

# Proyecto Fin de Máster Ingeniería Industrial

## Resolución metaheurística de un problema de rutas para ecommerce

Autor: José Juan Candón Hurtado

Tutor: Alejandro Escudero Santana

**Dpto. Organización Industrial y Gestión de Empresas II**  
**Escuela Técnica Superior de Ingeniería**  
**Universidad de Sevilla**

Sevilla, 2020





Proyecto Fin de Máster  
Ingeniería Industrial

# **Resolución metaheurística de un problema de rutas para ecommerce**

Autor:

José Juan Candón Hurtado

Tutor:

Alejandro Escudero Santana

Profesor titular

Dpto. Organización Industrial y Gestión de Empresas II

Escuela Técnica Superior de Ingeniería

Universidad de Sevilla

Sevilla, 2020



Proyecto Fin de Carrera: Resolución metaheurística de un problema de rutas para ecommerce

Autor: José Juan Candón Hurtado

Tutor: Alejandro Escudero Santana

El tribunal nombrado para juzgar el Proyecto arriba indicado, compuesto por los siguientes miembros:

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2020

El Secretario del Tribunal



*A mis padres y a mi hermano*



# Resumen

---

Este Trabajo Fin de Máster presenta una solución para un problema de rutado de vehículos dentro del entorno de comercio electrónico, concretamente para la parte denominada “última milla”. Este problema, llamado 3TW-VRP, tiene como características las propias de un VRPTW con ventanas temporales, pero además se le añade la posibilidad al cliente de seleccionar hasta tres localizaciones diferentes, con ventanas temporales y prioridades asociadas a cada una de estas localizaciones. De esta manera, no solo se busca minimizar los costes de operación, sino que también se busca maximizar la satisfacción del cliente. La solución propuesta en este trabajo se alcanza mediante un algoritmo metaheurístico de optimización: la Búsqueda Tabú combinada con un proceso VNS.



# Abstract

---

This final master's degree project presents a solution to a vehicle routing problem in the e-commerce area, especially to the section known as "the last mile". This problem, regarded as 3TW-VRP, has the characteristics of a VRPTW with time windows, with the addition of 3 possible destinations to be selected by the client, with time windows and priorities allotted to each one of them. This not only attempts to minimize operational costs, but it will also maximize the client's satisfaction. The solution given in this project is achieved by means of a metaheuristic optimization algorithm: Taboo Search combined with a VNS process.



# Índice

---

<b>Resumen</b>	<b>ix</b>
<b>Abstract</b>	<b>xi</b>
<b>Índice</b>	<b>xiii</b>
<b>Índice de Tablas</b>	<b>xv</b>
<b>Índice de Figuras</b>	<b>xvii</b>
<b>1 Introducción</b>	<b>1</b>
1.1. <i>Objetivos</i>	1
1.2. <i>Estructura del trabajo</i>	1
<b>2 Antecedentes</b>	<b>3</b>
2.1. <i>Comercio electrónico</i>	3
2.2. <i>Historia del comercio electrónico</i>	3
2.3. <i>Tipos de comercio electrónico</i>	4
2.4. <i>Ventajas y desventajas del comercio electrónico</i>	5
2.5. <i>Protocolos de reparto en comercio electrónico: VRP</i>	5
2.6. <i>Comercio electrónico en España</i>	7
2.7. <i>Tendencias del comercio electrónico</i>	8
<b>3 Planteamiento del problema</b>	<b>11</b>
3.1. <i>Problema VRP con ventanas temporales: VRPTW</i>	11
3.2. <i>Problema VRP con ventanas móviles: E-commerce VRP</i>	12
3.3. <i>Problema VRP con ventanas temporales y prioridades: 3TW-VRP</i>	12
<b>4 Metodologías de resolución para VRP</b>	<b>15</b>
4.1. <i>Metodología aplicada al problema VRP</i>	15
4.1.1 <i>Métodos exactos</i>	15
4.1.2 <i>Heurísticas</i>	16
4.1.3 <i>Metaheurísticas</i>	18
4.2. <i>Métodos de resolución para el VRPTW</i>	19
<b>5 Búsqueda Tabú</b>	<b>21</b>
5.1. <i>Búsqueda Local</i>	21
5.2. <i>Fundamentos</i>	21
5.3. <i>Algoritmo de Búsqueda Tabú</i>	22

5.4.	<i>Lista tabú</i>	23
<b>6</b>	<b>Implementación</b>	<b>25</b>
6.1.	<i>Definición de la solución</i>	25
6.1.1	Matriz de localizaciones seleccionadas	25
6.1.2	Definición de una ruta	26
6.1.3	Solución: lista de rutas	26
6.2.	<i>Interfaz de lectura de datos</i>	26
6.2.1	Estructura de los datos	26
6.2.2	Lectura de datos	27
6.2.3	Otros tipos de objetos: <i>Customer</i> y <i>Location</i>	28
6.3.	<i>Procedimientos de optimización</i>	28
6.3.1	Generar soluciones	28
6.3.2	VNS	29
6.3.3	Búsqueda Tabú	30
<b>7</b>	<b>Resultados</b>	<b>33</b>
7.1.	<i>Calibrado</i>	33
7.1.1	Proceso de calibración	33
7.1.2	Tamaño de lista tabú	34
7.1.3	Iteraciones de la búsqueda tabú	35
7.1.4	Iteraciones VNS	35
7.2.	<i>Alfa, Beta</i>	36
7.3.	<i>Batería de problemas</i>	38
7.4.	<i>Resolución de los problemas</i>	39
7.5.	<i>Análisis de las soluciones</i>	41
7.6.	<i>Ejemplo de la solución</i>	44
<b>8</b>	<b>Conclusiones</b>	<b>47</b>
8.1.	<i>Desarrollos futuros y aplicaciones</i>	48
	<b>Referencias</b>	<b>49</b>

# ÍNDICE DE TABLAS

---

Tabla 6-1: Matriz de localizaciones seleccionadas para 10 clientes.	25
Tabla 6-2: Solución admisible para 25 clientes.	26
Tabla 6-3: Conjunto de datos con el formato de Solomon.	27
Tabla 7-1: Pruebas para tamaño tabú.	35
Tabla 7-2: Pruebas para número de iteraciones de la búsqueda tabú.	35
Tabla 7-3: Pruebas de kmax para Ejecución 35	36
Tabla 7-4: Pruebas de kmax para Ejecución 36	36
Tabla 7-5: Prueba de Wilcoxon para kmax	36
Tabla 7-6: Resultados obtenidos para $\alpha = 1$ y $\beta = 0$ .	37
Tabla 7-7: Extracto de un problema R2 adaptado al problema de este trabajo	39
Tabla 7-8: Resultados para 25 clientes.	40
Tabla 7-9 : Resultados para 50 clientes.	40
Tabla 7-10: Resultados para 100 clientes	41
Tabla 7-11: Ejemplo solución para problema R2 de Solomon	45



# ÍNDICE DE FIGURAS

---

Figura 2-1: Problema VRP.	6
Figura 2-2: Evolución trimestral del volumen de negocio del comercio electrónico.	7
Figura 2-3: Sectores de actividad con mayor porcentaje de volumen de negocio del comercio electrónico.	8
Figura 2-4: Volumen de negocio del comercio electrónico segmentado geográficamente.	8
Figura 3-1: Ejemplo de 3TW-VRP a resolver con tres clientes.	13
Figura 3-2: Ejemplo de 3TW-VRP con tres clientes resuelto.	13
Figura 3-3: 3TW-VRP con 20 clientes resuelto.	14
Figura 4-1: Combinación de dos rutas.	17
Figura 5-1: Algoritmo Búsqueda Local.	21
Figura 5-2: Algoritmo Búsqueda Tabú.	22
Figura 6-1: Flujo general del algoritmo de optimización.	28
Figura 6-2: Función de generación de soluciones.	28
Figura 6-3: Criterio de ordenación de vector de índices.	30
Figura 6-4: Ejemplo ordenación de 25 clientes.	31
Figura 7-1: Valor obtenido para la función objetivo en función de los pesos.	37
Figura 7-2: Coste obtenido en función de los pesos.	38
Figura 7-3: Prioridad obtenida en función de los pesos.	38
Figura 7-4: Histograma de los resultados obtenidos por la función objetivo (25 clientes).	41
Figura 7-5: Histograma de los costes obtenidos (25 clientes).	42
Figura 7-6: Histograma de las prioridades obtenidas (25 clientes).	42
Figura 7-7: Histograma de los resultados obtenidos por la función objetivo (50 clientes).	42
Figura 7-8: Histograma de los costes obtenidos (50 clientes).	43
Figura 7-9: Histograma de las prioridades obtenidas (50 clientes).	43
Figura 7-10: Histograma de los resultados obtenidos por la función objetivo (100 clientes).	43
Figura 7-11: Histograma de los costes obtenidos (100 clientes).	44
Figura 7-12: Histograma de las prioridades obtenidas (100 clientes).	44



# 1 INTRODUCCIÓN

---

Comprar online es una de las opciones más planteadas a la hora de elegir una forma para adquirir un producto. De hecho, es la principal opción para un gran número de personas. Esto es debido a la rápida adaptación de los comercios electrónicos a las necesidades de los consumidores, mejorando el servicio constantemente y facilitando el acceso a un catálogo de productos cada vez más diverso.

Este crecimiento constante provoca un esfuerzo continuado en las áreas logísticas para mejorar el rendimiento de su operatividad debido a los altos volúmenes que manejan, al alto nivel de competencia existente y al rápido avance de las tecnologías, lo que hace que los consumidores también sean cada vez más exigentes.

En este trabajo se presenta una solución que se centra en mejorar el rendimiento de una de las partes más importantes del área logística: el transporte. En este caso, el problema se centrará en la última parte del proceso de transporte, que es en la que se pone el producto en manos del cliente, más conocida como “última milla”.

## 1.1. Objetivos

Este trabajo propondrá una solución a un problema de rutado de vehículos en un entorno de comercio electrónico con el objetivo de minimizar los costes de transporte. Para ello se utilizarán metodologías metaheurísticas de optimización.

Además, teniendo en cuenta que hoy en día con el auge de los comercios electrónicos, uno de los valores más importantes de estas empresas son la focalización en el cliente, en este trabajo se han añadido características que permiten adaptar el problema a dicho entorno y centrarse en el cliente. Con estas características se persigue que la resolución del problema maximice la satisfacción del cliente a la vez que minimiza costes de transporte. Por tanto, es importante destacar que otro objetivo principal de este trabajo es maximizar la satisfacción del cliente a la hora de resolver este tipo de problemas.

## 1.2. Estructura del trabajo

El presente trabajo está articulado en ocho partes con el propósito de cumplir los objetivos descritos anteriormente. A continuación, se describirá el contenido de cada capítulo.

En el presente capítulo, con intención introductoria, se describen los objetivos principales que este trabajo trata de alcanzar y la estructura propuesta.

En el segundo capítulo se presenta el comercio electrónico exponiendo su procedencia, sus tipos y las ventajas que supone. Además, se introduce al problema de rutado de vehículos (VRP) como solución para los protocolos de reparto empleados y un breve análisis del comercio electrónico en España acompañado de las tendencias actuales.

Con el tercer capítulo se pretende establecer las características del problema que plantea este trabajo. Partiendo del problema de rutado con ventanas temporales (VRPTW) para establecer las bases del problema relacionado con el comercio electrónico, se definen las restricciones y características del nuevo problema (3TW-VRP),

pasando por la definición del problema ya existente (EC-VRP) que sirve como punto de partida.

El propósito del cuarto capítulo es presentar la metodología utilizada para la resolución de los problemas VRP para dar paso al quinto capítulo en el que se presenta el método metaheurístico que se utilizará para resolver el problema del trabajo. En concreto, la búsqueda tabú será la metodología utilizada y se profundizará en sus características.

A lo largo del sexto capítulo se describe la hoja de ruta seguida para la implementación de la metodología propuesta con el objetivo de alcanzar una solución para el problema. En este capítulo se definen la forma de presentar la solución, la interfaz de lectura de los datos y los procedimientos de optimización que obtienen la solución. Estos se presentan conceptualmente para luego ser implementados en Python.

En el capítulo séptimo se realiza una calibración del modelo, es decir, se definen los valores de los parámetros que utiliza el modelo para resolver el problema. También se analiza la incidencia de la variación de los parámetros de peso en la función objetivo sobre los resultados. A continuación, se realizará una serie de pruebas resolviendo diferentes baterías de problemas para poder analizar las soluciones obtenidas y alcanzar las conclusiones que se detallarán en el último capítulo.

## 2 ANTECEDENTES

---

*El comercio electrónico, también conocido como e-commerce, consiste en la compra y venta de productos o de servicios a través de medios electrónicos, tales como redes sociales y otras páginas web.*

(Wikipedia, s.f.)

### 2.1. Comercio electrónico

El término se aplicaba originalmente a transacciones que se realizaban mediante cualquier medio electrónico. Con el avance de internet, a mediados de la década de 1990, comenzó a hacer referencia a la venta de productos y servicios por internet. La evolución de internet ha propiciado el crecimiento del comercio electrónico de manera exponencial.

### 2.2. Historia del comercio electrónico

Como señaló Adam Smith (Smith, 1776), el inicio de la economía fue hace ya más de 10.000 años con la aparición del trueque, intercambio de productos entre humanos sin mediar dinero, como método de pago y fue en 2.500 a.C. cuando se empezaron a utilizar los metales preciosos como forma de intercambio. La necesidad de los hombres para ser autosuficientes propició la aparición del intercambio de bienes y servicios que no podían fabricar o conseguir en sus comunidades para asegurar la subsistencia del ser humano y permitir el desarrollo de la humanidad. Las primeras monedas aparecen hacia el año 600 a.C. por la necesidad de establecer un medio de pago común para todos, aunque fue alrededor del siglo IX cuando aparecen los primeros billetes en China para luego extenderse por todo el mundo.

La primera aparición de negocio a distancia e inicio de lo que hoy se conoce como comercio electrónico fue hacia 1920 en los Estados Unidos donde surgió la venta por catálogo. La venta por catálogo mostraba a los clientes finales los productos mediante fotos ilustrativas. Esto supuso un gran avance para los consumidores, permitiendo comprar sin ver antes el producto y estableciendo la base logística del comercio de hoy en día. Este tipo de venta fue creado por comerciantes que ofrecían productos para quienes no tenían facilidad para desplazarse. El cliente seleccionaba el producto y mandaba las indicaciones por correo postal al proveedor para que le enviara los productos seleccionados a su domicilio.

En 1960 se inventó en Estados Unidos un método de Intercambio de Datos Electrónicos (EDI) que marca un hito en la evolución del comercio electrónico ya que permitió a las empresas realizar transacciones electrónicas e intercambio de información comercial.

En la década de los años 1970 se introdujo el uso de una computadora para realizar acciones comerciales, aunque ofrecían un servicio muy limitado.

En la década de 1980 se impulsó la venta por catálogo gracias a la televisión, lo que se denominó como “televenta”. La televisión dotó de más realismo a la venta por catálogo permitiendo exhibir los productos resaltando sus atributos y características. Este tipo de venta se realizaba mediante llamadas telefónicas y pagando con las tarjetas de crédito.

En 1981 se realiza la primera transacción online entre empresas (B2B). Una agencia de viaje conectó a todos sus agentes para homogeneizar catálogos y ofrecer reservas inmediatas a sus clientes. Esto permitía a los agentes conocer de forma inmediata la oferta de viajes para ofrecer a los clientes.

En 1984 una señora de Inglaterra compra desde su casa mantequilla, cereales y huevos a un supermercado solo con el mando a distancia de su televisor, lo que se considera como la primera venta online de empresa a consumidor (B2C).

Un año decisivo para el desarrollo de la tecnología y el avance del comercio electrónico fue 1989 cuando apareció la World Wide Web (WWW) de la mano de un trabajador del CERN, Tim Berners-Lee. Este nuevo método de transmisión de información renovó por completo la forma de comunicación y de comercializar, considerándose uno de los mayores hitos en la historia del comercio electrónico. En 1991 la NSF (National Science Foundation) permite que la red se utilice con objetivos comerciales.

Los primeros productos en venderse de manera oficial a través de Internet fueron un libro en 1992 de Stacks Unlimited, la primera librería online, y una pizza dos años después vendida por Pizza Hut, que suponía entrar en el mundo de los productos perecederos y con necesidad de entrega en un corto espacio de tiempo.

Otro hito para el comercio electrónico se trata de la aparición de SSL (Secure Sockets Layer) creado por Netscape para introducir la posibilidad de transferir datos e información de carácter personal de una forma totalmente segura.

A finales de la década de 1990 el comercio electrónico creció de manera significativa debido a la aparición de portales dedicados exclusivamente a esta actividad, ejemplo de ello son empresas tales como eBay o Amazon, multinacionales que siguen creciendo en la actualidad añadiendo nuevos servicios, ampliando productos, sistemas de reparto o almacenamiento. En 1997, la empresa Coca-Cola permite realizar compras y recibir en casa cualquiera de sus productos mediante SMS. Este se considera el primer salto del comercio electrónico al teléfono móvil. El servicio streaming de música, que hoy se conoce por iTunes o Spotify, vio sus inicios en 1998 con Ritmoteca. En esta década también surge el primer banco online, Presidential Bank, y el primer comparador de precios, Bargain Finder.

El vertiginoso avance del comercio electrónico en el siglo XXI se ha visto reflejado en la gran cantidad de compañías que han aumentado sus ventas gracias a la venta online. En 2007 el comercio electrónico tuvo un crecimiento del 25% respecto al año anterior. Más de 106 millones de compradores online se gastaron 180.000 millones de dólares. Más tarde, en 2012, estas cifras fueron superadas alcanzando las ventas online el billón (billón europeo) de dólares. En 2014 se establece el auge de las empresas que hoy más se conocen: Amazon, Aliexpress o Ebay, que han popularizado las compras a través de Internet. A pesar de este crecimiento, la inseguridad de los usuarios a pagar con tarjetas de créditos ha estado latente desde sus inicios, lo que propició el nacimiento de métodos de pagos alternativos más seguros como Trustly, Paypal, Google Wallet o Apple Pay.

En los últimos años se ha percibido la tendencia de sustituir el comercio tradicional por el comercio electrónico. La mejora continua de la infraestructura y la logística que exige este comercio ha permitido impulsar las ventas online, llegando en 2019 a los 2 billones (billón europeo) de dólares.

### 2.3. Tipos de comercio electrónico

- *Business to business* (B2B)

Es el comercio en el que se realiza una transacción comercial entre dos negocios sin la intervención de los consumidores. Este tipo de comercio es muy común cuando la tienda vende al por mayor o distribuye componentes o materias primas. Se necesita experiencia del mercado en el que se opera. Esta operación es un paso previo con el objetivo de vender a un consumidor.

- *Business to Consumer* (B2C)

La transacción comercial se lleva a cabo entre el negocio y el consumidor final, interesado en comprar un producto o adquirir un servicio. Es el tipo de comercio electrónico que realiza la mayoría de las tiendas online.

- *Business to Employee* (B2E)

Es el comercio dado entre una empresa y sus empleados. El objetivo de este comercio es la fidelización del empleado ofreciéndole precios especiales a través de su tienda online.

- *Consumer to Consumer (C2C)*

La transacción comercial es realizada entre dos consumidores. Uno de ellos ofrece en venta algún producto que ya no utiliza para que otro consumidor pueda reutilizarlo y encontrar productos con precios reducidos.

- *Government to Consumer (G2C)*

Este tipo de comercio se da cuando un gobierno municipal, estatal o federal permite que los ciudadanos realicen trámites en línea a través de un portal en internet. En la mayoría de las ocasiones se usa para pagar un trámite y acceder a la información en línea, por ello es considerado un tipo de comercio electrónico.

## 2.4. Ventajas y desventajas del comercio electrónico

El comercio electrónico presenta numerosas ventajas frente al comercio tradicional. La oportunidad que ofrece a los vendedores para tener un canal adicional de ventas, abrir nuevos mercados, promocionar productos o ampliar las posibilidades de negocio es una de las principales ventajas. Otras de las ventajas son:

- El coste de inicio y mantenimiento es menor que en el comercio tradicional. Al no requerir tienda física, el coste de la inversión es menor y se reducen costes de producción, administración y almacenamiento.
- Alcance geográfico, a nivel nacional e internacional.
- Mayor visibilidad gracias a internet que propicia la captación de clientes.
- Optimización del tiempo dedicado al negocio y a la atención al cliente por parte del vendedor.
- Mayor facilidad y rapidez para encontrar productos por el consumidor.
- El consumidor puede comprar en cualquier momento del día, desde casa y sin colas. Esto ahorra tiempo a la hora de realizar compras.
- Ofrece mucha más información al consumidor y permite una fácil comparación de productos.
- Ofrece métodos de pago flexibles.

Sin embargo, también existen desventajas en el comercio electrónico. A tener en cuenta:

- No existe comunicación física con el cliente.
- La competencia es mayor.
- Inseguridad del consumidor por no poder ver el producto o no confiar en los pagos online.
- Los gastos de envío incurridos son altos si el volumen de negocio es pequeño.
- La fidelización del cliente es más difícil, por lo que se requiere una buena estrategia profesional.
- Se debe proporcionar seguridad a los consumidores en el sitio online.
- Necesario un sistema eficaz de envío y devolución. Esto supone un reto logístico para las empresas de paquetería.

## 2.5. Protocolos de reparto en comercio electrónico: VRP

La logística en el comercio electrónico es la red de medios, métodos e infraestructuras combinadas para garantizar el almacenamiento, el transporte y la entrega de los bienes o servicios correspondientes a los pedidos demandados por el consumidor online. Este apartado estará centrado la parte correspondiente al transporte.

El transporte es un eslabón esencial en la cadena del comercio electrónico, pues no se finaliza la transacción hasta que el producto está en manos del consumidor. Los transportistas son los encargados de recoger el producto en el almacén y entregarlos a los clientes. La exigencia de los consumidores y la personalización, rapidez, flexibilidad y precisión de las compras hacen que la logística sea una clara ventaja competitiva debido a su influencia directa en la satisfacción y fidelización de los clientes. Es por esto por lo que las empresas de transporte se han desarrollado más durante los últimos años.

El objetivo principal de satisfacer al cliente se ve mermado si el producto no es entregado a tiempo o cuando el

cliente espera. Por ello, las empresas de reparto están introduciendo flexibilidades en sus servicios para facilitar la entrega en el primer intento y así adaptarse a las necesidades del cliente. De lo contrario, la satisfacción del cliente se vería muy reducida, además de los incrementos en los costes de volver a repartir el producto.

Ante la posible no entrega en el primer intento del producto, los principales inconvenientes del cliente son:

- El tiempo empleado para la gestión de la segunda entrega. Aunque las empresas de paquetería deben ofrecer a sus clientes un segundo intento de entrega, es usual que los clientes deban gestionar esta segunda entrega.
- El tiempo empleado en el caso de que deba desplazarse a la sucursal a recoger el producto.

Los motivos más comunes de las devoluciones del envío al almacén son:

- Destinatario ausente.
- Dirección errónea o incompleta.
- En envíos con reembolso, el destinatario no tiene dinero en efectivo.
- Rechazo del cliente.

El motivo principal por el que los repartos no se entregan es la ausencia del destinatario. Por esto, las empresas de transporte deben ofrecer servicios flexibles para asegurar que el cliente esté en su domicilio durante el primer intento de entrega. Además, las empresas de transporte deben cumplir las franjas horarias indicadas al destinatario. Estos servicios ofrecidos están en continuo desarrollo, incorporando cada vez más herramientas de flexibilidad de cara al cliente como, por ejemplo, la elección por parte del destinatario de la franja horaria para la entrega.

Con objeto de cumplir las necesidades del cliente, teniendo en cuenta las indicaciones de éste, y la optimización de los recursos de la empresa de reparto se utilizan modelos matemáticos para crear las rutas óptimas de los transportistas.

El problema VRP (Vehicle Routing Problem) o Problema de Rutado de Vehículos es un modelo utilizado para alcanzar este objetivo. El problema VRP consiste en diseñar rutas óptimas de reparto desde uno o varios depósitos (depot) hasta los clientes, dispersos geográficamente (nodos), cumpliéndose ciertas restricciones. De otra manera, el objetivo es repartir mercancía a una serie de clientes con demandas conocidas minimizando los costes y volviendo al punto de partida. Este problema es parte de la investigación de operaciones, donde se manejan restricciones para reflejar con mayor realismo la operación de los vehículos. Por esto, este problema resulta muy útil en una gran variedad de problemas reales relacionados con el transporte, la logística y la distribución.



Figura 2-1: Problema VRP. Fuente: Javier Távora Montero, Modelo de rutas de vehículos aplicado al e-commerce.

Existen diversas variantes de este problema. Múltiples restricciones se incorporan al problema para dar a lugar a cada una de ellas. Algunas son:

- VRP con capacidad limitada (CVRP): Problema VRP con restricción de capacidad para cada vehículo.

- VRP con ventanas de tiempo (VRPTW): Problema VRP en el que se debe realizar la entrega dentro de un intervalo de tiempo, considerado como ventana temporal.
- VRP con múltiples depósitos (MDVRP): Problema VRP en el que existen más de un depósito.
- VRP con demanda estocástica (SVRP): Problema VRP en el que los nodos de destino pueden tener una probabilidad de presencia.
- VRP de ida y vuelta (VRPB): Problema VRP en el que los clientes también pueden demandar un servicio de recogida.

## 2.6. Comercio electrónico en España

El comercio electrónico B2C es el más habitual. Gracias al comercio electrónico comprar o vender es cada vez más sencillo y rápido, lo que ha modificado los hábitos de consumo. En España, la Comisión Nacional de los Mercados y la Competencia (CNMC) publicó un informe en 2019 (CNMC, 2019) en el que realizaba un estudio sobre las ventas online hasta la fecha. Este estudio muestra resultados como los sectores que más han crecido gracias a esta actividad, el volumen de negocio a nivel geográfico o las actividades que concentran más ingresos.

En tan solo el primer trimestre de 2019, el e-commerce superó en España los 10.900 millones de euros, lo que suponía un 22.2% más que el año anterior.

EVOLUCIÓN TRIMESTRAL DEL VOLUMEN DE NEGOCIO DEL COMERCIO ELECTRÓNICO Y VARIACIÓN INTERANUAL (millones de euros y porcentaje)

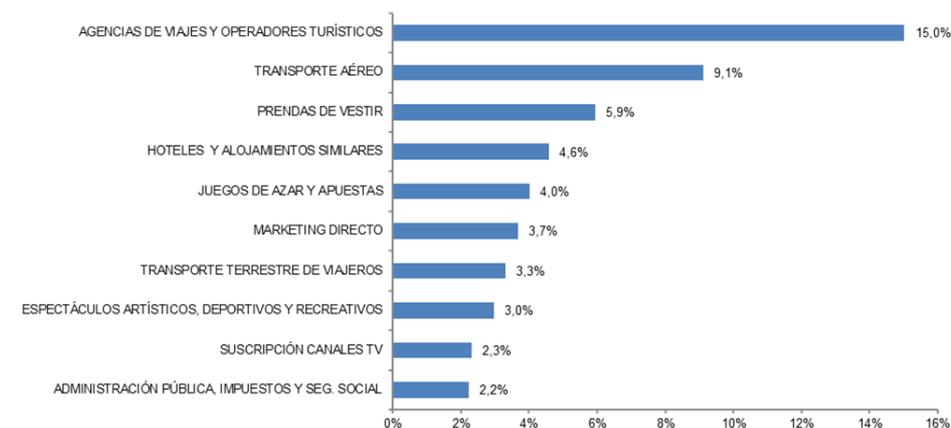


Fuente: CNMC

Figura 2-2: Evolución trimestral del volumen de negocio del comercio electrónico. Fuente: CNMC

Entre los sectores de actividad con mayor volumen de ingresos destacan el sector de las agencias de viajes y operadores turísticos, el transporte aéreo o el textil.

LAS DIEZ RAMAS DE ACTIVIDAD CON MAYOR PORCENTAJE DE VOLUMEN DE NEGOCIO DEL COMERCIO ELECTRÓNICO (I-19, porcentaje)



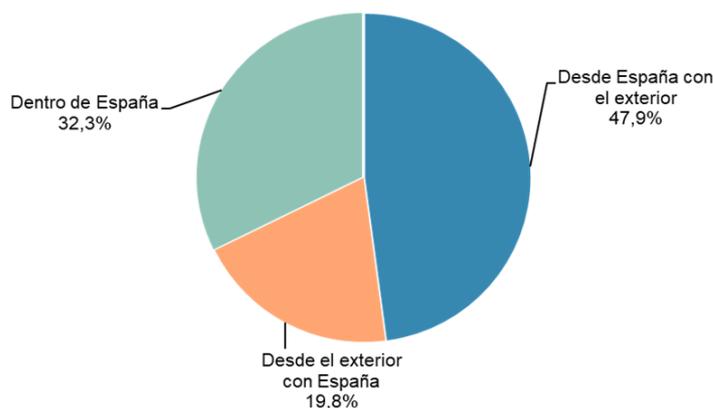
Fuente: CNMC

Figura 2-3: Sectores de actividad con mayor porcentaje de volumen de negocio del comercio electrónico.

Fuente: CNMC

Más de la mitad de los ingresos (52,15%) del primer trimestre de 2019 acabaron en webs de territorio nacional, mientras que el 47,9% acabaron en webs extranjeras.

VOLUMEN DE NEGOCIO DEL COMERCIO ELECTRÓNICO SEGMENTADO GEOGRÁFICAMENTE (I-19, porcentaje)



Fuente: CNMC

Figura 2-4: Volumen de negocio del comercio electrónico segmentado geográficamente. Fuente: CNMC

Los últimos datos disponibles corresponden al segundo trimestre del año 2019, que corresponden a 12.000 millones de euros, un incremento del 28% con respecto al mismo periodo del año anterior y un 9,4% con relación al primer trimestre. Este crecimiento demuestra la tendencia alcista que hace previsible la escalada de ventas online durante los próximos años.

## 2.7. Tendencias del comercio electrónico

Actualmente, el comercio electrónico se ha convertido en el método de compra preferido por los consumidores, especialmente para lo más jóvenes. La rapidez con la que crece el e-commerce va ligada al avance de la tecnología, que hace posible el desarrollo de los negocios online. Las tendencias del comercio electrónico en los próximos años pasan por la realidad aumentada, el uso de la inteligencia artificial o el pago mediante criptomonedas:

- *Realidad aumentada y realidad virtual*: El objetivo de esta tecnología es hacer ver a los compradores una idea más exacta del producto sin tener que esforzarse en imaginarse como quedará. Permite a los usuarios integrar los productos en su entorno, de modo que puedan sentir que tienen el producto en sus manos. Un ejemplo de ello es la aplicación lanzada por Ikea, en la que los consumidores pueden ver cómo quedan los muebles en su casa.
- *5G*: Cada vez son más las transacciones que se realizan a través del móvil. Los usuarios encuentran más productos en las aplicaciones que en el navegador móvil. Se prevé que durante 2020 las redes 5G estén disponibles, lo que permitirá ampliar las posibilidades del sector gracias a la velocidad de descarga para aumentar el contenido audiovisual y de realidad aumentada.
- *Inteligencia artificial y machine learning*: La inteligencia artificial permite analizar los comportamientos de los usuarios obteniendo información a partir de bases de datos o sensores. El avance a pasos agigantados de esta tecnología permitirá a los negocios online ofrecer una mejor experiencia a los compradores basándose en sus hábitos de compra o artículos más visitados.
- *Voice commerce*: En 2019 más del 50% de las búsquedas por internet se realizaron por voz, por lo tanto, es importante que los negocios de comercio electrónico estén preparados para facilitar las compras mediante voz.
- *Criptomonedas como método de pago*: Las criptomonedas y el blockchain son tecnologías en auge. Muchos comercios electrónicos incorporarán como método de pago el pago mediante criptomonedas, marcando diferencias y ganando a público objetivo.
- *Utilizar vídeos*: El uso de vídeos ilustrativos de los productos por los vendedores genera más confianza entre los clientes.
- *Tiendas probador*: Tener tiendas probador donde el cliente podrá ver, tocar y probarse los productos antes de comprarlos como nuevo modelo de negocio será una tendencia para tener en cuenta.



# 3 PLANTEAMIENTO DEL PROBLEMA

---

En el capítulo anterior se introduce el problema VRP como un modelo utilizado para solucionar los problemas de transporte y optimizar las rutas de los repartidores, de manera que se genera una serie de rutas que satisfacen de una sola vez el mayor número de pedidos posibles al menor coste.

En este capítulo se detallará el problema objeto de resolución, una nueva variante, el *Three Time Windows Vehicle Routing Problem* (3TW-VRP). En primer lugar, se explicará de manera detallada el problema del que parte: la variante VRP con ventanas temporales (VRPTW). A continuación, se comentará el problema EC-VRP intermedio para acabar detallando el problema 3TW-VRP.

## 3.1. Problema VRP con ventanas temporales: VRPTW

El problema introducido en (Cordone & Calvo, 2001), además de la restricción del problema VRP original, en el que cada ruta parte y acaba en una misma localización denominada depósito o depot, se incorpora la restricción para cumplir una estricta ventana temporal, dado en forma de intervalo  $[e_i, l_i]$ . El vehículo debe servir el pedido al nodo  $i$  en un instante de tiempo anterior a  $l_i$  y no antes de  $e_i$ . Aunque es posible llegar antes del instante  $e_i$ , el vehículo deberá esperar hasta que se alcance dicho instante para comenzar el servicio. Cuando comienza el servicio, se considera un *tiempo de servicio*  $s_i$ . Este problema también incorpora una restricción de capacidad para los vehículos, por lo que no podrán sobrepasar una determinada capacidad de carga,  $Q$ .

A continuación, se presenta la formulación matemática del problema VRPTW introducida por Cordone & Calvo en (Cordone & Calvo, 2001):

Sea el grafo  $G = (N, A)$ , siendo el conjunto de nodos  $N = \{0, 1, \dots, n\}$  y el conjunto de arcos  $A = \{(i, j) : i, j \in N, i \neq j\}$ . El nodo 0 representa el depósito o *depot* y  $N' = \{1, \dots, n\}$  representa los nodos que van a ser servidos. Cada arco  $(i, j)$  lleva asociado un coste por recorrido  $c_{ij} \geq 0$  y un tiempo de viaje  $t_{ij} \geq 0$ . Cada uno de los nodos  $i$  lleva asociado una demanda  $q_i$ , un tiempo de servicio  $s_i$ , y una ventana temporal  $[e_i, l_i]$ . La flota de vehículos es homogénea, por lo que todos tienen una capacidad  $Q$  y el mismo coste por el hecho de usarlo  $h \geq 0$ . Se debe estrechar la ventana temporal estableciendo  $e_i = \max(e_0 + t_{0i}, e_i)$  y  $l_i = \min(l_0 - t_{0i}, l_i)$ .

Se establece como  $x_{ij} = 1$  si se usa el arco  $(i, j)$  y  $x_{ij} = 0$  en otro caso;  $p_i$  el inicio del servicio en el nodo  $i$ ;  $y_i$  la carga del vehículo al abandonar el nodo  $i$ . La formulación del problema VRPTW queda:

$$\min \sum_{j \in N'} h x_{0j} + \sum_{j \in N'} c_{ij} x_{ij} \quad (1)$$

sujeto a

$$\sum_{j \in N} x_{ij} = 1 \quad \forall i \in N' \quad (2)$$

$$\sum_{i \in N} x_{ij} = 1 \quad \forall j \in N' \quad (3)$$

$$si \ x_{ij} = 1 \rightarrow p_i + s_i + t_{ij} \leq p_j \quad \forall (i, j) \in A \quad (4)$$

$$e_i \leq p_i \leq l_i \quad \forall i \in N' \quad (5)$$

$$si \ x_{ij} = 1 \rightarrow y_i + q_j \leq y_j \quad \forall (i, j) \in A \quad (6)$$

$$q_i \leq y_i \leq Q \quad \forall i \in N' \quad (7)$$

$$x_{ij} \in \{0,1\} \quad \forall (i, j) \in A \quad (8)$$

La función objetivo (1) es minimizar el coste total de desplazamiento, incluyendo el coste de usar el vehículo. Las restricciones (2) y (3) aseguran que se sirve cada cliente en una sola ruta. Las ventanas temporales se aseguran en las restricciones (4) y (5), mientras que las restricciones (6) y (7) son las de capacidad.

### 3.2. Problema VRP con ventanas móviles: E-commerce VRP

En 2017, Carlos Uribe introduce el problema *E-commerce VRP* (EC-VRP) en (Uribe & Escudero-Santana, 2017). Este problema tiene como base el problema VRP con ventanas temporales (VRPTW).

Teniendo en cuenta las características del problema VRPTW:

- Existe una serie de clientes dispersados geográficamente.
- Se debe satisfacer los pedidos de todos los clientes.
- Se dispone de una flota homogénea de vehículos, con las mismas características.
- La capacidad de los vehículos no es limitante, los productos se consideran de tamaño pequeño. Esto hace que un solo vehículo pueda proveer a todos los clientes de sus pedidos.
- Se supone una cantidad suficiente de vehículos para poder servir a todos los clientes.
- El depósito de salida y llegada de los vehículos tiene asociada una ventana temporal.

Para flexibilizar el servicio de reparto y mejorar la satisfacción del cliente, el problema EC-VRP incorpora la posibilidad de que el cliente pueda seleccionar tres localizaciones distintas para la entrega del pedido, teniendo cada una de estas localizaciones una ventana temporal conocida como intervalo de tiempo, en el que podrá ser entregado el pedido en dicha localización. Esto permite al cliente que cambia su situación a lo largo del día seleccionar hasta tres lugares diferentes para la entrega, asegurando que la entrega se realice en el primer intento. Si el cliente ha sido atendido en una de sus posibles localizaciones, las otras dejan de estar activas.<sup>1</sup>

### 3.3. Problema VRP con ventanas temporales y prioridades: 3TW-VRP

Este trabajo propone la incorporación de la posibilidad de establecer prioridades a cada una de las hasta tres localizaciones (con sus respectivas ventanas temporales) que indique el cliente. Estas prioridades se indicarán al establecer las localizaciones en el momento de realizar el pedido. Puede ser que el cliente indique como localizaciones su domicilio y su oficina, pero prefiera que le llegue al domicilio. Esta opción permitirá tener en cuenta la prioridad a la empresa distribuidora de manera que pueda maximizar la satisfacción del cliente.

El cliente podrá indicar la prioridad asignando un valor numérico de entre los valores “1”, “2” y “3” a cada

<sup>1</sup> Uribe introduce el problema EC-VRP con un ejemplo en su trabajo “Resolución de un VRP con ventanas móviles mediante el uso del algoritmo Cuckoo Search” en las páginas 17-18.

localización: el valor “1” indica la mayor prioridad y el valor “3” la menor. El cliente puede establecer la misma prioridad a más de una localización. Se asignará la prioridad siempre partiendo del valor “1”. Ejemplos:

- Localización 1: prioridad 1. Localización 2: prioridad 1. Localización 3: prioridad 2. (Máxima prioridad: localizaciones 1 y 2)
- Localización 1: prioridad 1. Localización 2: prioridad 2. Localización 3: prioridad 2. (Máxima prioridad: localización 1)
- Localización 1: prioridad 1. Localización 2: prioridad 1. Localización 3: prioridad 1. (Máxima prioridad: todas las localizaciones)
- Localización 1: prioridad 1. Localización 2: prioridad 2. (El cliente solo indica dos localizaciones, dando máxima prioridad a la localización 1)

El siguiente ejemplo cuenta con tres clientes, donde cada cliente establece tres localizaciones con sus ventanas temporales y las prioridades. En la figura puede observarse la localización y la prioridad.

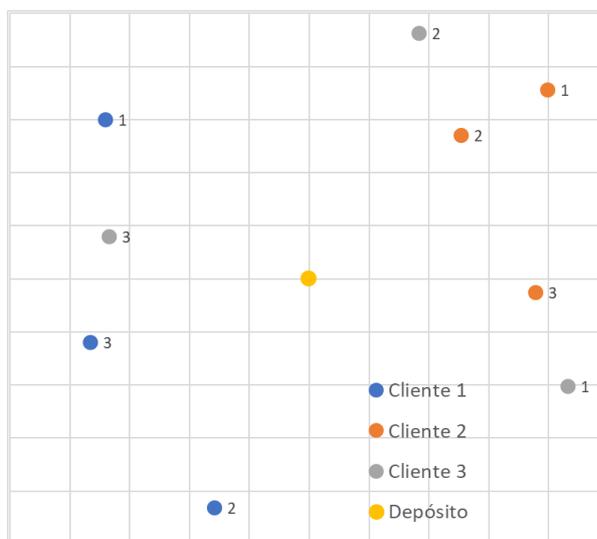


Figura 3-1: Ejemplo de 3TW-VRP a resolver con tres clientes.

En la figura siguiente se propone una solución al problema anterior.

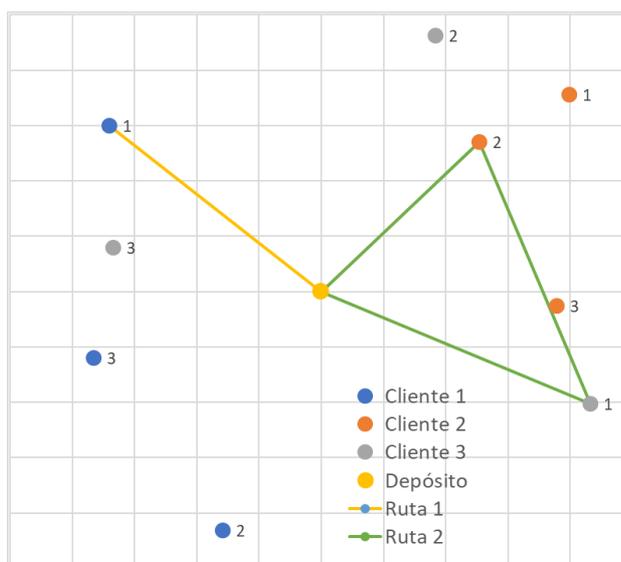


Figura 3-2: Ejemplo de 3TW-VRP con tres clientes resuelto.

A continuación, se muestra la solución a un problema que cuenta con 20 clientes, en el que intervienen varios vehículos, haciendo cada uno de estos una ruta diferente con el objetivo de entregar todos los pedidos a los clientes minimizando el coste total, cumpliendo las ventanas temporales y maximizando la satisfacción del cliente gracias a las prioridades establecidas.

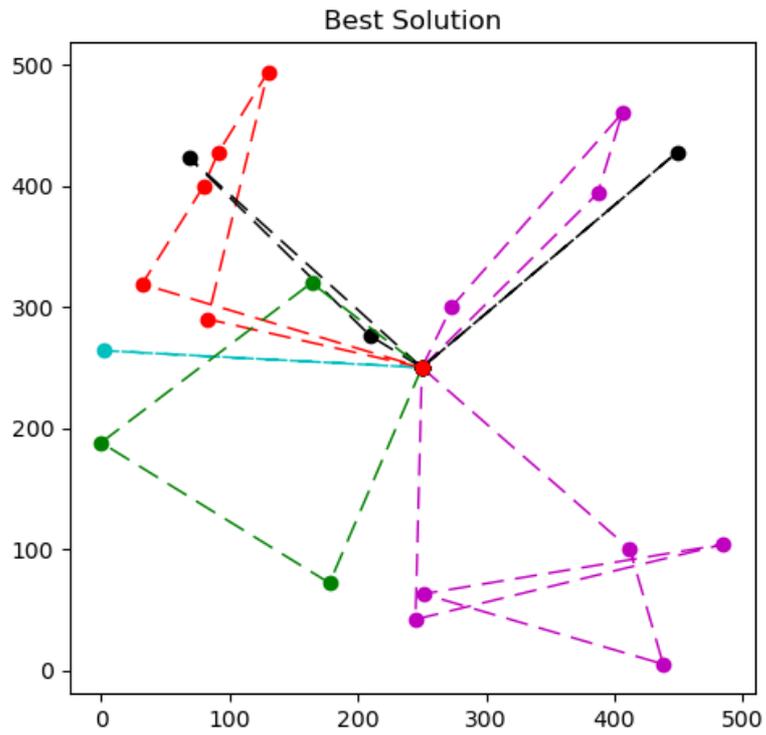


Figura 3-3: 3TW-VRP con 20 clientes resuelto.

# 4 METODOLOGÍAS DE RESOLUCIÓN PARA VRP

---

**E**n este capítulo se presenta la diversa metodología existente para la resolución de un problema VRP y los métodos más utilizados en la resolución de los problemas tipo VRPTW.

## 4.1. Metodología aplicada al problema VRP

La literatura aborda tres grandes categorías, agrupadas de la siguiente manera: métodos exactos, heurísticas y metaheurísticas.

### 4.1.1 Métodos exactos

Procedimiento que busca determinar todas las soluciones posibles hasta que se alcance la mejor. Estos métodos parten de una formulación como modelos de programación lineal y llegan a una solución factible gracias a algoritmos de acotamiento (Lüer, Benavente, Bustos, & Venegas, 2009). Los métodos exactos son eficientes en problemas de hasta 50 depósitos debido a restricciones de tiempo computacional. La clasificación para los métodos exactos se establece en tres grupos: búsqueda directa de árbol, programación dinámica y programación lineal y entera (Laporte, 1991).

#### Métodos de Búsqueda Directa de Árbol

La búsqueda se realiza sobre todos los nodos de un árbol de acuerdo con criterios específicos propios de cada método. Los métodos de búsqueda directa de árbol conocidos son: Asignación de cota inferior, algoritmo de ramificación y acotamiento, algoritmo de ramificación y corte y algoritmo de búsqueda de árbol.

- *Branch and bound (Ramificación y acotamiento)*

Este método se ha usado en los últimos años para resolver el CVRP y otras variantes. En muchas ocasiones, para el CVRP asimétrico y las distancias con restricciones (Toth & Vigo, 2002).

Con este método se pueden introducir variaciones realistas, como costes variables, ventanas temporales o flota heterogénea.

Se obtiene un conjunto de soluciones en subproblemas y luego se optimiza independientemente cada subproblema. Esto es, el problema está dividido en dos procesos: la ramificación del problema y las soluciones factibles de cada rama que conforman las soluciones del problema original.

En este método se construye un árbol con todas las posibles soluciones hasta llegar al punto en que una de las ramas ya no sea la mejor. Para acotar la solución se usan las cotas inferiores obtenidas. De este modo no se resuelven subproblemas que no mejorarán la solución actual.

## Programación Dinámica

Eilon, Watson-Gandy y Christofides proponen este método en 1971 (Eilon, Watson, Christofides, & de Neufville, 1974). Estos métodos aceleran los cálculos mediante una relajación del espacio de estados.

## Programación Lineal y Entera

Las técnicas que se utilizan en este grupo son: método de particiones y generaciones de columnas (Balinski & Quandt, 1964), formulación de flujo de vehículos de tres índices y formulación de flujo de vehículos de dos índices (Fisher & Jaikumar, 1981).

En el método de particiones y generaciones de columnas se considera un conjunto factible de rutas y un coeficiente binario que es igual a uno si un determinado depósito pertenece a una ruta. En la solución óptima, existe una variable binaria que es igual a uno si la ruta pertenece a ella. La solución se obtiene resolviendo un TSP.

Por otro lado, los métodos de flujo de vehículos de dos y tres índices se desarrollaron para el problema CVRPTW. Para el problema de dos índices, la variable  $X_{ij}$  representa el camino entre el depósito  $i$  y el depósito  $j$ . En los problemas de tres índices, la variable  $X_{ijk}$  representa el camino entre el depósito  $i$  y el depósito  $j$  utilizando el vehículo  $k$ . La solución óptima está garantizada en un número finito de iteraciones.

### 4.1.2 Heurísticas

Los métodos heurísticos consisten en algoritmos que permiten obtener soluciones factibles de buena calidad con tiempos menores de ejecución, pero sin llegar a la solución óptima. En 1964, fueron Clarke y Wright quienes propusieron por primera vez un algoritmo que resultó efectivo para resolver el VRP (Clarke & Wright, 1964). La mayoría de las heurísticas conocidas fueron estudiadas entre 1960 y 1990.

Los métodos heurísticos pueden clasificarse en los siguientes grupos: métodos constructivos, métodos de dos fases y heurísticas de mejora.

### Métodos constructivos

Un método constructivo construye la solución de manera gradual agregando en cada iteración un nodo al recorrido teniendo como objetivo minimizar el precio. Estos métodos no cuentan con fase de mejora, por lo que la solución factible obtenida es la aplicable. En este grupo se encuentran los algoritmos de los ahorros (Savings Algorithms) y las heurísticas de inserción.

- *Algoritmo de Clarke y Wright* (Clarke & Wright, 1964)

Está comprendido dentro de los algoritmos de los ahorros. Se aplica cuando el número de vehículos a utilizar es una variable de decisión. Si en una solución factible salen dos rutas y éstas pueden ser combinadas para dar una solución mejor, entonces se utilizará esta nueva ruta.

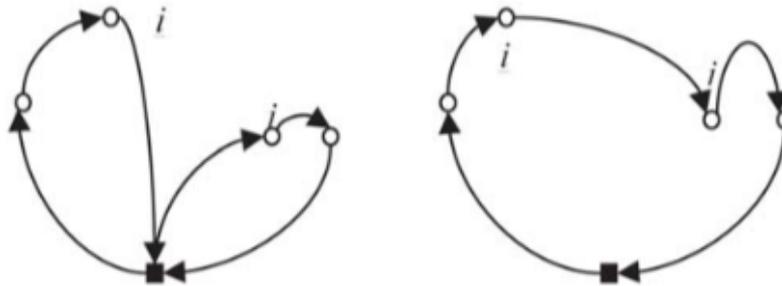


Figura 4-1: Combinación de dos rutas. Fuente: (Rocha, González, & Orjuela, 2011)

Este método es mejorado más adelante generalizando los ahorros mediante un parámetro llamado *Parámetro de Forma* que penaliza la unión de rutas con clientes lejanos (Toth & Vigo, 2002).

## Métodos de dos fases

En este grupo se encuentran diferentes métodos: asignación elemental, algoritmo de ramificación y acotamiento truncados, algoritmo de los pétalos, métodos de rutar primero y asignar después y los procedimientos de búsqueda local.

- *Métodos de asignación elemental*

Este tipo de métodos está formado por el algoritmo de barrido, el algoritmo basado en asignación generalizada y la heurística basada en localización.

Por un lado, el algoritmo de barrido funciona agrupando los clientes alrededor del depósito hasta violar la restricción de capacidad. En muchos casos es necesaria una fase de mejora.

Por otro lado, el algoritmo basado en asignación generalizada utiliza un problema de asignación generalizada (GAP). En la primera fase de este método se eligen los vértices llamados semillas alrededor de los cuales se construyen los agrupamientos. En la segunda fase se asignará los nodos a cada agrupamiento sin violar la restricción de capacidad resolviendo el GAP.

Por último, en la heurística basada en localización se forman inicialmente rutas, las llamadas semillas, como un problema de localización con capacidades. En la segunda fase se van añadiendo los nodos a cada ruta de manera gradual.

- *Algoritmo de ramificación y acotamiento truncados* (Christofides, Mingozzi, & Toth, 1979)

Este algoritmo forma tantos niveles como rutas de vehículos en el árbol de búsqueda y cada nivel contiene un conjunto de rutas de vehículos. El algoritmo determina una rama en cada nivel y una se descarta. Se puede construir un árbol limitado manteniendo pocas rutas en cada nivel.

- *Algoritmo de los pétalos* (Ryan, Hjorring, & Glover, 1993)

Procede del algoritmo de barrido y se utiliza para generar varas rutas llamadas pétalos con el fin de hacer una selección final. Se dispone de un conjunto de rutas  $R$  en la que cada cliente es visitado por varias rutas y se debe seleccionar un subconjunto de  $R$  que visite exactamente una vez cada cliente.

- *Métodos de ruteo primero y asignación después* (Beasley, 1983)

En la primera fase se determina una ruta que visite a todos los clientes resolviendo un TSP sin tener en cuenta las restricciones. En una segunda fase esta ruta se descompone en varias rutas factibles teniendo en cuenta las restricciones de capacidad.

- *Procedimientos de búsqueda local*

Estos procedimientos se aplican para mejorar una solución ya obtenida. Se define un conjunto de soluciones vecinas y se elige la que tenga menor coste. Este procedimiento se repite hasta que no se pueda mejorar la solución.

### 4.1.3 Metaheurísticas

Una metaheurística es una estrategia general para la resolución de una gran variedad de problemas para los que no existe un algoritmo confiable de resolución, ya sea por la complejidad del problema, o por falta de estudios en la resolución de éste (Lüer, Benavente, Bustos, & Venegas, 2009). Fueron desarrolladas a finales de la década de los 90. Dentro de este grupo se encuentran los algoritmos de: recocido simulado, redes neuronales, Búsqueda Tabú, algoritmos genéticos, colonia de hormigas y búsqueda de vecindades.

- *Algoritmos genéticos* (Holland, 1992)

Algoritmo inspirado en la teoría de la evolución de Darwin. Parte de una población inicial de individuos que representan soluciones factibles. El algoritmo evoluciona aplicando operadores evolutivos sobre la población que combinan y modifican los individuos creando una nueva población. Para cada individuo se define una función de aptitud  $f(i)$ , llamada fitness que califica su idoneidad. Los operadores utilizados son: selección, cruzamiento y mutación.

- *Colonia de hormigas* (Colorni, Dorigo, & Maniezzo, 1991)

Algoritmo inspirado en la estrategia que usan las colonias de hormigas en la búsqueda de alimentos. Las hormigas dejan rastro de feromonas en el camino al ir a una fuente de alimento. Éstas tienden a seguir los trayectos con mayor cantidad de feromonas ya que es más probable que sea el camino más rápido, lo que a su vez provoca un refuerzo de los mejores trayectos.

Para aplicar este algoritmo al VRP se coloca una hormiga en cada nodo y se utiliza una regla probabilística para construir los caminos. Esta regla asigna a cada nodo un cero si éste ya fue visitado o diferente a cero en caso contrario. La hormiga visitará el nodo que tenga la mayor probabilidad.

- *Búsqueda Tabú* (Glover, 1986)

Su principal característica es el uso de memoria adaptativa y de estrategias especiales en la resolución de problemas. Partiendo de una solución factible se genera un conjunto de soluciones vecinas mediante una búsqueda local. Se explora el espacio con estructuras de memoria flexible, restricciones estratégicas y niveles de aspiración. En cada iteración el algoritmo se mueve de una solución a otra, pero no necesariamente mejor. Se utiliza memoria de corto plazo que registre algunos atributos de soluciones ya visitadas. Las soluciones tabú son las soluciones ya exploradas que se registran en la memoria para no volver atrás. En algunos casos es necesario aceptar soluciones tabú porque poseen mejores atributos que otras soluciones y para eso se utiliza el criterio de aspiración, criterio que si se cumple permite volver a una solución tabú.

- *Recocido simulado* (Kirkpatrick, Gelatt, & Vecchi, 1983)

Este algoritmo está basado en un método de enfriamiento de materiales llamado Recocido. El Recocido es el enfriamiento de un material a un estado fundamental, un estado de mínima energía. Es una variante del método de búsqueda local que permite movimientos ascendentes para escapar de óptimos locales.

- *GRASP (Greedy Randomized Adaptive Search Procedure)* (Feo & Resende, 1989)

Este algoritmo consta de dos fases: fase de construcción y fase de mejora. En la fase de construcción se construye iterativamente una solución posible, considerando un elemento en cada paso. Para seleccionar el elemento a añadir en cada paso se utiliza una función greedy, que permite evaluar el beneficio de añadir cada uno de los elementos según la función objetivo. En la fase de mejora se realiza un proceso de búsqueda local a partir de la solución anterior hasta que no pueda mejorarse más.

- *Enjambre de partículas (Particle Swarm Optimization)* (Kennedy & Eberhart, 1995)

Es un algoritmo basado en organismos como bandadas de pájaros, enjambres de abejas y cardúmenes de peces donde existe un líder que marca el movimiento del grupo.

El algoritmo se inicia con un conjunto de soluciones aleatorias. A cada solución se le llama partícula. Se asignan velocidades aleatorias a cada partícula para que viajen hacia la solución óptima. La velocidad se va ajustando con el paso del tiempo mientras se evalúa cada partícula con la función fitness hasta encontrar la mejor solución.

## 4.2. Métodos de resolución para el VRPTW

Existen diferentes métodos de resolución que se han aplicado al problema VRPTW, desde métodos exactos, hasta técnicas heurísticas y metaheurísticas. Una de las técnicas metaheurísticas a destacar es la Búsqueda Tabú para el VRPTW propuesta por Semet y Taillard (1998) y Taillard (1997). También se han aplicado técnicas metaheurísticas con algoritmos genéticos para resolver el problema VRPTW como los que describen Homberger y Gehring (1999) o Berger, Barkaoui, y Bräysy (2003).



# 5 BÚSQUEDA TABÚ

El problema VRP es caracterizado por su alta complejidad computacional, por lo que es clasificado como un problema NP duro. Estos problemas se resuelven de manera aproximada con métodos metaheurísticos. En el caso que se presenta en este trabajo se utilizará el método metaheurístico de la Búsqueda Tabú. Se utilizará este método porque la Inteligencia Artificial le proporciona la capacidad de usar memoria, de manera que se logra direccionar la búsqueda basándose en la historia de la memoria, es decir, este método tiene en cuenta lo que ha sucedido previamente y toma decisiones en consecuencia.

## 5.1. Búsqueda Local

La Búsqueda Tabú está basada en los algoritmos de búsqueda local. Se trata de una técnica heurística sencilla de aplicar. Este algoritmo empieza con una solución factible y explora las soluciones vecinas en busca de una solución mejor. Este paso se hace tantas veces hasta que se no se encuentra una solución mejor en su vecindad<sup>2</sup>.

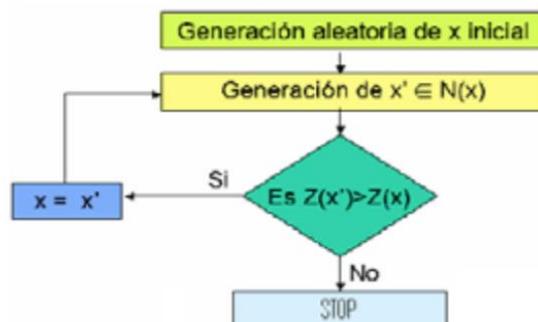


Figura 5-1: Algoritmo Búsqueda Local. Fuente: (Escudero-Santana & Aparicio-Ruíz, Búsqueda Local, 2018)

## 5.2. Fundamentos

Esta técnica basada en la Inteligencia Artificial e introducida por Fred Glover en 1986 ha resultado ser una técnica versátil y efectiva aplicada a problemas de optimización combinatorios. Su característica distintiva es la capacidad de tener memoria adaptativa y de usar estrategias especiales en la resolución de problemas.

El proceso de la Búsqueda Tabú parte de una solución inicial que puede ser aleatoria o generada mediante alguna técnica heurística. El algoritmo de resolución va modificando la solución de manera que construye un entorno de soluciones adyacentes que pueden ser alcanzadas desde la solución actual. Esta búsqueda guía la

<sup>2</sup> Se define *vecindad* como el entorno constituido por las soluciones accesibles desde la solución actual por uno de los movimientos posibles.

exploración a través de las regiones adyacentes imponiendo restricciones, donde algunas soluciones o transiciones se convierten en tabú. Esto significa que la Búsqueda Tabú continuará la búsqueda de soluciones incluso si la solución encontrada en una iteración es peor que la solución actual ya que las soluciones visitadas se almacenan como tabú para evitar iteraciones cíclicas.

El estado tabú de estas soluciones se mantiene durante un determinado número de iteraciones o hasta que se satisface una determinada condición, llamada *criterio de aspiración*.

A diferencia de otros métodos metaheurísticos como los algoritmos genéticos o recocido simulado, la Búsqueda Tabú cuenta con memoria adaptativa, su principal característica. La memoria adaptativa se basa en:

- Memoria a corto plazo (basada en lo reciente): almacena atributos de soluciones recientemente visitadas con el objetivo de explotar a fondo una región dada del espacio de soluciones.
- Memoria a largo plazo (basada en lo frecuente): almacena frecuencias u ocurrencias de atributos en las soluciones visitadas tratando de identificar o diferenciar regiones para:
  - Intensificar la búsqueda (memoria a medio plazo): permite volver a visitar zonas que contenían buenas soluciones. Se elige un número de mejores soluciones generadas durante un determinado período y sus características se almacenan para compararse. Esta característica es útil para la resolución de problemas de mayor envergadura ya que permite a la búsqueda centrarse en generar soluciones que son buenas.
  - Diversificar la búsqueda (memoria a largo plazo): permite explorar zonas aun no exploradas. Este mecanismo permite que la búsqueda no se centre solo en un pequeño espacio de soluciones, lo eliminaría la posibilidad de encontrar el óptimo global. Mientras que otras técnicas incorporan este mecanismo mediante la aleatorización, la Búsqueda Tabú penaliza los atributos que son frecuentes entre las soluciones intermedias y aquellos movimientos que se llevan a cabo con cierta frecuencia.

### 5.3. Algoritmo de Búsqueda Tabú

El algoritmo se inicia con una solución factible inicial  $x$ . Cada solución  $x$  tiene un entorno  $N(x)$  determinado por la propia solución  $x$ . Se genera la muestra de soluciones  $V^*$  contenidas en el entorno  $N(x)$  de la solución  $x$ , es decir, se genera un entorno reducido. Se elige la mejor solución ( $x^*$ ) del entorno reducido  $V^*$  y se convierte en la solución actual. El paso hasta la solución  $x^*$  se lleva a cabo, aunque esta solución sea peor que la actual  $x$ , permitiendo de esta manera escapar de óptimos locales. Si la condición de parada no se cumple, se actualizará la lista tabú. La lista tabú tiene la función de evitar volver a soluciones previas y contiene información de los movimientos realizados. A continuación, se repite el proceso partiendo de la actual solución  $x^*$ .



Figura 5-2: Algoritmo Búsqueda Tabú. Fuente: Escuerdo A., Aparicio P. Búsqueda Tabú, Modelado y Optimización de problemas de gestión, Universidad de Sevilla.

## 5.4. Lista tabú

Para evitar movimientos inversos y entrar en iteraciones cíclicas se almacena información acerca de los propios movimientos o atributos de los movimientos. Se consideran atributos de los movimientos aquellos aspectos que cambian como resultado de pasar de una solución actual a otra.

Almacenando el movimiento, se evitará dicho movimiento, pero si se almacenan los atributos del movimiento se evitarán movimientos con dichos atributos mientras permanezcan como estado tabú en la lista tabú.

El **tamaño de la lista tabú** es un parámetro que se debe establecer para realizar la Búsqueda Tabú. El tamaño de la lista tabú es el tiempo o número de iteraciones que un elemento (ya sea movimiento o atributo) permanece en la lista tabú. Este atributo puede definirse como fijo o variable.



# 6 IMPLEMENTACIÓN

---

En este capítulo se detallará el proceso de resolución que ha sido implementado para resolver el problema planteado. Este algoritmo de optimización ha sido codificado en lenguaje Python 3.8. Python es un lenguaje de programación interpretado, enfocado a una sintaxis que favorezca un código legible, lo que lo hace un lenguaje relativamente sencillo. Además, este lenguaje ofrece un amplio catálogo de bibliotecas para análisis de datos y optimización de problemas que ayudan a agilizar la resolución.

## 6.1. Definición de la solución

### 6.1.1 Matriz de localizaciones seleccionadas

En primer lugar, se detallará cómo se identificará qué localización se ha elegido para servir al cliente. Como se ha visto anteriormente, un cliente puede elegir hasta tres localizaciones asignándoles prioridad y ventana temporal a cada una de ellas.

Para el propósito de identificar qué localizaciones se han elegido en la solución admisible, se genera una matriz de dimensiones ( $n \times 3$ ) de tipo *booleana* (True o False), donde  $n$  es el número de clientes y el número de columnas hace referencia al número de localizaciones. Aunque un cliente no tiene la obligatoriedad de establecer 3 localizaciones, esta matriz sí se definirá con 3 columnas por ser el número máximo de localizaciones permitidas. De esta manera, cada fila de la matriz contendrá un solo valor positivo, que indicará cual es la localización seleccionada. A continuación, se muestra un ejemplo con 10 clientes. Se debe tener en cuenta que la primera fila de la columna hace referencia al depósito, por lo que el valor True estará siempre en la primera columna ya que el depósito en este problema solo tiene una localización.

```
[[ True False False]
 [False False True]
 [False True False]
 [ True False False]
 [False False True]
 [False True False]
 [False False True]]
```

Tabla 6-1: Matriz de localizaciones seleccionadas para 10 clientes.

La segunda fila de la matriz por tanto corresponde al cliente número 1 que en este caso recibirá su pedido en la localización número 3, por tener el valor True en la columna 3.

Es importante tener en cuenta que la prioridad no se corresponde con el número de la localización, por tanto, tampoco con el número de la columna.

### 6.1.2 Definición de una ruta

El propósito de este problema es obtener el menor número de rutas para satisfacer la demanda de todos los clientes. Es por esto por lo que la solución obtenida por el problema será una batería de rutas.

Para mantener la solución estructurada con toda la información necesaria se ha definido una clase de objetos llamada *Route*. Una clase es una plantilla genérica que define los atributos y métodos que tendrán los objetos de dicha clase (Escudero-Santana & Aparicio-Ruiz, Python: Orientación a objetos, 2018).

La clase *Route* contiene la información necesaria en de cada ruta contenida en sus atributos. Estos atributos que se han definido son:

- *path*: lista ordenada de los clientes visitados, especificando el número de identificación del cliente. Contiene el depósito al inicio y al final, ya que es restricción de este problema partir y acabar en el depósito.
- *totdis*: distancia total recorrida por el vehículo en esta ruta.
- *tottime*: tiempo total transcurrido desde el inicio hasta el final de dicha ruta.
- *totcost*: coste total imputado por dicha ruta. Incluye el coste de utilizar el vehículo.
- *starttime*: indica en qué momento debe salir el vehículo a realizar dicha ruta para cumplir las restricciones. Se indica como minutos tras la apertura de la ventana temporal del depósito.

### 6.1.3 Solución: lista de rutas

El algoritmo de resolución del problema agrupará las rutas obtenidas para una posible solución en una misma lista. Esta lista se considera la solución del problema. Por tanto, la salida final del proceso de resolución es una lista que contiene todas las rutas generadas para cumplir las restricciones del problema y satisfacer todas las demandas de los clientes. Esto se define en una lista de objetos de tipo *Route*.

A continuación, se muestra un ejemplo de la solución a un problema con 25 clientes mostrando información contenida en los atributos de los objetos que indican las rutas:

<i>path</i>	<i>totcost</i>	<i>tottime</i>	<i>starttime</i>
[0, 17, 4, 21, 20, 22, 25, 24, 0]	1634.32	1534.32	0
[0, 12, 7, 23, 13, 9, 0]	1050.86	950.86	0
[0, 18, 10, 0]	676.81	576.81	0
[0, 2, 19, 8, 0]	1008.20	908.20	1016
[0, 6, 14, 0]	832.59	732.59	365.33
[0, 16, 15, 3, 5, 0]	748.35	648.35	1025.42
[0, 1, 11, 0]	861.05	761.05	1056.28

Con un coste total de 6812.18 y 7 vehículos.

Tabla 6-2: Solución admisible para 25 clientes.

## 6.2. Interfaz de lectura de datos

La entrada de datos del problema se realizará mediante una función que los lee de un archivo externo y los guarda en un atributo de una nueva clase. Esta clase servirá para almacenar toda la información relevante del problema.

### 6.2.1 Estructura de los datos

Los datos se presentan en formato Excel, con el estilo que presenta Solomon en (Solomon, 1987). Estos conjuntos de datos están formados por: número de vehículos y capacidad (no se tendrán en cuenta en el problema que se presenta), número de cliente, coordenadas x e y, demanda, ventana temporal y tiempo de servicio. Adicionalmente se le ha añadido una nueva columna para indicar la prioridad. A continuación, se muestra un ejemplo del formato empleado:

r110_1								
VEHICLE								
NUMBER	CAPACITY							
250	200							
CUSTOMER								
CUST NO.	XCOORD.	YCOORD.	DEMAND	READY TIME	DUE DATE	SERVICE TIM	PRIORITY	
0	250	250	0	0	1925	0	1	
1	171	34	21	1	641	10	3	
2	67	190	25	642	1282	10	2	
3	80	400	10	1283	1923	10	1	

Tabla 6-3: Conjunto de datos con el formato de Solomon.

## 6.2.2 Lectura de datos

Para la carga inicial del problema se ha creado una clase llamada *Problem* que se encargará de inicializar el problema. Al inicializar el problema en un objeto de tipo *Problem* se inicializan los diferentes atributos que servirán para guardar la información necesaria para la solución:

- *customers*: matriz que contiene toda la información proveniente del fichero Excel con los datos de los clientes. Se inicializa mediante la función *listdata*. Esta función accede al archivo contenedor de los datos y los devuelve para ser almacenados.
- *depot*: variable que almacena toda la información obtenida sobre el depósito, punto de partida y llegada de todos los vehículos.
- *truckCost*: coste de utilizar un vehículo.
- *truckSpeed*: velocidad media de los vehículos.
- *kmcost*: coste por kilómetro para los vehículos.
- *servicetime*: indica si el problema tiene en cuenta o no el tiempo de servicio para su solución.
- *selectedLocations*: matriz de localizaciones seleccionadas, inicializada en False. La localización del depósito se inicializa en True.
- *solution*: inicializado en vacío. Almacenará la solución al problema.
- *solutionCost*: almacenará el coste total de la solución.
- *solutionFitness*: almacenará el fitness de la solución. Esta es la variable que se trata de minimizar en la función objetivo

Para inicializar un objeto de tipo *Problem* se requieren los parámetros:

- *problemClass*: hace referencia al tipo de problema, en este caso los tipos son los formulados por Solomon (*r1*, *r2*, *c1*, *c2*, *rc1* y *rc2*). Se corresponden con el título del archivo Excel que contiene los datos.
- *problemNum*: número del problema dentro del tipo. Cada tipo de problema contiene una colección numerada de conjuntos de datos. Se corresponde con el título de la hoja del archivo Excel.
- *problemSize*: número de clientes que contiene el problema.
- *truckCost*: coste de utilizar un vehículo. Se inicializa en 100 por defecto.
- *truckSpeed*: velocidad media de los vehículos. Se inicializa en 60 por defecto.
- *kmcost*: coste por kilómetro para los vehículos. Inicializado en  $1 \frac{u.m.}{km}$  por defecto

Un ejemplo de inicialización del problema:

```
myproblem = Problem("r1", "02", problemSize = 25)
```

### 6.2.3 Otros tipos de objetos: *Customer* y *Location*

Se han definido otros tipos de objetos para simplificar el acceso a los datos durante la resolución del problema. En el apartado anterior se comenta que la información de los clientes se almacena en el atributo *customers*. Se ha definido un objeto de tipo *Customer* que sirve para almacenar la información de un cliente. Por tanto, el atributo del problema *customers* será una lista de objetos de tipo *Customer*. A su vez, para contener la información de las localizaciones (coordenadas, ventana temporal y prioridad) se ha definido la clase *Location*. De esta manera, cada cliente (objeto *Customer*) contendrá una lista de tantos objetos de tipo *Location* como localizaciones tenga (el problema que se presenta puede contener hasta tres localizaciones).

## 6.3. Procedimientos de optimización

En esta sección se verá con más detalle el procedimiento de optimización que el algoritmo realiza para encontrar la solución al problema. En primer lugar, se presentará una vista general en formato de diagrama para ver de manera resumida las fases del procedimiento. A continuación, se detallarán cada una de las fases.

Como se ha comentado anteriormente, la metodología aplicada al problema será la Búsqueda Tabú. En este problema se ha implementado esta metodología junto con una búsqueda VNS que permitirá buscar en varias vecindades a lo largo de la optimización. Antes de entrar en la búsqueda, será necesario generar una solución inicial.



Figura 6-1: Flujo general del algoritmo de optimización.

### 6.3.1 Generar soluciones

Se ha definido una función que construye una solución dado un vector de índices (identificación de los clientes) y su matriz de localizaciones seleccionadas.

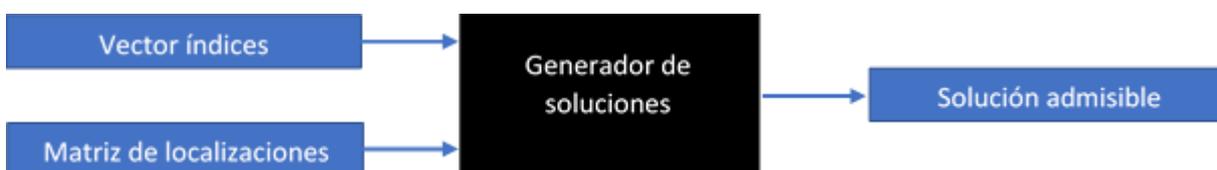


Figura 6-2: Función de generación de soluciones.

Esta función recorre el vector de índices añadiendo los clientes a una ruta en caso de que se cumplan las restricciones. Si al recorrer el vector completo aún quedan clientes por asignar a una ruta, se agregarán nuevas rutas estableciendo el momento de partida del vehículo en un momento diferente a 0 de manera que se puedan cumplir las restricciones. La función devuelve una solución admisible en forma de lista de objetos de tipo *Route*.

### 6.3.2 VNS

La Búsqueda de Entorno Variable o VNS (Variable Neighborhood Search) es una metaheurística desarrollada por Mladenovic y Hansen en (Mladenovic & Hansen, 1997). Esta metaheurística permite a los algoritmos de búsqueda local utilizar varias vecindades para salir de óptimos locales.

La Búsqueda Tabú ya incluye la característica que permite salir de mínimos locales gracias a la lista tabú, que hace accesible peores soluciones para explorar el entorno.

En este trabajo se ha utilizado VNS para poder diversificar la búsqueda, es decir, para explorar diferentes entornos. Cada entorno vendrá definido por una matriz de localizaciones seleccionadas, que se inicializa aleatoriamente (posteriormente se detallará el procedimiento de inicialización de dicha matriz). De esta manera, en cada iteración  $k$  de VNS se genera una nueva matriz de localizaciones iniciada aleatoriamente, que será la que se utilice dentro de la Búsqueda Tabú para generar las vecindades:

```

1: S = Generar matriz de localizaciones aleatoria
2: x = solución inicial (S)
3: k=1
4: mientras (k<=kmax)
5: Sk = Generar matriz de localizaciones aleatoria
6: x' = Solución aleatoria (Sk)
7: x'' = Búsqueda Tabú (x')
8: si (f(x'') < f(x))
9:     x=x''
10:     k=1
11: si no
12:     k=k+1

```

#### 6.3.2.1 Inicialización de matriz de localizaciones: probabilidades dinámicas

Para inicializar la matriz de localizaciones se ha definido una función que construye la matriz en función de probabilidades que se dan como parámetros de entrada. Dichas probabilidades corresponden a las prioridades de las diferentes localizaciones. En este problema el número máximo de localizaciones posibles para un cliente es tres, por tanto, para inicializar la matriz se deberán introducir tres probabilidades  $p_1$ ,  $p_2$  y  $p_3$ , donde  $p_1$  corresponde a la probabilidad de que la localización con prioridad 1 sea asignada,  $p_2$  corresponde a la probabilidad de que la localización con prioridad 2 sea asignada y  $p_3$  corresponde a la probabilidad de que la localización con prioridad 3 sea asignada. Así,  $p_1 + p_2 + p_3 = 1$ .

La matriz inicial para la solución inicial se inicia con  $p_1 = 1$ ,  $p_2 = 0$  y  $p_3 = 0$ . De esta manera se busca una solución admisible cuando todas las localizaciones seleccionadas son las de prioridad 1, por lo que en esta solución se estaría obteniendo la máxima satisfacción del cliente (la satisfacción del cliente es mayor cuando se le entrega el pedido en su primera prioridad).

A lo largo de las iteraciones de la búsqueda VNS van variando las probabilidades que se introducen en la función, es por esto por lo que se consideran probabilidades dinámicas. En cada iteración  $k$  de la búsqueda VNS, la probabilidad  $p_1$  decrementa y las probabilidades  $p_2$  y  $p_3$  incrementan de la siguiente manera:

```

1: p1 = 1, p2 = 0, p3 = 0
2: S = Matriz de localizaciones (p1, p2, p3)
3: x = Solución inicial (S)
4: k = 1
5: mientras (k<=kmax)
6:     Sk = Generar matriz de localizaciones (p1, p2, p3)
7:     x' = Solución aleatoria (Sk)
8:     x'' = Búsqueda Tabú (x')
9:     si (f(x'') < f(x))
10:         x=x''
11:         k = 1, p1 =1, p2 = 0, p3 = 0
12:     si no
13:         k = k + 1
14:         p1 = p1- 1/(2*kmax)
15:         p2 = (1 - p1) * 2/3, p3 = (1 - p1) * 1/3

```

### 6.3.3 Búsqueda Tabú

La Búsqueda Tabú permite realizar una búsqueda intensificada a partir de la solución aleatoria que se le introduce. En los apartados anteriores se ha indicado que esta solución aleatoria se genera antes de entrar en la búsqueda. En la resolución del problema, esta generación de la solución aleatoria se ha implementado dentro de la misma Búsqueda Tabú (líneas 6 y 7 del pseudocódigo anterior se realizan dentro de la Búsqueda Tabú). A continuación, se muestra de manera genérica y en pseudocódigo el procedimiento llevado a cabo por la Búsqueda Tabú:

```

1: Se inicializa la matriz de localizaciones en función de p1, p2 y p3
2: Se ordena el vector de índices según un criterio de ordenación
3: Se genera solución
4:
5: Se inicializa lista tabú
6:
7: Durante un número determinado de iteraciones:
8:   Recorrer vector de índices:
9:     Se crea un vecino
10:    Si el vecino creado no es tabú:
11:      Añadir vecino al vecindario
12:
13:   Criterio de aspiración
14:
15:   Se evalúa el vecindario según la función objetivo
16:
17:   Si la solución obtenida es mejor que la solución actual:
18:     Actualizar solución actual
19:   Actualizar lista tabú

```

#### 6.3.3.1 Criterio de ordenación de índices

Como se ha comentado antes, para inicializar la solución de la Búsqueda Tabú se genera una solución con la función de generar soluciones a raíz de un vector de índices que ha sido ordenado previamente.

El vector de índices, que corresponde a los índices de los clientes, se ordena de menor a mayor según el siguiente valor correspondiente a cada índice (cliente):

$$w * (l_j - e_j) - d_{0j}$$

Donde:

- $w = 100$  : Es un peso arbitrario usado para enfatizar la estrechez de la ventana en relación con la distancia del depósito.
- $l_j$  : cierre de la ventana temporal de  $j$
- $e_j$  : apertura de la ventana temporal de  $j$
- $d_{0j}$  : distancia de  $j$  al depósito

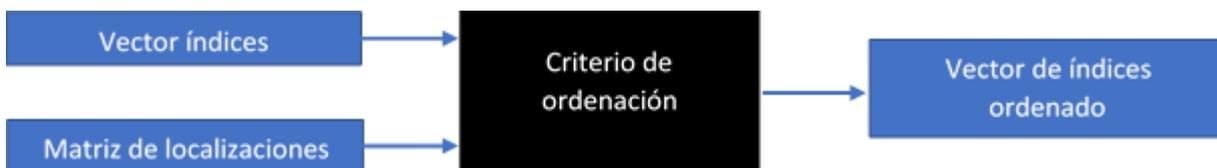


Figura 6-3: Criterio de ordenación de vector de índices.

Este criterio se ha aplicado de manera empírica. Se han establecido diferentes criterios de ordenación como el basado en la ventana temporal más temprana o en la mayor distancia al depósito, además de ordenar aleatoriamente. Se ha seleccionado este por ser el que mejores resultados ofrece (Chian & Rusell, 1997).

A continuación, se presenta un ejemplo de un conjunto de 25 clientes ordenados según dicho criterio teniendo inicializada la matriz de localizaciones:

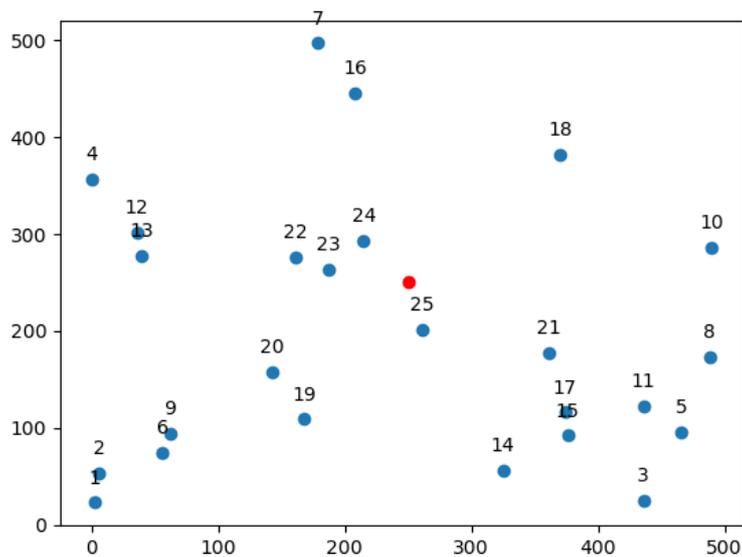


Figura 6-4: Ejemplo ordenación de 25 clientes.

### 6.3.3.2 Espacio de búsqueda

Los vecinos son soluciones generadas a partir de vectores de índices. Para generar diferentes vecinos se generan soluciones a partir de las posibles permutaciones del vector de índices de la solución actual. Estas permutaciones son las que definirán como tabú una solución, es decir, la lista tabú utiliza la permutación entre dos índices  $i$  y  $j$  para guardarla como tabú.

Partiendo del vector de índices de la solución actual, es decir, el vector con los clientes que se introduce en la función de generación de soluciones para crear la solución, se realizan permutaciones entre los diferentes índices y se evalúan los vectores resultantes, obteniendo las diferentes soluciones que conforman el vecindario. Este vecindario será evaluado según la función objetivo como se detalla a continuación.

### 6.3.3.3 Criterio de aspiración

Se ha considerado criterio de aspiración con el fin de evitar no encontrar vecinos no tabú. De esta manera, el criterio entra en funcionamiento una vez se ha generado el vecindario. Si el vecindario está vacío debido a que todos los vecinos posibles eran tabú, se añaden al vecindario dichos vecinos sin tener en cuenta la lista tabú.

### 6.3.3.4 Función objetivo

Una vez se ha generado el vecindario, se evalúan todos los vecinos para obtener el vecino que más minimice la función objetivo. La función objetivo tiene en cuenta tanto el coste de la solución como las prioridades de las localizaciones seleccionadas. Esta función permite minimizar el coste de la solución y maximizar la satisfacción del cliente (minimizando la suma de prioridades):

$$\text{Min } \alpha * \frac{c_i}{C} + \beta * \frac{p_i}{P}$$

Donde:

- $\alpha, \beta$ : pesos para coste y prioridades respectivamente  $\alpha + \beta = 1$
- $c_i$ : coste de la solución  $i$
- $C$ : coste máximo para la matriz de localizaciones establecida. Corresponde a la solución que establece una ruta para cada cliente.
- $p_i$ : suma de prioridades de las localizaciones seleccionadas de la solución  $i$
- $P$ : suma de las máximas prioridades de las localizaciones seleccionadas para el problema dado.

Los términos  $\frac{c_i}{C}$  y  $\frac{p_i}{P}$  normalizan los valores correspondientes a coste y prioridad para tenerlos en rango  $[0,1]$  ya que los intervalos de los diferentes valores se encuentran en diferentes escalas.

# 7 RESULTADOS

## 7.1. Calibrado

### 7.1.1 Proceso de calibración

Para ajustar los parámetros de la búsqueda tabú se ha llevado a cabo el proceso de ajuste definido en (Xu, Chiu, & Glover, 1998) que se explica a continuación. Se trata de un proceso basado en un árbol de decisión combinando dos pruebas estadísticas: prueba de Friedman y prueba de Wilcoxon.

Se llamarán **factores** a los parámetros que se van a calibrar y **tratamientos** a los valores que se evalúan para cada factor.

#### 7.1.1.1 Prueba de Friedman

La prueba de Friedman es un método estadístico que ordena los datos en función del resultado obtenido. Esto es, una vez definidos los tratamientos para un factor se obtienen múltiples soluciones para cada tratamiento. Estas soluciones son ordenadas por el método para medir el rendimiento.

La prueba de Friedman asume de partida la hipótesis nula, es decir, asume que las medias de las soluciones son iguales. El resultado de la prueba descartará la hipótesis nula y se procederá a la selección del tratamiento para el factor evaluado.

Para realizar la prueba, se obtendrá  $x_{ij}$  como el resultado para el tratamiento  $i$  en el problema  $j$  para  $i = 1, \dots, I$  y  $j = 1, \dots, J$ . Para cada problema  $j$ , se ordenarán los resultados  $x_{ij}$  ( $i = 1, \dots, I$ ) desde 1 a  $I$  de mejor a peor solución.  $R_{ij}$  denotará la posición en la clasificación (*rank*) de  $x_{ij}$ . Después, la prueba estadística se calculará con:

$$Fr = \frac{12}{IJ(J+1)} \sum_{i=1}^I R_i^2 - 3J(I+1)$$

donde  $R_i = \sum_{j=1}^J R_{ij}$ . Para descartar la hipótesis nula el valor de Friedman debe ser mayor al valor crítico obtenido por la distribución chi-cuadrado para  $I - 1$  grados de libertad y un nivel de confianza  $1 - \alpha$ :

$$Fr > \chi_{\alpha, I-1}^2$$

Si la hipótesis nula se cumple, puede usarse cualquier tratamiento evaluado para el factor que se está estudiando. Si se descarta la hipótesis, se sigue con el proceso de selección.

Una vez se ha descartado, se debe estudiar qué tratamientos dominan sobre el resto. Para ello se define  $R_{min}$  como el tratamiento con  $R_i$  más bajo. Se descartarán todos los tratamientos que cumplan:

$$R_i > R_{min} + z_{\alpha} * \sqrt{JI(I+1)/6}$$

donde  $z_{\alpha}$  es el percentil de  $(1 - \alpha)$  en la distribución normal.

Los tratamientos que no han sido descartado se consideran buenos valores para ser establecidos en el factor estudiado.

### 7.1.1.2 Prueba de los rangos con signo de Wilcoxon

La prueba de Wilcoxon compara dos muestras ( $X_A, X_B$ ) independientes para medir si existe diferencia entre ambas. Se supone inicialmente la hipótesis nula  $H_0: \mu_D = 0$ , donde  $D = X_A - X_B$  y  $\mu_D$  representa la media de  $D$ .

Para realizar la prueba se utiliza como medida el valor  $s_+$ , que se calculará como sigue:

1. Se calcula  $D$  para cada problema de prueba de las muestras  $A$  y  $B$ . El número de problemas probados se define como  $J$ .
2. Ignorando el signo de  $D$ , se clasifican estos valores de manera creciente.
3. Calcular  $s_+$  como la suma de la posición en la clasificación de los valores positivos de  $D$  (si todos son negativos, entonces  $s_+ = 0$ ).
4. Se descarta la hipótesis nula para el nivel de confianza  $1 - \alpha$  si  $s_+ > d_1$  o si  $s_+ \leq \frac{J(J+1)}{2} - d_1$  donde  $d_1$  es el valor crítico obtenido en (Devore, 1991).
5. Si se descarta la hipótesis nula:
  - a. Si  $s_+ > d_1$ , el tratamiento de  $B$  domina sobre  $A$ .
  - b. Si  $s_+ \leq \frac{J(J+1)}{2} - d_1$ , el tratamiento de  $A$  domina sobre  $B$ .

### 7.1.1.3 Método aplicado

*Paso 1* Definir los factores  $k$  para ajustar y sus tratamientos. Ordenar en orden decreciente de importancia estos factores:  $F = (F_1, \dots, F_k)$

*Paso 2* Iniciar el árbol de búsqueda considerando el factor  $F_1$  como el nodo raíz. Definir nivel  $i = 1$ .

*Paso 3* Para cada nodo en el nivel  $i$ , añadir tantas ramas como tratamientos tenga el factor  $F_i$ . Ejecutar los problemas de prueba para cada tratamiento y guardar los resultados en un nodo para cada rama en el nivel  $i + 1$ .

*Paso 4* Para cada nodo en el mismo nivel ( $i + 1$ ) realizar la prueba de Friedman en los nodos con mismo padre. Eliminar los nodos que resulten inferiores en la prueba.

*Paso 5* Para cada pareja de los nodos restantes que no comparten el nodo padre, realizar prueba de Wilcoxon. Descartar el nodo que resulte inferior en la prueba.

*Paso 6*  $i = i + 1$ . Si  $i > k$  ir al *Paso 7*, si no, ir al *Paso 3*.

*Paso 7* Cada nodo restante representa una de las mejores configuraciones para el algoritmo.

## 7.1.2 Tamaño de lista tabú

Es necesario establecer un adecuado tamaño para la lista tabú ya que un tamaño pequeño haría que el algoritmo se aproximase a una búsqueda local y un tamaño demasiado grande no permitiría volver a soluciones de calidad. Un tamaño adecuado permitirá diversificar la búsqueda.

Ya que se recomienda un tamaño del orden del número de atributos, en este caso el número de clientes, en este proceso de calibración se evaluarán los valores comprendidos en  $[\frac{n}{2} - \sqrt{n}, \frac{n}{2} + \sqrt{n}]$  siendo  $n$  el tamaño del problema, es decir, el número de clientes. Para el proceso de calibración se ha establecido  $n = 50$ .

Para el ajuste de este primer factor se han establecido provisionalmente el valor para las iteraciones de la búsqueda tabú en 50 y el valor para las iteraciones del algoritmo VNS en 1. Además, se establece el número de problemas  $J = 15$  para las pruebas estadísticas y para cada problema  $j$  se utiliza la misma matriz de localizaciones en todos los tratamientos evaluados.

Esta prueba se compone de dieciséis candidatos para el tamaño, denominado *tabu\_size*, que son los valores comprendidos en el intervalo [17, 32]. Los resultados de las dieciséis ejecuciones (*Ejecución 1, ... Ejecución 16*) se muestran en la Tabla 7-1: Pruebas para tamaño tabú..

Ejecución	1	2	3	4	5	6	7	8
<i>tabu_size</i>	17	18	19	20	21	22	23	24
$R_i$	113	116	142	129	154	108	130	132
Ejecución	9	10	11	12	13	14	15	16
<i>tabu_size</i>	25	26	27	28	29	30	31	32
$R_i$	87	128	146	134	159	118	107	137

Tabla 7-1: Pruebas para tamaño tabú.

Se establece un nivel de confianza del 90%, por lo que  $\alpha = 0.1$ . De los resultados obtenidos, se calcula el valor de Friedman  $F_r = 5.012$  que es menor que  $\chi_{0.1,15}^2 = 22.31$ .

Se concluye que se cumple la hipótesis nula y usar cualquiera de los valores evaluados no afecta al resultado. Por esto, se establece el tamaño tabú como  $\frac{n}{2}$ .

### 7.1.3 Iteraciones de la búsqueda tabú

Una vez ajustado el tamaño de la lista tabú se ajustará el número de iteraciones de la búsqueda tabú ( $n_{iterations}$ ). Para este factor se estudiarán los tratamientos comprendidos en el intervalo [10,200] con un paso de 10, esto son veinte ejecuciones. En la Tabla 7-2: Pruebas para número de iteraciones de la búsqueda tabú. se muestran los resultados obtenidos.

Ejecución	17	18	19	20	21	22	23	24	25	26
$n_{iterations}$	10	20	30	40	50	60	70	80	90	100
$R_i$	236	257	238	234	208	201	233	196	187	152
Ejecución	27	28	29	30	31	32	33	34	35	36
$n_{iterations}$	110	120	130	140	150	160	170	180	190	200
$R_i$	176	145	145	140	101	112	71	61	40	17

Tabla 7-2: Pruebas para número de iteraciones de la búsqueda tabú.

Para un nivel de confianza del 90%, es decir,  $\alpha = 0.1$  resulta el valor de Friedman  $F_r = 184.59$  que es mayor que  $\chi_{0.1,19}^2 = 27.20$ . Por esto, se descarta la hipótesis nula y se deduce que los diferentes tratamientos estudiados afectan a los resultados para un nivel de confianza del 90%. Se observa  $R_{min}$  en la ejecución 35. Se calcula el valor crítico de  $R$  para establecer la dominancia:

$$R_{min} + z_{\alpha} * \sqrt{\frac{JI(I+1)}{6}} = 58.53$$

Se descartan las ejecuciones cuyos  $R_i > 58.53$ . Por tanto, solo las ejecuciones 35 y 36 se siguen considerando para ser evaluadas.

### 7.1.4 Iteraciones VNS

Ahora se estudiará qué valor establecer como máximo para las iteraciones del algoritmo VNS ( $kmax$ ). Se establecen los tratamientos de este factor los comprendidos en el intervalo [5,10]. Se evaluarán estos tratamientos para cada tratamiento del factor  $n_{iterations}$  evaluado en el apartado anterior. En primer lugar, se evaluará para  $n_{iterations} = 190$ . En la Tabla 7-3: Pruebas de  $kmax$  para Ejecución 35 se muestran los resultados.

Ejecución	37	38	39	40	41	42
$kmax$	5	6	7	8	9	10
$R_i$	61	64	46	56	48	40

Tabla 7-3: Pruebas de  $kmax$  para Ejecución 35

Se establece un nivel de confianza del 90%, por lo que  $\alpha = 0.1$ . De los resultados obtenidos, se calcula el valor de Friedman  $F_r = 7.95$  que es menor que  $\chi_{0.1,5}^2 = 9.23$ . Por tanto, se cumple la hipótesis nula para  $n\_iterations = 190$  y se podrá seleccionar cualquier valor de  $kmax$  evaluado. Para  $n\_iterations = 190$  se establece  $kmax = 5$  (ejecución 37).

En segundo lugar, se evalúan los tratamientos de  $kmax$  para  $n\_iterations = 200$ . En la Tabla 7-4: Pruebas de  $kmax$  para Ejecución 36 se muestran los resultados.

Ejecución	43	44	45	46	47	48
$kmax$	5	6	7	8	9	10
$R_i$	70	65	54	44	49	33

Tabla 7-4: Pruebas de  $kmax$  para Ejecución 36

Para un nivel de confianza del 90%, es decir,  $\alpha = 0.1$  resulta el valor de Friedman  $F_r = 18.62$  que es mayor que  $\chi_{0.1,5}^2 = 9.23$ . Por esto, se descarta la hipótesis nula y se deduce que los diferentes tratamientos estudiados afectan a los resultados para un nivel de confianza del 90%. Se observa  $R_{min}$  en la ejecución 48. Se calcula el valor crítico de  $R$  para establecer la dominancia:

$$R_{min} + z_{\alpha} * \sqrt{\frac{JI(I+1)}{6}} = 46.13$$

Se descartan las ejecuciones cuyos  $R_i > 46.13$ . Por tanto, solo las ejecuciones 46 y 48 se siguen considerando para ser evaluadas.

Ahora solo quedan 3 nodos en este nivel, uno que desciende de la ejecución 35 y dos que descienden de la ejecución 36. Se aplicará la prueba de Wilcoxon a las parejas formadas por las ejecuciones (37, 46) y (37, 48). Los resultados de la prueba se muestran en la Tabla 7-5: Prueba de Wilcoxon para  $kmax$ .

Pareja	(37,46)	(37,48)
$s_+$	18	9

Tabla 7-5: Prueba de Wilcoxon para  $kmax$ 

El valor crítico  $d_1$  para  $\alpha = 0.094$  es 84 (ver Devore, 1991). Se descartará la hipótesis nula para cada pareja si  $s_+ > d_1 = 84$  o si  $s_+ \leq \frac{J(J+1)}{2} - d_1 = 36$ . En este caso se descarta la hipótesis nula para ambas y se descarta la pareja (37, 46) por ser su valor  $s_+$  mayor.

En la pareja (37, 48), dado que  $s_+$  es  $s_+ \leq \frac{J(J+1)}{2} - d_1 = 36$ , la ejecución 48 es descartada.

Se concluye de estas pruebas que la configuración que obtiene mejores resultados es la configuración de la ejecución 37:

- $tabu\_size = \frac{n}{2}$
- $n\_iterations = 200$
- $kmax = 5$

## 7.2. Alfa, Beta

Los parámetros  $\alpha$  y  $\beta$  son los pesos asociados al coste y a la prioridad respectivamente en la función objetivo. Es objeto del decisor establecer los valores en función de a qué se quiera dar preferencia tal que  $\alpha + \beta = 1$ .

En esta sección se llevará a cabo un análisis sobre la influencia de los parámetros  $\alpha$  y  $\beta$  en los resultados ya que, en función de los valores establecidos para dichos pesos, el problema varía su comportamiento. Para el análisis se ejecutará el escenario formado por el problema R201 de Homberger. Se detallará este tipo de problemas el siguiente apartado.

El primer comportamiento analizado se desarrolla si se establece  $\alpha = 1$  y  $\beta = 0$  el problema no tendrá en cuenta las prioridades de los clientes, por lo que se considera un problema 3TW-VRP sin prioridades. Este caso es similar al problema desarrollado en (Uribe & Escudero-Santana, 2017). A continuación, se muestran los resultados obtenidos tras 10 ejecuciones del escenario:

	Promedio	Mínimo	Desviación
Función objetivo	0.44	0.42	2%
Coste	5114.60	4819.84	3%
Prioridad	29.60	25.00	20%

Tabla 7-6: Resultados obtenidos para alfa = 1 y beta = 0.

En este caso se observa una pequeña desviación en el coste ya que la función objetivo se centra en este y por otro lado se observa una alta desviación en la prioridad. Esto evidencia el comportamiento del problema como un 3TW-VRP sin prioridades.

Otro comportamiento del problema que cabe destacar se desarrolla cuando se establece  $\alpha = 0$  y  $\beta = 1$ . En este caso el problema se centrará en minimizar solo las prioridades. Al centrarse solo en las prioridades seleccionará todas las primeras prioridades, convirtiéndose el problema en un problema VRPTW tradicional, cuyas ventanas temporales se corresponden con las establecidas como prioridad 1. En este caso, si el problema se ejecuta 10 veces se obtendrá siempre la misma solución ya que las localizaciones seleccionadas siempre serán las mismas y el criterio de ordenación previo a la generación de la solución no varía.

Por último, se mostrarán los resultados obtenidos para valores de  $\alpha \in [0,1]$  y  $\beta \in [0,1]$  tal que  $\alpha + \beta = 1$ .

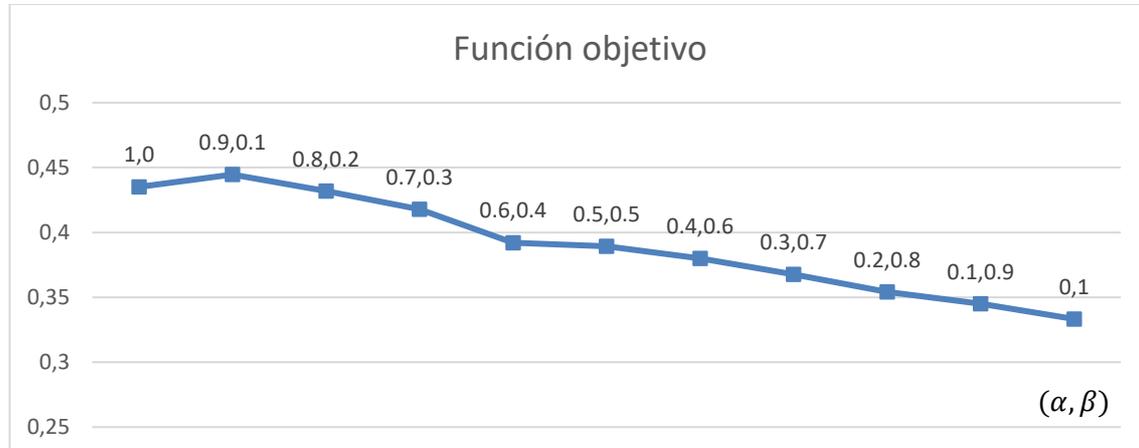


Figura 7-1: Valor obtenido para la función objetivo en función de los pesos.



Figura 7-2: Coste obtenido en función de los pesos.

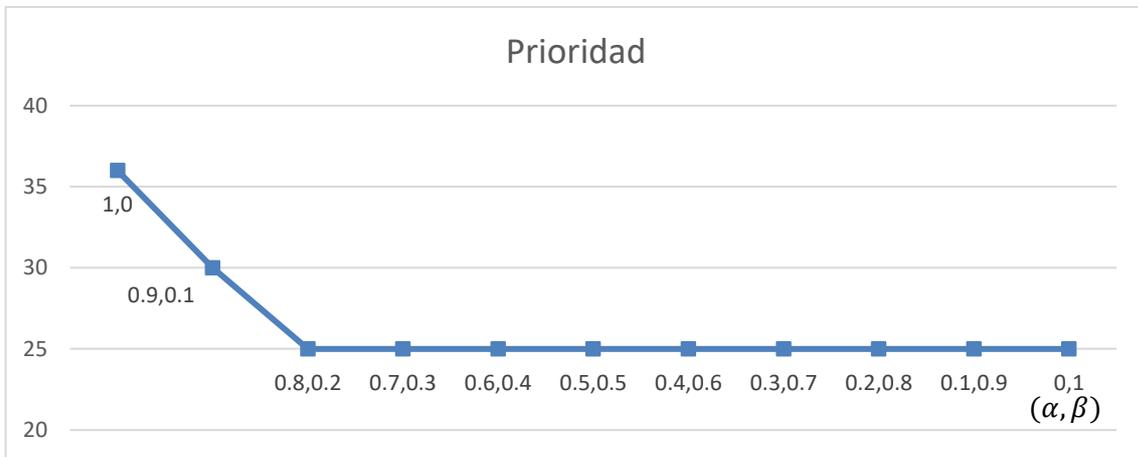


Figura 7-3: Prioridad obtenida en función de los pesos.

Con respecto al valor de la función objetivo se puede observar cómo decrece conforme  $\alpha$  decrece y  $\beta$  aumenta. Esto es debido a que el valor máximo  $\frac{c_i}{c}$  en el intervalo  $[0,1]$  es mayor que el valor máximo  $\frac{p_i}{p}$  en el intervalo  $[0,1]$ .

En relación con el coste, se puede observa cómo los valores son mas dispares en los extremos. En el caso  $\alpha = 1, \beta = 0$  se encuentra el valor mínimo de coste, ya que en este caso el problema solo se centra en minimizar costes sin tener en cuenta las prioridades. Por el contrario, en el caso  $\alpha = 0, \beta = 1$  se encuentra el valor máximo de coste, ya que aquí el problema se centra en minimizar las prioridades sin tener en cuenta el coste. En los puntos intermedios se observa un valor de coste estable y con poca variabilidad.

Por último, observando la evolución de las prioridades obtenidas se puede observar que hay un punto a partir del cuál el problema se estabiliza y ofrece el mínimo valor.

Para el análisis de resultados que se llevará a cabo en este capítulo se establecerá  $\alpha = 0.8$  y  $\beta = 0.2$ .

### 7.3. Batería de problemas

Para analizar el funcionamiento del algoritmo planteado se resolverán tres baterías de problemas, definidos por el número de clientes: 25, 50 y 100. Los problemas que se van a resolver son los correspondientes al tipo R2, generados por Solomon (Solomon, 1987) y extendidos por Homberger (Gehring & Homberger, 2002). Mientras que en los problemas de Solomon hay hasta 100 clientes, en los problemas extendidos pueden encontrarse hasta 1000 clientes.

En los problemas de tipo R2 los datos geográficos de los clientes han sido generados aleatoriamente y sus ventanas temporales son amplias en el horizonte, lo que permite que una ruta pueda atender a un mayor

número de clientes.

El formato que presenta estos problemas se ha explicado anteriormente en la sección de *Estructura de los datos*. Cabe destacar que se ha añadido una nueva columna para designar la prioridad de cada una de las localizaciones.

Para adaptar los datos al problema que se va a resolver se han agrupado los clientes en grupos de tres, de manera que ahora cada grupo de tres clientes se considera un cliente. Así, un cliente ahora pasa a tener tres localizaciones diferentes. Además, se han modificado las ventanas temporales para evitar solape entre ellas y a cada localización para cada cliente se le ha asignado una prioridad aleatoria. Se ha utilizado el problema R2 de Homburger de 1000 clientes (divididos en grupos de 3 quedan 333 clientes):

CUST NO.	XCOORD	YCOORD	DEMAND	READY TIME	DUE DATE	SERVIC E TIME	PRIORIT Y
0	250	250	0	0	7697	0	1
1	228	399	14	1	2565	10	3
1	325	56	13	2566	5130	10	2
1	290	145	5	5131	7695	10	1
2	340	291	22	1	2565	10	1
2	370	382	11	2566	5130	10	2
2	273	255	18	5131	7695	10	3
3	36	301	21	1	2565	10	1
3	117	178	28	2566	5130	10	3
3	295	283	26	5131	7695	10	2

Tabla 7-7: Extracto de un problema R2 adaptado al problema de este trabajo

Con respecto a la estructura de los datos, la primera columna indica el número de identificación del cliente. Es importante destacar que el cliente número 0 corresponde a los datos del depósito. La ventana temporal del depósito marca el intervalo total de operación, es decir, los vehículos deben servir a los clientes dentro del intervalo que marca el depósito, debiendo regresar al depósito antes del cierre. Las dos siguientes columnas hacen referencia a las coordenadas de las localizaciones. Las columnas quinta y sexta marcan el intervalo temporal de cada localización. Y la columna octava establece la prioridad a cada localización. La columna de demanda no se tiene en cuenta en este problema ya que la capacidad de los vehículos no es una restricción. A priori, el tiempo de servicio tampoco se considera en la resolución dado que se asume la entrega instantánea de los productos a los clientes, pero el algoritmo de resolución está desarrollado de manera que puede tenerlo en cuenta si el usuario lo desea.

## 7.4. Resolución de los problemas

Al inicializar los problemas, además de indicar el tipo de problema también se inicializan los parámetros del problema. En este caso se han dejado en su valor por defecto:

- $truckCost = 100$  u.m. (Coste de utilizar un vehículo)
- $truckSpeed = 60$  km/h (Velocidad media de los vehículos)
- $kmcost = 1$  u.m./km (Coste por cada kilómetro recorrido)

Las baterías de problemas se han separado por número de clientes. Para cada batería se ha obtenido el número de clientes correspondientes del mismo problema, es decir, del problema R2 de 1000 clientes, se ha extraído la información para 25, 50 y 100 clientes respectivamente. El problema tipo R2 contiene 10 variantes, por lo que cada variante será un escenario dentro de la batería de problemas. Por tanto, se evaluarán 30 escenarios (10 escenarios de tipo R2 por 3 baterías). Cada escenario se ha ejecutado un total de 10 veces y se presentarán los valores obtenidos por la función objetivo, además del coste y la prioridad para cada escenario como sigue:

- Valor medio en las 10 ejecuciones

- Mejor valor en las 10 ejecuciones
- Desviación en las 10 ejecuciones
- Tiempo medio de ejecución.

Los resultados obtenidos se muestran a continuación:

Problema	25 clientes									
	Función objetivo			Coste			Prioridad			T(s)
	Promedio	Min	Desv	Promedio	Min	Desv	Promedio	Min	Desv	
01	0.424	0.413	1.59%	5223.09	5014.87	2.39%	25.9	25	7.38%	12.84
02	0.413	0.388	2.96%	5195.87	4505.05	5.89%	26.6	25	12.05%	13.32
03	0.428	0.419	1.65%	5245.62	5115.03	1.95%	25.7	25	8.61%	13.23
04	0.408	0.393	2.64%	5530.29	5304.27	3.37%	25.2	25	2.51%	11.44
05	0.411	0.402	1.66%	5197.58	4998.00	2.25%	25.1	25	1.26%	14.01
06	0.411	0.405	1.08%	5251.48	5160.81	1.28%	25.0	25	0.00%	11.23
07	0.407	0.394	1.89%	5223.87	4967.22	3.02%	25.8	25	6.54%	14.41
08	0.431	0.418	1.83%	5614.30	5066.46	6.11%	27.7	25	16.85%	13.07
09	0.385	0.370	2.37%	4574.93	4363.56	2.87%	25.0	25	0.00%	8.97
10	0.398	0.387	1.71%	4815.47	4533.15	2.86%	25.5	25	4.98%	12.20

Tabla 7-8: Resultados para 25 clientes.

Problema	50 clientes									
	Función objetivo			Coste			Prioridad			T(s)
	Promedio	Min	Desv	Promedio	Min	Desv	Promedio	Min	Desv	
01	0.356	0.346	1.39%	8533.94	8260.79	1.80%	50.30	50	1.89%	47.82
02	0.364	0.355	1.26%	8607.28	8347.22	1.89%	51.50	50	7.39%	57.71
03	0.365	0.351	2.17%	9037.99	8165.41	3.94%	51.10	50	4.56%	46.88
04	0.363	0.347	3.19%	9190.26	8131.28	5.58%	51.10	50	6.81%	56.16
05	0.345	0.329	2.77%	8228.71	7767.57	3.43%	50.00	50	0.00%	49.53
06	0.357	0.348	1.46%	8941.46	8528.38	2.42%	50.60	50	3.75%	51.26
07	0.349	0.337	2.78%	8553.50	8177.82	3.42%	50.70	50	4.37%	50.53
08	0.365	0.355	2.21%	9369.44	8771.76	3.26%	50.60	50	3.12%	50.35
09	0.367	0.346	3.04%	8577.33	8020.42	4.00%	52.70	50	9.21%	52.02
10	0.357	0.348	1.65%	8821.00	8526.94	2.03%	50.00	50	0.00%	49.84

Tabla 7-9 : Resultados para 50 clientes.

Problema	100 clientes									
	Función objetivo			Coste			Prioridad			T(s)
	Promedio	Min	Desv	Promedio	Min	Desv	Promedio	Min	Desv	
01	0.353	0.341	2.37%	16851.63	16159.34	3.55%	102.80	100	5.81%	206.73
02	0.353	0.334	3.56%	16806.40	15673.85	4.45%	100.70	100	2.20%	202.48
03	0.353	0.343	1.54%	17144.93	16683.24	2.15%	103.60	100	5.94%	210.24
04	0.351	0.336	2.10%	17229.21	16397.92	2.82%	101.40	100	4.37%	204.38
05	0.343	0.331	3.07%	16270.05	15392.24	4.25%	102.80	100	8.61%	208.59
06	0.354	0.327	3.27%	16716.27	15018.89	5.36%	104.20	100	12.75%	215.06
07	0.362	0.341	3.54%	17520.29	15962.86	5.97%	106.90	100	16.10%	203.85
08	0.355	0.349	2.01%	17191.82	16492.24	2.90%	104.80	100	7.79%	199.28
09	0.365	0.343	2.74%	17151.35	15961.33	3.02%	102.80	100	8.61%	190.37
10	0.348	0.335	3.03%	16610.44	15859.48	3.70%	102.50	100	7.71%	209.98

Tabla 7-10: Resultados para 100 clientes

A la hora de analizar el tiempo de ejecución es importante tener en cuenta las características del equipo utilizado para llevar a cabo las diferentes pruebas. En este caso el equipo cuenta con las características siguientes:

- Sistema Operativo: Windows 10
- CPU: Intel Core i7 2.6 – 5 GHz
- RAM: 16GB 2666 MHz DDR4

## 7.5. Análisis de las soluciones

En esta sección se realizará un análisis para los resultados obtenidos para las 100 muestras de cada batería de problemas:

- 25 clientes

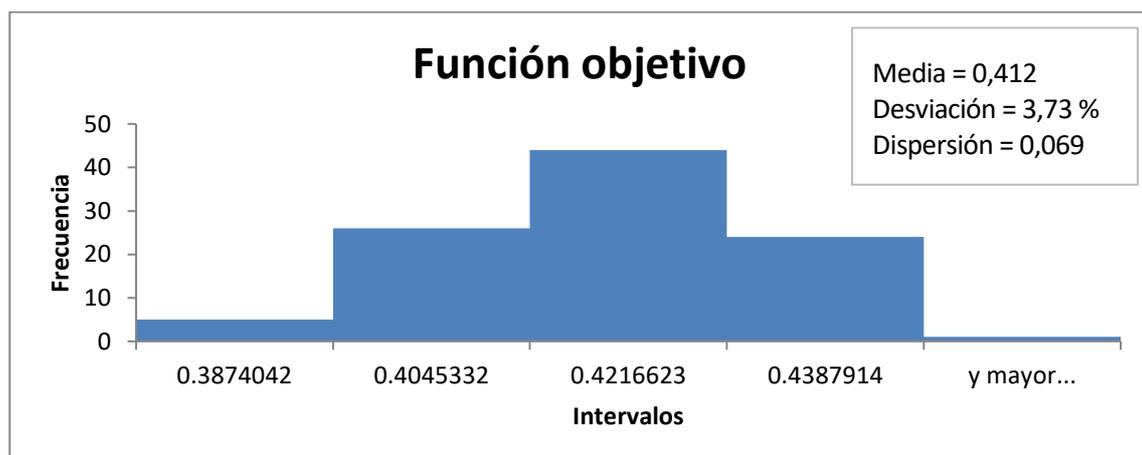


Figura 7-4: Histograma de los resultados obtenidos por la función objetivo (25 clientes).

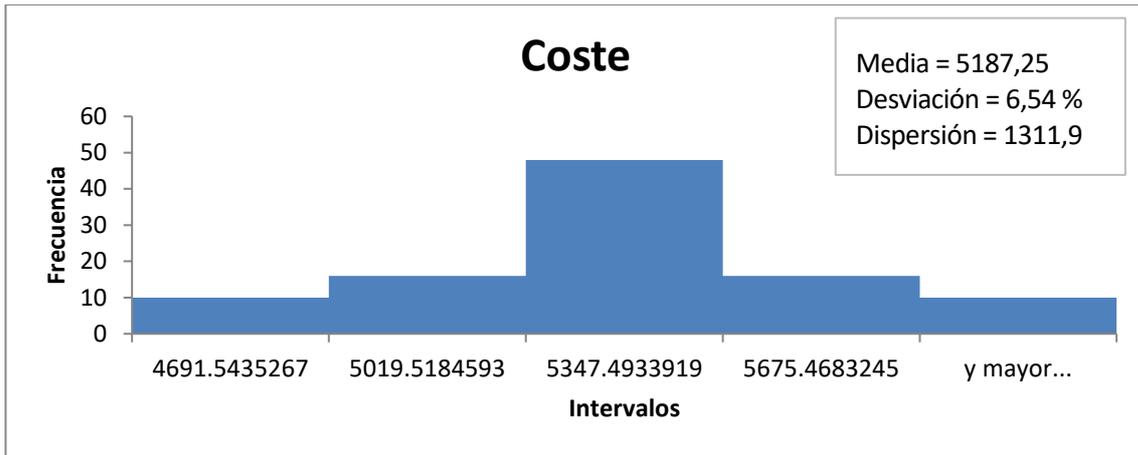


Figura 7-5: Histograma de los costes obtenidos (25 clientes).

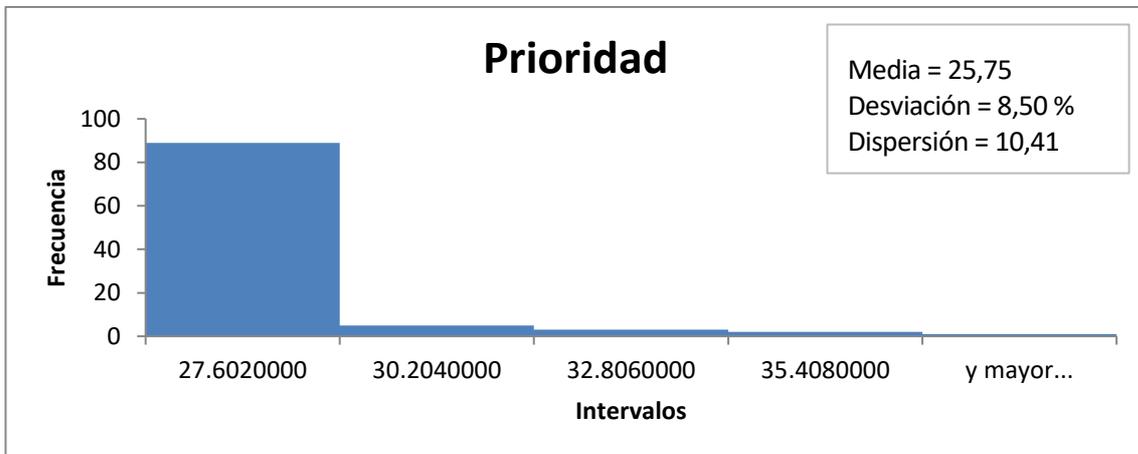


Figura 7-6: Histograma de las prioridades obtenidas (25 clientes).

- 50 clientes

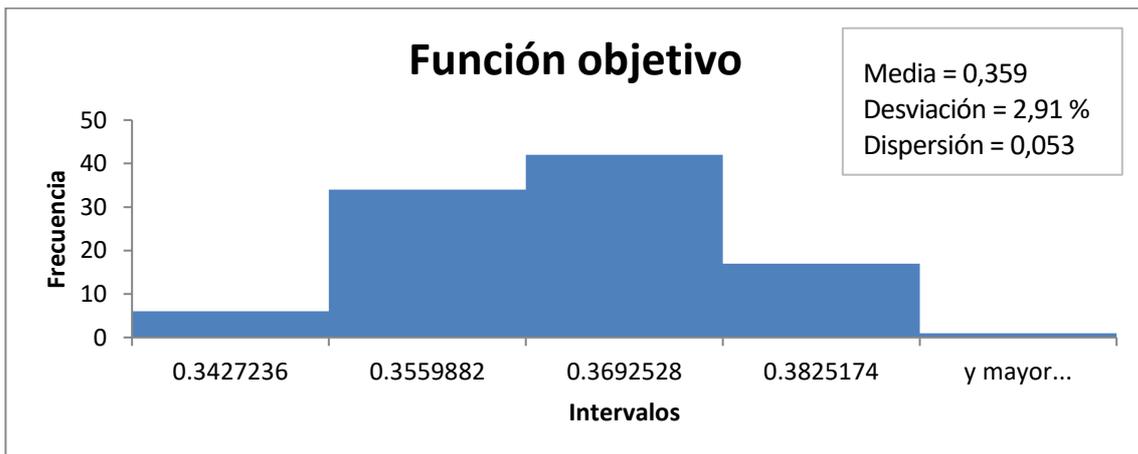


Figura 7-7: Histograma de los resultados obtenidos por la función objetivo (50 clientes).

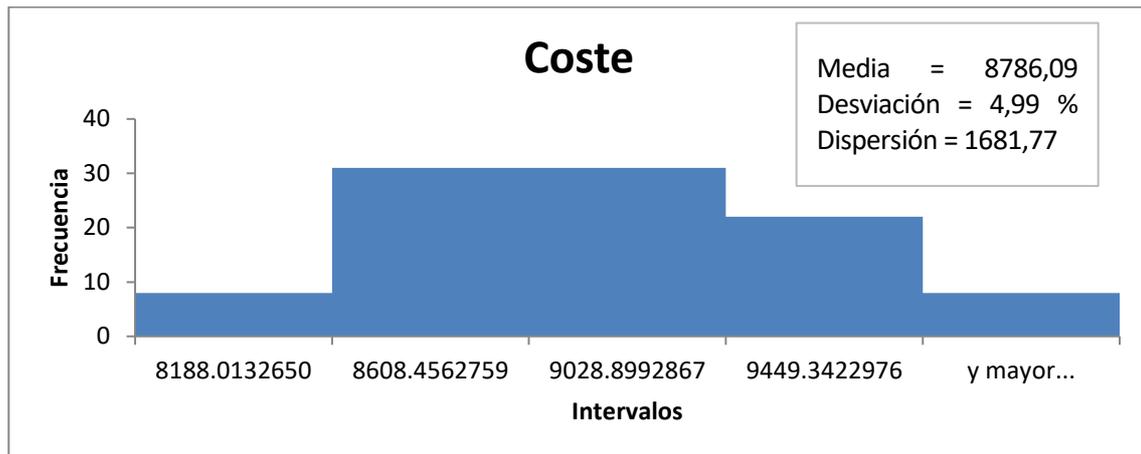


Figura 7-8: Histograma de los costes obtenidos (50 clientes).

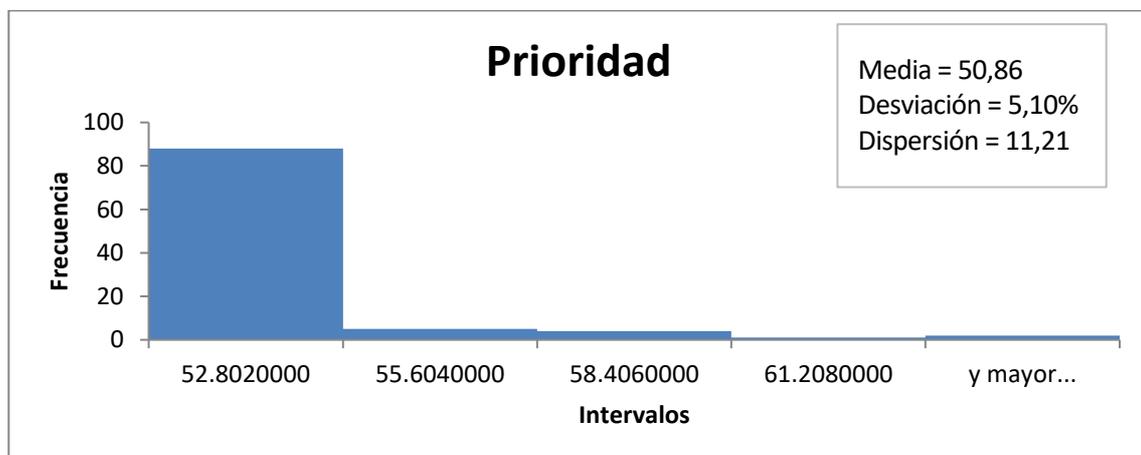


Figura 7-9: Histograma de las prioridades obtenidas (50 clientes).

- 100 clientes

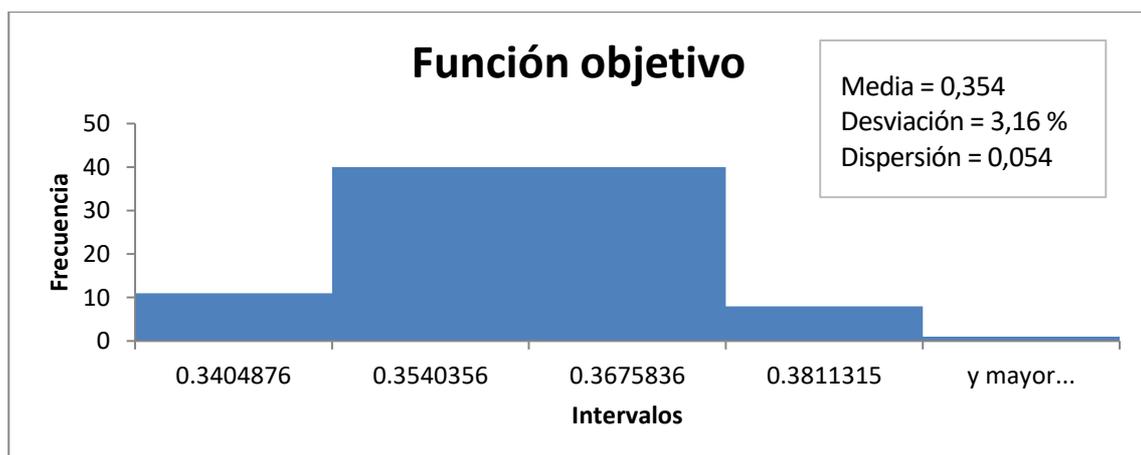


Figura 7-10: Histograma de los resultados obtenidos por la función objetivo (100 clientes).

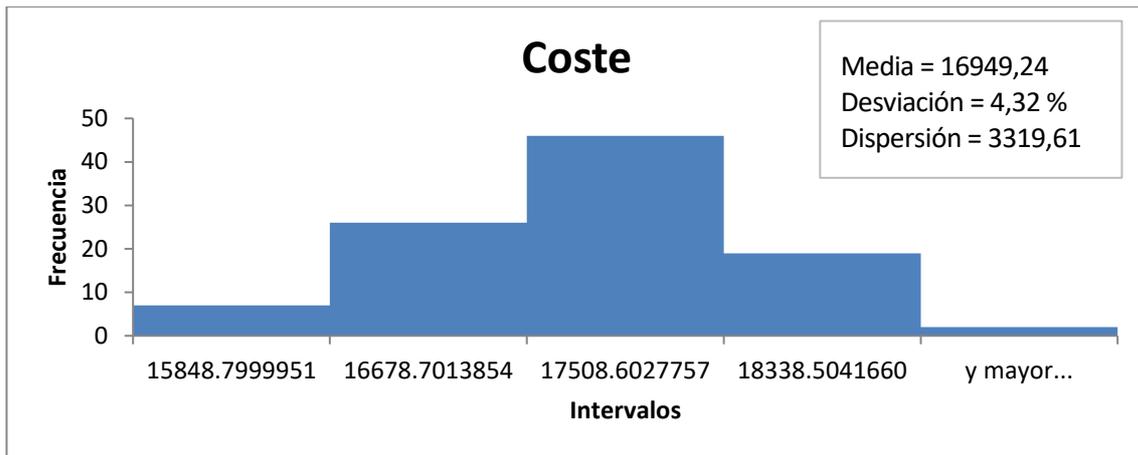


Figura 7-11: Histograma de los costes obtenidos (100 clientes).

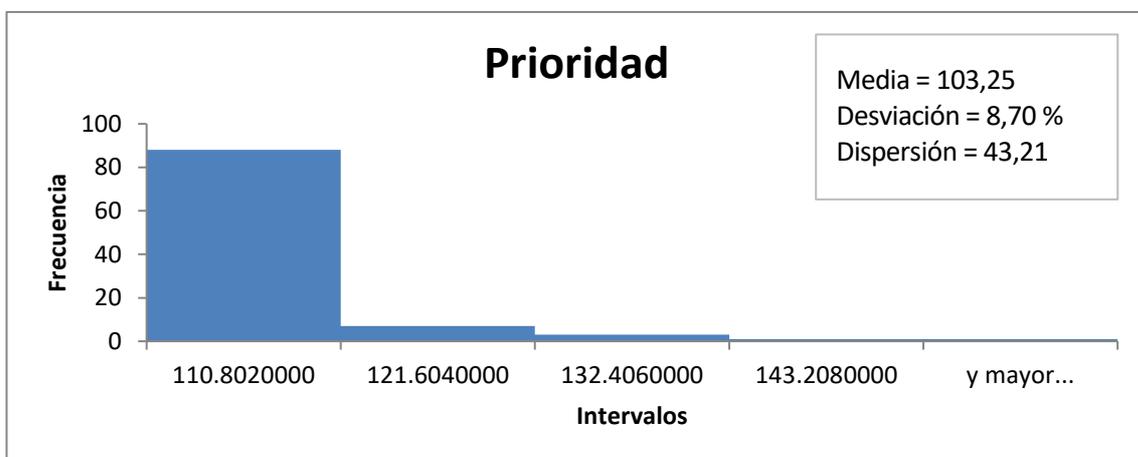


Figura 7-12: Histograma de las prioridades obtenidas (100 clientes).

En las tres baterías de problemas se pueden observar cómo la mayor parte de los valores de la función objetivo ocurren en torno al intervalo intermedio. La desviación obtenida en las tres baterías es lo suficientemente pequeña como para garantizar resultados estables en diferentes problemas con datos diferentes, inicializados aleatoriamente. Al igual que en los histogramas de los valores de la función objetivo, en los histogramas de los costes obtenidos pueden observarse el mismo comportamiento, con la mayor parte de los resultados en sus intervalos intermedios. Por otro lado, en los histogramas de las prioridades obtenidas, la mayor parte de los resultados se obtienen en el primer intervalo. Esto significa que el problema siempre trata de minimizar las prioridades y encuentra soluciones próximas al mínimo.

El número de muestras de cada batería de problemas (100) se considera suficiente para mostrar la distribución de los resultados. En este caso se observan distribuciones simétricas en los histogramas de los valores obtenidos por la función objetivo, asemejándose a distribuciones normales.

Si las distribuciones hubiesen resultado asimétricas, se estarían generando resultados muy dispersos con alta variabilidad, por lo que habría que ajustar el modelo para generar mejores resultados.

Por tanto, de las muestras analizadas se puede concluir que el modelo genera resultados estables, poco dispersos y con desviaciones pequeñas para un mismo tipo de problema y número de clientes, por lo que también se puede concluir que la calibración del modelo resulta adecuada.

## 7.6. Ejemplo de la solución

Para mostrar un ejemplo de la solución se resolverá un problema tipo R2 de Homberger con 25 clientes:

Número de vehículos: 5

Rutas:

[0, 3, 5, 16, 2, 20, 22, 23, 18, 0] Con coste 1850.71 u.m en 1750.71 minutos con salida en 0.00 minutos tras apertura.

[0, 11, 6, 8, 24, 0] Con coste 777.52 u.m en 677.52 minutos con salida en 4893.31 minutos tras apertura.

[0, 25, 4, 9, 1, 10, 17, 7, 0] Con coste 980.69 u.m en 880.69 minutos con salida en 2315.85 minutos tras apertura.

[0, 19, 15, 21, 14, 0] Con coste 745.30 u.m en 645.30 minutos con salida en 2354.14 minutos tras apertura.

[0, 12, 13, 0] Con coste 286.12 u.m en 186.12 minutos con salida en 5066.67 minutos tras apertura.

Con un coste total de 4640.34 u.m.

La función objetivo para esta solución es 0.40.

Las localizaciones seleccionadas son:

{1: (325, 56), 2: (340, 291), 3: (36, 301), 4: (436, 123), 5: (222, 499), 6: (398, 476), 7: (179, 286), 8: (371, 488), 9: (440, 77), 10: (309, 161), 11: (447, 383), 12: (187, 263), 13: (161, 276), 14: (168, 109), 15: (45, 166), 16: (348, 476), 17: (295, 247), 18: (15, 11), 19: (40, 278), 20: (322, 290), 21: (34, 169), 22: (393, 98), 23: (303, 78), 24: (215, 293), 25: (488, 173)}

Porcentaje de prioridades seleccionadas:

P1	P2	P3
0.8	0.16	0.04

Tiempo de ejecución: 16.76 segundos

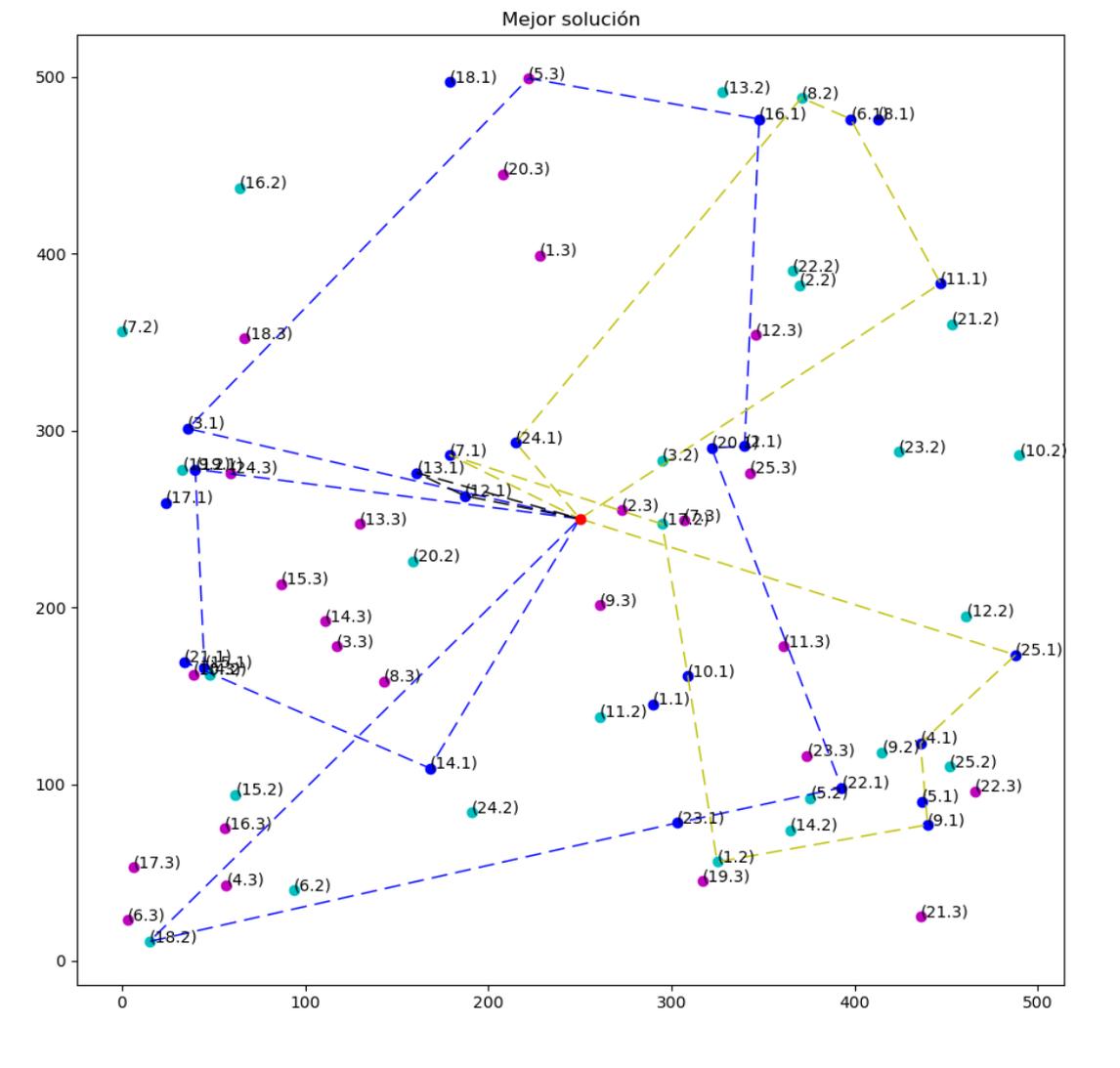


Tabla 7-11: Ejemplo solución para problema R2 de Solomon

En primer lugar, se indica el número de vehículos utilizados. A continuación, cada una de las rutas indicando la secuencia de clientes visitados, su coste, tiempo y momento en el que se debe iniciar la ruta. Tras la lista de rutas se muestra el coste total de la solución y el valor obtenido por la función objetivo para esta solución. Con respecto a las localizaciones seleccionadas, se indica qué localización se ha seleccionado para cada cliente y el

porcentaje (tanto por uno) de las prioridades que se han seleccionado. En el gráfico se observa visualmente las rutas obtenidas.

## 8 CONCLUSIONES

---

Antes de exponer las conclusiones se hará un breve repaso a la hoja de ruta seguida por este trabajo para llegar hasta aquí:

- En primer lugar, se hace un repaso del comercio electrónico a través de su historia, desde sus inicios hasta lo que es hoy en día, pasando por los hitos más relevantes en su evolución. A su vez, se presenta cómo este sector lleva a cabo sus protocolos de reparto a través de la resolución de problemas VRP.
- A continuación, se plantea el problema que se resuelve en el presente trabajo detallando sus características. Además, se introducen las metodologías que se utilizan en este tipo de problema y se detalla la metodología elegida para la resolución en este trabajo, la *Búsqueda Tabú*.
- Posteriormente, se explica lo que puede considerarse el núcleo de este trabajo, la implementación del algoritmo de resolución. Se detallan conceptualmente los procesos que sigue el algoritmo y su codificación en el lenguaje de programación *Python*.
- Una vez que se ha implementado el algoritmo y se ha calibrado, se realizan diferentes pruebas con el fin de analizar las soluciones obtenidas para medir el rendimiento del método.

El objetivo principal de este trabajo es proponer una solución que ofrezca más flexibilidad al cliente de manera que aumente su confianza para recibir su producto en la localización deseada en el momento deseado. De esta manera se reduce la probabilidad de realizar una entrega fallida en el primer intento. La ausencia del destinatario es la mayor causa de las no entregas en el primer intento. Para resolver este problema se ha propuesto la resolución de un VRP (problema de rutado de vehículos) que incluye la posibilidad de seleccionar tres localizaciones diferentes con ventanas temporales y asignarles orden de prioridad, de manera que no solo se reduce la probabilidad de no entregar el producto, sino que también se maximiza la satisfacción del cliente teniendo en cuenta sus prioridades.

Para resolver el problema propuesto se ha utilizado la metaheurística *Búsqueda Tabú*, combinada con un proceso de búsqueda variable *VNS* que ha permitido diversificar la búsqueda. Tras la aplicación del método propuesto y observando los resultados se puede concluir:

- Las soluciones obtenidas con este método se consideran válidas y aceptables en tiempos computacionales relativamente cortos, lo que hace el método eficiente. Propone rutas que cumplen las restricciones teniendo en cuenta las prioridades de los clientes, siendo el porcentaje de las localizaciones seleccionadas con primera prioridad siempre muy significativo.
- No solo se obtienen buenas soluciones, sino que también, como se puede ver en la sección de resultados, el análisis de las soluciones constata que las soluciones obtenidas se distribuyen de manera normal, por lo que se puede definir el método aplicado como un método estable y seguro. Esto demuestra la eficacia del método propuesto para este tipo de problemas.

En conclusión, este trabajo presenta un método eficaz y eficiente para la resolución del problema propuesto para satisfacer las necesidades que los protocolos de reparto en el sector del comercio electrónico presentan.

## 8.1. Desarrollos futuros y aplicaciones

Este tipo de problema presenta actualmente un gran catálogo de desarrollos y aplicaciones. Este trabajo utiliza con una metodología muy concreta, pero es importante destacar que puede ser ampliado y desarrollado de diversas formas. En este caso las ideas que se presentan estarán enfocadas al desarrollo de una logística más sostenible.

Por un lado, en términos de desarrollos futuros, cabe destacar que el algoritmo de *Búsqueda Tabú* podría ser mejorado funcionalmente aplicando estrategias diversas de intensificación y diversificación, así como diversas técnicas para obtener la solución inicial, de manera que se obtenga una solución inicial muy aceptable y reduzca el tiempo de búsqueda. Para ampliar el alcance del problema podría introducirse la optimización simultánea de la eficiencia energética de las rutas, de manera que se tenga en cuenta el consumo energético y el impacto ambiental de los vehículos para minimizarlos en la misma función objetivo. Esto puede considerarse una combinación del problema Green VRP con el problema 3TW-VRP.

Por otro lado, en términos de aplicabilidad, más allá de su aplicación trivial en protocolos de repartos en el área logística actual, se podría destacar alguna aplicación en un entorno protagonizado por la mejora continua, el rápido desarrollo tecnológico y la preocupación por la sostenibilidad de las soluciones. Este problema sería bastante útil en un entorno tecnológicamente desarrollado como puede ser una *smart city* donde se podría tener información en tiempo de real de factores influyentes como tráfico, obras, señalizaciones o la calidad del aire de manera que las rutas pueden actualizarse de manera instantánea.

# REFERENCIAS

---

- Balas, E., & Toht, P. (1983). *Branch and bound methods for the traveling salesman problem*. Carnegie Mellon University Graduate School of Industrial Administration.
- Balinski, M., & Quandt, R. (1964). On a Integer Program for a Delivery Problem. *Operations Research*, 12(2), 187-376.
- Beasley, J. (1983). Route first-Cluster second methods for vehicle routing. *Omega*, 11(4), 403-408.
- Chian, W., & Rusell, R. (1997). A Reactive Tabu Search Metaheuristic for the Vehicle Routing Problem with Time Windows. *INFORMS Journal on Computing*, 9(4), 417-430.
- Christofides, N., Mingozzi, A., & Toth, P. (1979). *The Vehicle Routing Problem*.
- Clarke, G., & Wright, J. (1964). Scheduling of Vehicle Routing Problem from a Central Depot to a Number of Delivery Points. *Operations Research*, 12, 568-581.
- CNMC. (04 de octubre de 2019). *CNMC*. Obtenido de <https://www.cnmc.es/prensa/comercio-electr%C3%B3nico-IT2019-cnmc>
- Colomi, A., Dorigo, M., & Maniezzo, V. (1991). Distributed Optimization by Ant Colonies. En F. Varela , & P. Bourguine, *Proceedings of the European Conference on Artificial Life* (págs. 134-142). Paris: ECAL'91.
- Cordone, R., & Calvo, R. (2001). A Heuristic for the Vehicle Routing Problem with Time Windows. *Journal of Heuristics*, 7(2), 107-129.
- Devore, J. (1991). *Probability & Statistics for Engineering and the Sciences, Eighth Edition*. Pacific Grove, California: Books/Cole Publishing Company.
- Eilon, S., Watson, C., Christofides, N., & de Neufville, R. (1974). Distribution management: Mathematical modelling and practical analysis. *IEEE Transactions on Systems Man and Cybernetics*, 21(6), 589-589. doi:10.1109/TSMC.1974.4309370
- Escudero-Santana, A., & Aparicio-Ruiz, P. (2018). *Búsqueda Local*. Sevilla: Modelado y Optimización de Problemas de Gestión. Universidad de Sevilla.
- Escudero-Santana, A., & Aparicio-Ruiz, P. (2018). *Python: Orientación a objetos*. Sevilla: Modelado y optimización de problemas de gestión. Universidad de Sevilla.
- Feo, T., & Resende, M. (1989). A probabilistic heuristic for a computationally difficult set covering problem. *Operations Research Letters*, 8, 67-71.
- Fisher, M., & Jaikumar, R. (1981). A generalized assignment heuristic for vehicle routing. *Networks*, 11, 109-124.
- Gehring, H., & Homberger, J. (2002). Parallelization of a two-phase metaheuristic for routing problems with time windows. *J. Heuristics* , 8, 251-276.
- Gispert, B. (04 de enero de 2020). *La Vanguardia*. Obtenido de

- <https://www.lavanguardia.com/economia/20200104/472683449884/comercio-electronico-espana-ecommerce-datos.html>
- Glover, F. (1986). Future Paths for Integer Programming and Links to Artificial Intelligence. *Computers and Operations Research*, 5, 533-549.
- Golan, P. (2020 de enero de 2020). *Shopify*. Obtenido de <https://www.shopify.es/blog/12621205-los-5-tipos-de-comercio-electronico#example1>
- Gutiérrez, A. (24 de octubre de 2018). *ecommercenews*. Obtenido de <https://ecommerce-news.es/la-evolucion-de-los-pagos-del-trueque-al-pago-invisible-87386>
- He, M. (13 de mayo de 2019). *2open*. Obtenido de <http://www.2open.es/los-mayores-hitos-de-la-historia-del-comercio-online/>
- Hipertextual*. (s.f.). Obtenido de <https://hipertextual.com/presentado-por/visa/historia-comercio-electronico>
- Holland, J. (1992). *Adaptation in Natural and Artificial Systems*. Bradford Books.
- Kennedy, J., & Eberhart, R. (1995). Particle swarm optimization. *International Conference on Neural Networks*, (págs. 1942-1948). Perth.
- Kirkpatrick, S., Gelatt, C., & Vecchi, M. (1983). Optimization by Simulated Annealing. *Science*, 220, 671-680.
- Laporte, G. (1991). The Vehicle Routing Problem: An overview of exact and approximate algorithms. *European Journal of Operational Research*, 59, 345-358.
- Liong, C., Wan, I., & Omar, K. (2008). Vehicle Routing Problem: Models and Solutions. *Journal of Quality Measurement and Analysis*, 4, 205-218.
- Lüer, A., Benavente, M., Bustos, J., & Venegas, B. (2009). El problema de rutas de vehículos: Extensiones y métodos de resolución, estado del arte. *WORKSHOP INTERNATIONAL*. Temuco.
- Lynkoo*. (22 de octubre de 2012). Obtenido de <https://www.lynkoo.com/la-historia-del-comercio-electronico/>
- Mladenovic, N., & Hansen, P. (1997). Variable neighborhood search. *Computers & Operations Research*, 24(11), 1097-1100.
- Rocha, L., González, C., & Orjuela, J. (2011). Una revisión al estado del arte del problema de ruteo de vehículos: Evolución histórica y métodos de solución. *Ingeniería*, 16(2), 35-55.
- Rodríguez, E. (s.f.). *Actualidad eCommerce*. Obtenido de <https://www.actualidadecommerce.com/ventajas-y-desventajas-del-ecommerce-frente-al-comercio-tradicional/>
- Ryan, D., Hjorring, C., & Glover, F. (1993). Extensions of the Petal Method for Vehicle Routing. *J Oper Res Soc*, 44, 289-296.
- Salvador, J., Escudero-Santana, A., & Rodríguez, M. (2018). *Tendencias en el Comercio Electrónico. Estudio de las preferencias de los usuarios*. Sevilla: Trabajo Fin de Grado. Grado en Ingeniería de Organización Industrial. Universidad de Sevilla. Departamento de Organización Industrial y Gestión de Empresas II.
- Smith, A. (1776). *La riqueza de las naciones*. William Strahan.
- Solomon, M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research*, 35(2), 254-265.
- Toth, P., & Vigo, D. (2002). *The Vehicle Routing Problem*. Society of Industrial and Applied Mathematics. doi:10.1137/1.9780898718515
- Trujillo, S. (02 de diciembre de 2019). *IEBS*. Obtenido de Previsiones y tendencias de eCommerce para 2020: <https://www.iebschool.com/blog/previsiones-y-tendencias-de-ecommerce-para-2020-e-commerce/>
- Uribe, C., & Escudero-Santana, A. (2017). *Resolución de un VRP con ventanas móviles mediante el uso del algoritmo Cuckoo Search*. Sevilla: Trabajo Fin de Máster. Máster en Ingeniería Industrial. Universidad de Sevilla. Departamento de Organización Industrial y Gestión de Empresas II.

*Wikipedia*. (s.f.). Obtenido de [https://es.wikipedia.org/wiki/Comercio\\_electr%C3%B3nico](https://es.wikipedia.org/wiki/Comercio_electr%C3%B3nico)

Xiao, Y., Zhao, Q., Kaku, I., & Xu, Y. (2012). Development of a fuel consumption optimization model for the capacited vehicle routing problem. *Computers and Operations Research* , 37(7), 1419-1431.

Xu, J., Chiu, S., & Glover, F. (1998). Fine-tuning a Tabu Search Algorithm with Statistical Tests. *International Transactions in Operational Research* , 5(3), 233-244.



