado a la carga de la batería y el coste fijo por el uso de cada camión. De nuevo, al tratarse de costes, esta función se mide en euros.

Como se puede observar, la formulación es muy similar a la presentada en la Sección 4.1, con algunas diferencias clave para adaptarse al nuevo escenario. A las variables que hacen referencia a los camiones, se les añadió el superíndice k, que permite distinguir entre los distintos vehículos. Asimismo, se añadió una nueva restricción (5.10) que limita la distancia máxima que puede recorrer cada camión, asegurando que no se exceda la autonomía diaria del vehículo.

Cabe destacar que, si la ruta óptima no supera los 400 km, el modelo empleará un solo camión, y por tanto, la solución será equivalente a la del modelo original. Es decir, se ha desarrollado una formulación más general, adaptándose a una nueva casuística.

5.2. Métodos y técnicas seleccionadas

El objetivo de este apartado es analizar la escalabilidad de las técnicas empleadas anteriormente para instancias mayores. Por lo tanto, se hará uso de los mismos métodos: el modelo exacto mediante SCIP, el algoritmo del vecino más próximo como heurística constructiva y el algoritmo *2-opt* como heurística de mejora.

Para implementar el modelo exacto, se ha empleado nuevamente la librería OR-Tools de Google. Se ajustaron los parámetros del modelo, manteniendo los valores de $L = 10^5$ y el tiempo máximo de duración del recorrido de 15h y a mayores se añadió el límite de distancia máxima d_{MAX}, que se adaptará a las limitaciones del vehículo empleado. Como se está trabajando con instancias de gran tamaño, se fijó un tiempo máximo de ejecución de 20 minutos, que es un tiempo que la empresa considera razonable. Con ese tiempo, el *solver* es posible que no devuelva la mejor solución existente, pero se garantiza que el método termine en un tiempo razonable.

Se emplearán los mismos criterios del capítulo anterior: la calidad de la solución obtenida, los tiempos de cómputo y la escalabilidad de los algoritmos.

Para adaptar la heurística constructiva a los nuevos datos, se ha incluido en el código una comprobación de la distancia máxima. El proceso es muy similar al algoritmo con un solo vehículo: se genera una ruta con los nodos aún no visitados más cercanos, comprobando en cada paso que no se excede la distancia máxima fijada. Si se alcanza el límite establecido y el vehículo no ha visitado todos los nodos necesarios, se crea una nueva ruta con un nuevo camión desde el punto de partida. Esta lógica se repite hasta que se han visitado todos los nodos de carga y descarga.

En el Algoritmo 3 se muestra el pseudocódigo del vecino más próximo adaptado a múltiples vehículos.

Algoritmo 3 Algoritmo del vecino más próximo para múltiples vehículos

```
Inicializar solución vacía
Marcar todos los loadings y unloadings como no visitados
while existan nodos por visitar do
    Iniciar nueva ruta desde el parking
    current_loc ← parking
    distancia_ruta ← 0
                                                                    ▷ Loadings
    while existan loadings no visitados do
        Elegir el loading más cercano a current_loc
        if Se supera la distancia de la ruta máxima then
            break                  ▷ Cerrar ruta: no cabe este loading respetando el límite
        end if
        Añadir loading a la ruta
        Marcar loading como visitado
        distancia_ruta ← distancia_ruta + dist(current_loc, loading)
        current_loc ← loading elegido
    end while
                                                                  ▷ Unloadings
    while existan unloadings no visitados do
        Seleccionar el unloading no visitado más cercano a current_loc
        if Se supera la distancia de la ruta máxima then
            break                ▷ Cerrar ruta: no cabe este unloading respetando el límite
        end if
        Añadir unloading a la ruta
        Marcar unloading como visitado
        distancia_ruta ← distancia_ruta + dist(current_loc), unloading)
        current_loc ← unloading
    end while
                                                                ▷ Insertar charger
    if hay chargers disponibles then
        Elegir el charger más cercano al último nodo
        Buscar posición que minice el aumento de distancia
        Insertar charger
    end if
    Terminar ruta en parking
    Añadir ruta completa a la solución
end while
Verificar restricciones de factibilidad globales
```

El algoritmo *2-opt* para los nuevos datos es muy similar al algoritmo original. Para implementarlo con múltiples vehículos, se aplica el algoritmo para cada ruta, verificando en cada paso que se satisfacen las restricciones de distancia impuestas. Es decir, se tratan las diferentes rutas de los camiones como si fuesen soluciones iniciales independientes. Aunque, desde un punto de vista académico, sería posible extender el algoritmo *2-opt* incorporando intercambios entre rutas de vehículos distintos, esta mejora queda fuera del alcance del trabajo y se marca como línea de trabajo futura.

Se puede ver el pseudocódigo de la heurística de mejora en el Algoritmo 4.

Algoritmo 4 Algoritmo *2-opt* para múltiples vehículos

```
Inicializar nueva solución vacía
for each ruta (vehículo) en la solución inicial do
    Copiar la ruta como best_route
    mejorar ← True
    while mejorar do
        mejorar ← False
        for i ← 1 until penúltimo nodo antes del final do
            for j ← i + 1 until penúltimo nodo do
                Seleccionar segmento entre i y j                ▷ Respetar restricciones
                if mezcla loadings y unloadings then
                    continue
                end if
                if más de un charger then
                    continue
                end if
                Invertir el segmento para obtener new_route
                Calcular distancia total de new_route
                if existe un límite de km por vehículo and se supera then
                    continue
                end if
                Calcular distancia total de best_route
                if new_route mejora la distancia de best_route then
                    best_route ← new_route
                    mejorar ← True
                    break                                       ▷ Reiniciar búsqueda en la nueva ruta
                end if
            end for
            if mejorar then
                break                                           ▷ Volver a iterar
            end if
        end for
    end while
    Agregar best_route a la nueva solución
end for
Verificar factibilidad global de todas las rutas
return solución optimizada
```

5.2.1. Datos empleados

Para evaluar el rendimiento de los métodos seleccionados en situaciones de mayor escala, *Trucksters* proporcionó un conjunto de datos ampliado basado en la operativa real. Estos datos representan posibles puntos de recogida dentro de la provincia de Barcelona, siguiendo patrones coherentes con los históricos de entregas de la empresa. La estructura de los datos es la siguiente:

- 1 nodo correspondiente a la base operativa.
- 1 nodo de descarga.
- 5 nodos que representa estaciones de recarga.

- 129 nodos ampliados que actúan como nodos de recogida.

En total, disponemos de 136 nodos, lo que permite generar instancias de mayor tamaño para evaluar el comportamiento de los algoritmos. En concreto, se han generado 30 instancias diferentes, seleccionando un tamaño de $|N|$, correspondiente a los nodos de recogida, de 10, 15 y 20. Para cada tamaño, se han creado 10 instancias distintas seleccionando los nodos de forma aleatoria.

Para el cálculo de las distancias, los costes y los tiempos se emplearon los mismos métodos usados con anterioridad.

5.3. Comparativa

En esta sección se realiza una comparativa de los tres métodos implementados con las técnicas de la Sección 5.2, utilizando la base de datos ampliada.

Se ha evaluado el comportamiento de los algoritmos con tres distancias máximas permitidas para los vehículos. La primera $d_{MAX} = 400$, se corresponde con la limitación actual permitida para el camión eléctrico disponible. La segunda y tercera, $d_{MAX} = 200$ y $d_{MAX} = 150$, son una extrapolación de las características del vehículo a posibles camiones eléctricos.

Para seguir la estructura de la Sección 4.3, se han recogido en las Tablas 5.1, 5.2 y 5.3 los resultados medios de las 10 instancias para cada $|N|$ (el coste medio de la solución obtenida y los tiempos de ejecución). A mayores, se ha incluido una columna en cada método que indica los camiones medios que han sido necesarios para calcular la ruta ($|K|$) y el número de instancias resueltas por cada uno (S.). Se pueden ver las tablas con los resultados completos en el Apéndice B.

Se presentan además en las Figuras 5.1 y 5.2, 5.3, boxplots relativos al tiempo de cómputo, al coste de la solución y a la diferencia porcentual de la solución de las heurísticas respecto al método exacto por cada método y cada tamaño de $|N|$.

	Modelo exacto				**Vecino más próximo**				***2-opt***			
$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	S.	Coste (€)	Tiempo (s)	$\|K\|$	S.	Coste (€)	Tiempo (s)	$\|K\|$	S.
10	304.303	$1.200 \cdot 10^3$	1	10	331.975	$9.2 \cdot 10^{-5}$	1	10	313.732	$1.321 \cdot 10^{-3}$	1	10
15	349.398	$1.200 \cdot 10^3$	1	10	378.621	$1.2 \cdot 10^{-4}$	1	10	349.123	$5.757 \cdot 10^{-3}$	1	10
20	403.713	$1.200 \cdot 10^3$	1	10	385.656	$1.59 \cdot 10^{-4}$	1	10	362.249	$1.110 \cdot 10^{-3}$	1	10

Tabla 5.1: Comparación de los tres métodos con nodos ficticios y $d_{MAX} = 400$. Se recogen los valores medios de las funciones objetivo, los tiempos medios de ejecución, el número medio de camiones ($|K|$) y el número de instancias resueltas (S.) para cada tamaño.

	Modelo exacto				**Vecino más próximo**				***2-opt***			
$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	S.	Coste (€)	Tiempo (s)	$\|K\|$	S.	Coste (€)	Tiempo (s)	$\|K\|$	S.
10	391.732	$1.202 \cdot 10^3$	1.4	10	508.8	$2.050 \cdot 10^{-4}$	1.9	10	496.9	$1.421 \cdot 10^{-3}$	1.9	10
15	652.574	$1.202 \cdot 10^3$	2.3	10	590.1	$2.901 \cdot 10^{-4}$	2	10	576.8	$5.656 \cdot 10^{-3}$	2	10
20	569.585	$1.204 \cdot 10^3$	2	2	634.1	$1.203 \cdot 10^{-3}$	2.2	10	622.2	$2.435 \cdot 10^{-3}$	2.2	10

Tabla 5.2: Comparación de los tres métodos con nodos ficticios y $d_{MAX} = 200$. Se recogen los valores medios de las funciones objetivo, los tiempos medios de ejecución y el número medio de camiones ($|K|$) y el número de instancias resueltas (S.) para cada tamaño.

	Modelo exacto				**Vecino más próximo**				***2-opt***			
$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	S.	Coste (€)	Tiempo (s)	$\|K\|$	S.	Coste (€)	Tiempo (s)	$\|K\|$	S.
10	494.47	$1.200 \cdot 10^3$	1.9	10	582.0	$1 \cdot 10^{-3}$	2.2	10	573.2	$1.921 \cdot 10^{-3}$	2.2	10
15	831.034	$1.200 \cdot 10^3$	3.2	5	799.0	$2.45 \cdot 10^{-4}$	3	10	786.8	$4.757 \cdot 10^{-3}$	3	10
20	-	-	-	0	873.7	$4.59 \cdot 10^{-4}$	3.3	10	864.0	$8.410 \cdot 10^{-3}$	3.3	10

Tabla 5.3: Comparación de los tres métodos con nodos ficticios y $d_{MAX} = 150$. Se recogen los valores medios de las funciones objetivo, los tiempo de ejecución y el número medio de camiones ($|K|$) y el número de instancias resueltas (S.) para cada tamaño.

En este nuevo experimento, se fijó un tiempo máximo de 20 minutos de ejecución para el método exacto. Se puede ver en la Tablas 5.1, 5.2 y 5.3 como, de nuevo, el *solver* emplea el tiempo máximo establecido. Para la distancia de 400 km, se observa como el método exacto da lugar a soluciones buenas para los conjuntos de $|N| = 10$ y $|N| = 15$, sin embargo, en el caso de $|N| = 20$, ambas heurísticas consiguen reducir el coste de la ruta calculada.

Para los casos de límite de distancia de 200 km y 150 km, el *solver* ya no es capaz de resolver de forma eficaz el problema. Esto es debido a que la región factible se hace más pequeña y el problema se vuelve más complejo, lo que provoca que 20 minutos no sean suficientes para su resolución. Para solventar esto, sería necesario ampliar el tiempo límite de ejecución. Además, esto justifica la incorporación de la columna relativa a las soluciones encontradas. Como podemos ver en las Tablas 5.2 y 5.3, el método exacto tiene dificultades a la hora de encontrar soluciones para los tamaños $|N| = 15$ y $|N| = 20$.

Para $|N| = 10$, el método exacto presenta las mejores soluciones en media de las tres técnicas, mientras que para $|N| = 15$, es mejorada por ambas heurísticas. Destaca el rendimiento eficaz en media que presenta el *solver* para $|N| = 20$ con $d_{MAX} = 200$, debido a que se está haciendo la media para tan solo dos instancias con las que se obtuvo solución con este método.

En el caso de la heurística del vecino más próximo, este sigue siendo el método que peores soluciones proporciona de forma general. Este comportamiento se puede apreciar de manera muy clara en la Figura 5.3, donde las diferencias respecto al método exacto para $d_{MAX} = 400$ son, por lo general, mayores a 0. En los casos en los que se disminuye la distancia permitida, podemos ver algunos casos donde la heurística consigue mejorar al *solver*, aunque, de nuevo, se debe tener en cuenta que no se comparan todas las posibles soluciones. En cuanto a tiempos de ejecución, la heurística constructiva sigue presentando los tiempos más pequeños, siendo el más competitivo en este aspecto para todos los

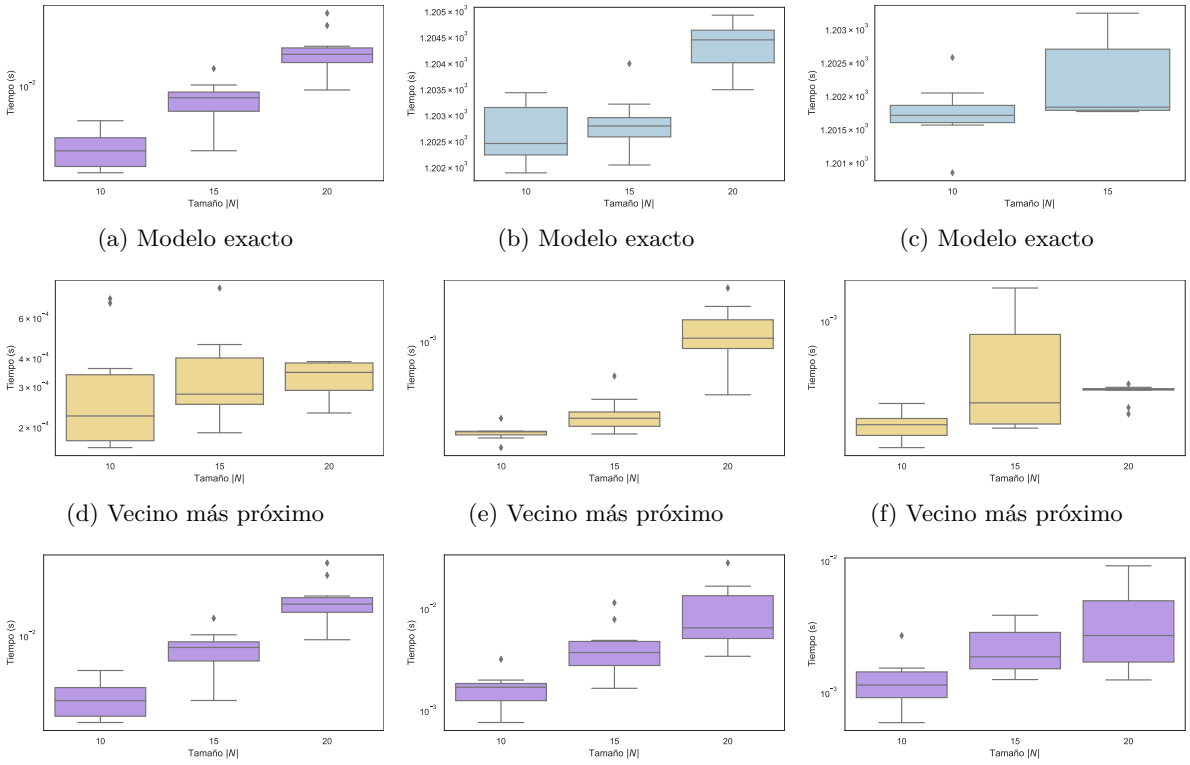
experimentos realizados.

Por último, el algoritmo *2-opt* parece proporcionar de nuevo un equilibrio entre tiempo y calidad de soluciones. En las instancias de 10 nodos de recogida para $d_{MAX} = 400$, esta técnica no mejora de media las soluciones del *solver*, sin embargo, a medida que aumentamos los nodos, se aprecia como estas diferencias empiezan a ser menores, consiguiendo la heurística reducir los costes de la ruta en algunas ocasiones(Figura 5.3f). Para los otros dos tamaños, el algoritmo vuelve a ser muy eficiente para reducir la solución. En ambos casos cumple su cometido de reducir el coste del algoritmo del vecino más próximo, con un aumento de tiempo despreciable.

En la Figura 5.2 se aprecia que, a medida que aumenta el tamaño de las instancias, y por tanto la longitud de la ruta, se incrementa a su vez el coste para ambas heurísticas. No obstante, esta diferencia parece mayor entre las instancias con $|N| = 10$ y $|N| = 15$, que con $|N| = 15$ y $|N| = 20$. Esto puede ser debido a que la disposición de los puntos de recogida es más próxima a medida que aumentamos su número. Es por esto que no se plantea en este Trabajo Fin de Máster realizar pruebas con un mayor número de nodos, que además, conllevaría unos tiempos de ejecución muy elevados.

Cabe destacar que, a pesar de haber adaptado todos los métodos para incluir múltiples vehículos, para el caso de $d_{MAX} = 400$, en la mayoría de los casos bastaría con usar la formulación planteada en la Sección 4, dado que se emplea únicamente un camión para llegar a la solución en 29 de 30 instancias. Para $d_{MAX} = 200$, el número de vehículos parece estar muy próximo a dos, aumentando ligeramente su número según se incrementa el número de puntos de recogida. En el caso de $d_{MAX} = 150$, se necesita un vehículo más para satisfacer la demanda de los clientes. La empresa *Trucksters* tendría que hacerse con dos camiones eléctricos que pudiesen recorrer un máximo de 150 km para poder resolver esta casuística.

En conclusión, según los criterios establecidos de calidad de solución, tiempo de cómputo y escalabilidad, el algoritmo *2-opt* presenta un buen desempeño. La calidad de las soluciones para un tiempo límite de 20 minutos es, por lo general, la mejor, reduciendo los costes respecto al método exacto en varios casos, que no llega a alcanzar la solución óptima. El tiempo empleado es de apenas unos milisegundos, por lo que proporciona soluciones inmediatas. A mayores, esta técnica presenta un buen funcionamiento para instancias de mayor tamaño y diferentes valores de d_{MAX}. El algoritmo *2-opt* satisface los requerimientos tanto técnicos, como de negocio, siendo el idóneo para incorporarse a la operativa de la empresa.

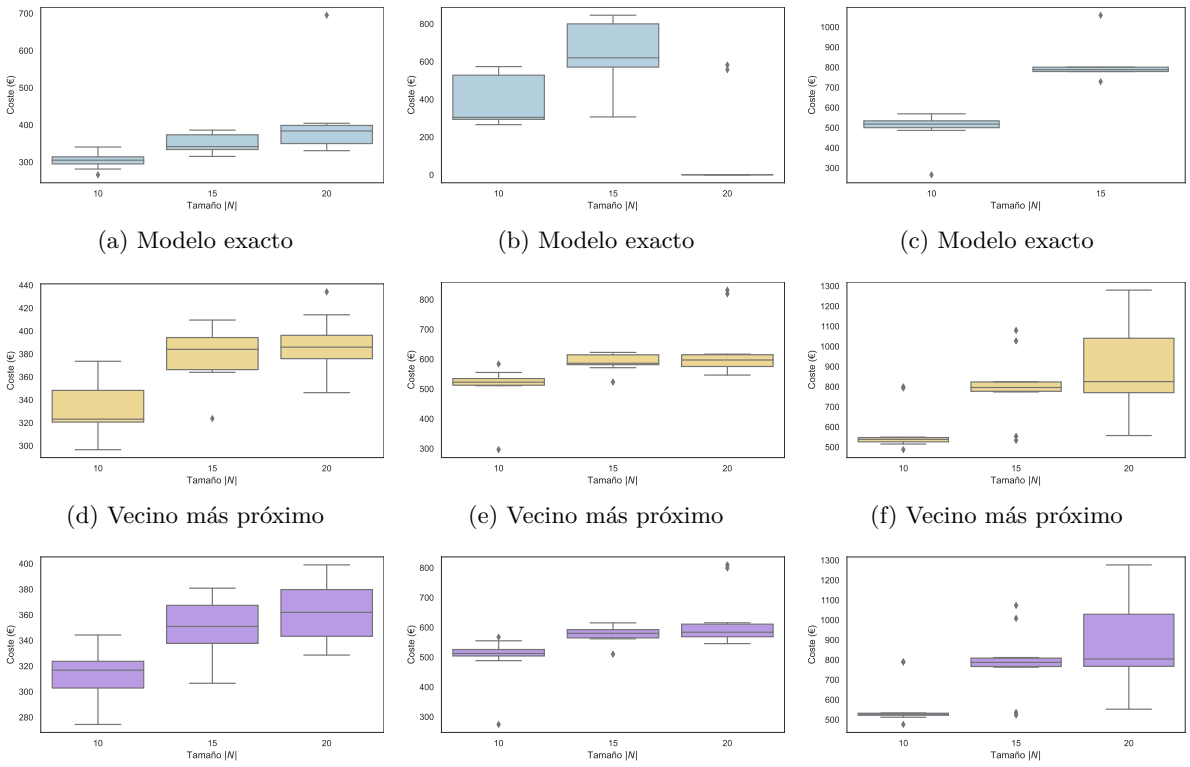


(a) Modelo exacto (b) Modelo exacto (c) Modelo exacto

(d) Vecino más próximo (e) Vecino más próximo (f) Vecino más próximo

(g) *2-opt* (h) *2-opt* (i) *2-opt*

Figura 5.1: Boxplots de tiempos de ejecución. En la primera columna, los resultados para $d_{MAX} = 400$; en la segunda, para $d_{MAX} = 200$; y en la tercera, para $d_{MAX} = 150$.

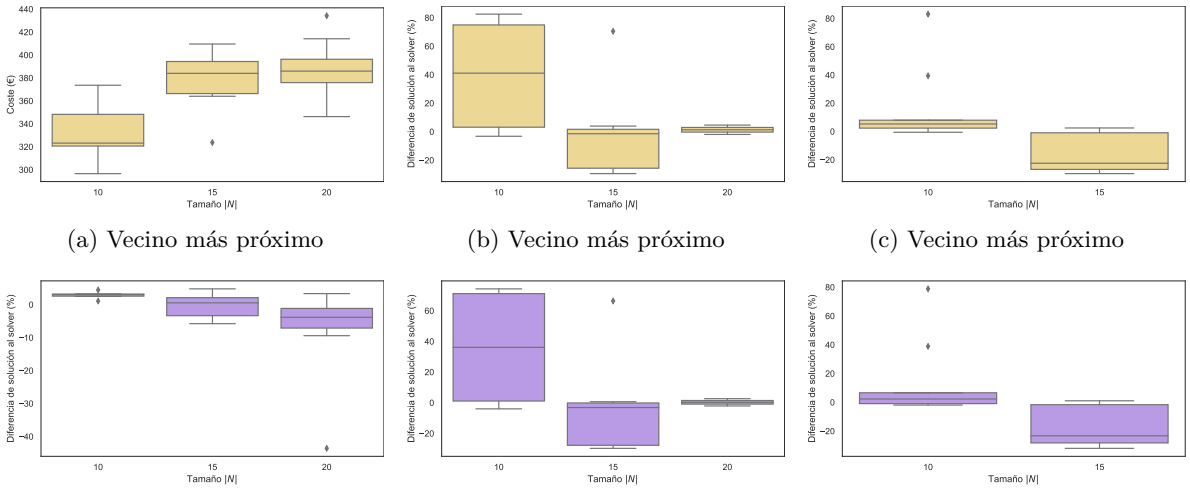


(a) Modelo exacto (b) Modelo exacto (c) Modelo exacto

(d) Vecino más próximo (e) Vecino más próximo (f) Vecino más próximo

(g) *2-opt* (h) *2-opt* (i) *2-opt*

Figura 5.2: Boxplots de los costes asociados a la solución. En la primera columna, los resultados para $d_{MAX} = 400$; en la segunda, para $d_{MAX} = 200$; y en la tercera, para $d_{MAX} = 150$.



(a) Vecino más próximo (b) Vecino más próximo (c) Vecino más próximo

(d) *2-opt* (e) *2-opt* (f) *2-opt*

Figura 5.3: Boxplots de las diferencias porcentuales de cada heurística respecto al modelo exacto.En la primera columna, los resultados para $d_{MAX} = 400$; en la segunda, para $d_{MAX} = 200$; y en la tercera, para $d_{MAX} = 150$.

Capítulo 6

Conclusiones y líneas de trabajo futuro

A lo largo de este trabajo se ha desarrollado un modelo de optimización para resolver el problema de rutas de vehículos eléctricos planteado por la empresa *Trucksters*. El objetivo era planificar el itinerario de un camión eléctrico que partía de una base operativa, visitaba varios puntos de recogida, realizaba la descarga en una localización fijada y regresaba de nuevo a la base. A mayores, el vehículo debía parar en una estación de recarga antes de agotar la batería del mismo.

La finalidad de este Trabajo Fin de Máster es la comparación de diferentes metodologías seleccionadas. Para ello, en un primer momento se llevó a cabo una modelización del problema planteado en la Sección 4.1. La formulación recogía las restricciones necesarias para que el camión realizase el recorrido por los diferentes puntos, cumpliendo las necesidades de los vehículos eléctricos. Posteriormente se implementó la casuística para resolverlo mediante los métodos seleccionados: un método exacto con el *solver* SCIP, la heurística del vecino más próximo y el algoritmo *2-opt*. Para ello se desarrolló un código de elaboración propia en el lenguaje de programación Python.

Por último, se realizó un análisis comparativo de los resultados obtenidos con las tres técnicas y dos conjuntos de datos diferentes, adaptando los métodos para ambos casos. Las soluciones mostraban que el modelo exacto proporcionaba, para los casos con un único vehículo, de forma general, los menores costes para el problema, pero conllevando tiempos de ejecución elevados. Sin embargo, con más de un vehículo, el *solver* ya no era tan competente. El algoritmo del vecino más próximo ofrecía una solución constructiva rápida, con tiempos muy bajos, aunque con una calidad de solución moderada para todos los experimentos. La heurística *2-opt* proporcionaba un equilibrio óptimo entre calidad de solución y tiempo de ejecución, teniendo el mejor rendimiento tanto para instancias pequeñas como de mayor tamaño, y conviertiéndose en el algoritmo seleccionado para implementar en la empresa.

Se han alcanzado los objetivos y los resultados esperados por los algoritmos, aún así, existen ámbitos donde el trabajo podría seguir mejorándose. En primer lugar, las rutas que se han estudiado son en un entorno limitado, dentro de la provincia de Barcelona. Una posible línea de trabajo futuro sería el análisis de rutas nacionales o internacionales, que impliquen mayores distancias y tiempos de viaje. Así, se podrían implementar nuevas restricciones y variables que se adapten a las exigencias de estos escenarios. Por otro lado, en la formulación empleada se eliminaron las ventanas temporales y las restricciones relativas a la capacidad energética del vehículo. Una posible mejora podría ser la implementación de estas restricciones al modelo, adaptándolo aún más a las necesidades reales de los clientes. En relación con la modelización del problema, se comentó que la selección de L se

hizo de forma que se garantizase la validez de las restricciones de tipo big-M. En este contexto, esta elección se podría realizar de forma más precisa, siendo una posible área en la que mejorar el modelo. Además, las distancias utilizadas para evaluar las rutas se calculan mediante la distancia geodésica, sin considerar el relieve del terreno, la estructura real de la red de carreteras ni posibles desvíos. Una mejora relevante para futuros trabajos sería incorporar distancias reales obtenidas de mapas o servicios de geolocalización. Por último, se ha implementado un algoritmo *2-opt* para múltiples vehículos que mejoraba únicamente las rutas ya elegidas por la heurística constructiva. Se podría mejorar de forma que se permitiesen los intercambios entre rutas de diferentes vehículos.

En conclusión, en este Trabajo Fin de Máster se han revisado los conceptos necesarios para abordar el problema de rutas de vehículos eléctricos, se ha propuesto una formulación que satisface las necesidades de la empresa y se han diseñado dos algoritmos heurísticos, implementados con un código de elaboración propia. Además, los resultados alcanzados pueden considerarse satisfactorios, ya que se reducen los tiempos de planificación marcados en un principio. En definitiva, se ha demostrado que la automatización de la planificación de rutas para vehículos eléctricos es viable y beneficiosa.

Apéndice A

Resultados de los métodos

En la Sección 4.3 se incluyó un resumen de los resultados obtenidos para el experimento. En este apéndice se recogerán los datos al completo para una mejor comprensión.

Se pueden ver en la Tabla A.1 estos resultados. Se omite el número de vehículos empleados en cada instancia por ser en todas 1.

		Modelo exacto		**Vecino más próximo**		***2-opt***	
	$\|N\|$	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)
Instancia 1	3	268.74	$3.0052 \cdot 10^{2}$	292.52	$8.7400 \cdot 10^{-5}$	277.71	$1.3230 \cdot 10^{-4}$
Instancia 2	3	231.35	$3.0073 \cdot 10^{2}$	248.09	$8.2000 \cdot 10^{-5}$	240.34	$1.9490 \cdot 10^{-4}$
Instancia 3	3	231.19	$3.0089 \cdot 10^{2}$	240.46	$7.3900 \cdot 10^{-5}$	240.18	$1.3560 \cdot 10^{-4}$
Instancia 4	3	212.75	$2.6220 \cdot 10^{2}$	222.72	$8.0300 \cdot 10^{-5}$	221.65	$1.6490 \cdot 10^{-4}$
Instancia 5	3	263.86	$3.0057 \cdot 10^{2}$	284.82	$8.6400 \cdot 10^{-5}$	272.26	$2.1230 \cdot 10^{-4}$
Instancia 6	3	253.54	$3.0090 \cdot 10^{2}$	264.82	$9.1200 \cdot 10^{-5}$	262.44	$1.3880 \cdot 10^{-4}$
Instancia 7	3	258.94	$3.0084 \cdot 10^{2}$	278.59	$8.3500 \cdot 10^{-5}$	267.91	$2.6040 \cdot 10^{-4}$
Instancia 8	3	239.47	$3.0093 \cdot 10^{2}$	251.00	$8.7300 \cdot 10^{-5}$	247.87	$1.8270 \cdot 10^{-4}$
Instancia 9	3	259.40	$3.0092 \cdot 10^{2}$	276.66	$8.3900 \cdot 10^{-5}$	268.37	$2.5500 \cdot 10^{-4}$
Instancia 10	3	242.58	$3.0075 \cdot 10^{2}$	260.62	$8.8700 \cdot 10^{-5}$	251.53	$1.9780 \cdot 10^{-4}$
Instancia 11	4	256.01	$3.0089 \cdot 10^{2}$	274.94	$8.8200 \cdot 10^{-5}$	264.87	$3.3240 \cdot 10^{-4}$
Instancia 12	4	265.19	$3.0081 \cdot 10^{2}$	284.84	$8.3300 \cdot 10^{-5}$	274.16	$3.0380 \cdot 10^{-4}$
Instancia 13	4	226.36	$3.0090 \cdot 10^{2}$	234.42	$9.3900 \cdot 10^{-5}$	234.23	$1.9100 \cdot 10^{-4}$
Instancia 14	4	269.85	$3.0083 \cdot 10^{2}$	308.71	$1.0100 \cdot 10^{-4}$	278.48	$2.8180 \cdot 10^{-4}$
Instancia 15	4	254.94	$3.0093 \cdot 10^{2}$	274.44	$1.2400 \cdot 10^{-4}$	263.93	$3.3950 \cdot 10^{-4}$
Instancia 16	4	242.58	$3.0081 \cdot 10^{2}$	260.63	$9.1900 \cdot 10^{-5}$	251.54	$3.3830 \cdot 10^{-4}$
Instancia 17	4	264.56	$3.0092 \cdot 10^{2}$	285.69	$9.5900 \cdot 10^{-5}$	273.36	$2.2490 \cdot 10^{-4}$
Instancia 18	4	253.01	$3.0091 \cdot 10^{2}$	263.41	$1.2000 \cdot 10^{-4}$	261.98	$3.3650 \cdot 10^{-4}$
Instancia 19	4	237.63	$3.0085 \cdot 10^{2}$	247.64	$9.7800 \cdot 10^{-5}$	246.43	$2.2470 \cdot 10^{-4}$

		Modelo exacto		**Vecino más próximo**		***2-opt***	
	$\|N\|$	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)
Instancia 20	4	240.86	$3.0091 \cdot 10^{2}$	251.13	$8.5800 \cdot 10^{-5}$	250.16	$2.4940 \cdot 10^{-4}$
Instancia 21	5	265.83	$3.0084 \cdot 10^{2}$	290.68	$9.3100 \cdot 10^{-5}$	274.82	$4.7820 \cdot 10^{-4}$
Instancia 22	5	255.20	$3.0091 \cdot 10^{2}$	271.60	$8.6900 \cdot 10^{-5}$	264.19	$4.5880 \cdot 10^{-4}$
Instancia 23	5	256.46	$3.0087 \cdot 10^{2}$	274.13	$1.8190 \cdot 10^{-4}$	265.04	$6.2650 \cdot 10^{-4}$
Instancia 24	5	242.20	$3.0087 \cdot 10^{2}$	258.35	$9.7800 \cdot 10^{-5}$	251.20	$4.9910 \cdot 10^{-4}$
Instancia 25	5	253.11	$3.0090 \cdot 10^{2}$	264.26	$9.4000 \cdot 10^{-5}$	262.08	$5.0050 \cdot 10^{-4}$
Instancia 26	5	269.76	$3.0084 \cdot 10^{2}$	294.31	$1.5920 \cdot 10^{-4}$	278.73	$4.3250 \cdot 10^{-4}$
Instancia 27	5	274.46	$3.0083 \cdot 10^{2}$	283.55	$8.6700 \cdot 10^{-5}$	283.26	$4.1760 \cdot 10^{-4}$
Instancia 28	5	247.96	$3.0094 \cdot 10^{2}$	265.98	$9.4900 \cdot 10^{-5}$	256.96	$4.7410 \cdot 10^{-4}$
Instancia 29	5	253.10	$3.0086 \cdot 10^{2}$	263.52	$8.4800 \cdot 10^{-5}$	262.07	$3.2360 \cdot 10^{-4}$
Instancia 30	5	245.71	$3.0087 \cdot 10^{2}$	263.10	$7.3200 \cdot 10^{-5}$	254.70	$3.7210 \cdot 10^{-4}$
Instancia 31	6	242.22	$3.0082 \cdot 10^{2}$	258.48	$7.3200 \cdot 10^{-4}$	251.22	$5.1440 \cdot 10^{-4}$
Instancia 32	6	255.07	$3.0085 \cdot 10^{2}$	266.07	$8.1800 \cdot 10^{-5}$	264.64	$6.4480 \cdot 10^{-4}$
Instancia 33	6	275.10	$3.0091 \cdot 10^{2}$	283.78	$1.1080 \cdot 10^{-4}$	283.50	$4.5580 \cdot 10^{-4}$
Instancia 34	6	275.05	$3.0083 \cdot 10^{2}$	283.78	$1.2740 \cdot 10^{-4}$	283.45	$4.3470 \cdot 10^{-4}$
Instancia 35	6	241.83	$3.0229 \cdot 10^{2}$	252.05	$1.3080 \cdot 10^{-4}$	250.80	$5.3540 \cdot 10^{-4}$
Instancia 36	6	271.51	$3.0131 \cdot 10^{2}$	282.67	$3.0570 \cdot 10^{-4}$	280.14	$5.6290 \cdot 10^{-4}$
Instancia 37	6	281.94	$3.0085 \cdot 10^{2}$	310.12	$9.7100 \cdot 10^{-5}$	290.90	$5.2790 \cdot 10^{-4}$
Instancia 38	6	273.31	$3.0087 \cdot 10^{2}$	297.14	$1.0590 \cdot 10^{-4}$	282.31	$6.5490 \cdot 10^{-4}$
Instancia 39	6	281.04	$3.0082 \cdot 10^{2}$	309.27	$1.0260 \cdot 10^{-4}$	290.00	$5.9610 \cdot 10^{-4}$
Instancia 40	6	267.20	$3.0088 \cdot 10^{2}$	286.85	$1.1570 \cdot 10^{-4}$	276.17	$5.6300 \cdot 10^{-4}$
Instancia 41	7	255.93	$3.0097 \cdot 10^{2}$	265.63	$1.3130 \cdot 10^{-4}$	263.34	$1.2044 \cdot 10^{-3}$
Instancia 42	7	258.53	$3.0086 \cdot 10^{2}$	275.00	$1.0530 \cdot 10^{-4}$	267.53	$8.7570 \cdot 10^{-4}$
Instancia 43	7	276.02	$3.0076 \cdot 10^{2}$	284.70	$8.8600 \cdot 10^{-5}$	284.42	$5.6700 \cdot 10^{-4}$
Instancia 44	7	275.51	$3.0091 \cdot 10^{2}$	284.66	$5.9020 \cdot 10^{-4}$	284.31	$7.2950 \cdot 10^{-4}$
Instancia 45	7	257.10	$3.0065 \cdot 10^{2}$	274.47	$1.0180 \cdot 10^{-4}$	266.09	$2.2124 \cdot 10^{-3}$
Instancia 46	7	272.47	$3.0080 \cdot 10^{2}$	300.93	$1.3200 \cdot 10^{-4}$	281.46	$1.1582 \cdot 10^{-3}$
Instancia 47	7	282.65	$3.0087 \cdot 10^{2}$	310.77	$1.2160 \cdot 10^{-4}$	291.64	$1.4790 \cdot 10^{-3}$
Instancia 48	7	282.07	$3.0080 \cdot 10^{2}$	310.25	$1.5520 \cdot 10^{-4}$	291.03	$7.6330 \cdot 10^{-4}$
Instancia 49	7	272.12	$3.0088 \cdot 10^{2}$	300.53	$1.0070 \cdot 10^{-4}$	281.08	$8.1910 \cdot 10^{-4}$
Instancia 50	7	248.25	$3.0196 \cdot 10^{2}$	267.05	$2.1500 \cdot 10^{-4}$	257.24	$7.3110 \cdot 10^{-4}$
Instancia 51	8	277.35	$3.0185 \cdot 10^{2}$	301.90	$6.8930 \cdot 10^{-4}$	286.32	$1.1091 \cdot 10^{-3}$
Instancia 52	8	282.73	$3.0182 \cdot 10^{2}$	310.85	$2.6150 \cdot 10^{-4}$	291.72	$1.0713 \cdot 10^{-3}$
Instancia 53	8	259.37	$3.0179 \cdot 10^{2}$	276.85	$1.0490 \cdot 10^{-4}$	268.36	$2.1588 \cdot 10^{-3}$
Instancia 54	8	259.60	$3.0180 \cdot 10^{2}$	276.90	$9.6900 \cdot 10^{-5}$	268.52	$2.4769 \cdot 10^{-3}$

		Modelo exacto		**Vecino más próximo**		***2-opt***	
	$\|N\|$	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)	Coste (€)	Tiempo (s)
Instancia 55	8	272.31	$3.0181 \cdot 10^2$	300.63	$1.4136 \cdot 10^{-3}$	281.27	$1.0472 \cdot 10^{-3}$
Instancia 56	8	282.68	$3.0094 \cdot 10^2$	310.85	$1.4110 \cdot 10^{-4}$	291.67	$8.7800 \cdot 10^{-4}$
Instancia 57	8	277.40	$3.0168 \cdot 10^2$	298.39	$1.3490 \cdot 10^{-4}$	282.41	$9.9170 \cdot 10^{-4}$
Instancia 58	8	273.82	$3.0084 \cdot 10^2$	302.14	$1.1340 \cdot 10^{-4}$	282.78	$9.7460 \cdot 10^{-4}$
Instancia 59	8	281.33	$3.0069 \cdot 10^2$	309.45	$1.3000 \cdot 10^{-4}$	290.32	$9.1380 \cdot 10^{-4}$
Instancia 60	8	282.81	$3.0169 \cdot 10^2$	310.90	$1.0720 \cdot 10^{-4}$	291.77	$1.0700 \cdot 10^{-3}$
Instancia 61	9	283.79	$3.0178 \cdot 10^2$	311.88	$1.0930 \cdot 10^{-4}$	292.75	$1.0357 \cdot 10^{-3}$
Instancia 62	9	283.81	$3.0174 \cdot 10^2$	311.77	$2.6190 \cdot 10^{-4}$	292.76	$1.1075 \cdot 10^{-3}$
Instancia 63	9	283.63	$3.0182 \cdot 10^2$	311.83	$1.3470 \cdot 10^{-4}$	292.59	$1.2773 \cdot 10^{-3}$
Instancia 64	9	287.92	$3.0183 \cdot 10^2$	311.77	$1.5200 \cdot 10^{-4}$	292.64	$1.8108 \cdot 10^{-3}$
Instancia 65	9	273.84	$3.0197 \cdot 10^2$	302.27	$1.4780 \cdot 10^{-4}$	282.80	$2.0908 \cdot 10^{-3}$
Instancia 66	9	262.62	$3.0419 \cdot 10^2$	276.92	$1.4940 \cdot 10^{-4}$	268.54	$2.4307 \cdot 10^{-3}$
Instancia 67	9	282.86	$3.0180 \cdot 10^2$	310.98	$1.2490 \cdot 10^{-4}$	291.85	$1.2360 \cdot 10^{-3}$
Instancia 68	9	282.28	$3.0198 \cdot 10^2$	310.48	$1.2240 \cdot 10^{-4}$	291.24	$1.2708 \cdot 10^{-3}$
Instancia 69	9	280.37	$3.0159 \cdot 10^2$	302.03	$1.3620 \cdot 10^{-4}$	286.45	$1.1942 \cdot 10^{-3}$
Instancia 70	9	287.43	$3.0173 \cdot 10^2$	311.01	$9.6700 \cdot 10^{-5}$	292.68	$1.0766 \cdot 10^{-3}$

Tabla A.1: Comparación de los tres métodos de las instancias reales de la Sección 4.3.

Apéndice B

Nuevo caso planteado

En la Sección 5.3 se incluyó un resumen de los resultados obtenidos para los experimentos. En este apéndice se recogerán los datos al completo para una mejor comprensión.

Se pueden ver en las Tablas B.1, B.2 y B.3 estos resultados.

		Modelo exacto			**Vecino más próximo**			***2-opt***		
	$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$
Instancia 1	10	340.53	$1.2020 \cdot 10^{3}$	1	351.64	$1.6260 \cdot 10^{-4}$	1	344.30	$2.5404 \cdot 10^{-3}$	1
Instancia 2	10	334.93	$1.2022 \cdot 10^{3}$	1	373.68	$2.7280 \cdot 10^{-4}$	1	343.48	$4.4978 \cdot 10^{-3}$	1
Instancia 3	10	294.64	$1.2018 \cdot 10^{3}$	1	317.24	$1.6530 \cdot 10^{-4}$	1	302.03	$2.2629 \cdot 10^{-3}$	1
Instancia 4	10	315.47	$1.2021 \cdot 10^{3}$	1	350.25	$3.5990 \cdot 10^{-4}$	1	324.95	$2.2303 \cdot 10^{-3}$	1
Instancia 5	10	304.74	$1.2029 \cdot 10^{3}$	1	342.37	$2.4010 \cdot 10^{-4}$	1	314.69	$3.2033 \cdot 10^{-3}$	1
Instancia 6	10	281.28	$1.2022 \cdot 10^{3}$	1	322.16	$7.2640 \cdot 10^{-4}$	1	288.81	$1.8245 \cdot 10^{-3}$	1
Instancia 7	10	305.34	$1.2017 \cdot 10^{3}$	1	324.32	$1.9800 \cdot 10^{-4}$	1	318.96	$1.3801 \cdot 10^{-3}$	1
Instancia 8	10	296.25	$1.2017 \cdot 10^{3}$	1	320.90	$6.9440 \cdot 10^{-4}$	1	305.45	$4.0359 \cdot 10^{-3}$	1
Instancia 9	10	265.83	$1.2025 \cdot 10^{3}$	1	296.60	$2.0690 \cdot 10^{-4}$	1	274.43	$1.3590 \cdot 10^{-3}$	1
Instancia 10	10	310.33	$1.2020 \cdot 10^{3}$	1	320.59	$1.6640 \cdot 10^{-4}$	1	320.22	$1.4856 \cdot 10^{-3}$	1
Instancia 11	15	338.61	$1.2031 \cdot 10^{3}$	1	385.30	$2.4820 \cdot 10^{-4}$	1	337.47	$6.6489 \cdot 10^{-3}$	1
Instancia 12	15	372.51	$1.2019 \cdot 10^{3}$	1	382.83	$4.5790 \cdot 10^{-4}$	1	380.87	$3.6046 \cdot 10^{-3}$	1
Instancia 13	15	386.04	$1.2024 \cdot 10^{3}$	1	395.90	$1.9940 \cdot 10^{-4}$	1	371.64	$5.2416 \cdot 10^{-3}$	1
Instancia 14	15	373.78	$1.2026 \cdot 10^{3}$	1	401.81	$2.8890 \cdot 10^{-4}$	1	352.25	$7.5320 \cdot 10^{-3}$	1
Instancia 15	15	332.25	$1.2028 \cdot 10^{3}$	1	364.07	$1.8880 \cdot 10^{-4}$	1	338.52	$7.7401 \cdot 10^{-3}$	1
Instancia 16	15	322.12	$1.2025 \cdot 10^{3}$	1	323.78	$2.6760 \cdot 10^{-4}$	1	306.59	$8.0076 \cdot 10^{-3}$	1
Instancia 17	15	338.77	$1.2034 \cdot 10^{3}$	1	409.61	$8.0710 \cdot 10^{-4}$	1	355.02	$1.0230 \cdot 10^{-2}$	1
Instancia 18	15	315.25	$1.2030 \cdot 10^{3}$	1	366.27	$3.8230 \cdot 10^{-4}$	1	322.20	$1.4934 \cdot 10^{-2}$	1
Instancia 19	15	344.80	$1.2034 \cdot 10^{3}$	1	366.83	$2.5990 \cdot 10^{-4}$	1	349.75	$8.9265 \cdot 10^{-3}$	1
Instancia 20	15	385.36	$1.2034 \cdot 10^{3}$	1	389.81	$4.0610 \cdot 10^{-4}$	1	376.92	$2.2560 \cdot 10^{-3}$	1

		Modelo exacto			**Vecino más próximo**			***2-opt***		
	$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$
Instancia 21	20	346.97	$1.2031 \cdot 10^3$	1	346.33	$2.3030 \cdot 10^{-4}$	1	330.98	$9.1254 \cdot 10^{-3}$	1
Instancia 22	20	404.46	$1.2034 \cdot 10^3$	1	386.80	$3.3260 \cdot 10^{-4}$	1	383.10	$1.0857 \cdot 10^{-2}$	1
Instancia 23	20	390.42	$1.2033 \cdot 10^3$	1	397.81	$3.8110 \cdot 10^{-4}$	1	360.34	$5.3415 \cdot 10^{-2}$	1
Instancia 24	20	348.45	$1.2047 \cdot 10^3$	1	385.21	$3.0350 \cdot 10^{-4}$	1	339.31	$2.4952 \cdot 10^{-2}$	1
Instancia 25	20	401.43	$1.2043 \cdot 10^3$	1	391.86	$3.8590 \cdot 10^{-4}$	1	363.66	$2.1208 \cdot 10^{-2}$	1
Instancia 26	20	386.12	$1.2033 \cdot 10^3$	1	434.27	$3.8450 \cdot 10^{-4}$	1	399.08	$2.0240 \cdot 10^{-2}$	1
Instancia 27	20	354.09	$1.2047 \cdot 10^3$	1	375.71	$3.7900 \cdot 10^{-4}$	1	355.26	$4.0223 \cdot 10^{-2}$	1
Instancia 28	20	330.75	$1.2042 \cdot 10^3$	1	347.66	$2.8410 \cdot 10^{-4}$	1	328.63	$1.6703 \cdot 10^{-2}$	1
Instancia 29	20	695.25	$1.2039 \cdot 10^3$	2	414.19	$3.5960 \cdot 10^{-4}$	1	392.23	$1.8511 \cdot 10^{-2}$	1
Instancia 30	20	381.59	$1.2048 \cdot 10^3$	1	376.72	$2.6260 \cdot 10^{-4}$	1	369.90	$2.1208 \cdot 10^{-2}$	1

Tabla B.1: Comparación de los tres métodos para las instancias sintéticas de la Sección 5.3 con $d_{MAX} = 400$.

		Modelo exacto			**Vecino más próximo**			***2-opt***		
	$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$
Instancia 1	10	545.02	$1.2034 \cdot 10^3$	2	583.92	$1.5790 \cdot 10^{-4}$	2	567.56	$7.6400 \cdot 10^{-4}$	2
Instancia 2	10	573.73	$1.2021 \cdot 10^3$	2	555.41	$1.8690 \cdot 10^{-4}$	2	555.09	$1.7118 \cdot 10^{-3}$	2
Instancia 3	10	293.68	$1.2033 \cdot 10^3$	1	510.94	$1.9530 \cdot 10^{-4}$	2	503.03	$1.5974 \cdot 10^{-3}$	2
Instancia 4	10	525.83	$1.2019 \cdot 10^3$	2	535.58	$2.0260 \cdot 10^{-4}$	2	528.11	$8.6540 \cdot 10^{-4}$	2
Instancia 5	10	304.04	$1.2023 \cdot 10^3$	1	532.86	$2.6270 \cdot 10^{-4}$	2	519.18	$1.8551 \cdot 10^{-3}$	2
Instancia 6	10	280.37	$1.2034 \cdot 10^3$	1	511.72	$2.1010 \cdot 10^{-4}$	2	488.50	$1.9912 \cdot 10^{-3}$	2
Instancia 7	10	305.34	$1.2022 \cdot 10^3$	1	521.16	$2.0690 \cdot 10^{-4}$	2	513.38	$1.1354 \cdot 10^{-3}$	2
Instancia 8	10	294.55	$1.2022 \cdot 10^3$	1	524.29	$2.0870 \cdot 10^{-4}$	2	512.02	$3.1842 \cdot 10^{-3}$	2
Instancia 9	10	265.97	$1.2026 \cdot 10^3$	1	296.60	$2.1000 \cdot 10^{-4}$	1	274.43	$1.8051 \cdot 10^{-3}$	1
Instancia 10	10	528.79	$1.2028 \cdot 10^3$	2	515.11	$2.0930 \cdot 10^{-4}$	2	507.75	$1.6757 \cdot 10^{-3}$	2
Instancia 11	15	788.59	$1.2032 \cdot 10^3$	3	580.58	$2.6640 \cdot 10^{-4}$	2	567.54	$3.8923 \cdot 10^{-3}$	2
Instancia 12	15	846.00	$1.2028 \cdot 10^3$	3	621.43	$2.5970 \cdot 10^{-4}$	2	606.39	$1.9200 \cdot 10^{-3}$	2
Instancia 13	15	628.97	$1.2030 \cdot 10^3$	2	607.81	$2.1080 \cdot 10^{-4}$	2	594.29	$2.9392 \cdot 10^{-3}$	2
Instancia 14	15	578.53	$1.2040 \cdot 10^3$	2	585.56	$2.9470 \cdot 10^{-4}$	2	576.28	$3.5244 \cdot 10^{-3}$	2
Instancia 15	15	568.39	$1.2028 \cdot 10^3$	2	571.37	$3.6610 \cdot 10^{-4}$	2	564.12	$7.8215 \cdot 10^{-3}$	2
Instancia 16	15	306.83	$1.2023 \cdot 10^3$	1	523.18	$5.4880 \cdot 10^{-4}$	2	510.44	$1.1411 \cdot 10^{-2}$	2
Instancia 17	15	611.27	$1.2020 \cdot 10^3$	2	622.64	$2.9130 \cdot 10^{-4}$	2	615.28	$4.4073 \cdot 10^{-3}$	2
Instancia 18	15	562.13	$1.2026 \cdot 10^3$	2	584.41	$2.2620 \cdot 10^{-4}$	2	561.80	$4.8786 \cdot 10^{-3}$	2
Instancia 19	15	831.08	$1.2027 \cdot 10^3$	3	586.86	$2.0030 \cdot 10^{-4}$	2	583.95	$2.7081 \cdot 10^{-3}$	2

		Modelo exacto			**Vecino más próximo**			***2-opt***		
	$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$
Instancia 20	15	803.95	$1.2029 \cdot 10^3$	3	616.78	$2.3670 \cdot 10^{-4}$	2	587.61	$1.6529 \cdot 10^{-3}$	2
Instancia 21	20	-	-	-	574.17	$1.3362 \cdot 10^{-3}$	2	567.65	$6.8537 \cdot 10^{-3}$	2
Instancia 22	20	-	-	-	579.29	$9.5070 \cdot 10^{-4}$	2	571.61	$3.4052 \cdot 10^{-3}$	2
Instancia 23	20	-	-	-	831.78	$1.8451 \cdot 10^{-3}$	3	809.85	$1.6579 \cdot 10^{-2}$	3
Instancia 24	20	-	-	-	569.26	$8.7410 \cdot 10^{-4}$	2	547.90	$2.8056 \cdot 10^{-2}$	2
Instancia 25	20	-	-	-	608.10	$9.2470 \cdot 10^{-4}$	2	595.57	$6.9831 \cdot 10^{-3}$	2
Instancia 26	20	-	-	-	616.82	$1.5039 \cdot 10^{-3}$	2	615.43	$6.0953 \cdot 10^{-3}$	2
Instancia 27	20	-	-	-	586.39	$1.1722 \cdot 10^{-3}$	2	572.13	$5.5166 \cdot 10^{-3}$	2
Instancia 28	20	557.14	$1.2044 \cdot 10^3$	2	546.55	$4.8530 \cdot 10^{-4}$	2	545.60	$4.9498 \cdot 10^{-3}$	2
Instancia 29	20	-	-	-	819.10	$2.5423 \cdot 10^{-3}$	3	799.21	$1.5520 \cdot 10^{-2}$	3
Instancia 30	20	582.03	$1.2046 \cdot 10^3$	2	609.08	$3.9620 \cdot 10^{-4}$	2	597.49	$4.6097 \cdot 10^{-3}$	2

Tabla B.2: Comparación de los tres métodos para instancias sintéticas de la Sección 5.3 fijando $d_{MAX} = 200$.

		Modelo exacto			**Vecino más próximo**			***2-opt***		
	$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$
Instancia 1	10	0.00	$1.2015 \cdot 10^3$	0	799.15	$2.8240 \cdot 10^{-4}$	3	789.25	$8.9170 \cdot 10^{-4}$	3
Instancia 2	10	568.97	$1.2016 \cdot 10^3$	2	793.48	$2.4200 \cdot 10^{-4}$	3	790.22	$9.9100 \cdot 10^{-4}$	3
Instancia 3	10	534.18	$1.2018 \cdot 10^3$	2	533.91	$2.0270 \cdot 10^{-4}$	2	529.29	$1.5342 \cdot 10^{-3}$	2
Instancia 4	10	536.53	$1.2017 \cdot 10^3$	2	549.41	$1.6060 \cdot 10^{-4}$	2	526.24	$5.9280 \cdot 10^{-4}$	2
Instancia 5	10	517.96	$1.2008 \cdot 10^3$	2	536.43	$1.4350 \cdot 10^{-4}$	2	529.57	$1.3645 \cdot 10^{-3}$	2
Instancia 6	10	486.56	$1.2019 \cdot 10^3$	2	515.26	$2.0500 \cdot 10^{-4}$	2	512.08	$6.9290 \cdot 10^{-4}$	2
Instancia 7	10	513.29	$1.2016 \cdot 10^3$	2	540.70	$1.6790 \cdot 10^{-4}$	2	525.05	$2.7045 \cdot 10^{-3}$	2
Instancia 8	10	500.15	$1.2020 \cdot 10^3$	2	540.25	$2.1170 \cdot 10^{-4}$	2	533.30	$1.0807 \cdot 10^{-3}$	2
Instancia 9	10	266.14	$1.2026 \cdot 10^3$	1	487.33	$1.8860 \cdot 10^{-4}$	2	475.94	$1.4591 \cdot 10^{-3}$	2
Instancia 10	10	526.44	$1.2016 \cdot 10^3$	2	523.60	$2.2850 \cdot 10^{-4}$	2	521.26	$1.2020 \cdot 10^{-3}$	2
Instancia 11	15	779.00	$1.2018 \cdot 10^3$	3	798.60	$1.9350 \cdot 10^{-4}$	3	787.28	$3.8526 \cdot 10^{-3}$	3
Instancia 12	15	-	-	-	1079.57	$3.0300 \cdot 10^{-4}$	4	1073.31	$1.2560 \cdot 10^{-3}$	4
Instancia 13	15	1058.39	$1.2018 \cdot 10^3$	4	820.62	$2.0180 \cdot 10^{-4}$	3	811.90	$1.3017 \cdot 10^{-3}$	3
Instancia 14	15	800.58	$1.2018 \cdot 10^3$	3	793.33	$2.8000 \cdot 10^{-4}$	3	787.48	$2.0904 \cdot 10^{-3}$	3
Instancia 15	15	-	-	-	786.12	$2.0550 \cdot 10^{-4}$	3	777.42	$1.5895 \cdot 10^{-3}$	3
Instancia 16	15	728.82	$1.2027 \cdot 10^3$	3	533.30	$2.9030 \cdot 10^{-4}$	2	523.02	$2.3044 \cdot 10^{-3}$	2
Instancia 17	15	-	-	-	824.32	$1.0948 \cdot 10^{-3}$	3	799.59	$3.8193 \cdot 10^{-3}$	3
Instancia 18	15	788.38	$1.2032 \cdot 10^3$	3	553.45	$2.0750 \cdot 10^{-4}$	2	536.05	$3.0413 \cdot 10^{-3}$	2

		Modelo exacto			**Vecino más próximo**			***2-opt***		
	$\|N\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$	Coste (€)	Tiempo (s)	$\|K\|$
Instancia 19	15	-	-	-	773.93	$1.6587 \cdot 10^{-3}$	3	763.39	$1.6432 \cdot 10^{-3}$	3
Instancia 20	15	-	-	-	1027.20	$9.8480 \cdot 10^{-4}$	4	1008.48	$1.4918 \cdot 10^{-3}$	4
Instancia 21	20	-	-	-	562.96	$3.6170 \cdot 10^{-4}$	2	552.52	$3.3831 \cdot 10^{-3}$	2
Instancia 22	20	-	-	-	1044.09	$3.4820 \cdot 10^{-4}$	4	1033.19	$1.5522 \cdot 10^{-3}$	4
Instancia 23	20	-	-	-	802.55	$3.4670 \cdot 10^{-4}$	3	794.86	$5.4869 \cdot 10^{-3}$	3
Instancia 24	20	-	-	-	765.41	$2.4050 \cdot 10^{-4}$	3	761.92	$9.1246 \cdot 10^{-3}$	3
Instancia 25	20	-	-	-	1029.59	$3.5470 \cdot 10^{-4}$	4	1016.66	$2.1591 \cdot 10^{-3}$	4
Instancia 26	20	-	-	-	1279.72	$3.4880 \cdot 10^{-4}$	5	1276.66	$1.2482 \cdot 10^{-3}$	5
Instancia 27	20	-	-	-	848.12	$3.8080 \cdot 10^{-4}$	3	813.87	$1.4862 \cdot 10^{-3}$	3
Instancia 28	20	-	-	-	557.24	$3.5320 \cdot 10^{-4}$	2	554.85	$6.1889 \cdot 10^{-3}$	2
Instancia 29	20	-	-	-	1062.05	$3.5160 \cdot 10^{-4}$	4	1050.61	$2.6396 \cdot 10^{-3}$	4
Instancia 30	20	-	-	-	785.41	$2.6530 \cdot 10^{-4}$	3	784.47	$2.7698 \cdot 10^{-3}$	3

Tabla B.3: Comparación de los tres métodos para instancias sintéticas de la Sección 5.3 fijando $d_{MAX} = 150$.

Bibliografía

[1] Bazaraa MS, Sherali HD, Shetty CM (2006) Nonlinear Programming: Theory and Algorithms. John Wiley & Sons, 3rd Edition.

[2] Benders JF (1962) Partitioning procedures for solving mixed-variables programming problems. Numerische Mathematik 4:238–252.

[3] Carlsson F, Johansson-Stenman O (2003) Costs and benefits of electric vehicles: A 2010 perspective. Journal of Transport Economics and Policy 37(1):1–28.

[4] Clausen J (1999) Branch and Bound Algorithms – Principles and Examples. Technical Report, Department of Computer Science, University of Copenhagen.

[5] Conejo AJ, Castillo E, Minguez R, Garcia-Bertrand R (2006) Decomposition Techniques in Mathematical Programming. Springer.

[6] Croes GA (1958) A method for solving traveling salesman problems. Operations Research 6:791-812.

[7] Dantzig GB, Ramser JH (1959) The truck dispatching problem. Management Science 6(1):80–91.

[8] Ding N, Prasad K, Lie TT (2017) Environmental noise reduction through the transition from internal combustion engine vehicles (ICEVs) to electric vehicles (EVs): The electric vehicle – a review. International Journal of Electric and Hybrid Vehicles 9(1):49–69.

[9] Fernández E, Roca-Riu M, Speranza MG (2017) The shared customer collaboration vehicle routing problem. European Journal of Operational Research 265(3):1078–1093.

[10] Folium Developers (2025) Folium: Python data, Leaflet.js maps. Disponible en: https://python-visualization.github.io/folium/latest/. Accedido el 4 de diciembre de 2025.

[11] geopy Developers (2025) geopy: Python client for geocoding libraries. Disponible en: https://pypi.org/project/geopy/. Accedido el 4 de diciembre de 2025.

[12] González Díaz J (2021). *Apuntes de la asignatura "Programación lineal y entera"*. Curso 2020–2021, Universidade de Santiago de Compostela.

[13] Google Developers (2025) OR-Tools: Optimization tools for Python. Disponible en: https://developers.google.com/optimization/introduction/python?hl=es-419. Accedido el 4 de diciembre de 2025.

[14] Harahap RF, Sawaluddin (2020) Study vehicle routing problem using Nearest Neighbor Algorithm. Departement of Mathematics, Universitas Sumatera Utara, Medan - Indonesia 20155.

[15] pandas Developers (2025) pandas: Python-based data analysis library. Disponible en: https://pandas.pydata.org/. Accedido el 4 de diciembre de 2025.

[16] Kim G (2024) Electric vehicle routing problem with states of charging stations. Sustainability 16:3439.

[17] López Álvarez FA (2014) Automatización de horarios como un problema de flujo en redes. Trabajo Fin de Máster, Máster en Técnicas Estadísticas, Universidade de Vigo.

[18] Matai R, Singh SP, Mittal ML (2010) Traveling salesman problem: Theory and Applications 1:28.

[19] Ministerio para la Transición Ecológica y el Reto Demográfico (MITECO) (2025) Transporte y emisiones de gases de efecto invernadero en España. Disponible en: https://www.miteco.gob.es/es/cambio-climatico/temas/mitigacion-politicas-y-medidas/transporte.html#emisiones-de-gases-de-efecto-invernadero-correspondientes-al-sector-en-espana. Accedido el 15 de octubre de 2025.

[20] Álvarez Tejero N (2025) TFM: Repositorio del Trabajo Fin de Máster. Disponible en: https://github.com/Nati-design/TFM.git. Accedido el 4 de diciembre de 2025.

[21] NumPy Developers (2025) NumPy: fundamental package for scientific computing with Python. Disponible en: https://numpy.org/. Accedido el 4 de diciembre de 2025.

[22] Olivera A (2004) Heurísticas para Problemas de Ruteo de Vehículos. Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay.

[23] Requia WJ, Mohamed M, Higgins CD, Arain A, Ferguson M (2018) How clean are electric vehicles? Evidence-based review of the effects of electric mobility on air pollutants, ForestGreenhouse gas emissions and human health. Atmospheric Environment 185:64–77.

[24] SCIP Optimization Suite (2025) SCIP: Solving Constraint Integer Programs. Disponible en: https://www.scipopt.org/. Accedido el 4 de diciembre de 2025.