Trabajo Fin de Master
Máster en Ingeniería Industrial

Desarrollo y programación de una aplicación multihilo para planificación de rutas de drones

Autor: Miguel Antonio Sanz Pérez
Tutor: José Miguel León Blanco

Dpto. Organización Industrial y Gestión de Empresas I
Escuela Técnica Superior de Ingeniería

Sevilla, 2021

Trabajo Fin de Máster
Máster en Ingeniería Industrial

Desarrollo y programación de una aplicación multihilo para planificación de rutas de drones

Autor:
Miguel Antonio Sanz Pérez

Tutor:
José Miguel León Blanco
Profesor contratado doctor

Dpto. de Organización Industrial y Gestión de Empresas I
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2021

Trabajo Fin de Máster: Desarrollo y programación de una aplicación multihilo para planificación de rutas de drones

Autor: Miguel Antonio Sanz Pérez

Tutor: José Miguel León Blanco

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2021

Agradecimientos

Agradecer a todos mis amigos y compañeros que me han acompañado durante este periodo de mi vida tan bueno y fructífero, pero que sin embargo es la base para un futuro que espero me permita vivir nuevas experiencias y conocer a muchas personas que me aporten tanto como ellos me han aportado a mí.

Mención especial a todos los profesores que me han ayudado a llegar hasta aquí y haberme inculcado la importancia de la educación, desde primaria hasta la universidad.

Y como no podría ser de otra manera, a mi familia; mi padre, mi madre y mi hermana por apoyarme en todo momento y confiar en mi incluso cuando ni yo confiaba en mis posibilidades. Por aconsejarme y educarme para llegar a lo que soy y lo que el día de mañana seré. Muchas gracias.

Miguel Antonio Sanz Pérez

Sevilla, 2021

Resumen

Este trabajo trata de profundizar en los conocimientos existentes sobre el uso combinado de vehículos terrestres como camiones de reparto y nuevos vehículos aéreos no tripulados como lo son los UAVs o los drones en tareas de reparto a domicilio. Este tipo de tareas han ganado popularidad en la última década debido al incremento de las ventas a través de Internet y su correspondiente necesidad de enviar los paquetes a sus clientes. El bajo coste de este tipo de vehículos, sumado al ahorro de tiempo por trayecto al evitar el tráfico en las ciudades ha permitido que se hayan elaborado numerosos trabajos que buscan modelar y resolver este tipo de problemas.

La planificación de las rutas es un problema NP-Hard, por lo que para problemas a gran escala (cientos e incluso miles de nodos) es necesario emplear algoritmos heurísticos que permitan hallar una solución cercana al óptimo sin que el tiempo de computación se eleve exponencialmente. Por ello, se ha tomado como referencia el estado del arte actual y se ha tratado de mejorar un algoritmo heurístico mediante el uso de ejecución en paralelo.

La ejecución en paralelo o también conocida como multihilo permite acelerar la ejecución de algoritmos con alto coste computacional de tal forma que se emplean los recursos computacionales de forma óptima. Para conseguir esta reducción del tiempo de ejecución de un algoritmo, es necesario en cambio sacrificar otros aspectos de la heurística original, y en este trabajo se tratará de medir cuales son las ventajas y desventajas de emplear esta técnica.

Abstract

This work tries to deepen the existing knowledge on the combined use of land vehicles such as delivery trucks and new unmanned aerial vehicles such as UAVs or drones in home delivery tasks. These types of tasks have gained popularity in the last decade due to the increase in sales over the Internet and the corresponding need to send the packages to their customers. The low cost of this type of vehicles, added to the saving of time per journey by avoiding traffic, has led to the development of numerous works that seek to model and solve this type of problems.

Route planning is a NP-Hard problem, so for large-scale problems (hundreds and even thousands of nodes) it is necessary to use heuristic algorithms that allow finding an optimal solution without the computation time rising exponentially. Therefore, the current state of art has been taken as a reference and has been made an attempt to improve a heuristic algorithm through the use of parallel execution.

The execution in parallel or also known as multithreading allows to accelerate the execution of algorithms with high computational cost in such a way that computational resources are used optimally. To achieve this reduction in the execution time of an algorithm, it is necessary to sacrifice other aspects of the heuristics. This work tries to measure the advantages and disadvantages of using this technique.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

Glosario

UAV	Unmanned Aerial Vehicle
TSP	Travelling Salesman Problem
VRP	Vehicle Routing Problem
NP	Nondeterministic Polynomial
TW	Time Window
MILP	Mixed Integer Linear Programming
ILP	Integer Linear Programming
FSTSP	Flying Sidekick Traveling Salesman Problem
mFSTSP	multiple Flying Sidekick Traveling Salesman Problem
MVDRP	Multi Visit Drone Routing Problem

1 INTRODUCCIÓN

En las últimas décadas, la aparición y desarrollo de los robots móviles como los UAVs o los drones, ha provocado que múltiples empresas y organismos traten de innovar en su forma de producir o de dar servicios a sus usuarios mediante el uso de estas tecnologías. Este tipo de vehículos no tripulados son ya una realidad en multitud de ámbitos como en la aviación militar o la automatización de almacenes de paquetería.

La aplicación de robots móviles voladores de corto rango y alimentados por baterías a tareas de reparto de paquetes se ha comenzado a contemplar como una posibilidad en la última década, consiguiéndose grandes avances en la planificación de rutas óptimas que deben ser recorridas por estos vehículos. Sin embargo, el uso únicamente de drones en entornos urbanos para este tipo de tareas presenta ciertos inconvenientes que se verán a lo largo de este trabajo. Es por ello, que surge la necesidad de combinar el uso de drones que agilicen las tareas de reparto con vehículos terrestres (camiones) que sirvan para realizar tanto tareas de reparto como de apoyo a los drones.

1.1 Objetivo

El presente proyecto trata de analizar y mejorar un problema de actualidad como es el reparto de paquetes individuales a pequeños clientes. Estos clientes pueden ser domicilios o negocios que adquieran productos a una empresa de distribución para que esta a través de sus medios de transporte les haga llegar estos paquetes. En este caso nos centraremos en el reparto de estos paquetes en núcleos urbanos donde exista un almacén desde el que salgan y al que lleguen los vehículos de distribución o mensajería. Este problema se modelará como un sistema multiagente en el que los agentes representan los drones y el camión empleados para las tareas de reparto

Por tanto, el objetivo del trabajo es realizar un estudio acerca de las distintas formulaciones y aproximaciones que se han realizado en los últimos años y seleccionar un problema que tenga aplicabilidad práctica para proceder a su implementación. En la actualidad, el esfuerzo principal se centra en diseñar algoritmos heurísticos que traten de alcanzar una ruta lo más cercana al óptimo sin que esto conlleve un coste computacional muy elevado. Para este trabajo, se ha buscado realizar una mejora de estos algoritmos heurísticos, no desde un punto de vista de la optimización de la ruta obtenida, sino desde la eficiencia computacional de la misma.

Para conseguir tal objetivo, se ha planteado la segregación de un algoritmo heurístico en distintas partes que permitan ser ejecutadas en paralelo mediante el uso de múltiples hilos. Si bien esta técnica permitirá mejorar el cálculo de las rutas óptimas en cuanto al tiempo de ejecución, también será necesario realizar un estudio sobre el efecto que tiene esta implementación sobre la precisión de los resultados obtenidos respecto al óptimo y respecto a los resultados obtenidos en caso de no emplear dicha técnica.

Todo esto, permitirá conocer si este tipo de técnicas son aplicables a los problemas de optimización de rutas o en caso contrario no es una buena metodología. Para ello, se plantearán más adelante distintos escenarios donde se evaluará la efectividad y la eficiencia de la implementación

1.2 Alcance

Una vez visto el objetivo del presente trabajo, se procederá a realizar una breve descripción del alcance del mismo. El trabajo abarcará el estudio del efecto que tienen distintas implementaciones, tanto mediante un algoritmo exacto, como por una heurística, ejecutándose esta última en paralelo. Además, se realizarán simulaciones para los distintos parámetros que permiten modelar tanto la heurística como para distintos tipos de problemas:

- Número de nodos y distancia entre ellos
- Número de UAVs
- Tipos de UAVs (velocidad, capacidad, autonomía)
- Iteraciones realizadas por la heurística

En cambio, se considerará el mismo modelo de consumo de combustible para todas las simulaciones (modelo no lineal), puesto que su efecto sobre el tiempo de ejecución es mínimo. Tampoco se le incluirán modificaciones a la heurística que introduzcan nuevos parámetros como la posibilidad de realizar operaciones tanto de entrega como de recogida, o la posibilidad de emplear múltiples almacenes de origen/destino. La implementación en paralelo de la heurística tiene como único objetivo mejorar los tiempos de ejecución del algoritmo, reduciéndolos en lo máximo posible, por lo que no se tendrá en cuenta el efecto que dicha modificación tenga sobre el valor óptimo alcanzado a la hora de diseñar la implementación, aunque si se comentará cuál es el efecto sobre dicho valor.

Otra simplificación introducida es que el número de camiones será siempre uno, dicho vehículo tendrá que estar disponible en el almacén para que comience la operación de reparto, eliminando la posibilidad de que los UAVs realicen operaciones de entrega cuando dicho vehículo no está disponible inicialmente en el depósito. Se ha tomado esta decisión para simplificar el problema y evitar añadir una casuística más al estudio del efecto de la programación en paralelo sobre el tiempo de ejecución. Otro parámetro que se mantendrá constante será que el conductor deberá estar siempre disponible cuando un UAV vaya a realizar una operación de despegue o aterrizaje, impidiendo realizar al conductor una entrega de un paquete mientras un UAV realiza una tarea de despegue/aterrizaje. Por tanto, la disponibilidad del conductor quedará fuera del alcance de este trabajo. Esta casuística es más realista en la actualidad, ya que en caso contrario sería necesario que los UAVs fuesen capaces de aterrizar/despegar autónomamente desde el camión, al igual que ser capaces de cambiar sus baterías cada vez que aterrizan de forma autónoma.

El contenido del documento es el siguiente:

- *Capítulo 2:* en primer lugar, tras realizar la introducción, se recopilará y se resumirá toda la información existente acerca del problema que se quiere optimizar, pasando por las distintas fases y mejoras que han ido teniendo. Además, se comentarán algunas aplicaciones reales de la planificación de rutas mediante drones.
- *Capítulo 3:* se realizará un análisis profundo del problema a considerar, su formulación matemática y las distintas características que provocan que el problema tenga una complejidad computacional tan elevada, siendo necesario la implementación de un algoritmo heurístico en vez de un algoritmo exacto.
- *Capítulo 4:* Implementación en Python del algoritmo heurístico mediante multihilos que permitan ejecutar en paralelo. Esto permitirá una ganancia en cuanto al tiempo de ejecución del programa respecto al algoritmo inicial sin la posibilidad de ejecución en paralelo. Descripción detallada de las estrategias implementadas para llevar a cabo la ejecución en paralelo.
- *Capítulo 5:* en este capítulo se recogerán los datos obtenidos de las distintas simulaciones realizadas para los distintos escenarios planteados: uso de distinto número de drones para un mismo conjunto clientes-almacén, comparativa entre la implementación exacta (casos pequeños), implementación heurística inicial y las heurísticas con ejecución en paralelo para un mismo conjunto clientes-almacén; y por último, replicar estas simulaciones para distintos conjuntos.

- *Capítulo 6:* finalmente se obtendrán unas conclusiones sobre los efectos que tiene dicha implementación sobre el algoritmo heurístico y sobre los resultados (tiempo de ejecución y solución óptima). Además, se propondrán posibles trabajos futuros que permitan ampliar el uso de la ejecución en paralelo a problemas similares, así como posibles mejoras del propio algoritmo implementado.

2 ESTADO DEL ARTE

En este capítulo se va a realizar una revisión de los distintos tipos de problemas que se encuentran relacionados con el objeto de este trabajo; es decir, problemas de transporte de mercancías mediante el uso combinado de vehículos terrestres y UAVs. También se realizará una breve descripción sobre las distintas aproximaciones que se han realizado para resolver estos problemas, y cómo estas han servido de soporte de conocimiento para conseguir implementar un algoritmo que permita cubrir las restricciones de estos problemas.

2.1 Aplicaciones

El uso de vehículos aéreos autónomos en tareas de distribución de productos tiene una gran aplicabilidad y un futuro muy prometedor en distintos ámbitos, tanto civiles como militares. A continuación, se muestran algunos de los sectores en los que la aplicación de los drones o UAVs se encuentra en fase de pruebas e incluso sectores en los que dicha aplicación es una realidad.

- *Sector sanitario:* los UAVs permiten transportar objetos poco pesados a gran velocidad y económicamente. De esto se aprovechan múltiples estudios y experimentos en el sector médico, ya que permite transportar muestras de sangre entre hospitales y laboratorios que se encuentran en regiones urbanas y que por lo tanto el tráfico es un gran inconveniente. Para ello, se están llevando a cabo experimentos como el descrito por Roca-Riu et al. [1] en el que hospitales suizos intercambian muestras de sangre y otros materiales sanitarios mediante el uso de un camión y un dron. Otra aplicación en el ámbito sanitario y de actualidad es el trabajo realizado por Adwibowo, A. [2] donde se expone la posibilidad de usar drones para transportar vacunas contra el virus COVID-19 en el Sudeste Asiático. Con esto se conseguiría llegar a zonas densamente pobladas pero cuyo acceso sea imposible por medios terrestres que sean capaces de mantener las vacunas en un intervalo de temperaturas muy bajo.

Figura 1. Experimento en el sector sanitario [1]

- *Ayuda humanitaria en caso de desastre natural:* el uso de drones no tripulados en áreas que han sufrido un desastre natural es una solución que conseguiría evitar que determinados grupos de población vulnerable no puedan recibir suministros básicos. En este ámbito se han desarrollado trabajos como el de Rabta et al. [3] en los que se emplean ejemplos como el terremoto de Haití de 2011 o inundaciones en países subdesarrollados. Para solventar los problemas de distribución de alimentos, medicinas u otros suministros de especial necesidad, este trabajo describe el uso de una flota de drones que repartan a aquellas zonas que han quedado incomunicadas debido a la imposibilidad de acceder mediante medios terrestres cuya accesibilidad tenga un alto factor de riesgo para el personal encargado de proveer la ayuda.
- *Logística para empresas de distribución:* empresas como Amazon o FedEx ya están buscando formas de distribuir las compras que sus clientes realizan a través de drones, dando una mayor flexibilidad y rapidez a las tareas de reparto de paquetes, redundando en un mejor servicio a sus clientes. Es en este ámbito donde se engloba este trabajo. Existen múltiples trabajos que buscan dar una solución al problema de repartir desde un único almacén a múltiples clientes que se encuentran en una parcela cercana al propio almacén, y es por ello que en los siguientes apartados de este capítulo se van a describir distintas aproximaciones para dar una respuesta a este problema. Este tipo de aplicaciones tienen por contrapartida la legislación actual que restringe el vuelo de UAVs sobre zonas pobladas.
- *Usos militares:* por último, uno de los usos que más atención está creando sobre este tipo de problemas son las posibles aplicaciones militares que tengan estos algoritmos. Entre estas aplicaciones se encuentra el uso en inteligencia, reconocimiento y vigilancia por parte de las fuerzas aéreas de los Estados Unidos recogidos en el trabajo de Weinstein et al. [4]

2.2 Un vehículo terrestre y un dron

Tras realizar una breve descripción sobre la aplicabilidad del presente trabajo, así como el de otros trabajos relacionados con el mismo, se continuará exponiendo el estado del arte en cuanto a la aproximación que se realiza para alcanzar una solución al problema de la distribución mediante el uso de drones.

Para ello, se comenzará con el análisis de la combinación de un vehículo terrestre con un único dron. Y es en este campo de estudio donde destaca el trabajo de Murray et al. [5], el cual es el trabajo que ha servido de base para futuros trabajos en el campo de estudio de los problemas relacionados con la logística entre un almacén y clientes. Además, también ha permitido que trabajos como el presente documento implementen una formulación similar a la que inicialmente se empleó en este trabajo a la que se le han añadiendo nuevas restricciones y características que han permitido dar cobertura a un mayor número de posibilidades y exigencias por parte de las empresas de la logística.

En [5] se emplea la variante del problema del viajero conocida como Flying Sidekick Traveling Salesman Problem (FSTSP) donde un dron asistido por un vehículo terrestre (en este caso un camión) trabajan cooperativamente para realizar tareas de entrega de paquetes. En esta variante se plantea el uso de una formulación MILP (Mixed Integer Linear Programming) y una metodología heurística para resolver casos en los que existen una gran cantidad de clientes que han de ser servidos por los vehículos de transporte. Además del trabajo cooperativo entre un dron y un vehículo convencional, también se incluye en este trabajo la posibilidad de emplear varios drones que sirvan a clientes que se encuentren cerca del almacén sin necesidad de intervención de un medio convencional.

A continuación se muestra una breve descripción de la formulación MILP y su importancia desde el punto de vista de problemas como el estudiado en este trabajo.

2.2.1 Formulación MILP

La formulación MILP recoge aquellos problemas cuya formulación matemática se puede realizar mediante una combinación entre variables de decisión continuas y variables discretas (variables binarias o enteros). Esta formulación es un caso concreto de la programación lineal, la cual a su vez consiste en modelar problemas de optimización mediante el uso de inecuaciones lineales.

La programación lineal entera ILP es un subconjunto de la programación lineal en la que se fuerza a que las

variables empleadas sean enteras. Esto provoca que el problema sea NP-completo y por lo tanto se puedan emplear algoritmos que están diseñados para resolver de forma eficiente (bajo uso de memoria y tiempo) este tipo de problemas.

Para el caso del problema TSP, puesto que se trata de un problema NP-Hard como se expondrá en futuros capítulos, la formulación MILP permite emplear algoritmos que se basan en esta formulación matemática para obtener resultados cercanos al óptimo en tiempos de ejecución bajos. Por eso surge la necesidad de emplear esta formulación para el caso de este trabajo, permitiendo simplificar la implementación de un algoritmo heurístico que busque en tiempos de ejecución bajos una solución cuando el número de clientes que deben ser suministrados toman valores elevados y para los que otros algoritmos como el Branch & Bound no tiene la capacidad de resolver en tiempos bajos.

2.3 Múltiples drones sin vehículo terrestre

Una vez revisado algunos de los problemas de un vehículo terrestre cooperando con un dron, el siguiente paso sería realizar un repaso sobre los trabajos que extienden este problema al uso de múltiples drones junta a un único vehículo terrestre, pero para ello es necesario realizar una etapa intermedia en la que se describa la formulación llevada a cabo para casos en los que se emplean múltiples drones pero sin la colaboración con un vehículo terrestre convencional.

Este grupo de trabajos quedan más alejados del objetivo del presente trabajo, que como ya se ha definido en el capítulo primero, es realizar la implementación de un algoritmo en el que se vean involucrados distintos medios de transporte y que estos trabajen de forma conjunta, permitiendo explotar las ventajas de cada uno de estos medios, y mitigar sus inconvenientes. A pesar de ello, sirven para dar un contexto sobre la formulación empleada en futuros capítulos de este trabajo.

El trabajo de Thibbotuwawa et al. [6] establece restricciones para el peso de la carga y para la autonomía del dron mediante un modelado del consumo de energía de la batería y su estado de carga. Además, se tiene en cuenta la posibilidad de que un mismo dron puede realizar múltiples tareas mediante el cambio de baterías cuando vuelve al almacén de salida. Esto permite sentar las bases de futuros trabajos, como se verá más adelante, en los que se incluyen estas restricciones en cuanto al uso de los drones y las necesidades de proveer de tiempos de espera cuando estos finalizan una tarea para que pueda ser cambiada su batería por una que esté en plena carga.

2.4 Múltiples drones con un vehículo terrestre

El uso de múltiples drones cooperando con un vehículo terrestre o convencional añade un grado de dificultad, ya que combina la necesidad de tener en cuenta el uso de múltiples drones donde cada uno de ellos puede tener características diferentes (autonomía, capacidad máxima de peso, volumen de la carga, velocidad máxima de funcionamiento, etc.) y la cooperación con un agente de naturaleza completamente distinta a un dron como puede ser por ejemplo un camión. Esto produce que surjan nuevos problemas de cara a su implantación en el mundo real como puede ser la necesidad de aterrizar de múltiples drones al mismo tiempo sobre el camión.

Para estos problemas que surgen del uso combinado de estos dos sistemas de distribución de paquetes aparecen trabajos que intentan dar solución a los mismos como el trabajo de Murray y Raj [7] donde se plantea el uso de una formulación similar a la empleada en su trabajo anterior [5], pero aplicada a múltiples drones. A esta variación se la conoce como mFSTSP (multiple Flying Sidekick Traveling Salesman Problem). Este trabajo se analizará más profundamente en futuros capítulos del trabajo, ya que el problema identificado en su paper, así como su algoritmo heurístico serán la base del diseño del algoritmo empleado en este trabajo, y por ende este paper llevado a cabo por Murray servirá de lugar de partida a la hora de realizar el diseño de la programación en paralelo.

En este trabajo se tienen en cuenta múltiples restricciones que afectan tanto a la disponibilidad de drones para servir a un determinado cliente, como sus capacidades de transporte o los tiempos que tardan en realizarlas. Además, para solventar el problema de varios drones que tratan de despegar y aterrizar desde un mismo vehículo convencional, se plantea el uso de una cola en la que se van almacenando las peticiones de cada uno de los drones, de forma que se pueda realizar de forma ordenada y secuencial todas las operaciones necesarias para

una correcta colaboración dron-vehículo.

2.5 Múltiples drones y múltiples vehículos terrestres

En los últimos años, fruto del creciente desarrollo e interés por parte de grandes empresas del sector de la logística, se han comenzado a realizar estudios sobre el uso de múltiples drones trabajando de forma cooperativa con múltiples vehículos convencionales. Estos trabajos tienen una complejidad mayor que todos los recogidos a lo largo de este capítulo debido al aumento en el número de combinaciones que surgen fruto de introducir un mayor número de posibilidades a la hora de realizar las tareas de distribución. Es por ello que muchos de los artículos que están surgiendo son elaborados para grupos reducidos de clientes o relajando restricciones que se habían implantado en iteraciones anteriores como el uso de drones con distintas capacidades-características.

En este sentido, se puede destacar el artículo de Ham [8] donde se lleva a cabo una formulación que integra múltiples camiones, múltiples drones y múltiples almacenes restringidos por ventanas de tiempo, clientes de entrega-recogida y por drones que pueden realizar múltiples visitas en un solo trayecto.

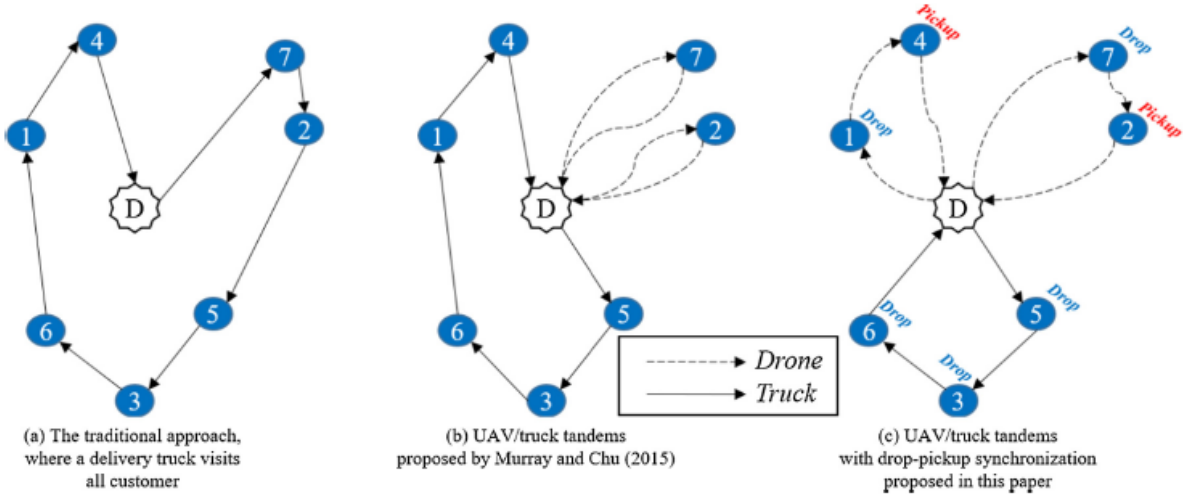

Figura 2. Comparativa entre TSP básico, 1 camión + 1 dron, y el trabajo de Ham [8]

En la Figura 2 se muestra una comparativa entre la ruta planificada por un algoritmo que resuelva el problema TSP básico; es decir, un único vehículo se encarga de visitar a todos los clientes con dos aproximaciones distintas en el uso de drones junto con vehículos terrestres. En primer lugar (imagen central) tenemos el resultado para esa misma red de clientes y almacén para el estudio realizado por Murray y Chu en 2015 [5] donde existe un único vehículo (camión) y por otra parte, un dron que puede visitar a clientes que se encuentren cerca del almacén. Por último, la tercera imagen hace referencia al artículo de Ham donde los clientes pueden requerir tanto una operación de recogida como de envío de un paquete. Además, cada dron puede realizar múltiples visitas (envío + recogida).

A pesar de que estos problemas son más completos desde el punto de vista del uso de drones y vehículos terrestres, carecen de otras ventajas que tienen los trabajos de los apartados anteriores, ya que las restricciones en cuanto a autonomía, velocidad de los drones o tiempos de despegue/aterrizaje no son tenidas en cuenta debido a que estos trabajos se encuentran en fases tempranas de desarrollo. Es por ello que se ha seleccionado un problema que involucre un único vehículo terrestre con múltiples drones, ya que permitirá obtener resultados más realistas en cuanto a la disponibilidad de drones que pueden existir en la actualidad en una empresa (alta variabilidad entre las características de drones) y además, la heurística está en fases más desarrolladas, lo que a su vez permite que se implementen mejoras en tiempo de ejecución mediante la ejecución en paralelo (mediante multihilos), que es el objetivo a alcanzar en este trabajo.

2.6 Clientes servidos únicamente por drones

Para finalizar con este capítulo, también cabe destacar un caso concreto de este tipo de problemas, y es el uso del conjunto vehículo terrestre-dron para realizar las tareas de distribución de paquetes pero con la peculiaridad de que los clientes son únicamente servidos por los drones, dejando a los vehículos terrestres como vehículos auxiliares que dan soporte a los drones.

Este problema está también relacionado con un grupo de artículos vistos anteriormente, uso únicamente de drones, sin vehículos terrestres, para las tareas de transporte y entrega de paquetes. A este tipo de problemas se les conoce con las siglas MVDRP (Multi Visit Drone Routing Problem). Entre ellos destaca el artículo de Poikonen et al. [9] donde se emplea un conjunto de un vehículo terrestre junto con múltiples drones, pero como se ha mencionado anteriormente, el vehículo terrestre sirve para proveer de los paquetes a los drones y de batería en caso de ser necesaria. Por otra parte, este paper también tiene en cuenta la posibilidad de realizar múltiples visitas por un mismo dron, combinando algunas de las funcionalidades descritas en los apartados anteriores por artículos de otros investigadores.

3 PROBLEMA

En el presente capítulo se expondrá el problema seleccionado y su formulación matemática que permitirá elaborar un algoritmo heurístico. Además, se describirá la base teórica en la que se sustenta la heurística y las distintas fases de las que se compone la misma.

Antes de comenzar con ello, es necesario introducir brevemente la diferencia que existe entre un problema TSP y un problema VRP, ya que la formulación del problema del presente trabajo se basa en un problema TSP.

3.1 TSP vs VRP

El problema del viajante (TSP) fue formulado en 1930 y su objetivo trata de buscar la ruta óptima que debe ser trazada por un viajero a través de distintos nodos. Esta ruta óptima puede ser seleccionada mediante el uso de distintos objetivos, como el coste total de realizar cada uno de los arcos que conecta los nodos o minimizar el tiempo necesario para recorrer todos los clientes (nodos) a los que es necesario visitar.

El TSP es uno de los problemas de optimización combinatoria más conocidos debido a su amplia aplicabilidad y su dificultad computacional. Se trata de un problema NP-Hard, lo que quiere decir que la búsqueda del óptimo no puede ser realizada en tiempo polinomial con los algoritmos que existen en la actualidad. Esto provoca que conforme aumenta la complejidad del problema (en nuestro caso podría ser un aumento del número de clientes que hay que atender), aumenta también el tiempo necesario para encontrar el óptimo, pero este aumento no se corresponde con una curva polinómica, si no que se puede asimilar más bien por una curva exponencial. Por ello, en la actualidad se utilizan algoritmos heurísticos que permitan acercarnos a la solución óptima sin necesidad de tener que calcular todas las posibles combinaciones del problema, reduciendo con ello el coste computacional asociado.

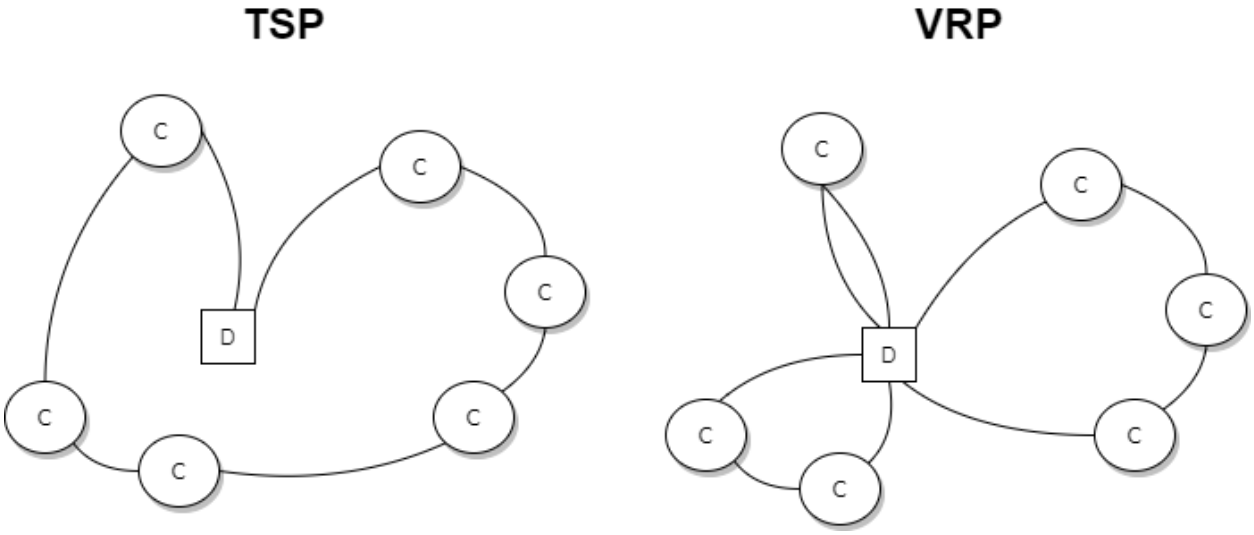

Figura 3. Representación gráfica de los problemas TSP y VRP

Sin embargo, en 1959 surgió por parte de George Dantzig y John Ramser la primera definición del problema de enrutamiento de vehículos [1] o VRP de sus siglas en inglés (Vehicle Routing Problem) el cual busca minimizar el coste de visitar una serie de clientes o nodos, pero a diferencia del TSP, el problema de enrutamiento de vehículos contempla el uso de múltiples vehículos realizando las tareas de reparto a los distintos clientes de la ruta. Esta diferencia en la formulación del problema ha permitido que en la actualidad se diferencien los algoritmos diseñados para la resolución de problemas de selección de una ruta óptima para repartir a una serie de clientes dados desde un local/almacén en estos dos problemas.

En el caso de este proyecto, el problema seleccionado es el TSP, y más concretamente una formulación mFTSP (multiple Flying Sidekick Traveling Salesman Problem). Aunque esta formulación incluye el uso de varios vehículos (UAVs + camión) para realizar las operaciones de reparto, estas pueden ser simplificadas. Esta simplificación se basa en reducir el problema a un único vehículo principal que es el encargado de realizar las tareas de reparto y transportar a vehículos auxiliares. Mientras que dichos vehículos auxiliares se encargarán de reducir el número de clientes que deben ser visitados por el vehículo principal.

3.2 Modelo matemático

En siguiente lugar, se procederá a definir el modelo matemático del problema mFSTSP. Esta formulación del problema tiene varias características, como que cada UAV únicamente será capaz de servir a un único cliente, y para ello, necesitará de un nodo de despegue (inicio de la subruta), el nodo de servicio, y un nodo de aterrizaje (fin de la subruta del UAV). Es por ello que cada subruta llevada a cabo por un UAV tendrá asociado el número del UAV (identificador) que se encargará de realizar dicha operación, y de los 3 nodos que caracterizan su subruta. Además, cada UAV podrá realizar múltiples subrutas a lo largo de la operación de reparto, constando cada una de ellas de esta tupla de 4 elementos.

Un UAV tampoco puede repetir un mismo nodo como nodo de despegue y de aterrizaje, siendo únicamente posible para el UAV aterrizar en un nodo que sea visitado por el camión, o en su defecto, en el propio almacén donde se inicia la ruta de reparto y donde finaliza. Todo esto queda resumido en las tablas siguientes (Tabla 1 y Tabla 2) donde se muestran los parámetros de la formulación matemática y las variables de decisión del problema.

Tabla 1. Parámetros de la formulación matemática

V	Conjunto de UAVs
C	Conjunto de clientes/nodos
$\widehat{C_v}$	Conjunto de clientes que pueden ser servidos por UAVs
N	Conjunto de todos los nodos incluido el almacén como nodo 0 y nodo c+1
N_0	Conjunto de nodos disponibles para iniciar una ruta (orígen) $N_0 = \{0,1,2,\dots,c\}$
N_+	Conjunto de nodos que serán visitados (destino) de una ruta $N_+ = \{1,2,\dots,c+1\}$
τ_{ij}	Tiempo de viaje del camión del nodo i al nodo j
τ'_{vij}	Tiempo de viaje de un UAV (v) desde un nodo i hasta un nodo j
$s^L_{v,i}$	Tiempo de despegue de un UAV desde un nodo i
$s^R_{v,k}$	Tiempo de aterrizaje + recogida de un UAV en un nodo k
σ_k	Tiempo de servicio empleado por un camión en un nodo k; $k \in N_+$ $donde\ \sigma_{c+1} \equiv 0$

$\boldsymbol{\sigma'_{vk}}$	Tiempo de servicio empleado por un UAV en un nodo k; k $\in N_+$ $donde\ \sigma'_{v,c+1} \equiv 0$
P	Conjunto $\langle v, i, j, k \rangle$ que especifica todas las posibles agrupaciones de 3 nodos que pueden ser visitados por un UAV
$\boldsymbol{e_{vijk}}$	Autonomía de un UAV para viajar de un nodo i a un nodo j y regresar a un nodo k (en unidades de tiempo)

Tabla 2. Variables de decisión

$\boldsymbol{x_{ij} \in \{0,1\}}$	$\boldsymbol{x_{ij} = 1}$ **si el camión viaja inmediatamente desde el nodo i al nodo j**
$\boldsymbol{p_{ij} \in \{0,1\}}$	$p_{ij} = 1$ si el nodo i aparece antes que el nodo j en la ruta del camión
$\boldsymbol{y_{vijk} \in \{0,1\}}$	$y_{vijk} = 1$ si el UAV v viaja desde el nodo i para visitar el nodo j y se junta con el camión en el nodo k
$\boldsymbol{\check{t}_i \geq 0}$	Tiempo de llegada del camión al nodo i
$\boldsymbol{\bar{t}_i \geq 0}$	Tiempo de servicio del camión en nodo i
$\boldsymbol{\hat{t}_i \geq 0}$	Tiempo de finalización de la tarea en el nodo i por parte del camión
$\boldsymbol{\check{t}'_{vi} \geq 0}$	Tiempo de llegada del UAV al nodo i
$\boldsymbol{\hat{t}'_{vi} \geq 0}$	Tiempo de finalización de la tarea en el nodo i por parte del UAV
$\boldsymbol{z^R_{v_1,v_2,k} \in \{0,1\}}$	$z^R_{v_1,v_2,k} = 1$ si el UAV v_1 es recogido en el nodo k antes que el UAV v_2
$\boldsymbol{z^R_{0,v,k} \in \{0,1\}}$	$z^R_{0,v,k} = 1$ si el camión finaliza su servicio en el nodo k antes que el UAV regrese
$\boldsymbol{z^R_{v,0,k} \in \{0,1\}}$	$z^R_{v,0,k} = 1$ si el UAV regresa antes que el camión finalice su servicio en el nodo k
$\boldsymbol{z^L_{v_1,v_2,i} \in \{0,1\}}$	$z^L_{v_1,v_2,i} = 1$ si el UAV v_1 despega en el nodo i antes que el UAV v_2
$\boldsymbol{z^L_{0,v,i} \in \{0,1\}}$	$z^L_{0,v,i} = 1$ si el camión finaliza su servicio en el nodo i antes que el UAV despegue
$\boldsymbol{z^L_{v,0,i} \in \{0,1\}}$	$z^L_{v,0,i} = 1$ si el UAV despega antes que el camión finalice su servicio en el nodo i
$\boldsymbol{z'_{v_1,v_2,k} \in \{0,1\}}$	$z'_{v_1,v_2,k} = 1$ si el UAV v_1 despega desde el nodo i antes que v_2 aterrice en i
$\boldsymbol{z''_{v_1,v_2,k} \in \{0,1\}}$	$z'_{v_1,v_2,k} = 1$ si el UAV v_1 aterriza en el nodo i antes que v_2 despegue desde i
$\boldsymbol{1 \leq u_i \leq c+2}$	Indica el orden de visitas relativo al nodo i

El objetivo del mFSTSP es minimizar el tiempo necesario para servir a todos los nodos/clientes, partiendo desde el almacén y volviendo al mismo. Para que la ruta finalice, deben volver al almacén tanto el camión como todos los UAVs que participen en las tareas de reparto.

A continuación se muestran las ecuaciones que rigen el modelo matemático:

Min $\hat{\boldsymbol{t}}_{\boldsymbol{c+1}}$ (1)

s.a.

$$\sum_{\substack{i\in N_0\\ i\neq j}} x_{ij} + \sum_{v\in V}\sum_{\substack{i\in N_0\\ i\neq j}}\sum_{\langle v,i,j,k\rangle\in P} y_{vijk} = 1 \;\; \forall j \in C \;\; , \tag{2}$$

$$\sum_{j\in N_+} x_{0j} = 1, \tag{3}$$

$$\sum_{i\in N_0} x_{i,c+1} = 1, \tag{4}$$

$$\sum_{\substack{i\in N_0\\ i\neq j}} x_{ij} = \sum_{\substack{k\in N_+\\ k\neq j}} x_{jk} \;\; \forall j \in C\,, \tag{5}$$

$$\sum_{\substack{j\in C\\ i\neq j}}\sum_{\langle v,i,j,k\rangle\in P} y_{vijk} \leq 1 \;\; \forall i \in N_0, \forall v \in V, \tag{6}$$

$$\sum_{\substack{i\in N_0\\ i\neq k}}\sum_{\langle v,i,j,k\rangle\in P} y_{vijk} \leq 1 \;\; \forall k \in N_+, \forall v \in V, \tag{7}$$

$$2y_{vijk} \leq \sum_{\substack{h\in N_0\\ h\neq i}} x_{hi} + \sum_{\substack{l\in C\\ l\in k}} x_{lk} \;\; \forall v \in V, i \in C, j \in \{C: j \neq i\}, k \in \{N_+: \langle v,i,j,k\rangle \in P\}, \tag{8}$$

$$2y_{v0jk} \leq \sum_{\substack{h\in N_0\\ h\neq k}} x_{hk} \;\; \forall v \in V, j \in C, k \in \{N_+: \langle v,0,j,k\rangle \in P\}, \tag{9}$$

$$u_k - u_i \geq 1 - (c+2)\left(1 - \sum_{\substack{j\in C\\ \langle v,i,j,k\rangle\in P}} y_{vijk}\right) \;\; \forall v \in V, i \in C, k \in \{N_+: k \neq i\}, \tag{10}$$

$$u_i - u_j + 1 \leq (c+2)(1 - x_{ij}) \;\; \forall i \in C, j \in \{N_+: j \neq i\}, \tag{11}$$

$$u_i - u_j \geq 1 - (c+2)p_{ij} \;\; \forall i \in C, j \in \{C: j \neq i\}, \tag{12}$$

$$u_i - u_j \leq -1 + (c+2)(1 - p_{ij}) \;\; \forall i \in C, j \in \{C: j \neq i\}, \tag{13}$$

$$p_{ij} + p_{ji} = 1 \;\; \forall i \in C, j \in \{C: j \neq i\}. \tag{14}$$

El objetivo es minimizar el tiempo de finalización de la tarea de reparto del camión en el nodo c+1, el cuál se corresponde con el nodo final del almacén. Este es el tiempo total que conlleva una determinada ruta de reparto, ya que en el tiempo del camión están incluidos los tiempos de los UAVs al tener que esperar el camión a que todos los UAVs vuelvan al camión antes de dar por finalizado su tiempo en el nodo c+1.

La restricción de la ecuación (2) exige que cada cliente sea visitado una única vez, ya sea tanto por un UAV como por el camión. Las restricciones (3) y (4) establecen que los nodos de inicio y final (0 y c+1) son los nodos del almacén.

La restricción (5) determina que el camión tendrá que salir desde cada nodo que ha visitado, mientras que las restricciones (6) y (7) determinan respectivamente que cada UAV solo podrá despegar/aterrizar una vez desde cada nodo, incluido el almacén. La restricción (8) exige que si un UAV despega en un nodo i y aterriza en un nodo k, el camión tiene que tener asignados ambos nodos en su ruta. La restricción (9) se encarga del caso concreto en el que el UAV sea lanzado desde el depósito, en cuyo caso el nodo k de aterrizaje tiene que ser visitado por el camión. De igual forma, la restricción (10) obliga a que el nodo i sea precedente al nodo k en la ruta del camión en caso de que un UAV despegue desde i y aterrice en k.

La restricción (11) elimina subrutas para el camión, mientras que las restricciones (12) – (14) se encargan de determinar un valor apropiado para p_{ij}.

En el trabajo desarrollado por Murray [7] también se plantea la formulación de las restricciones temporales para los UAVs y el camión, pero quedan fuera del ámbito de estudio de este trabajo profundizar más en la formulación matemática del modelo completo. De igual forma, puede ser consultada toda la formulación desarrollada para la secuencia de aterrizajes, despegues y tiempo de servicio del camión a un cliente de un nodo concreto.

3.3 Variantes

Además de la formulación descrita anteriormente, también se contemplan distintas variaciones a la formulación matemática que le dan una mayor versatilidad al problema. Aunque estas variantes pueden tener grandes implicaciones en cuanto al tiempo óptimo de la función objetivo, su efecto sobre los tiempos de ejecución es despreciable, por lo que como se expondrá más adelante en el capítulo de los resultados, estas variantes han quedado fuera del estudio llevado a cabo en este trabajo. A pesar de esto último, es necesario exponer brevemente estas variantes, ya que pueden ser puntos de partida de futuros trabajos que puedan ampliar los resultados obtenidos en este estudio.

3.3.1 Camión no requerido en el depósito

Esta variante relaja las restricciones que obligan a que tanto cuando parte el camión del almacén (nodo 0), como cuando este vuelve al almacén después de recorrer la ruta (nodo c+1), los drones no tienen que realizarlo en el mismo instante. Esto permite que tanto los UAVs puedan realizar operaciones desde el nodo 0 como origen, como rutas que tengan como nodo de destino el nodo c+1.

Esto provoca también que la función objetivo varíe debido a que el último vehículo en llegar al almacén (c+1) dejará de ser siempre el camión. Puesto que los UAVs pueden volver de forma independiente al almacén, es posible que la ruta del camión sea modificada para que vuelva al almacén antes de lo planificado si no existiese esta variante, siendo el UAV el que determinaría el tiempo total empleado en una ruta concreta.

3.3.2 Sistema de lanzamiento y aterrizaje automático

En segundo lugar, se plantea la posibilidad de modificar la formulación matemática original de tal forma que los UAVs puedan despegar y aterrizar en el camión sin necesidad de que el conductor tenga que manipularlos, Esto permite un ahorro en el tiempo necesario para llevar a cabo una ruta, al permitir que el conductor del camión pueda estar sirviendo a un determinado cliente mientras que los UAVs despegan/aterrizan de forma autónoma en el camión.

Al igual que en la variante anterior, esta variante proporciona una mejora en el valor de la función objetivo, pero no tiene ningún efecto sobre el tiempo de ejecución. Además, esta variante necesita de un sistema incorporado en el camión que permita realizar dichas maniobras, lo que disminuye la aplicabilidad del algoritmo implementado a camiones con dicha tecnología.

3.4 Heurística

Tras mostrar la formulación matemática del problema y algunas variantes del mismo, se procederá a exponer la heurística empleada en este trabajo. Como se ha comentado anteriormente, esta heurística está basada en el trabajo desarrollado por Murray y Raj [7].

En su trabajo, estos autores desarrollan una heurística para solventar el problema conocido como mFSTSP (multiple Flying Sidekick Traveling Salesman Problem). Esta heurística está dividida en 3 fases, las cuales se proceden a describir a continuación.

3.4.1 Fase I

En la primera fase se trata de dividir a los clientes entre aquellos que pueden ser servidos tanto por UAVs como por el camión, y aquellos que solo pueden ser servidos por el camión. En la Figura 4 se muestra un ejemplo donde los nodos en verde son clientes que pueden ser servidos por cualquier vehículo, mientras que los nodos azules solo pueden ser servidos por el camión.

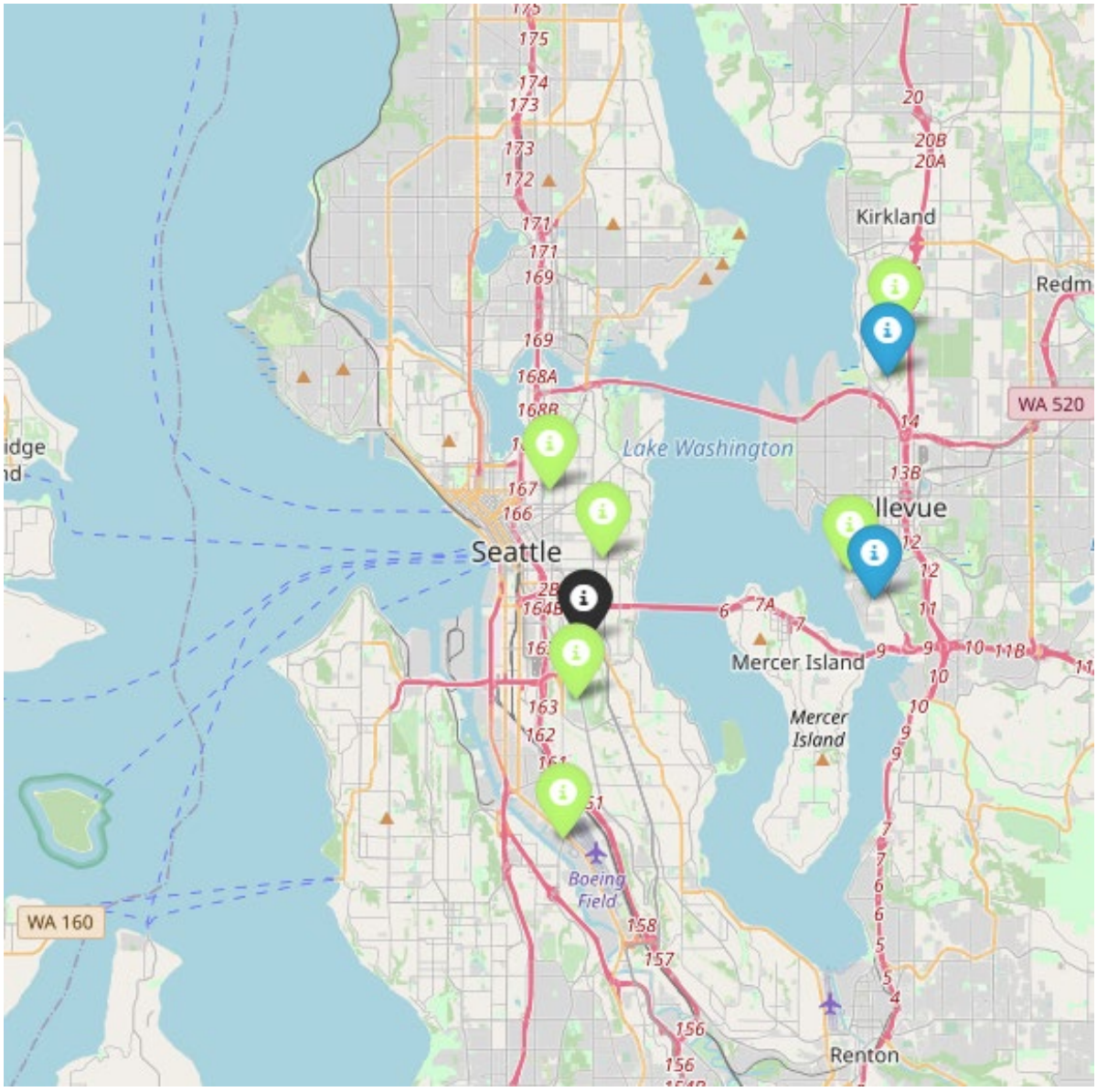

Figura 4. Ejemplificación de clientes servidos por distintos vehículos[1]

Para conocer el número mínimo de clientes que deben ser visitados por el camión, se calcula un parámetro conocido como LTL o *Lower Truck limit,* el cuál se obtiene a partir del número de clientes total y el número de vehículos (UAVs) empleados en la ruta.

$$LTL_0 = \left| \frac{|C| - |V|}{|V| + 1} \right| \tag{15}$$

[1] Elaboración propia

De tal forma que si en una ruta determinada consta de 50 clientes y el número de UAVs empleados es de 4, el número mínimo de clientes que deberán ser visitados por el camión será de 10. Además de esto, será necesario tener en cuenta el número de clientes que pueden ser servidos únicamente por el camión, ya sea por criterios de peso del paquete, como por cualquier otro criterio que se establezca.

A lo largo de la ejecución de la heurística, el parámetro LTL irá incrementando desde su valor mínimo, hasta el máximo, el cuál coincide con el número total de clientes; es decir, que todos los clientes serán servidos por el camión. Con esto se consigue asignar en cada paso del bucle, cuantos nodos son servidos por cada tipo de vehículo, de tal forma que para valores bajos de LTL (primeros pasos del bucle), se empleará en mayor medida los UAVs para servir a los clientes, mientras que en los últimos pasos, será el camión el que realice la mayoría de movimientos, siendo el resultado del bucle la ruta con menor duración de las calculadas.

Con todo esto, lo que realmente se está realizando en esta fase es la resolución de un problema TSP únicamente para aquellos clientes que serán servidos por el camión, calculándose cuáles deben de ser esos nodos para mejorar el tiempo total de la ruta del camión. Al finalizar este cálculo, es necesario realizar también una comprobación de que todos los restantes nodos pueden ser servidos por los UAVs, ya que si por ejemplo, existe un cliente muy alejado de los restantes clientes, pero este nodo ha sido identificado como nodo a ser servido por un UAV, es posible que ningún UAV tenga la autonomía necesaria para llegar hasta ese cliente. Es por ello, que se realiza dicha comprobación, y en caso de que algún nodo no pueda ser alcanzado por los UAVs, se fuerza a que dicho nodo sea visitado por el camión en vez de por los UAVs, calculándose de nuevo una solución para el problema TSP con esa nueva restricción.

3.4.2 Fase II

Una vez calculada la ruta que desempeñará el camión, es el turno de calcular las rutas llevadas a cabo por los UAVs, sus nodos de salida, de servicio y de vuelta al camión. Esto se realiza en la fase 2. En esta fase se busca reducir el tiempo de espera tanto del camión como de los UAVs en los nodos de despegue y aterrizaje de los UAVs.

Por su parte, la fase 2 irá asignando a cada nodo que debe ser servido por un UAV, el UAV concreto que se encargará de realizar dicho servicio, quedando identificados al final de esta fase cuáles serán los nodos visitados por UAVs y cuales por el camión, y en caso de ser visitados por UAV, cuál será el UAV encargado de dicha ruta, y cuál será el nodo desde el que despegará, y cual será el nodo en el que aterrice el UAV para juntarse de nuevo con el camión. Nuevamente se realizará un cálculo para comprobar si existe algún nodo que no tiene asignado un UAV, y en caso afirmativo, será asignado al camión. Puesto que desde la fase 3 se puede volver a la fase 2, en caso de que una ruta no sea realizable, pero dicha ruta sea consecuencia de un cálculo realizado en la fase 3, en vez de realizar la sustitución de dicho nodo para que sea servido por el camión, se saltará directamente a la fase 1 y se incrementará el valor del parámetro LTL, pasando de esta forma al siguiente paso del bucle.

Esta fase se caracteriza por ser la fase con un menor peso en cuanto al coste computacional de la heurística, por lo que como se verá más adelante, la importancia de esta fase desde el punto de vista de este trabajo será muy reducida respecto a las fases 1 y 3, las cuales soportan la mayor parte del coste computacional.

3.4.3 Fase III

Por último, en la fase 3 se llevan a cabo varios cálculos finales para obtener una ruta subóptima para el valor de LTL de ese paso del bucle. Es en esta fase, donde se resuelve un MILP (Mixed Integer Linear Prorgamming) para determinar el momento en el que se realiza el despegue y el aterrizaje de los UAVs en sus respectivos nodos.

En siguiente lugar, se comprueba si la solución obtenida es mejor que la solución subóptima hallada hasta al momento, y en caso afirmativo, se guarda dicha ruta. Tras esto, se intenta cambiar un nodo que sea servido por el camión a un nodo que sea servido por un UAV. En este proceso se evalúan todos los nodos visitados por el camión y se busca el nodo que proporciona un mayor ahorro de tiempo. En caso de existir un nodo con esta posibilidad, se realiza el cambio y se vuelve a la fase 2 para realizar dicha modificación.

Una vez finalizada estas operaciones, se realiza una búsqueda local para intentar mejorar la solución obtenida. Para esta búsqueda local se trata de reducir el tiempo de espera del camión a que los UAVs aterricen en un determinado nodo, tratando de establecer como nodo de recogida el siguiente nodo en la ruta del camión, para

que de esta forma se disminuya el tiempo de espera del camión. Esto tiene por contrapartida que el tiempo que el UAV tendrá que esperar en el nodo siguiente de la ruta aumente, pero puesto que el recurso limitante es el camión, esta acción permitirá reducir el tiempo total de la solución obtenida. Sin embargo, esto propone otros problemas tecnológicos que deberán ser resueltos, como que varios UAVs aterricen de forma prematura en un cliente antes de que el propio camión haya llegado a dicho cliente.

Finalmente, la heurística vuelve a la fase 1 e incrementa el valor de LTL, pasando al siguiente paso del bucle. A continuación se muestra un esquema-resumen sobre la heurística y sus distintas fases y toma de decisiones.

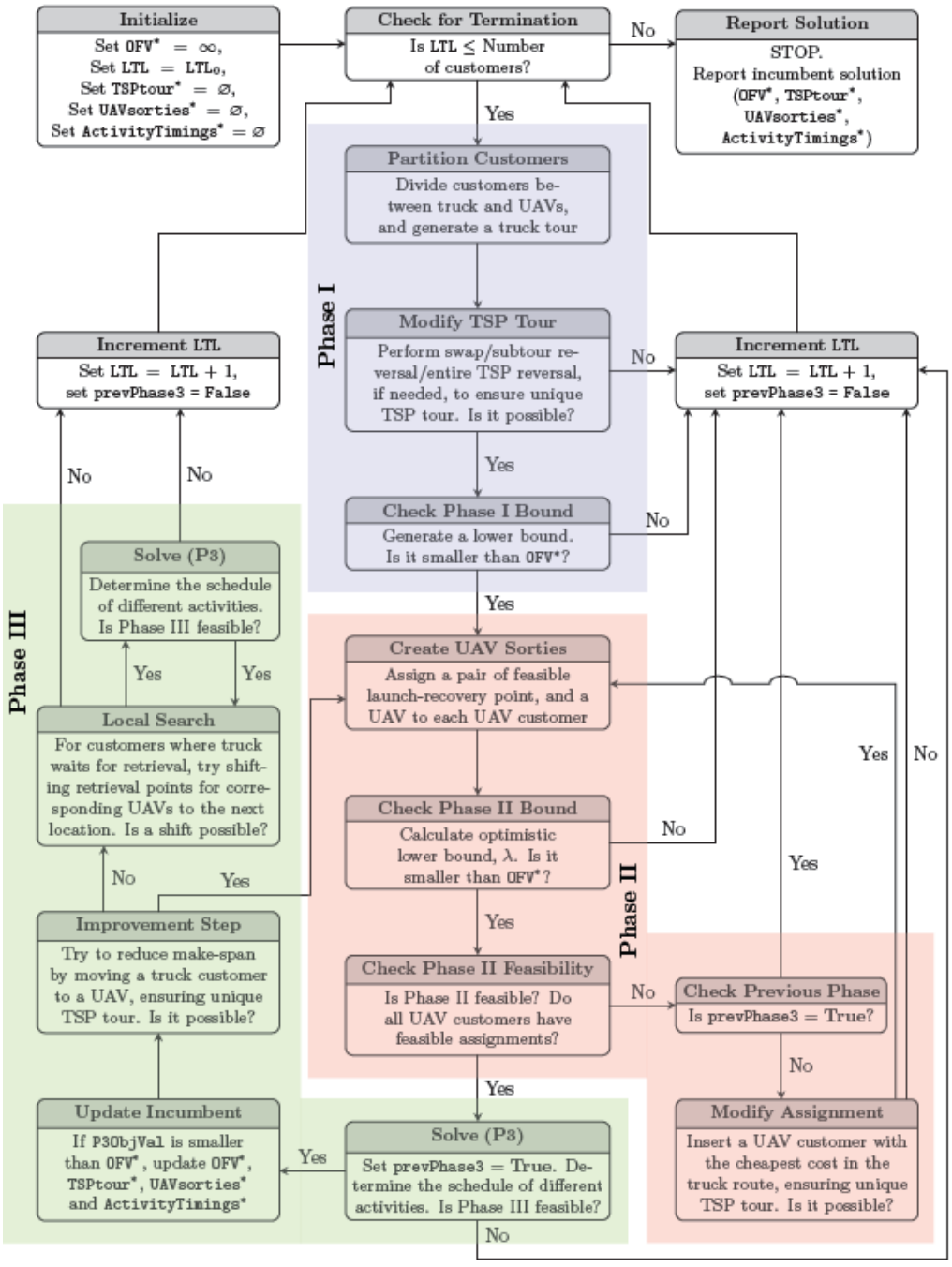

Figura 5. Esquema- resumen de las fases de la heurística [7]

4 ALGORITMO

En este capítulo se expondrá la implementación llevada a cabo para poder solucionar el problema descrito anteriormente. Para ello, la implementación será tanto de un algoritmo exacto como de uno heurístico. Es en este último donde se centra este trabajo. Mejorar el tiempo de ejecución de dicho algoritmo mediante el uso de programación en paralelo.

Inicialmente se realizará una breve descripción sobre el algoritmo exacto implementado y cómo ha sido implementado, para posteriormente realizar lo mismo con el algoritmo heurístico de partida. Es aquí donde, una vez expuesta la base de la heurística, se separará entre el algoritmo inicial del que se parte y el algoritmo que se ejecuta en paralelo.

4.1 Algoritmo exacto

Los algoritmos exactos buscan obtener la mejor ruta posible para satisfacer a todos los clientes y las restricciones que se incluyan en el problema (ventanas de tiempo, consumo de combustible, etc…). Aunque este tipo de algoritmos permiten conocer cuál es la mejor solución posible, conllevan un alto coste computacional debido a la necesidad de comprobar todas las rutas posibles y descartar aquellas que no satisfacen alguna restricción o arrojan valores de tiempo de ruta muy elevados desde un inicio. Además, este coste computacional aumenta rápidamente conforme el número de clientes a satisfacer o nodos de la red aumenta, siendo estos algoritmos una solución poco útil para resolver este tipo de problemas.

A pesar de ello, existen una gran cantidad de algoritmos aplicados a los problemas de enrutamientos, entre los que se puede destacar el algoritmo Branch and Bound (Ramificación y poda). Este algoritmo permite descartar ramas de soluciones o rutas cuyos tiempos son muy elevados respecto a una solución hallada previamente, evitando de esta forma tener que computar todas las posibles rutas involucradas. Otras metodologías como la programación lineal han sido aplicadas a este tipo de problemas con resultados buenos para pequeños grupos de clientes.

En la Figura 6 se muestra un esquema del funcionamiento del algoritmo Branch and Bound, donde los vértices marcados en verde se refieren a una ruta completa cuyo coste final es de 28, mientras que los vértices rojos son otras ramas cuyas soluciones son peores. Como se puede observar, la mejor solución hasta el momento tiene un coste de 28, todas las ramas que se calculen con posterioridad a conocer dicho valor se compararán con este valor. En caso de que sea superior, se descarta esa rama, evitando consumir tiempo de ejecución en una rama que ya se conoce que su valor será peor que el mejor valor hasta ese momento.

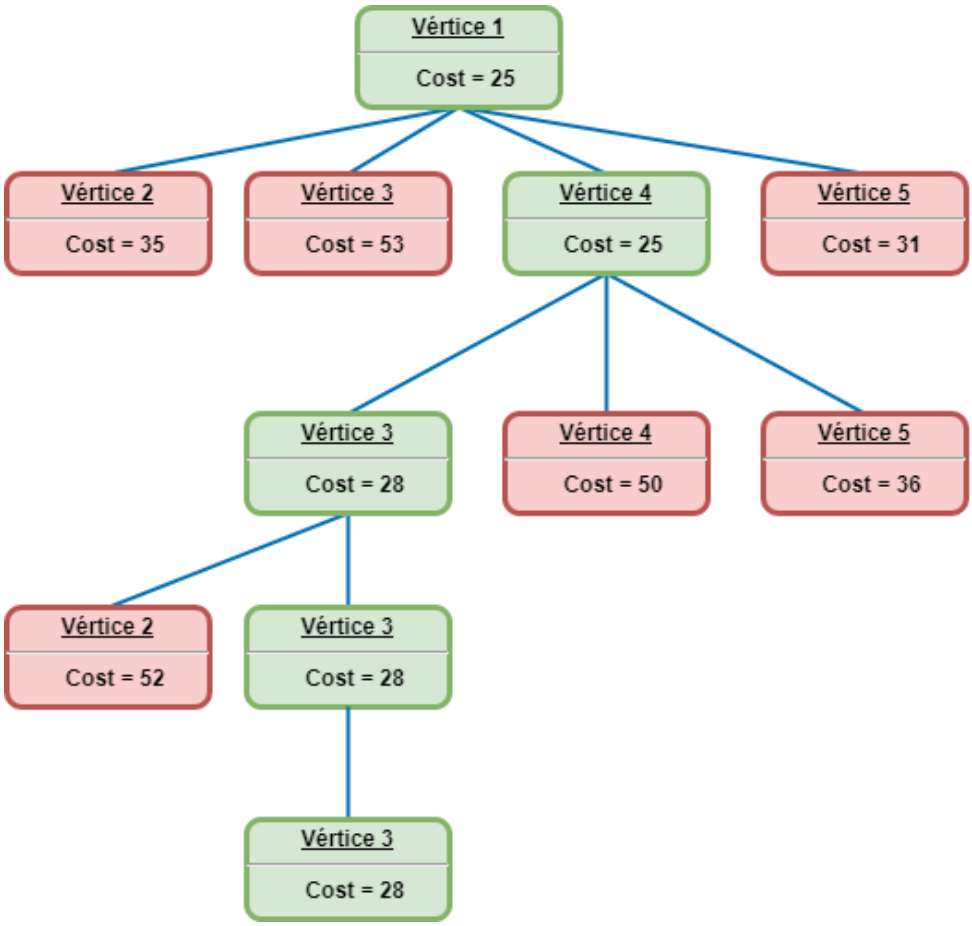

Figura 6. Branch and Bound aplicado a TSP[2]

4.1.1 Gurobi

En este trabajo, la implementación del algoritmo exacto se ha llevado a cabo a través de Python, al igual que el algoritmo heurístico, pero en el caso del algoritmo exacto se ha empleado un software de solución de problemas de optimización conocido como Gurobi. Este software permite implementar algoritmos como el Branch and Bound de una forma más sencilla y con mayor poder de cálculo que si fuese implementado desde cero. Además, esta herramienta informática consta de librerías en Python, siendo únicamente necesario obtener una licencia de su página web para poder emplear esta herramienta de forma gratuita.

La ejecución del algoritmo exacto se basa en un cálculo inicial para todos los vehículos sobre si son capaces o no de realizar una determinada ruta, almacenándose en caso afirmativo como una de las posibles soluciones, y descartándose en caso negativo. En siguiente lugar se asignan los tiempos de despegue y aterrizaje a cada vehículo y los tiempos de servicio para cada cliente. Finalmente se añaden las restricciones al modelo de Gurobi y se resuelve con la función *model.optimize()*.

Esta función devuelve tres posibles soluciones:

- No se ha encontrado ninguna solución o no existe una solución posible para las restricciones introducidas
- Se ha encontrado solución, pero no es la solución óptima debido al tiempo de cutoff.
- Se ha encontrado la solución óptima

El tiempo de cutoff o tiempo de corte es un parámetro que se le pasa al modelo de Gurobi para que en caso de que el tiempo de ejecución en una determinada rama de soluciones supere un tiempo máximo, el algoritmo se detenga y finalice su ejecución. Con esto se evita que el algoritmo exacto pueda excederse más de un valor

[2] Elaboración propia

límite. En caso de que la simulación finalice debido a este parámetro, la solución mínima calculada hasta ese momento será la solución que se muestre en los resultados. Para este trabajo, se ha tomado un tiempo de cutoff de 10s.

4.2 Heurística secuencial

Por otra parte, los algoritmos heurísticos son empleados para obtener una solución cercana al óptimo pero sin recorrer todas las rutas posibles para un determinado número de clientes. Es por ello, que este tipo de algoritmos es empleado para agrupaciones de clientes mayores que los algoritmos exactos o para pequeñas agrupaciones pero que es necesario obtener una solución en un tiempo de ejecución muy bajo, sin importar que la solución obtenida no sea la mejor posible. Una de las metodologías más empleadas es el algoritmo de colonias de hormigas, el cual se basa en simular el proceso de buscar la ruta óptima a través de un gran número de hormigas cuyo objetivo es buscar la ruta óptima (o la más cercana al óptimo) para los clientes dados. Esta metodología fue descrita por Marco Dorigo [11] en 1993, ya que mediante la simulación de las feromonas que dejan las hormigas a su paso, se puede inferir cual es la ruta óptima.

En este trabajo, el algoritmo empleado es el descrito en el paper [7]. Esta heurística, como se ha descrito en el capítulo anterior, consta de 3 fases. Para el caso en el que el algoritmo se ejecute de forma secuencial, el orden de ejecución es el que viene descrito en el capítulo 3, y más concretamente en la Figura 5.

4.3 Heurística en paralelo

Finalmente se llega a la heurística en paralelo. Esta heurística recibe su nombre debido a que se ejecuta de forma paralela, mediante el uso de múltiples hilos que se encargan de realizar operaciones computacionales de cada una de las fases. Es este algoritmo el objeto principal de este trabajo, mientras que los dos algoritmos anteriores servirán como referencia para comparar los resultados obtenidos con esta implementación respecto a las implementaciones existentes en la actualidad.

4.3.1 Estudio preliminar

Para realizar el diseño de este algoritmo ha sido necesario llevar a cabo un estudio preliminar sobre el uso de recursos computacionales del algoritmo heurístico base (secuencial) y observar cuales eran las fases que tenían un mayor consumo de recursos. De esta forma, se centrarían los esfuerzos en reducir los tiempos de dichas fases.

En la Tabla 3 se muestran los resultados obtenidos fruto del estudio preliminar. En este estudio se ha empleado un único UAV con una configuración de alta velocidad y baja autonomía. Se ha realizado una única iteración para cada simulación. En esta tabla se muestran los resultados para 4 problemas distintos, cada uno de ellos con un número de clientes/nodos distintos, variando por tanto su complejidad y su uso de recursos computacionales.

A partir de la tabla se desprende que para valores bajos de clientes, el uso de multihilos tendrá un efecto inapreciable debido al ya de por sí, bajo tiempo de ejecución. Sin embargo, conforme aumenta el número de clientes, y con ello el tiempo de ejecución, se comienza a observar ciertos patrones. El más importante es la importancia que tiene la fase 1 sobre el tiempo total de ejecución, siendo cada vez más importante.

Por su parte la fase 3 es la siguiente fase con mayor efecto sobre el tiempo de ejecución, mientras que la fase 2 tiene un efecto casi nulo sobre el valor final del tiempo de ejecución.

Tabla 3. Resultados del estudio preliminar

ID PROBLEMA	Nº CLIENTES	T1 [SEC]	T2 [SEC]	T3 [SEC]	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	8	0,02	-	0,01	0,03	3425,66
20170606T113038113409	25	0,42	0,01	0,44	0,87	8799,86
20170606T114511221132	50	3,79	0,29	3,29	7,37	12030,11
20170606T115437348436	100	59,76	3	25,99	88,76	19335,41

Con todo esto, se diseña el algoritmo heurístico en paralelo, ejecutándose la fase 1 en paralelo respecto a las otras dos fases, las cuales se ejecutarán en un solo hilo. Esto se debe a varios motivos. En primer lugar, usar un tercer hilo para la fase 2 no conlleva un ahorro elevado en el tiempo de ejecución debido a sus bajos tiempos de ejecución. En segundo lugar, y como se ha expuesto en el capítulo 3, se puede pasar de la fase 3 a la fase 2 para refinar la solución, por lo que sería necesario implementar una comunicación muy compleja y que provocaría una ralentización de la ejecución. Además, puesto que inicialmente se desconoce si es necesario volver a la fase 2 una vez el algoritmo se encuentra en la fase 3, es imposible que el hilo encargado de la fase 2 sepa que cálculos debe hacer en paralelo, con la fase 3, ya que hasta que la fase 3 no finaliza, la fase 2 no conoce cuáles son los valores de entrada. De igual forma, la fase 3 no puede ir en paralelo a la fase 2, ya que hasta que no finaliza esta última, la fase 3 no tiene toda la información necesaria para comenzar a realizar cálculos.

Es por ello, que el algoritmo se separa en dos subconjuntos, la fase 1 que se ejecutará en un hilo por separado, y las fases 2 y 3 que se ejecutarán en otro hilo. El funcionamiento de la heurística en paralelo quedaría de la siguiente forma:

- Paso 1: Se crea un hilo para la fase 1, mientras que las fases 2 y 3 se ejecutan en el hilo principal.
- Paso 2: Se inicia el bucle para el LTL mínimo tanto en el hilo de la fase 1 como en el hilo principal.
- Paso 3: Puesto que al inicio no existe ninguna solución por parte de la fase 1 (hilo auxiliar), las fases 2 y 3 permanecerán en espera hasta que la fase 1 obtenga una primera solución para el TSP. En ese momento, el hilo de la fase 1 comunicará al hilo principal la solución del TSP
- Paso 4: El hilo principal sale de la espera en el que se encontraba y comienza a ejecutar las fases 2 y 3, calculando los tiempos de lanzamiento, servicio y aterrizaje, así como posibles mejoras de la solución calculada por la fase 1. En paralelo, la fase 1 ha pasado al siguiente paso del bucle de LTL y ha incrementado el valor de LTL, comenzando de esta forma a calcular una nueva solución, pero en este caso para un LTL superior.
- Paso 5: Al finalizar el hilo principal de ejecutar la solución, esta se le comunica al hilo auxiliar (fase 1) para que este lo tenga en cuenta. En caso de que el hilo auxiliar haya conseguido calcular la segunda solución del bucle LTL, el hilo principal recibirá dicha solución, incrementando su LTL y por lo tanto, el paso del bucle, y comenzando de nuevo a realizar los cálculos. En caso de que no haya recibido todavía la solución de la fase 1, se quedará en espera hasta que se le comunique la solución de la fase 1.
- Paso 6: Se repiten los pasos 4 y 5 hasta que se recorre el bucle al completo.

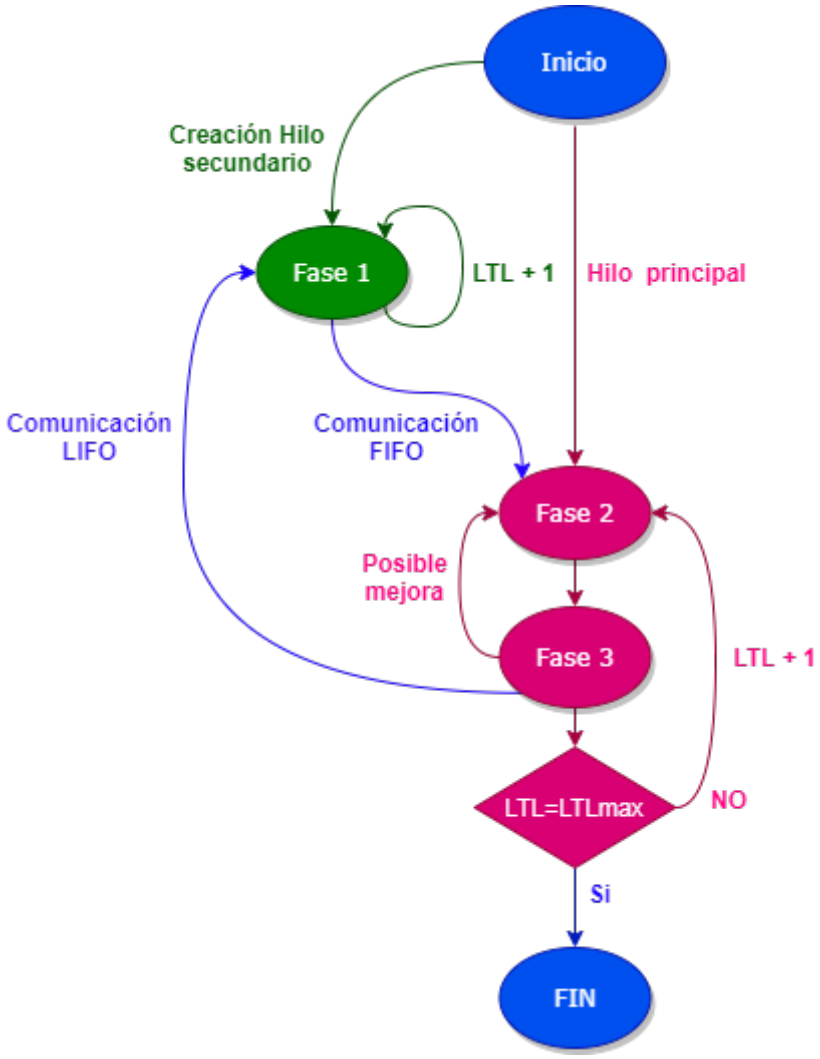

Figura 7. Esquema resumen heurística en paralelo[3]

4.3.2 Comunicación entre hilos

Esto plantea varios retos entre los que destaca la necesidad de realizar una comunicación robusta entre los hilos. Esta debe ser capaz de comunicar los hilos entre sí para que en todo momento tengan la información correcta, sin importar que el hilo de la fase 1 vaya muy adelantado al hilo principal.

Para resolverlo, se emplearán 2 colas que comunicarán a los dos hilos de forma asíncrona. A continuación se describirá más concretamente como se llevará a cabo esta implementación.

4.3.3 Cola FIFO

En primer lugar, la comunicación desde el hilo auxiliar (fase 1) hacia el hilo principal (fases 2 y 3) es la más importante, ya que se encarga de comunicar al hilo principal las soluciones calculadas por la fase 1. Pero esta información debe ser enviada ordenadamente, ya que la fase 1 podría ir varios pasos por delante del hilo principal, por lo que es necesario emplear una cola donde se vaya guardando la información, y de donde el hilo principal pueda ir leyendo la información.

Además, esta cola debe de mostrar la información en orden, de tal forma que la solución de la fase 1 para el paso 1 (LTL inicial) sea la primera solución de la cola, y así con el resto de pasos del bucle. Es por ello que se decide emplear una cola tipo FIFO (First Input First Output). Esta cola se comunicará asíncronamente entre los hilos, de tal forma que cuando el hilo principal lee un valor, este valor es eliminado, y su posición en la memoria es

[3] Elaboración propia

sustituida por el siguiente valor calculado en la fase 1 y que se encontraba en la segunda posición en la cola.

De esta forma se consigue que la fase 1 pueda ejecutarse completamente en paralelo, y cada vez se obtiene una solución, únicamente tiene que añadir la solución a la cola y comenzar a calcular otra solución para un valor mayor de LTL.

4.3.4 Cola LIFO

En segundo lugar, será necesario establecer una comunicación desde el hilo principal hacia el hilo auxiliar. Esta comunicación se empleará para que el hilo auxiliar sepa en todo momento cuál es la solución óptima calculada por el hilo principal, y en caso de que calcule una solución cuyo valor sea superior al óptimo hasta ese momento, la solución sea descartada directamente y el hilo auxiliar pueda pasar al siguiente paso del bucle.

Esta comunicación no será tan importante como la anterior, ya que la heurística puede seguir funcionando, aunque el tiempo de ejecución puede ser peor, además de que la solución calculada por la fase 1 puede ser peor, ya que al no compararla respecto a un valor, puede dar por buena una solución que esté muy alejada del valor óptimo, provocando que las fases 2 y 3 inviertan mucho tiempo en dicha solución tratando de mejorarla.

A diferencia de la cola anterior, en este caso será una cola LIFO (Last Input First Output), ya que lo que se busca con esta cola es obtener la información más actualizada posible, sin importar los valores antiguos.

4.3.5 Otras consideraciones

Aunque desde un primer momento se ha establecido que dividir las fases para realizar la ejecución en paralelo era la mejor estrategia, no se ha justificado mucho el motivo de emplear esa estrategia en vez de otra posible estrategia, como sería ejecutar cada paso del bucle del parámetro LTL en un hilo distinto. Esta estrategia tiene sentido inicialmente, ya que el ahorro de tiempo podría ser mucho mayor, al emplear una gran cantidad de hilos ejecutándose todos ellos en paralelo.

Sin embargo, esta estrategia tiene múltiples problemas cuando se comienza a plantear su implementación. En primer lugar, el uso de hilos conlleva la necesidad de comunicarse entre ellos, y puesto que el objetivo de este trabajo es mejorar los tiempos de ejecución para problemas de una complejidad media-alta (50-100 clientes), si se aumenta el número de pasos del bucle, se aumenta también el número de hilos y el número de colas de comunicación, lo cual llega un punto en el que se hace insostenible.

En siguiente lugar, e incluso más importante, es que los tiempos de ejecución de cada paso del bucle no están equitativamente distribuidos. Los primeros pasos (cercanos a LTL inicial) son los que tienen un mayor número de clientes servidos por UAVs, por lo que la complejidad aumenta, siendo estos pasos muy costosos computacionalmente hablando, pero en cambio, también son estos pasos los que obtienen valores más cercanos al óptimo absoluto, ya que permiten que más vehículos (UAVs) ayuden al camión a realizar tareas de reparto, y por lo tanto, más rápido se realicen las tareas de reparto. Esto permite que una vez se haya calculado un valor muy cercano al óptimo, los siguientes pasos del bucle se descarten rápidamente al superar el valor subóptimo calculado hasta ese momento. Si en cambio, esto se realizase en paralelo, los pasos del bucle para LTL altos no serían descartados rápidamente, ya que los pasos bajos son más costosos computacionalmente, y por lo tanto serían los últimos en acabar, provocando que los pasos altos del bucle estén "quitando" recursos computacionales a los pasos del bucle que son mejores (desde un punto de vista de la solución obtenida), provocando que el tiempo de ejecución final sea muy superior con este tipo de ejecuciones en paralelo.

4.4 Python

Por último, se va a describir brevemente el lenguaje de programación en el que se ha implementado el algoritmo, así como algunas de las librerías empleadas. El lenguaje de programación empleado ha sido Python como se ha mencionado anteriormente. Se ha escogido este lenguaje por su gran accesibilidad a documentación y librerías desarrolladas por otros programadores y que permiten ahorrar una gran cantidad de tiempo durante la programación. Tiene por contrapartida que es un lenguaje lento, pero puesto que todas las simulaciones realizadas en este trabajo están realizadas en el mismo ordenador, bajo unas mismas condiciones y con el mismo lenguaje de programación, al comparar los valores obtenidos entre los distintos algoritmos se podrá observar la diferencia entre ellos sin que el efecto de la lentitud de Python distorsione los resultados.

Otro aspecto a mencionar es el uso de Spyder como entorno de desarrollo integrado para llevar a cabo tanto la programación como las posteriores simulaciones.

4.4.1 Librerías empleadas

A continuación se muestran algunas de las librerías empleadas para implementar la algoritmia descrita anteriormente:

- Pandas: permite analizar grandes cantidades de información, así como almacenarla en archivos. Ha sido empleada tanto en la ejecución de los distintos algoritmos como en la posterior obtención de resultados.
- Gurobipy: esta librería ha sido desarrollada a partir del software Gurobi y ha sido empleada en la resolución del algoritmo exacto
- Queue: librería encargada de la comunicación entre hilos mediante el uso de colas asíncronas (algoritmo heurístico en paralelo).
- Threading: permite crear múltiples hilos que se ejecutan en paralelo (algoritmo heurístico en paralelo).
- Math: permite emplear funciones matemáticas que Python no tiene por defecto.
- Time: se ha empleado para medir el tiempo de ejecución de cada fase del algoritmo heurístico, así como los tiempos de ejecución del algoritmo exacto.
- Matplotlib y numpy: se han empleado para obtener los resultados que se muestran en el capítulo 5 (gráficas).

5 RESULTADOS

Una vez visto los distintos algoritmos que se han implementado en Python, se procederá a mostrar los resultados que se obtienen de las simulaciones. Además, se mostrarán los resultados para cada uno de los algoritmos, así como una comparativa entre los mismos, de tal forma que se pueda conocer con mayor profundidad el funcionamiento de cada algoritmo y las diferencias que existen entre ellos.

Puesto que las simulaciones se realizan en un ordenador con escasos recursos computacionales, para evitar que factores externos a la simulación afecten a los resultados, se han realizado 5 repeticiones para cada simulación. A estas repeticiones se le aplicará una media para obtener el valor final, el cuál será el que aparezca tanto en las tablas como en las gráficas de este capítulo.

Como se vio en el capítulo 3, el problema que se trata de resolver en este trabajo depende de multitud de variables que pueden ser parametrizadas para dar mayor versatilidad al algoritmo, y por lo tanto, dar una mayor cantidad de casuísticas que puedan ser evaluadas en las simulaciones. A pesar de ello, en este trabajo se ha seleccionado un parámetro del modelo heurístico y varios dependientes de los problemas planteados. Estos parámetros son los que provocan un mayor efecto a nivel computacional (tiempo de ejecución), por lo que el objetivo de este documento quedará satisfecho. Estos parámetros serán:

- Número de clientes (nodos) según distintas agrupaciones:
  - 8
  - 25
  - 50
  - 100
- Tipología de los UAVs:
  - Alta velocidad y baja autonomía (101)
  - Alta velocidad y autonomía (102)
  - Baja velocidad y autonomía (103)
  - Baja velocidad y alta autonomía (104)
- Número de UAVs:
  - 1
  - 4
- Número de Iteraciones de la heurística:
  - 1
  - 10

Esta categorización permitirá tratar por separado cómo afecta cada una de estas variables tanto a los valores absolutos de tiempo de ejecución y valor objetivo de la función, como la diferencia que existe entre los distintos algoritmos. Los valores que irán tomando estos parámetros han sido tomados del trabajo de Murray y Raj [7]. A continuación se muestra una tabla resumen sobre la tipología empleada en los UAVs.

Tabla 4. Parámetros de los UAVs según su tipología

VEHICLEID	**1**	**101**	**102**	**103**	**104**
VEHICLETYPE	1	2	2	2	2
TAKEOFFSPEED [M/S]	-1	156.464	156.464	78.232	78.232
CRUISESPEED [M/S]	-1	312.928	312.928	156.464	156.464
LANDINGSPEED [M/S]	-1	78.232	78.232	39.116	39.116
YAWRATEDEG [DEG/SEC]	-1	360	360	360	360
CRUISEALT [M]	-1	50	50	50	50
CAPACITY [LBS]	-1	5	5	5	5
LAUNCHTIME [SEC]	-1	60	60	60	60
RECOVERYTIME [SEC]	-1	30	30	30	30
SERVICETIME [SEC]	30	60	60	60	60
BATTERYPOWER [JOULE]	-1	457503	904033	291094	562990
RANGE	NA	low	high	low	high

Como se puede observar en la tabla anterior, los UAVs que pertenecen al grupo '101' son UAVs con bajo rango pero con una elevada velocidad de crucero, a diferencia de los UAVs pertenecientes al grupo '102', los cuales se caracterizan por una elevada autonomía y velocidad de crucero. Por otra parte, la columna de los vehículos tipo '1' se refiere al camión encargado de tanto transportar los UAVs como de ayudar en las tareas de reparto a los clientes. Esta columna tiene como valor -1 para casi todas sus filas, ya que se establece por defecto que el camión podrá cargar toda la carga necesaria sin atender a su peso, o que su autonomía es superior a la distancia recorrida para realizar la ruta de reparto.

En siguiente lugar se muestran en dos imágenes la distribución espacial (mapa) de los clientes de un determinado problema, así como las restricciones impuestas a los vehículos que pueden visitar cada nodo.

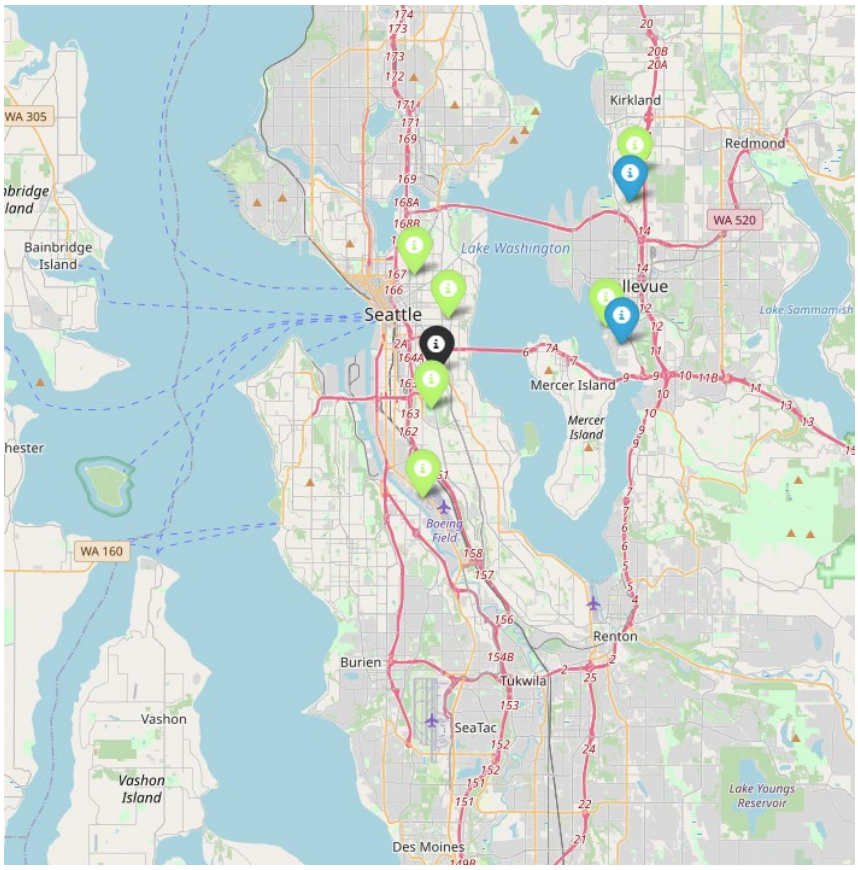

Figura 8. Ejemplo de un mapa de 8 clientes[4]

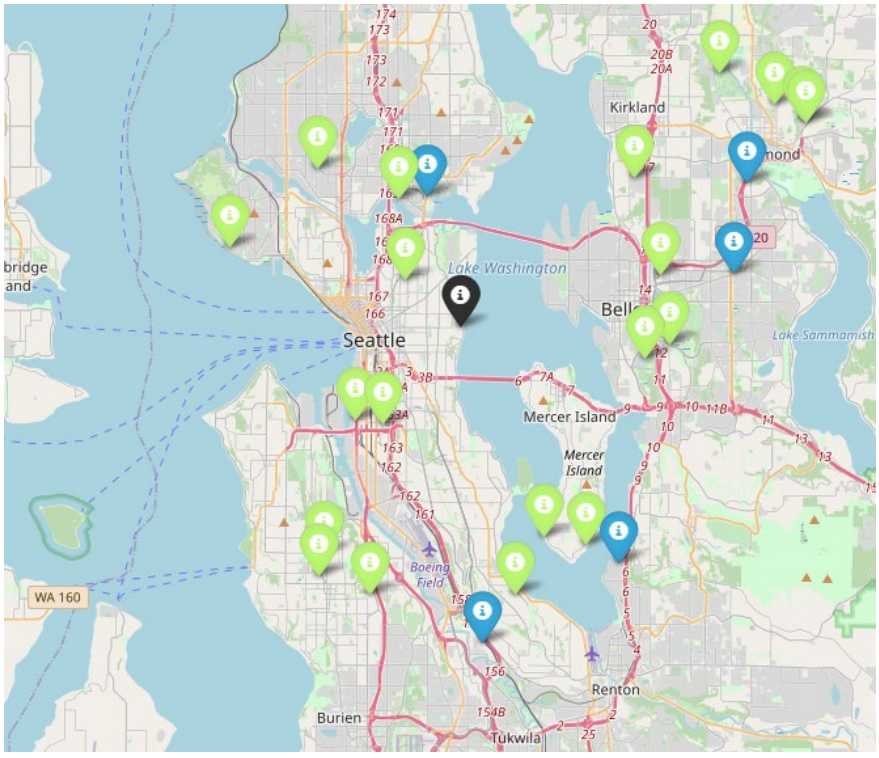

Figura 9. Ejemplo de un mapa de 25 clientes[5]

[4] Elaboración propia
[5] Elaboración propia

En las figuras anteriores se representan dos configuraciones de problemas distintos. En la primera, el número de clientes a atender son 8, de los cuales 2 clientes tienen que ser visitados obligatoriamente por el camión (nodos azules) y 6 clientes pueden ser visitados tanto por el camión como por un UAV (nodos verdes). El nodo negro se refiere al almacén desde el cual partirán y llegarán los vehículos involucrados en la ruta de reparto.

De igual forma, en la segunda figura se representa una configuración de 25 clientes, de los cuales 5 tienen que ser visitados por el camión, mientras que los demás nodos no tienen esa restricción. A partir de ahora, las restantes figuras sobre la distribución de los clientes que se representen en un mapa emplearán el mismo código de colores para identificar los nodos que tienen que ser visitados por el camión, o por camión/UAV.

Por otra parte, otros parámetros se han tomado constantes, ya que su efecto sobre el tiempo de ejecución es mínimo cuando se trata de comparar entre los distintos algoritmos. Estos parámetros son:

- Requerir camión en el almacén: este parámetro puede ser tanto 1 como 0. Para el caso de tomar un valor nulo, se considera que el camión no tiene que estar necesariamente en el almacén para que los drones comiencen a realizar entregas. En cambio, si toma el valor 1 (nuestro caso), se forzará a que el inicio de la operación de entrega de paquetes no sea hasta que el camión esté disponible en el almacén.
- Requiere conductor: este parámetro también podrá tomar únicamente dos valores (1 o 0). Para un valor de 1, el conductor tendrá que estar disponible cada vez que un dron haga una operación de aterrizaje/despegue desde el camión, mientras que si el valor es nulo, los UAVs podrán operar de forma autónoma.
- Consumo de energía: existen distintos modelos de consumo de energía por parte de los UAVs, pero para este documento se ha decidido escoger un modelo no lineal. Otras opciones serían el uso de modelos lineales, constantes o basados en distancia recorrida.

5.1 Modelo exacto

En primer lugar se van a mostrar los resultados obtenidos para las simulaciones del modelo exacto, y puesto que no se trata del objetivo principal de este trabajo evaluar dicho modelo exacto, se expondrán brevemente los resultados obtenidos. Además, debido a los elevados tiempos de ejecución necesarios para poder resolver los problemas de forma exacta, se ha limitado el estudio para rutas de 8 clientes, y donde además se ha impuesto tiempo de corte al algoritmo para evitar que se extienda en exceso evaluando una determinada ramificación del problema, por lo que cuando el tiempo de ejecución es elevado, el valor objetivo obtenido puede no ser el valor óptimo, como se mostrará más adelante.

En la primera tabla (Tabla 5) se muestran los resultados obtenidos para 5 problemas distintos cuando se emplea tanto 1 UAV como 4 UAVs. Estos UAVs tienen una configuración de alta velocidad de crucero y baja autonomía, lo cual provoca que en función de la configuración del problema (proximidad de los nodos entre sí), el efecto de incrementar el número de UAVs no se vea reflejado en el valor objetivo. Sin embargo, al aumentar el número de UAVs a 4, el tiempo de ejecución aumenta significativamente debido al aumento de la complejidad, y por ende, al número de posibles combinaciones/rutas que deben ser analizadas por el algoritmo.

Este efecto se puede observar claramente en los dos últimos problemas, cuyo valor objetivo no varía entre el uso de 1 o 4 UAVs. Esto se debe a que en las rutas calculadas, se usa únicamente un UAV o incluso ninguno, ya que la distancia entre los nodos es muy elevada y la baja autonomía de los UAVs no permite emplear los drones para colaborar con el camión. A continuación se muestran dos imágenes, en la primera se representan los nodos del primer problema de la tabla, el cuál si tiene obtiene una mejora al aumentar el número de UAVS, ya que sus nodos están cerca entre sí, y por lo tanto los UAVs pueden realizar tareas de reparto. En cambio, en la segunda imagen se muestra el último problema, el cual tiene sus nodos muy alejados entre sí, impidiendo que los UAVs con baja autonomía puedan colaborar en las tareas de reparto.

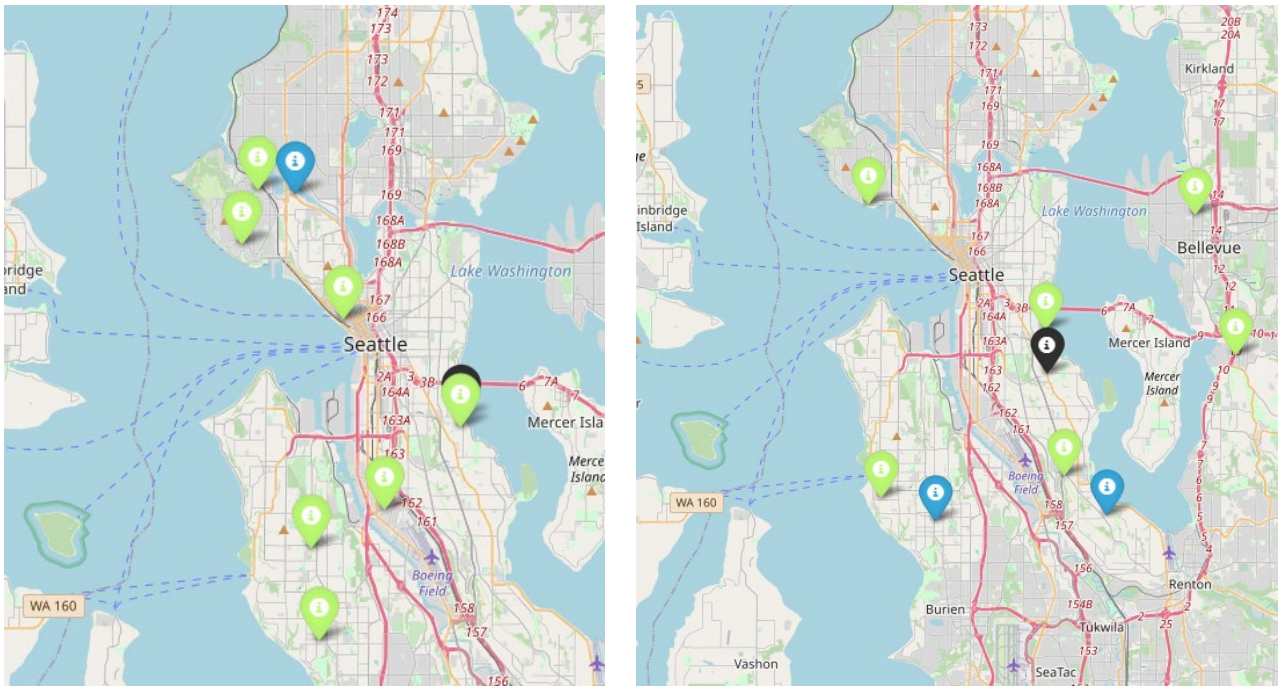

Figura 10. Izquierda: Representación de los clientes del problema ***20170608T121355407419***
Derecha: Representación de los clientes del problema ***20170608T121458174165***[6]

Tabla 5. Resultados para UAVs de alta velocidad y baja autonomía

ID PROBLEMA	Nº UAVS	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	1	3,68	3408,71
20170608T121355407419	4	11,08	3268,92
20170608T121411132375	1	7,91	3916,98
20170608T121411132375	4	11,25	3856,98
20170608T121426910678	1	5,02	3192,12
20170608T121426910678	4	11,53	3115,57
20170608T121442695307	1	6,31	3127,97
20170608T121442695307	4	11,81	3127,97
20170608T121458174165	1	4,87	5527,23
20170608T121458174165	4	8,04	5527,23

[6] Elaboración propia

En la Tabla 6 se muestran los resultados para los mismos problemas que en la tabla anterior, pero con UAVs de alta velocidad y autonomía. Puesto que en esta ocasión los UAVs tienen una mayor autonomía, el tiempo de ejecución aumenta al ser necesario evaluar una mayor cantidad de combinaciones. Por otra parte, se observa cómo los tiempos totales empleados en la ruta disminuyen respecto a la tabla anterior, ya que se pueden emplear más UAVs.

Sin embargo, se observa otro fenómeno que podría resultar extraño en un primer momento, pero que tras analizarlo, muestra la debilidad que tiene el uso de algoritmos exactos en este tipo de problemas. Este fenómeno es el aumento del valor objetivo cuando se emplean 4 UAVs en vez de 1. Esto carece de sentido, ya que aumentar el número de UAVs no debería de repercutir negativamente sobre el tiempo total empleado en realizar en una ruta, si no que debería de mejorar el valor, o como se ha visto anteriormente, en el peor de los casos, mantener el mismo valor objetivo.

En esta ocasión, este aumento se debe a la utilización de un tiempo de cutoff o corte cuando el algoritmo exacto está durante demasiado tiempo en una misma rama de la solución. Esto permite que el algoritmo exacto no eleve en exceso su tiempo de ejecución, pero tiene la contrapartida de que se dejan de calcular rutas/subrutas, cuya solución pueda ser la óptima. Debido al aumento de la complejidad al emplear un mayor número de UAVs, el efecto que tiene este tiempo de cutoff se hace cada vez más importante, hasta llegar a provocar distorsiones como las que se pueden ver en la tabla siguiente.

Tabla 6. Resultados para UAVs de alta velocidad y alta autonomía

ID PROBLEMA	Nº UAVS	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	1	7,81	2831,6
20170608T121355407419	4	15,49	3009,5
20170608T121411132375	1	10,29	3060,52
20170608T121411132375	4	15,34	3180,81
20170608T121426910678	1	10,32	2895,11
20170608T121426910678	4	15,62	2826,96
20170608T121442695307	1	10,37	2428,75
20170608T121442695307	4	16,87	1909,75
20170608T121458174165	1	10,21	4351,94
20170608T121458174165	4	13,57	5085,69

Para finalizar, las dos últimas configuraciones establecidas en este proyecto; baja velocidad y autonomía, y baja velocidad y alta autonomía se muestran en las tablas siguientes (Tabla 7 y Tabla 8). En estas tablas se pueden observar los efectos mencionados anteriormente para las dos configuraciones de UAVs anteriores.

Además, a partir de la comparación con las configuraciones anteriores, se desprende que el valor objetivo se ve influenciado en mayor medida por la autonomía de los UAVs que por la velocidad de los mismos, ya que para igualdad de velocidades y distintas autonomías, el cambio observado en el valor objetivo es mucho mayor que cuando se comparan dos configuraciones de UAVs con autonomías similares, pero velocidades distintas.

Tabla 7. Resultados para UAVs de baja velocidad y baja autonomía

ID PROBLEMA	Nº UAVS	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	1	10,19	3498,92
20170608T121355407419	4	13,12	3422,62
20170608T121411132375	1	10,19	3789,38
20170608T121411132375	4	13,11	4117,35
20170608T121426910678	1	10,22	3422,1
20170608T121426910678	4	13,79	3530,81
20170608T121442695307	1	10,25	2870,11
20170608T121442695307	4	14,41	2477,53
20170608T121458174165	1	6,09	5133,03
20170608T121458174165	4	11,8	5133,03

Tabla 8. Resultados para UAVs de baja velocidad y alta autonomía

ID PROBLEMA	Nº UAVS	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	1	10,36	3285,22
20170608T121355407419	4	16,79	2553,81
20170608T121411132375	1	10,32	3638,32
20170608T121411132375	4	15,95	2840,07
20170608T121426910678	1	10,33	3150,78
20170608T121426910678	4	16,13	2932,32
20170608T121442695307	1	10,39	3008,53
20170608T121442695307	4	17,2	2001,78
20170608T121458174165	1	10,3	4582,01
20170608T121458174165	4	15,56	3230

5.2 Heurística secuencial vs paralela

Una vez expuestos los resultados para el algoritmo exacto, el siguiente paso es mostrar los resultados obtenidos mediante la implementación de la heurística descrita en capítulos anteriores, así como mostrar una comparativa entre la heurística secuencial, paralela y el algoritmo exacto. Para ello, se han dividido estos resultados por los distintos factores que pueden afectar tanto al valor objetivo como al tiempo de ejecución. Estos son:

- Número de clientes o nodos
- Tipología de los UAVs
- Número de UAVs empleados
- Número de iteraciones del algoritmo heurístico

5.2.1 Comparación por número de clientes (nodos)

En primer lugar, se comenzará por el estudio de los resultados obtenidos en función del número de clientes o nodos. Puesto que este parámetro es el principal en cuanto al efecto que provoca sobre el tiempo de ejecución y el valor objetivo, se realizará un análisis más detallado, descomponiendo los resultados en problemas de 8, 25 50 y 100 clientes.

5.2.1.1 8 clientes

En la Tabla 9 se muestran los resultados de la heurística secuencial para el caso de que los UAVs empleados tengan una alta velocidad y una baja autonomía. En esta tabla se observan los resultados para distintos problemas en función del número de UAVs y del número de iteraciones del algoritmo heurístico. Estos resultados vienen descritos por los tiempos empleados en las distintas fases de la heurística, el tiempo total de ejecución y el valor objetivo obtenido.

Desde un primer momento, contrastan los valores de tiempo de ejecución obtenidos mediante el algoritmo heurístico frente al algoritmo exacto, aunque esto podrá observase mejor más adelante. Otro aspecto que se puede observar, es que para ciertos problemas, el valor objetivo es independiente del número de UAVs o de iteraciones. Esto se debe que al igual que el algoritmo exacto tenía sus ventajas y desventajas, la heurística permite acercarnos al valor objetivo en un tiempo de ejecución muy bajo, pero sin embargo, puede descartar ciertas ramas de soluciones que inicialmente puedan parecer poco probables de ser la solución óptima, llevando a que en ciertos casos, incluso aumentando el número de iteraciones, el algoritmo no sea capaz de aproximarse más a la solución óptima. Esto se verá en mayor profundidad en el apartado destinado al estudio del efecto del número de iteraciones sobre el valor objetivo y el tiempo de ejecución.

Un aspecto a destacar de los resultados mostrados es el tiempo empleado en la fase 2. Como se había mencionado anteriormente, esta fase de la heurística es la menos relevante desde un punto de vista del tiempo de ejecución, y esto queda claramente evidenciado en esta tabla, ya que a pesar de que el tiempo total es muy bajo, la influencia de la fase 2 es prácticamente nula.

De igual forma, la fase 1 tiene un elevado peso en el tiempo de ejecución, sobre todo cuando se incrementa el número de iteraciones, permaneciendo casi invariable el tiempo empleado en la fase 3. Como se irá viendo conforme se aumente el número de clientes a visitar, el peso de cada fase irá variando, llegando a pasar a ser la fase 3 la fase con mayor peso en el tiempo total de ejecución. Sin embargo, y como se ha comentado anteriormente, la fase 2 permanecerá siempre en un segundo plano en lo que a tiempo de ejecución se refiere.

Tabla 9. Heurística secuencial: UAVs de alta velocidad y baja autonomía para 8 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	1	1	0,02	-	0,01	0,03	3425,66
20170608T121355407419	4	1	0,03	-	0,01	0,03	3282,74
20170608T121355407419	1	10	0,08	-	0,01	0,09	3425,66
20170608T121355407419	4	10	0,11	0	0,01	0,12	3282,74
20170608T121411132375	1	1	0,02	0	0,02	0,04	4024,26
20170608T121411132375	4	1	0,02	0	0,03	0,05	4024,26
20170608T121411132375	1	10	0,05	0	0,04	0,09	4024,26
20170608T121411132375	4	10	0,08	0	0,04	0,12	4024,26
20170608T121426910678	1	1	0,02	0	0,03	0,05	3222,12
20170608T121426910678	4	1	0,02	0	0,04	0,06	3192,12
20170608T121426910678	1	10	0,08	0	0,05	0,14	3205,57
20170608T121426910678	4	10	0,1	0	0,05	0,15	3192,12
20170608T121442695307	1	1	0,02	0	0,01	0,03	3168,29
20170608T121442695307	4	1	0,03	0	0,03	0,06	3251,66
20170608T121442695307	1	10	0,1	-	0,01	0,11	3168,29
20170608T121442695307	4	10	0,1	0	0,03	0,13	3251,66
20170608T121458174165	1	1	0,01	-	0	0,01	5527,23
20170608T121458174165	4	1	0,01	-	0	0,01	5527,23
20170608T121458174165	1	10	0,01	-	0	0,01	5527,23
20170608T121458174165	4	10	0,02	-	0	0,02	5527,23

En la siguiente tabla se muestran los resultados de la heurística en paralelo para la misma tipología de UAVs que en la tabla anterior, alta velocidad y baja autonomía. Esto permitirá comparar los resultados obtenidos entre ambas simulaciones sin tener que atender al efecto que provoca la tipología de los UAVs sobre los resultados.

Como era de esperar, al tratarse de problemas de 8 clientes únicamente, los valores tanto de tiempo de ejecución como de valor objetivo no varían significativamente respecto a la implementación secuencial. Sin embargo, se pueden observar ciertas tendencias que en problemas de mayor complejidad se escenificarán con mayor claridad. Se trata de la disminución del tiempo de la fase 1, gracias a la implementación en paralelo de dicha fase, pero sin embargo, se produce un empeoramiento en los tiempos de las fases 2 y 3, ya que los cálculos realizados por la fase 1 no son los óptimos, al no contar con toda la información. Esto se debe a que la comunicación existente entre la fase 1 y las fases 2 y 3 permite que estas dos últimas fases siempre tengan la información actualizada

sobre las rutas calculadas por la fase 1, pero sin embargo la fase 1 recibe los resultados de las fases 2 y 3 con retraso. Esto provoca que los cálculos llevados por la fase 1 no sean tan buenos como los realizados por la heurística secuencial, forzando a las fases 2 y 3 a realizar más cálculos intermedios hasta poder llegar a una solución subóptima.

Tabla 10. Heurística en paralelo: UAVs de alta velocidad y baja autonomía para 8 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170608T121355407419	1	1	0,02	-	0,01	0,03	3425,66
20170608T121355407419	4	1	0,02	-	0,01	0,03	3282,74
20170608T121355407419	1	10	0,08	-	0,01	0,09	3425,66
20170608T121355407419	4	10	0,1	-	0,02	0,11	3282,74
20170608T121411132375	1	1	0,01	-	0,02	0,03	4024,26
20170608T121411132375	4	1	0,01	0	0,03	0,04	4024,26
20170608T121411132375	1	10	0,03	-	0,05	0,07	4024,26
20170608T121411132375	4	10	0,05	-	0,05	0,1	4024,26
20170608T121426910678	1	1	0,01	0	0,03	0,04	3222,12
20170608T121426910678	4	1	0	0	0,04	0,05	3192,12
20170608T121426910678	1	10	0,04	0	0,07	0,11	3205,57
20170608T121426910678	4	10	0,07	0	0,06	0,13	3192,12
20170608T121442695307	1	1	0,01	-	0,01	0,03	3168,29
20170608T121442695307	4	1	0,01	0	0,03	0,04	3251,66
20170608T121442695307	1	10	0,09	-	0,01	0,1	3168,29
20170608T121442695307	4	10	0,08	0	0,05	0,12	3251,66
20170608T121458174165	1	1	0,01	-	0	0,01	5527,23
20170608T121458174165	4	1	0,01	-	0	0,01	5527,23
20170608T121458174165	1	10	0,01	-	0	0,01	5527,23
20170608T121458174165	4	10	0,01	-	0,01	0,02	5527,23

Por último en la Tabla 11 se muestra una comparativa entre la heurística en paralelo y la secuencial. Además, se han añadido los resultados del algoritmo exacto para poder comparar los valores de los tres algoritmos. Los parámetros de los UAVs son los mismos que los empleados anteriormente: alta velocidad y baja autonomía.

A partir de los valores mostrados en la tabla, se puede observar la clara mejora que ofrece el algoritmo heurístico frente al algoritmo exacto, que incluso para problemas pequeños (8 clientes), la reducción del tiempo de ejecución es muy notable, sin que esto perjudique en exceso al valor objetivo del algoritmo exacto.

Esto se produce para ambos algoritmos heurísticos, tanto cuando se ejecuta secuencialmente como cuando se ejecuta en paralelo. Sin embargo, al tratarse de problemas de una complejidad baja, el efecto del tiempo de ejecución en paralelo es prácticamente nulo, al igual que el efecto sobre el valor objetivo. Es por ello que si el objetivo de este trabajo fuese mejorar el tiempo de ejecución de este tipo de problemas, la aproximación llevada a cabo con este algoritmo de ejecución en paralelo no sería la más adecuada. Sin embargo, más adelante se mostrará la importante mejora que ofrece esta implementación para problemas de mayor magnitud (50 o más clientes) frente a la ejecución secuencial, lo cual se alinea con el propósito inicial del trabajo.

Tabla 11. Heurística secuencial vs paralela vs algoritmo exacto: UAVs de alta velocidad y baja autonomía para 8 clientes

ID PROBLEMA	TIEMPO EJECUCIÓN PARALELO [SEC]	VALOR OBJETIVO PARALELO [SEC]	TIEMPO EJECUCIÓN SECUENCIAL [SEC]	VALOR OBJETIVO SECUENCIAL [SEC]	TIEMPO EJECUCIÓN EXACTO [SEC]	VALOR OBJETIVO EXACTO [SEC]
20170608T121355407419	0,03	3425,66	0,03	3425,66	3,5	3408,7
20170608T121355407419	0,03	3282,74	0,03	3282,74	11,1	3268,9
20170608T121355407419	0,09	3425,66	0,09	3425,66	3,5	3408,7
20170608T121355407419	0,11	3282,74	0,12	3282,74	11,1	3268,9
20170608T121411132375	0,03	4024,26	0,04	4024,26	7,9	3917,0
20170608T121411132375	0,04	4024,26	0,05	4024,26	11,3	3857,0
20170608T121411132375	0,07	4024,26	0,09	4024,26	7,9	3917,0
20170608T121411132375	0,1	4024,26	0,12	4024,26	11,3	3857,0
20170608T121426910678	0,04	3222,12	0,05	3222,12	4,7	3192,1
20170608T121426910678	0,05	3192,12	0,06	3192,12	11,4	3115,6
20170608T121426910678	0,11	3205,57	0,14	3205,57	4,7	3192,1
20170608T121426910678	0,13	3192,12	0,15	3192,12	11,4	3115,6
20170608T121442695307	0,03	3168,29	0,03	3168,29	5,5	3128,0
20170608T121442695307	0,04	3251,66	0,06	3251,66	11,7	3128,0
20170608T121442695307	0,1	3168,29	0,11	3168,29	5,5	3128,0
20170608T121442695307	0,12	3251,66	0,13	3251,66	11,7	3128,0
20170608T121458174165	0,01	5527,23	0,01	5527,23	4,4	5527,2
20170608T121458174165	0,01	5527,23	0,01	5527,23	7,3	5527,2
20170608T121458174165	0,01	5527,23	0,01	5527,23	4,4	5527,2
20170608T121458174165	0,02	5527,23	0,02	5527,23	7,3	5527,2

5.2.1.2 25 clientes

En siguiente lugar se evaluarán problemas de una mayor complejidad al tratarse de un total de 25 clientes. Para mostrar los resultados, se seguirá el mismo procedimiento que para el caso de 8 clientes, pero en esta ocasión no podrá llevarse a cabo la comparación con los resultados del algoritmo exacto, ya que únicamente se han realizado simulaciones de este algoritmo para 8 clientes debido al elevado tiempo de ejecución que conllevaba simular problemas más complejos con dicho algoritmo.

Tabla 12. Heurística secuencial: UAVs de alta velocidad y baja autonomía para 25 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170606T113038113409	1	1	0,42	0,01	0,44	0,87	8799,86
20170606T113038113409	4	1	0,5	0,01	0,43	0,94	8055,31
20170606T113038113409	1	10	2,32	0,03	1,84	4,19	8795,83
20170606T113038113409	4	10	3,22	0,01	0,66	3,9	8055,31
20170606T113251786976	1	1	0,22	0	0,14	0,36	10320,36
20170606T113251786976	4	1	0,34	0	0,07	0,41	9877,68
20170606T113251786976	1	10	2,14	0	0,14	2,28	10320,36
20170606T113251786976	4	10	3,67	0,01	0,29	3,96	9854,06
20170606T113339368121	1	1	0,46	0,01	0,67	1,14	8589,96
20170606T113339368121	4	1	0,41	0,03	1,02	1,46	8516,88
20170606T113339368121	1	10	3,45	0,08	4,82	8,36	8610,46
20170606T113339368121	4	10	3,9	0,21	7,93	12,04	8469,16
20170606T113427164164	1	1	0,38	0	0,04	0,42	9046,31
20170606T113427164164	4	1	0,6	0	0,16	0,77	9252,96
20170606T113427164164	1	10	2,57	0	0,04	2,6	9046,31
20170606T113427164164	4	10	5,85	0	0,16	6,02	9252,96
20170606T113515209066	1	1	0,19	0	0,18	0,38	9171,13
20170606T113515209066	4	1	0,36	0,01	0,29	0,66	9069,47
20170606T113515209066	1	10	3	0,01	0,39	3,4	9171,13
20170606T113515209066	4	10	4,24	0,01	0,29	4,55	9069,47

En la Tabla 12 se muestran los resultados de la heurística cuando se ejecuta de forma secuencial para UAVs de alta velocidad y baja autonomía. Como era de esperar, tanto el tiempo de ejecución como el valor objetivo se han incrementado respecto a las simulaciones con 8 clientes. Un aspecto importante a destacar es que a pesar de haber triplicado el número de clientes, el tiempo de ejecución empleado por el algoritmo heurístico se ha incrementado en aproximadamente un orden de 100 veces. Aunque esto pueda llevar a pensar que el algoritmo heurístico no permite resolver problemas de gran complejidad debido a su elevado incremento en el tiempo de ejecución, si se compara con el algoritmo exacto, este último ni siquiera fue capaz de arrojar datos después de más de una hora de ejecución. Por ello, la ganancia es más que evidente, ya que el algoritmo heurístico únicamente ha precisado de 12 segundos en el peor de los casos. Además, ha de tenerse en cuenta los limitados recursos computacionales en los que ha sido llevada a cabo la simulación, siendo muy mejorables estos tiempos en ordenadores de mayor capacidad computacional.

Al igual que sucedía con los problemas de 8 clientes, y como se ha comentado en capítulos anteriores, la fase 2 apenas tiene influencia sobre el tiempo final de ejecución, siendo las otras dos fases las que más peso ejercen sobre el valor final. También se observa que la fase 1 suele ser la predominante, sobre todo cuando se eleva el número de iteraciones, mientras que incrementar el número de UAVs no tiene un efecto directo sobre ninguna de las dos fases, afectando en unos problemas en mayor medida a la fase 1 y en otros a la fase 3. Esto se debe a la propia heurística, ya que mientras que la fase 1 siempre es ejecutada, y por lo tanto a mayor número de iteraciones, mayor es su tiempo de ejecución, la fase 3 depende de los valores calculados por las fases previas, por lo que si los resultados de las fases anteriores se encuentran cercanos a un óptimo local, el tiempo invertido en la fase 3 será reducido, mientras que si debido a la configuración/distribución de los nodos en el mapa provoca que las fases previas no sean eficaces en el cálculo de una ruta subóptima, será la fase 3 la que tenga un mayor tiempo de ejecución.

Tabla 13. Heurística en paralelo: UAVs de alta velocidad y baja autonomía para 25 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170606T113038113409	1	1	0,08	0,01	0,6	0,69	8799,86
20170606T113038113409	4	1	0,16	0,01	0,58	0,75	8055,31
20170606T113038113409	1	10	0,9	0,03	2,42	3,34	8795,83
20170606T113038113409	4	10	2,63	0,03	0,89	3,55	8055,31
20170606T113251786976	1	1	0,14	0	0,16	0,3	10320,36
20170606T113251786976	4	1	0,28	0,02	0,09	0,38	9877,68
20170606T113251786976	1	10	2,01	0	0,21	2,22	10320,36
20170606T113251786976	4	10	3,44	0,01	0,38	3,83	9854,06
20170606T113339368121	1	1	0,05	0,01	0,82	0,88	8589,96
20170606T113339368121	4	1	0,05	0,05	1,1	1,2	8516,88
20170606T113339368121	1	10	1,33	0,08	5,44	6,85	8610,46
20170606T113339368121	4	10	0,5	0,21	9,5	10,21	8469,16
20170606T113427164164	1	1	0,36	0	0,04	0,41	9046,31
20170606T113427164164	4	1	0,45	0	0,24	0,69	9252,96

20170606T113427164164	1	10	2,53	0	0,04	2,57	9046,31
20170606T113427164164	4	10	5,77	0	0,24	6,01	9252,96
20170606T113515209066	1	1	0,08	0	0,2	0,29	9171,13
20170606T113515209066	4	1	0,13	0,01	0,38	0,53	9069,47
20170606T113515209066	1	10	2,74	0,01	0,47	3,22	9171,13
20170606T113515209066	4	10	4,02	0,01	0,42	4,45	9069,47

En la tabla anterior (Tabla 13) se muestran los resultados para el algoritmo paralelo. Al tratarse de problemas de mayor envergadura, y por lo tanto, de obtener tiempo de ejecución más elevados que en los problemas para 8 clientes, la ganancia de tiempo entre un algoritmo y otro comienza a ser más visible. Esto se debe a la reducción del tiempo de ejecución de la fase 1, ya que esta fase es la que se ejecuta en paralelo con las otras dos, por lo que mientras se están realizando cálculos en las fases 2 y 3, la fase 1 puede ir ejecutando en paralelo y obteniendo nuevas subrutas. Esto tiene la contrapartida, como ya se ha expuesto en el capítulo anterior de que la información sobre las rutas calculadas que tiene la fase 1 está retrasada, por lo que las rutas calculadas tienden a ser peores que en la heurística secuencial, forzando a las fases posteriores a invertir más tiempo de ejecución para llegar hasta el valor subóptimo de la rama en la que se encuentra el algoritmo.

A pesar de ello, el resultado neto es positivo, permitiendo obtener un ahorro en el tiempo de ejecución total. Por otra parte, para los problemas en los que el tiempo de la fase 3 es muy reducido frente al tiempo de la fase 1, la capacidad del algoritmo heurístico ejecutado en paralelo se ve rápidamente decrementada, ya que la fase limitante es en todo momento la fase 1, por lo que no existe una ejecución en paralelo de forma continuada, si no que se realizan cálculos en paralelo de forma intermitente.

Tabla 14. Heurística secuencial vs heurística en paralelo: UAVs de alta velocidad y baja autonomía para 25 clientes

ID PROBLEMA	TIEMPO EJECUCIÓN SECUENCIAL [SEC]	VALOR OBJETIVO SECUENCIAL [SEC]	TIEMPO EJECUCIÓN PARALELO [SEC]	VALOR OBJETIVO PARALELO [SEC]
20170606T113038113409	0,87	8799,86	0,69	8799,86
20170606T113038113409	0,94	8055,31	0,75	8055,31
20170606T113038113409	4,19	8795,83	3,34	8795,83
20170606T113038113409	3,9	8055,31	3,55	8055,31
20170606T113251786976	0,36	10320,36	0,3	10320,36
20170606T113251786976	0,41	9877,68	0,38	9877,68
20170606T113251786976	2,28	10320,36	2,22	10320,36
20170606T113251786976	3,96	9854,06	3,83	9854,06
20170606T113339368121	1,14	8589,96	0,88	8589,96
20170606T113339368121	1,46	8516,88	1,2	8516,88
20170606T113339368121	8,36	8610,46	6,85	8610,46

20170606T113339368121	12,04	8469,16	10,21	8469,16
20170606T113427164164	0,42	9046,31	0,41	9046,31
20170606T113427164164	0,77	9252,96	0,69	9252,96
20170606T113427164164	2,6	9046,31	2,57	9046,31
20170606T113427164164	6,02	9252,96	6,01	9252,96
20170606T113515209066	0,38	9171,13	0,29	9171,13
20170606T113515209066	0,66	9069,47	0,53	9069,47
20170606T113515209066	3,4	9171,13	3,22	9171,13
20170606T113515209066	4,55	9069,47	4,45	9069,47

Por último, en la Tabla 14 se muestra una comparativa de los resultados para los dos algoritmos evaluados donde en todas las simulaciones llevada a cabo existe un ahorro en el tiempo de ejecución. Además, el valor objetivo es el mismo que para el algoritmo secuencial, demostrando que la ejecución en paralelo no tiene una repercusión negativa sobre la ruta subóptima cuando el problema es de tamaño medio-bajo (25 clientes).

5.2.1.3 50 clientes

En siguiente lugar se evaluarán 5 problemas/rutas con un total de 50 clientes a visitar. Es en este tipo de problemas donde se comienza a observar más claramente el efecto de emplear la ejecución en paralelo en vez de la ejecución secuencial para obtener una ruta subóptima en menor tiempo. A continuación se muestran las tablas con los resultados de la heurística secuencial, en paralelo, y una comparativa de ambas.

Tabla 15. Heurística secuencial: UAVs de alta velocidad y baja autonomía para 50 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170606T114000833192	1	1	3,58	0,04	2,47	6,09	12892,43
20170606T114000833192	4	1	6,53	0,05	1,34	7,92	12806,86
20170606T114000833192	1	10	13,7	0,3	16,27	30,27	12892,43
20170606T114000833192	4	10	100,01	0,05	1,39	101,45	12806,86
20170606T114145593946	1	1	3,76	0,13	3,1	6,99	13715,21
20170606T114145593946	4	1	9,11	0,25	29,84	39,2	12726,22
20170606T114145593946	1	10	12,29	1,25	18,67	32,21	13430,95
20170606T114145593946	4	10	172,59	1,21	95,96	269,76	12638,89
20170606T114330252507	1	1	3,52	0,07	3,12	6,71	12980,97
20170606T114330252507	4	1	8,39	0,04	2,75	11,18	12515,7
20170606T114330252507	1	10	17,81	0,7	26,3	44,81	13014,65

20170606T114330252507	4	10	165,96	0,05	3,11	169,11	12515,7
20170606T114511221132	1	1	3,79	0,29	3,29	7,37	12030,11
20170606T114511221132	4	1	6,77	0,1	13,37	20,24	10954,97
20170606T114511221132	1	10	14,29	2,78	28,19	45,26	11977,18
20170606T114511221132	4	10	96,58	0,32	49,88	146,78	10925,85
20170606T114654882472	1	1	4,3	0,11	1,39	5,81	13197,26
20170606T114654882472	4	1	5,78	0,03	0,85	6,66	12678,58
20170606T114654882472	1	10	19,11	0,98	11,73	31,82	13006,57
20170606T114654882472	4	10	73,51	0,09	1,95	75,56	12629,29

Tabla 16. Heurística en paralelo: UAVs de alta velocidad y baja autonomía para 50 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170606T114000833192	1	1	1,91	0,05	3,11	5,07	12941,2
20170606T114000833192	4	1	4,67	0,1	2,31	7,08	12806,86
20170606T114000833192	1	10	1,69	0,35	21,31	23,35	12886,16
20170606T114000833192	4	10	100,21	0,1	2,37	102,68	12806,86
20170606T114145593946	1	1	1,59	0,15	4,58	6,31	13715,21
20170606T114145593946	4	1	1,1	0,23	34,44	35,76	12726,22
20170606T114145593946	1	10	0,66	2,07	23,65	26,39	13430,95
20170606T114145593946	4	10	70,07	1,79	141,59	213,46	12638,89
20170606T114330252507	1	1	0,94	0,11	4,45	5,51	12980,97
20170606T114330252507	4	1	5,51	0,09	4,22	9,82	12515,7
20170606T114330252507	1	10	2,31	0,83	33,79	36,93	13014,65
20170606T114330252507	4	10	164,72	0,11	4,73	169,56	12515,7
20170606T114511221132	1	1	1,27	0,36	4,64	6,28	12030,11
20170606T114511221132	4	1	0,25	0,12	16,88	17,25	10954,97
20170606T114511221132	1	10	0,45	4,28	34,16	38,9	11977,18
20170606T114511221132	4	10	58,68	0,43	64,99	124,1	10925,85

20170606T114654882472	1	1	3,07	0,18	2,01	5,26	13197,26
20170606T114654882472	4	1	5,04	0,08	1,26	6,38	12678,58
20170606T114654882472	1	10	8,43	1,67	15,82	25,93	13006,57
20170606T114654882472	4	10	72,69	0,15	2,66	75,5	12629,29

Tabla 17. Heurística secuencial vs heurística en paralelo: UAVs de alta velocidad y baja autonomía para 50 clientes

ID PROBLEMA	TIEMPO EJECUCIÓN SECUENCIAL [SEC]	VALOR OBJETIVO SECUENCIAL [SEC]	TIEMPO EJECUCIÓN PARALELO [SEC]	VALOR OBJETIVO PARALELO [SEC]
20170606T113038113409	6,09	12892,43	5,07	12941,2
20170606T114000833192	7,92	12806,86	7,08	12806,86
20170606T114000833192	30,27	12892,43	23,35	12886,16
20170606T114000833192	101,45	12806,86	102,68	12806,86
20170606T114000833192	6,99	13715,21	6,31	13715,21
20170606T114145593946	39,2	12726,22	35,76	12726,22
20170606T114145593946	32,21	13430,95	26,39	13430,95
20170606T114145593946	269,76	12638,89	213,46	12638,89
20170606T114145593946	6,71	12980,97	5,51	12980,97
20170606T114330252507	11,18	12515,7	9,82	12515,7
20170606T114330252507	44,81	13014,65	36,93	13014,65
20170606T114330252507	169,11	12515,7	169,56	12515,7
20170606T114330252507	7,37	12030,11	6,28	12030,11
20170606T114511221132	20,24	10954,97	17,25	10954,97
20170606T114511221132	45,26	11977,18	38,9	11977,18
20170606T114511221132	146,78	10925,85	124,1	10925,85
20170606T114511221132	5,81	13197,26	5,26	13197,26
20170606T114654882472	6,66	12678,58	6,38	12678,58
20170606T114654882472	31,82	13006,57	25,93	13006,57
20170606T114654882472	75,56	12629,29	75,5	12629,29

En la Tabla 17 se muestra una comparativa de los resultados entre la heurística secuencial y la heurística paralela. En primer lugar, destaca que se obtengan valores superiores de tiempo de ejecución para la ejecución paralela frente a la secuencial en dos ocasiones. Ambas simulaciones se corresponden con dos problemas que son evaluados con 4 UAVs y 10 iteraciones, lo cual provoca que la fase 1 aumente considerablemente su tiempo de ejecución. Como ya se ha comentado anteriormente, este hecho provoca que la ejecución en paralelo pierda efectividad. Además, en estos dos casos sucede que el tiempo de la fase 1 ya era el predominante en el algoritmo secuencial, por lo que la introducción del algoritmo en paralelo tiene un efecto adverso, ya que debido a la concepción del mismo y su necesidad de comunicarse entre los distintos hilos ejecutándose en paralelo, se introduce una pérdida de tiempo que en el algoritmo secuencial no existía, dando lugar a un efecto negativo en el cómputo global del tiempo de ejecución.

Por otra parte, el valor objetivo permanece en la mayoría de casos inmutable entre ambos algoritmos, pero en dos ocasiones el valor es inferior para la ejecución secuencial, y en otras dos ocasiones lo es para la ejecución en paralelo. A pesar de esta variación, la diferencia es muy pequeña y por lo tanto no debe ser tenido en cuenta a la hora de sacar conclusiones. Este efecto se debe a la propia heurística del problema, la cual puede llevar a que ante variaciones en la subruta calculada por la fase 1 (paralelo o secuencial), los resultados difieran en mayor medida que para otros problemas, llevando a las fases 2 y 3 a alcanzar resultados distintos.

5.2.1.4 100 clientes

Por último, se llega a los problemas de 100 clientes, los cuales serán los problemas más complejos que sean objeto de estudio en este trabajo. Debido a la magnitud de estos problemas, se ha decidido reducir las simulaciones realizadas a estos problemas a las distintas tipologías de UAVs, tomando constante tanto el número de UAVs como el número de iteraciones. Esto evita que los tiempos de ejecución sea muy elevados tanto para la heurística secuencial como paralela.

Tabla 18. Heurística secuencial: UAVs de alta velocidad y baja autonomía para 100 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170606T115437348436	1	1	59,76	3	25,99	88,76	19335,41
20170606T115823934453	1	1	34,4	2,91	29,87	67,18	19144,83
20170606T120227545709	1	1	41,68	4,24	18,72	64,64	20037,26
20170606T121241353494	1	1	32,61	2,75	47,4	82,76	19014,19
20170606T121632081849	1	1	40,53	6,03	22,41	68,97	18689,6

Tabla 19. Heurística en paralelo: UAVs de alta velocidad y baja autonomía para 100 clientes

ID PROBLEMA	Nº UAV	ITER	T1	T2	T3	TIEMPO EJECUCIÓN [SEC]	VALOR OBJETIVO [SEC]
20170606T115437348436	1	1	38,77	4,47	39,41	82,64	19335,41
20170606T115823934453	1	1	7,03	4,36	46,6	57,99	19144,83
20170606T120227545709	1	1	24,89	6,66	27,19	58,74	20037,26
20170606T121241353494	1	1	8,58	3,93	61,64	74,15	19030,22
20170606T121632081849	1	1	19,21	8,62	32,76	60,59	18689,6

Tabla 20. Heurística secuencial vs heurística en paralelo: UAVs de alta velocidad y baja autonomía para 100 clientes

ID PROBLEMA	TIEMPO EJECUCIÓN SECUENCIAL [SEC]	VALOR OBJETIVO SECUENCIAL [SEC]	TIEMPO EJECUCIÓN PARALELO [SEC]	VALOR OBJETIVO PARALELO [SEC]
20170606T115437348436	88,76	19335,41	82,64	19335,41
20170606T115823934453	67,18	19144,83	57,99	19144,83
20170606T120227545709	64,64	20037,26	58,74	20037,26
20170606T121241353494	82,76	19014,19	74,15	19030,22
20170606T121632081849	68,97	18689,6	60,59	18689,6

En la Tabla 20 se comparan los valores de las simulaciones, destacando nuevamente un valor objetivo superior en la ejecución en paralelo frente a la ejecución secuencial. Al igual que sucedía para algunos problemas de 50 clientes, este efecto es causado por la propia heurística y su dependencia del valor inicial calculado por la fase 1, la cual a su vez depende de la comunicación establecida entre los hilos ejecutados en paralelo, y puesto que la información es transferida con retraso, esto puede llegar a causar este tipo de fenómenos. Sin embargo, y al igual que sucedía anteriormente, su efecto es prácticamente despreciable sobre el valor objetivo.

5.2.1.5 Comparativa

Para finalizar este apartado del estudio se realizará una breve recopilación de la información mostrada hasta ahora en tablas en forma de gráficas de barras que permitan resumir lo expuesto anteriormente.

Para llevar a cabo esta comparativa, se dividirá este apartado en:

- Evaluación de fases:
  - Fase 1
  - Fases 2 y 3
- Tiempo total de ejecución
- Valor objetivo

A su vez, se atenderá tanto a distintos tipos de problemas (diferente cantidad de clientes) como distintos tipos de UAVs: alta/baja velocidad y alta/baja autonomía.

En la primera figura se representan los tiempos de ejecución de la fase 1 para cada una de las tipologías de UAVs y para cada agrupación de problemas por número de clientes. En los apartados anteriores se han mostrado en tablas las simulaciones siempre para una misma tipología de UAV, la cual se correspondía con la tipología 101 (alta velocidad y baja autonomía). En cambio, en esta primera imagen se pueden extraer conclusiones de los restantes tipos de UAVs.

En primer lugar, para valores elevados de T1 para el algoritmo heurístico secuencial, el algoritmo paralelo pierde eficiencia, ya que valores altos de T1 significa que su peso sobre el valor total del tiempo de ejecución es muy predominante, impidiendo acelerar la obtención de la ruta subóptima. Esto último se podrá observar y comprender mejor con las gráficas de las fases 2 y 3 del algoritmo que se muestran en la Figura 11.

En siguiente lugar, es necesario comentar que estos son los valores medios, sin atender a otros criterios como el número de iteraciones, puesto que esta casuística será estudiada más adelante en el apartado destinado para ello.

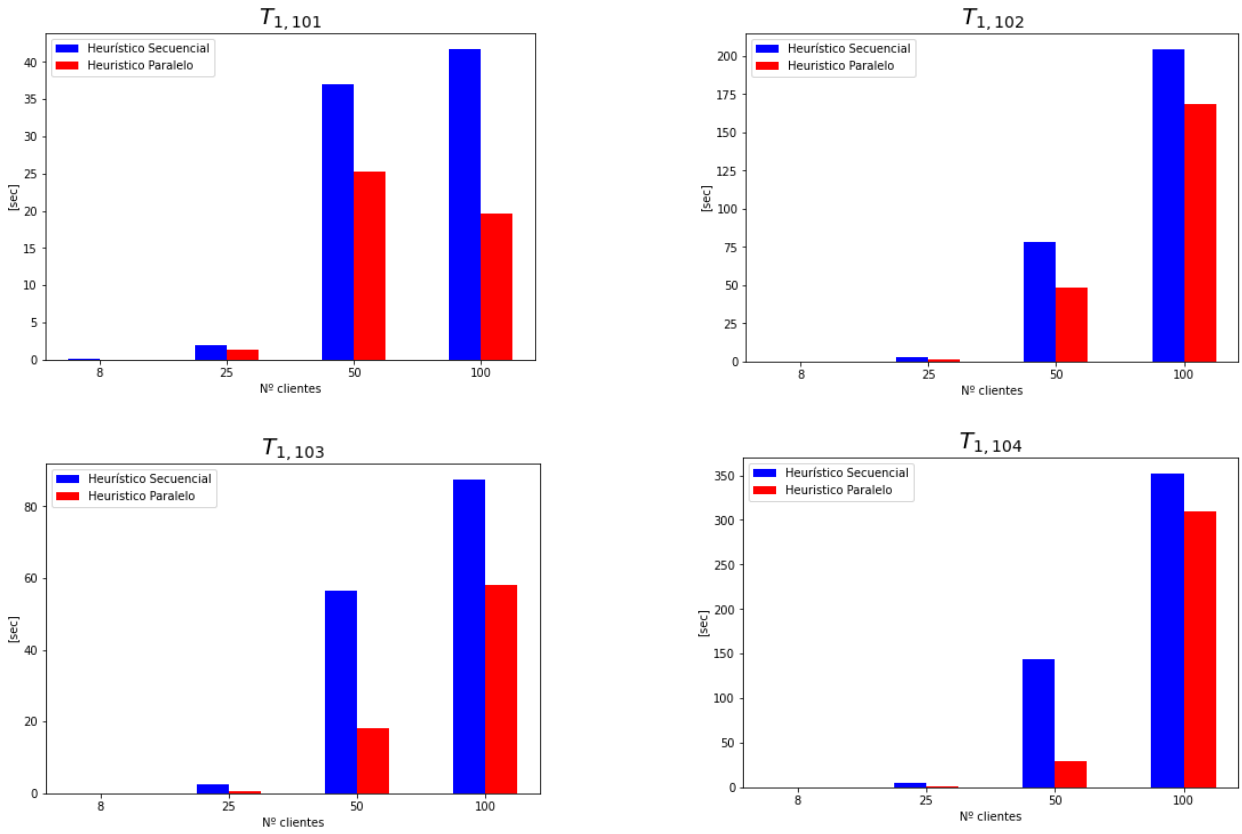

Figura 11. Tiempo de ejecución en la fase 1 para distintos números de clientes y tipologías de UAVs

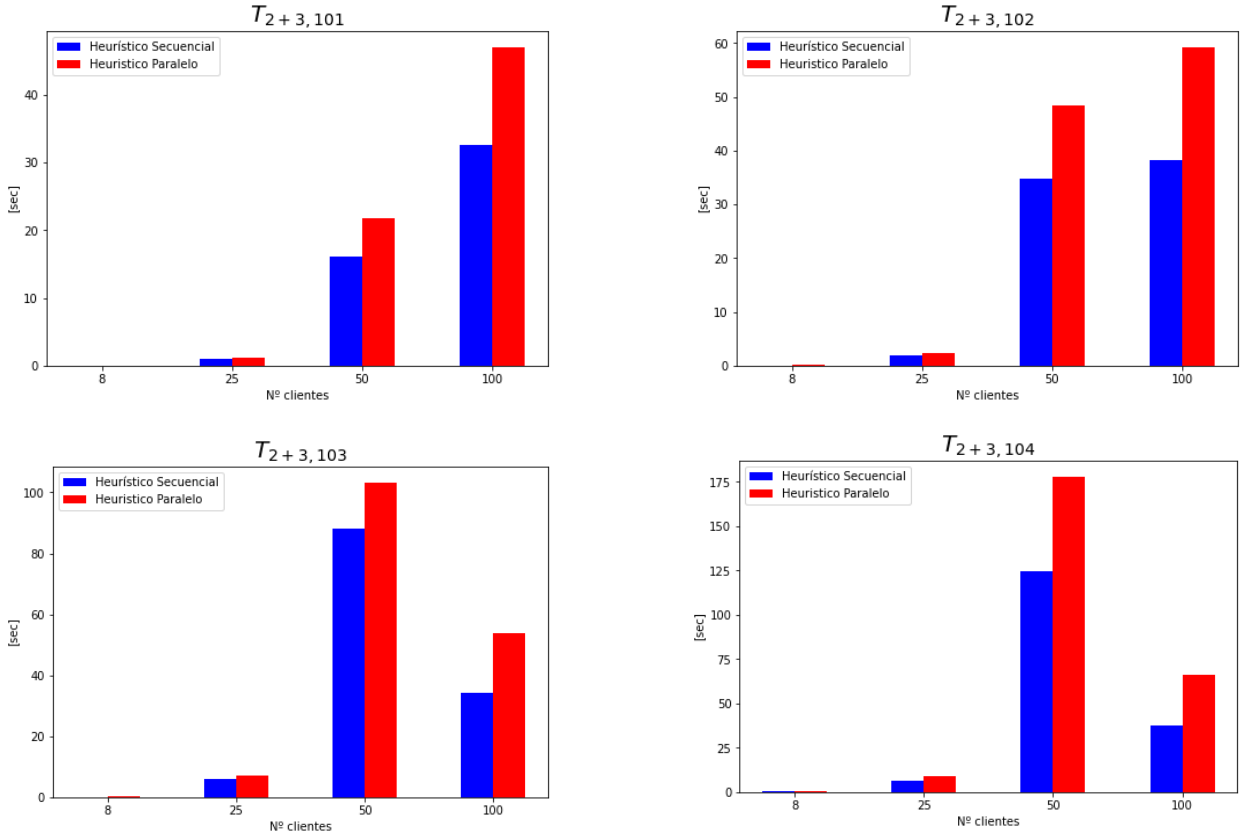

Figura 12. Tiempo de ejecución en las fases 2 y 3 para distintos números de clientes y tipologías de UAVs

En la Figura 11 se muestran los resultados de los tiempos de ejecución medios de las fases 2 y 3. Como era de esperar, estos tiempos son superiores para el algoritmo ejecutado en paralelo que para el algoritmo ejecutado de forma secuencial. Esto se debe al empeoramiento de las subrutas calculadas por la fase 1 cuando se ejecuta en paralelo, como ya se ha expuesto anteriormente.

Un aspecto importante y que ha sido mencionado en los resultados de la fase 1 es el efecto que tiene la predominancia de la fase 1 sobre el resto de fases cuando se evalúan los tiempos de ejecución, ya que cuando esta predominancia es muy elevada, como puede ser el caso de la gráfica para los UAVs de tipo 104 y 100 clientes, donde la diferencia entre la fase 1 y las fases 2+3 es muy elevada, provocando que el ahorro de tiempo en la fase 1 sea muy bajo, mientras que si nos fijamos en esa misma gráfica (104), pero para 50 clientes, las fases 2+3 representan prácticamente la mitad del tiempo de ejecución, lo que permite que el ahorro de tiempo de ejecución en la fase 1 sea muy elevado sin que exista un perjuicio muy elevado en las fases 2 y 3.

En siguiente lugar se muestran las gráficas para el tiempo total de ejecución (Figura 12), la cual permitirá dar una visión completa sobre el efecto del número de clientes sobre el tiempo de ejecución.

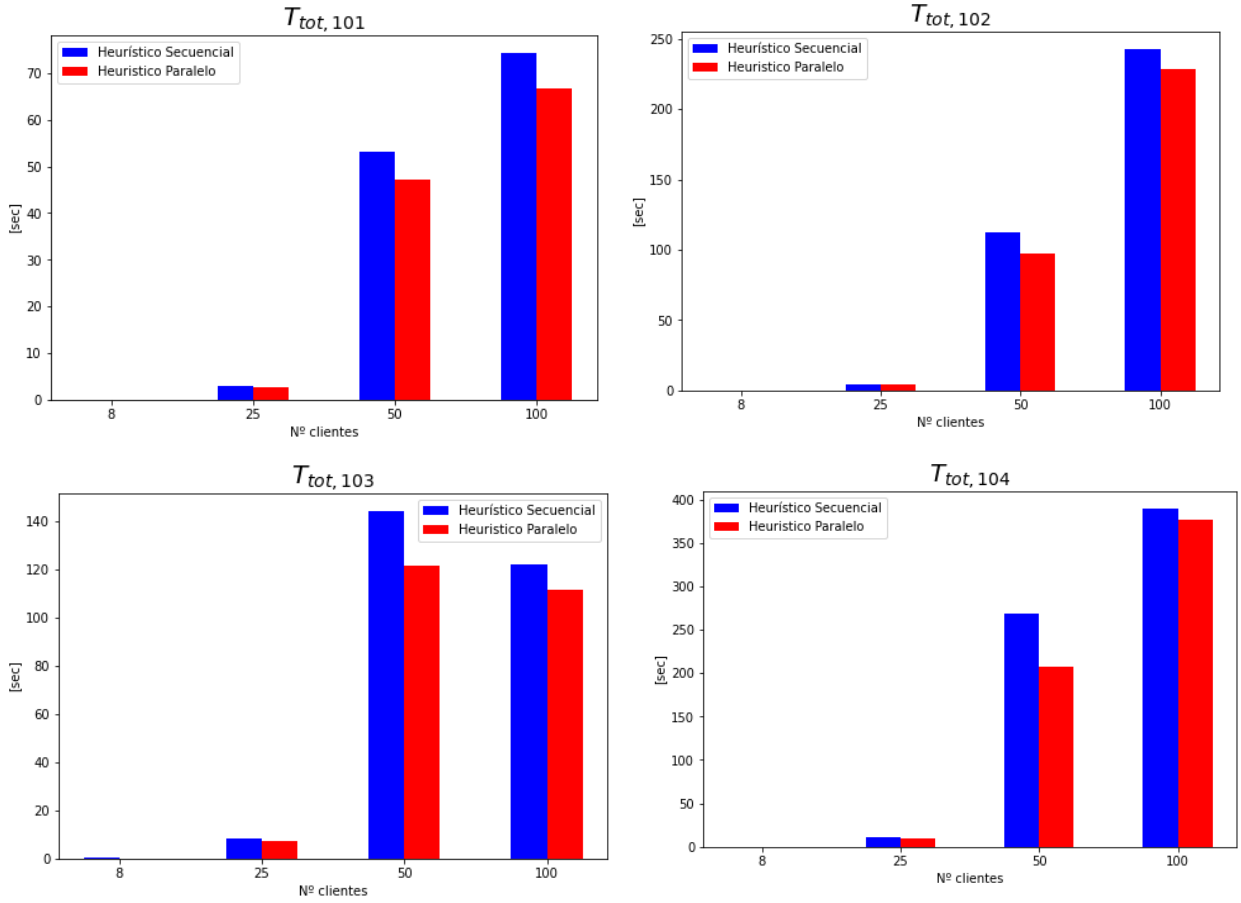

Figura 13. Tiempo de ejecución total para distintos números de clientes y tipologías de UAVs

Al igual que sucedía en las figuras de los tiempos de ejecución por fases, para el tiempo total de ejecución se cumple que en aquellos casos donde los tiempos de la fase 1 y las fases 2+3 son similares, el tiempo total de ejecución se ve reducido en mayor medida que en los casos en los que la fase 1 ejerce una fuerte predominancia sobre las otras dos fases.

Finalmente se muestran las gráficas del valor objetivo para las distintas clasificaciones por clientes y por tipología de UAVs. Al igual que sucedía en los apartados anteriores, donde este valor objetivo no se veía influenciado por el uso del algoritmo en paralelo para UAVs del tipo 101, ahora se puede observar también en la Figura 13 que para el resto de tipologías los resultados son los mismos para un algoritmo que para el otro.

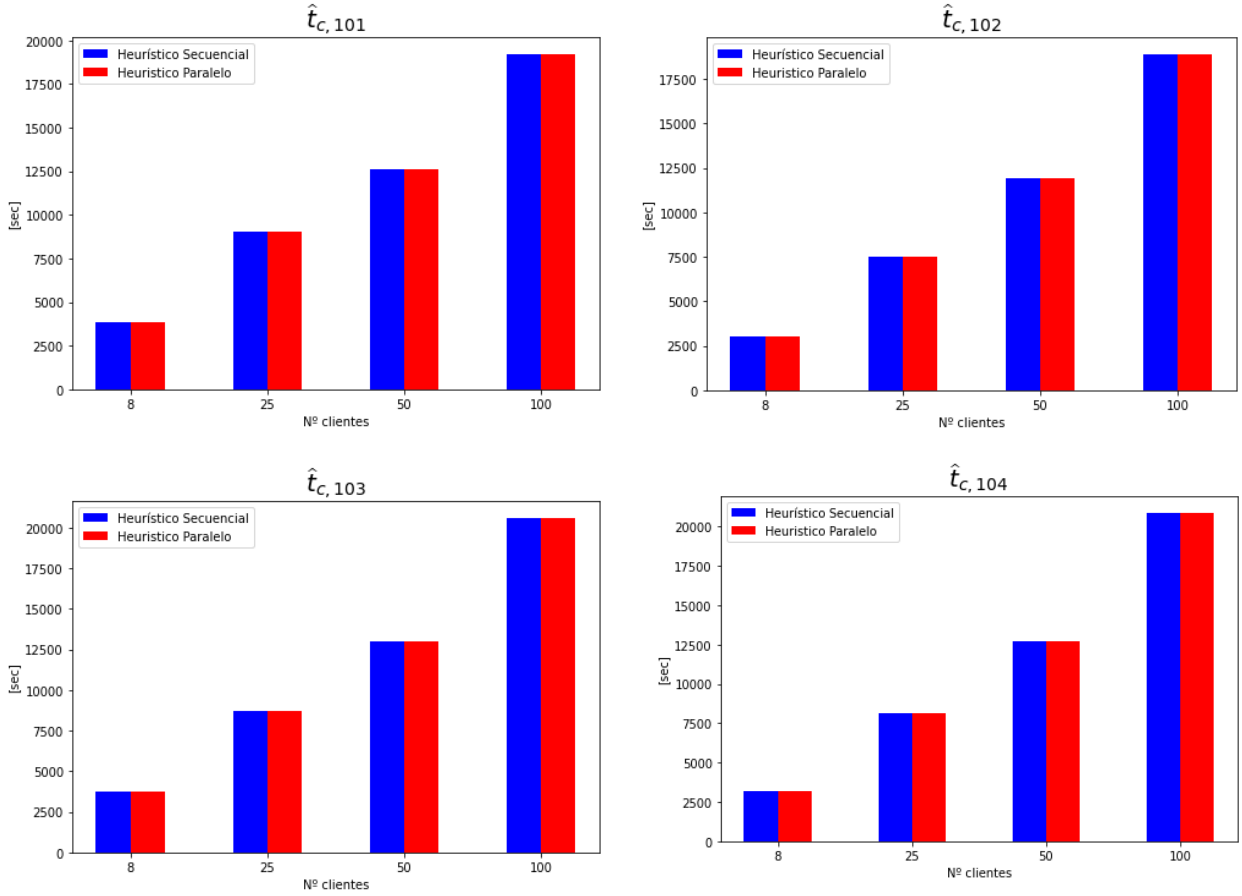

Figura 14. Valor objetivo para distintos números de clientes y tipologías de UAVs

5.2.2 Comparación por tipo de UAV

Una vez finalizado el apartado de resultados en función del número de clientes, se explicará la influencia que ejercen los distintos tipos de UAVs sobre el tiempo total de ejecución y sobre el valor objetivo. Este análisis ya se ha realizado brevemente en el apartado anterior, ya que ejercía una gran influencia. Es por ello que en esta ocasión se representará la información de forma distinta a anteriormente.

En las gráficas que se mostrarán en este apartado no se mostrarán los valores absolutos de tiempo de ejecución y de valor objetivo, si no que se mostrará su variación porcentual respecto a un valor de referencia. Este valor de referencia será el valor de las simulaciones para la tipología 101 de UAV, de la que se ha extraído casi toda la información del apartado anterior. Es por ello, que al tener perfectamente identificado el comportamiento del algoritmo ante ese tipo de UAVs, se mostrarán los resultados del resto respecto a dicho valor.

En la Figura 14 se muestra la comparativa de las distintas tipologías de UAVs por su tiempo total de ejecución en función del número de clientes, siendo para el valor de 8 clientes el único en el que se incluye también el algoritmo exacto. A partir de las gráficas se observa que la tipología 101 es siempre la que conlleva un menor tiempo de ejecución, siendo la más cercana la tipología 102 para pocos clientes (8 y 25), y la tipología 103 para problemas más complejos (100 clientes). En contrapartida, la tipología 104 es la más costosa en cuanto a recursos computacionales, ya que es la que conlleva más tiempo de ejecución. Esto tiene sentido ya que esta tipología (baja velocidad y alta autonomía) es justo la contraria a la tipología 101 (alta velocidad y baja autonomía).

Con todo esto, se puede extraer que cuando se emplean UAVs de alta velocidad (101 y 102), los tiempos de ejecución son bajos para problemas simples, mientras que valores bajos de autonomía (101 y 103) permiten reducir el tiempo de ejecución cuando se trata de un problema más complejo, ya que se reducen el número de posibilidades al descartarse subrutas al no ser alcanzables por los UAVs.

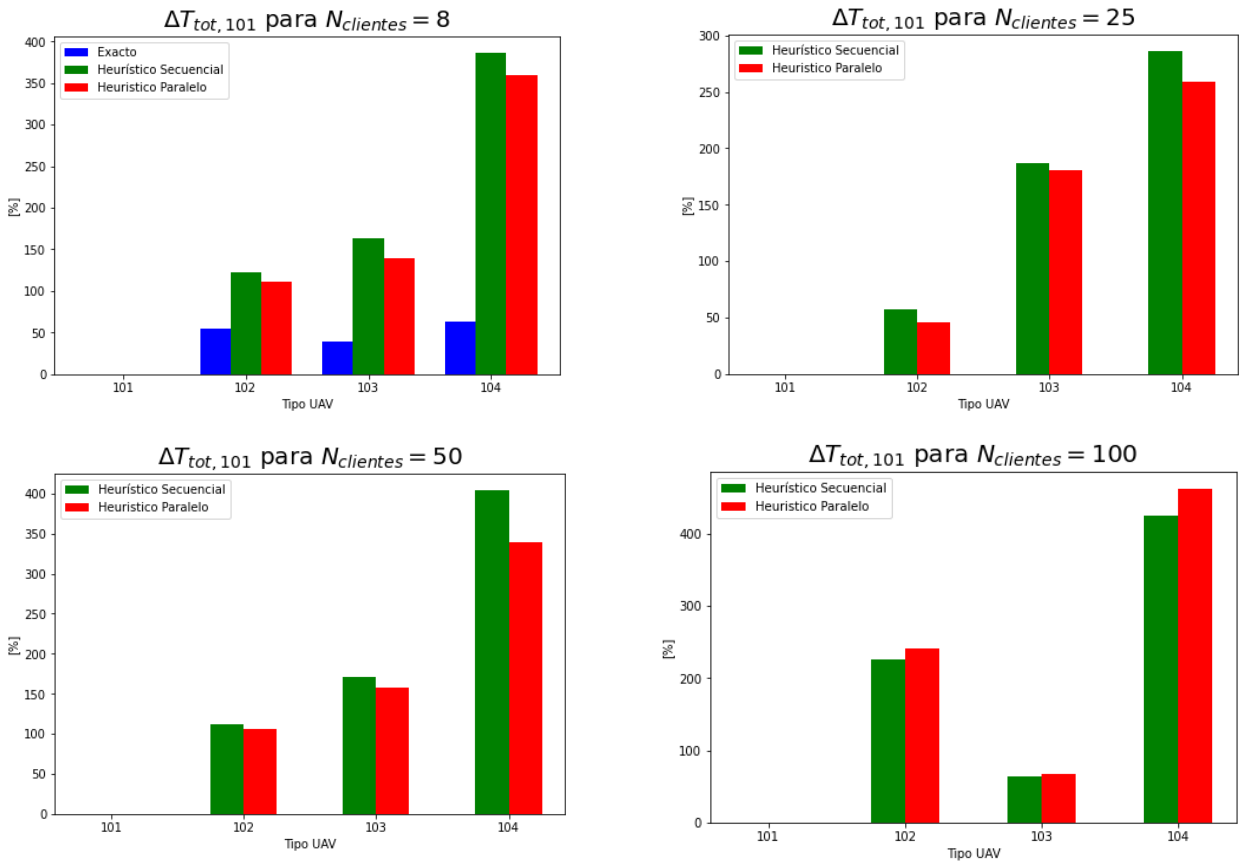

Figura 15. Comparativa por tipología UAV: Variación tiempo total respecto a tipología 101

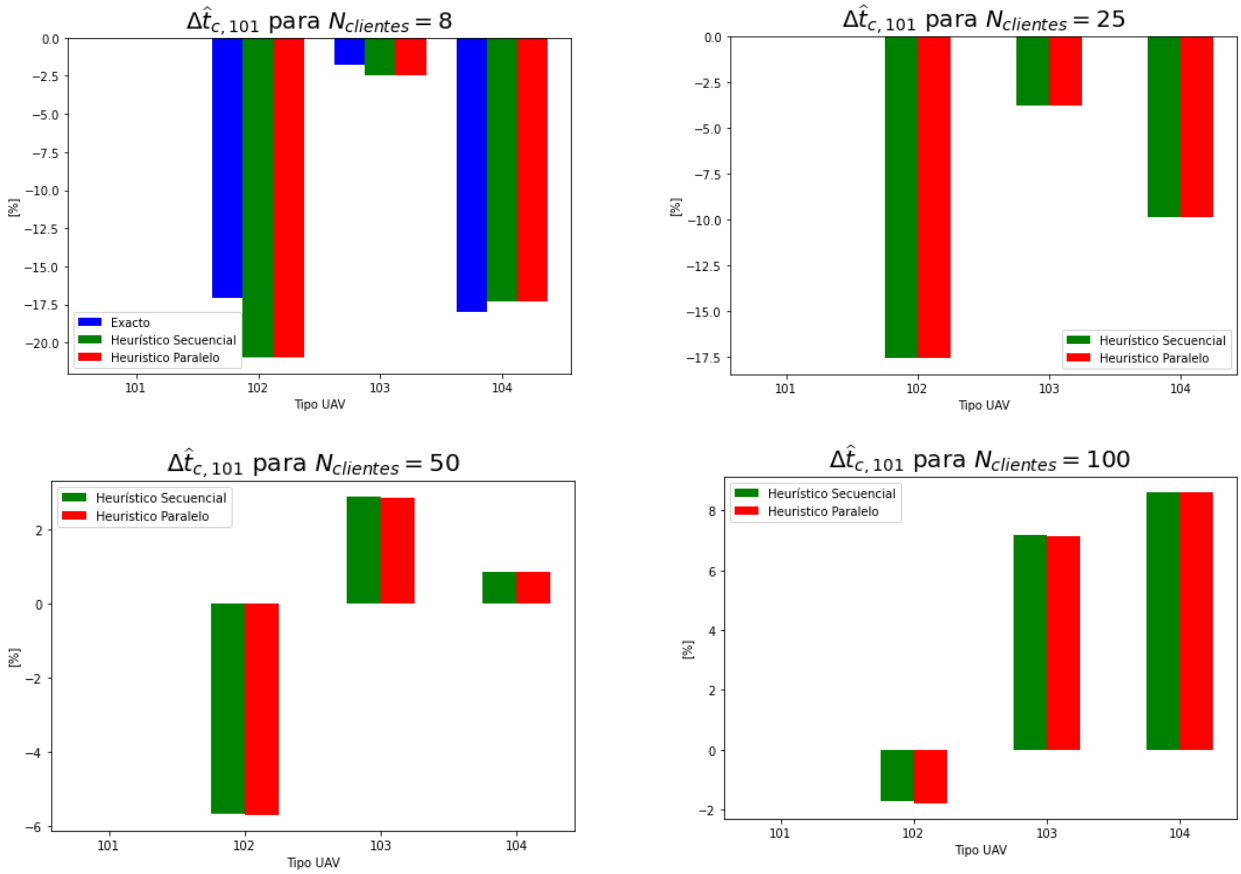

Figura 16Figura 14. Comparativa por tipología UAV: Variación valor objetivo respecto a tipología 101

Por último, en la Figura 15 se muestran las mismas gráficas pero para los valores objetivos. En esta ocasión, tener valores bajos de velocidad (103-104) provoca que aumente el valor objetivo cuando el problema es complejo (50-100 clientes). En cambio, para problemas simples (8-25), todas las tipologías ofrecen valores objetivos inferiores a los de la tipología de referencia (101).

5.2.3 Comparación por número de UAVs

La siguiente comparativa que se va a realizar es el efecto que tiene la variable *número de UAVs* sobre las simulaciones. Para ello, se va a emplear nuevamente la tipología de UAV 101 (alta velocidad, baja autonomía) para comparar los resultados obtenidos.

En primer lugar se muestra la gráfica para 1 UAV y para 4 UAVs con sus tiempos medios en la fase 1 para distintos números de clientes. Se puede observar que los valores tanto para el algoritmo secuencial como para el paralelo son superiores cuando se emplean 4 UAVs que cuando se emplea 1, ya que al tener más UAVs, el problema se vuelve más complejo y aparecen más posibles combinaciones a evaluar.

Por otra parte, y como se ha comentado anteriormente, debido al elevado coste computacional que tiene simular problemas de 100 clientes, se ha decidido simular únicamente estos problemas para 1 iteración, 1 uav y todas las tipologías de UAV, por lo que como se puede observar en la gráfica para 4 UAVs, no existen valores para 100 clientes.

En cuanto a la diferencia entre la heurística secuencial y paralela, se observa que el ahorro de tiempo en la fase 1 es muy superior para 1 UAV que para 4, ya que el tiempo invertido en las fases 2 y 3 es similar al tiempo de la fase 1, como se puede observar comparando las dos gráficas de fase 1 (Figura 16) con las de las fases 2 y 3 (Figura 17). Este efecto de ahorro de tiempo también puede observarse en las gráficas del tiempo total de ejecución, aunque la diferencia entre 1 UAV y 4 es inferior que en la fase1, sigue obteniéndose un ahorro mayor en simulaciones con 1 UAV que con 4.

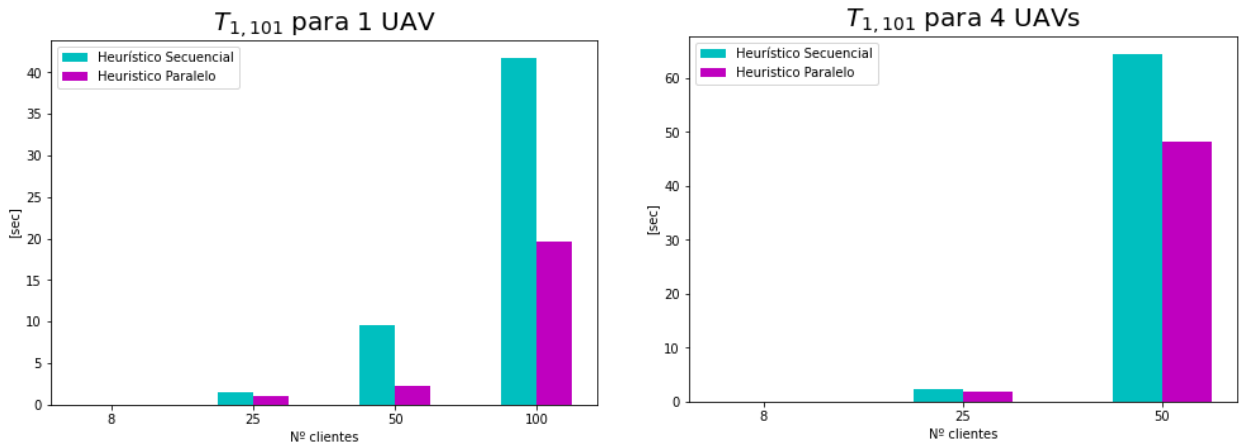

Figura 17Figura 14. Comparativa por número de UAV: Tiempo fase 1

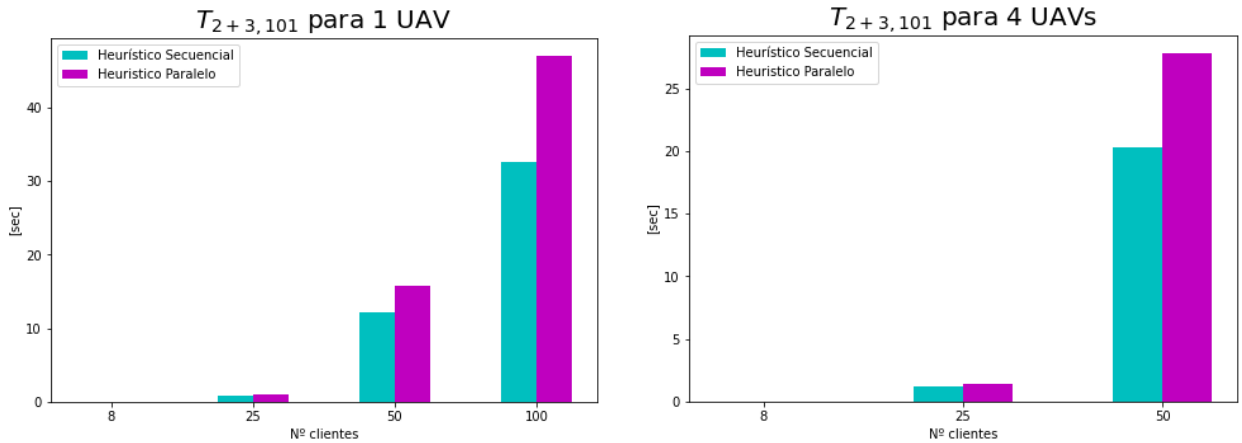

Figura 18. Figura 14Comparativa por número de UAV: Tiempo fases 2+3

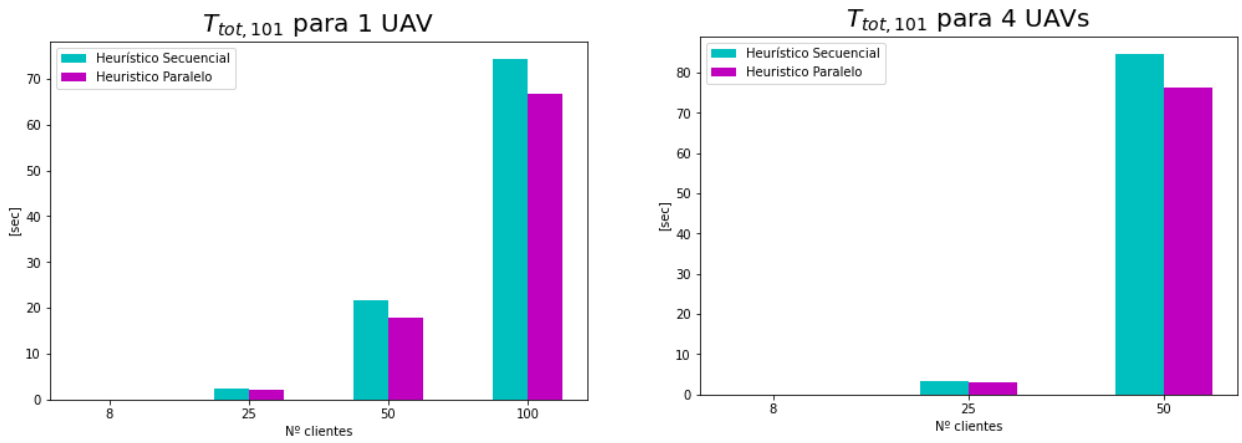

Figura 19. Comparativa por número de UAV: Tiempo total

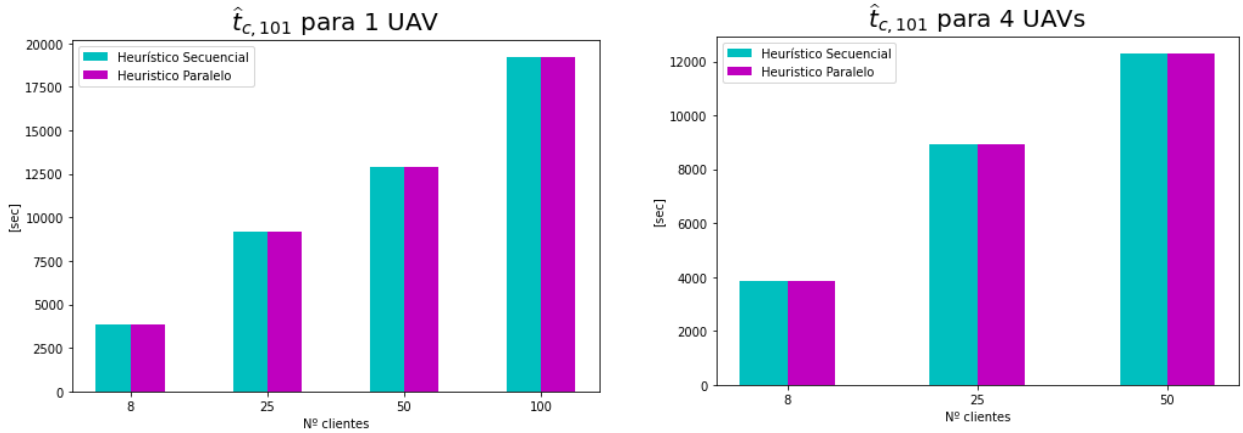

Figura 20. Comparativa por número de UAV: Valor objetivo

Por último, como se lleva viendo en todos los resultados anteriores, cambiar el número de UAVs no afecta directamente en una mejora o un empeoramiento del valor objetivo.

5.2.4 Comparación por número de iteraciones

Finalmente, se va a repetir el mismo estudio que para el número de UAVs, pero en este caso para el número de iteraciones del algoritmo heurístico. En las gráficas siguientes se puede observar que los resultados obtenidos, y por lo tanto, las conclusiones que se pueden extraer son muy similares a las de la comparativa por número de UAVs.

La fase 1 tiene un mayor ahorro para los casos en los que se simula una única iteración, ya que el peso de la fase 1 es muy cercano al peso de las fases 2 y 3, conllevando por tanto una mejora del tiempo total de ejecución. Por otra parte, los valores absolutos de cada una de las gráficas son superiores en el caso de 10 iteraciones que en el caso de 1 iteración. Esto era de esperar, ya que al aumentar el número de iteraciones del algoritmo, se incrementa el tiempo de ejecución, sin embargo, a pesar de que estas iteraciones se aumenten en 10 veces, el tiempo de ejecución no se incrementa de forma proporcional. Esto se debe a que una vez obtenida una ruta subóptima, las nuevas rutas se van comparando continuamente con dicha ruta, y en el momento que una subruta y su rama de posibles soluciones supera en tiempo a dicha ruta subóptima, se pasa al siguiente paso en el bucle, consiguiendo reducir drásticamente el tiempo de ejecución de cada nueva iteración que se calcula.

En contrapartida, cada iteración que se añade incrementa el tiempo total de ejecución, pero sin embargo no ofrece una mejora sobre el valor objetivo significativa, por lo que este fenómeno puede deberse tanto a un efecto positivo como uno negativo. El punto negativo sería que la heurística empleada en este trabajo no es capaz de encontrar las rutas óptimas cuando el problema crece en complejidad, quedándose en mínimos locales, pero siendo incapaz de alcanzar el mínimo global. El punto positivo podría ser que la ruta subóptima calculada con la primera iteración ya se acerca mucho al óptimo global, por lo que aumentar el número de iteraciones genera un beneficio muy pequeño.

Para poder conocer cuál de las dos opciones es la correcta, sería necesario ejecutar el algoritmo exacto para problemas de mayor complejidad, pero como se ha comentado anteriormente, incluso para problemas de complejidad media-baja (25 clientes), simular el algoritmo exacto ha sido inviable con los medios computacionales habilitados para este proyecto.

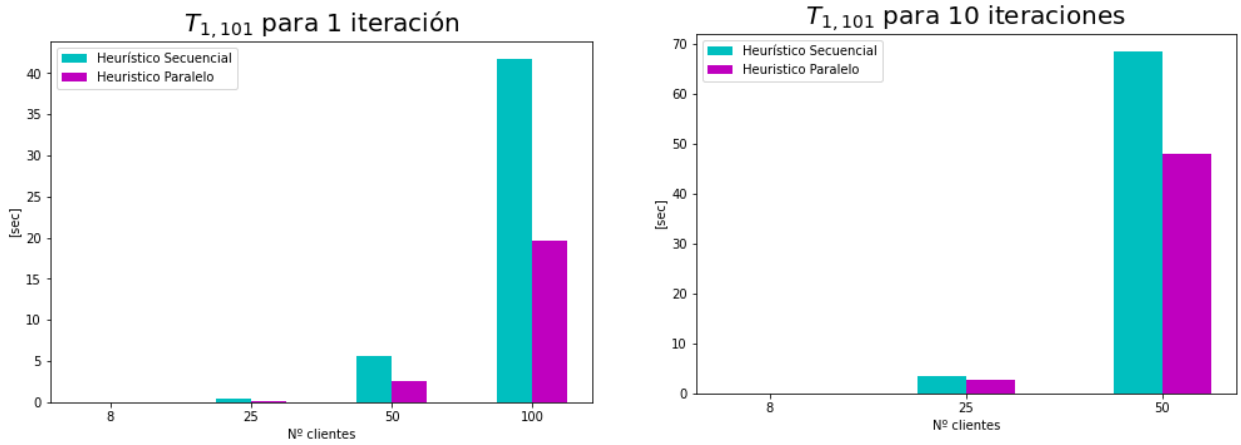

Figura 21. Comparativa por número de iteraciones: Tiempo fase 1

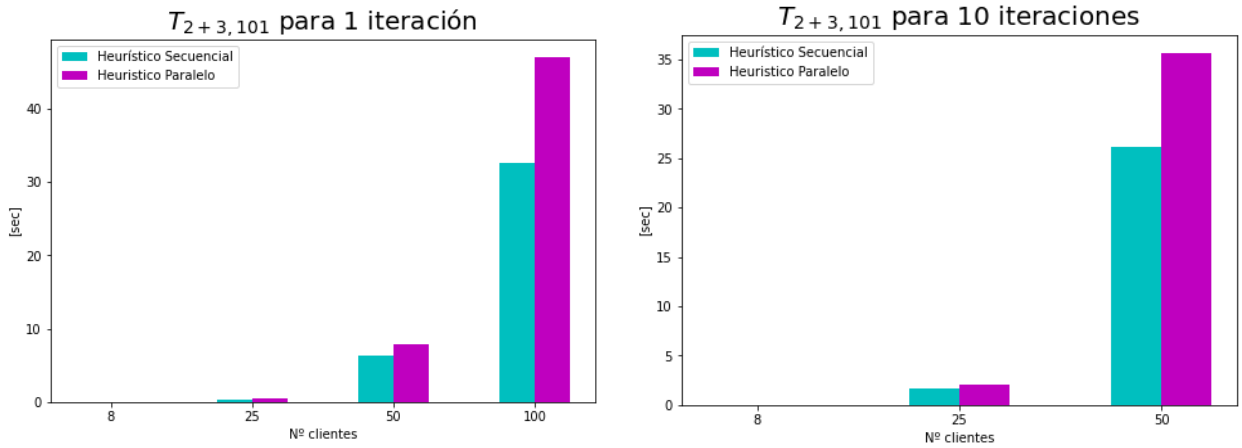

Figura 22. Comparativa por número de iteraciones: Tiempo fases 2+3

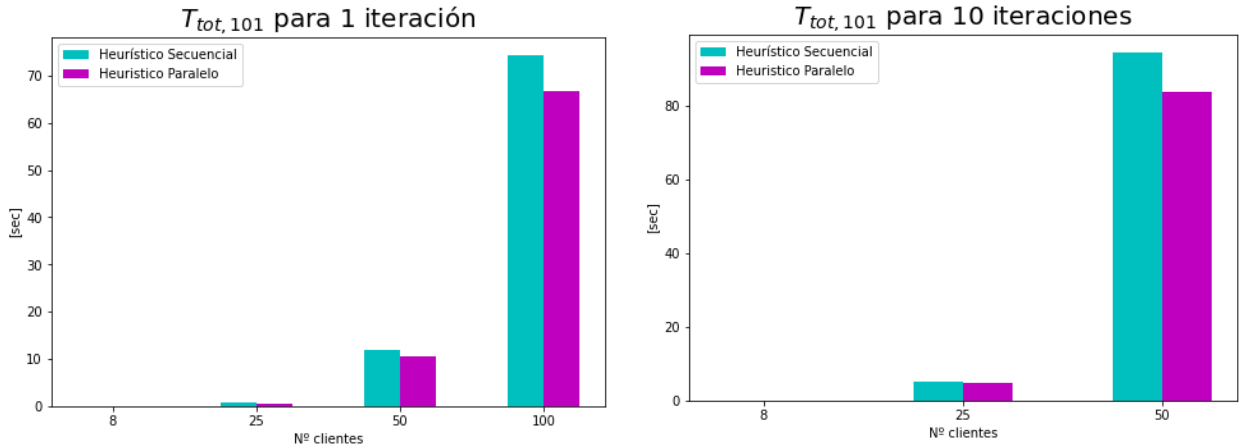

Figura 23. Comparativa por número de iteraciones: Tiempo total

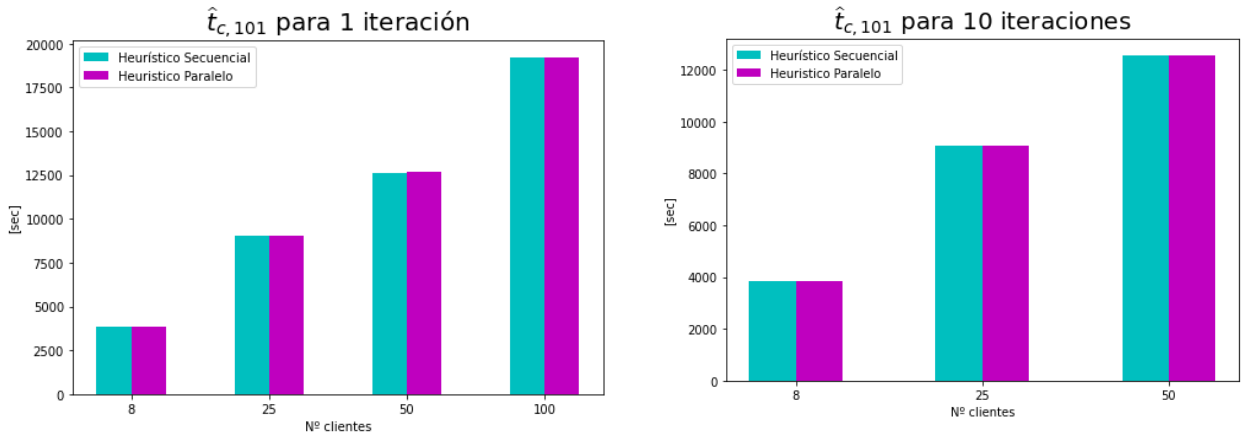

Figura 24. Comparativa por número de iteraciones: Valor objetivo

Por último, se incluyen dos tablas en las que se observa la baja efectividad que se obtiene aumentando el número de iteraciones, ya que mientras que el óptimo local mejora muy levemente, el tiempo de ejecución de la heurística aumenta considerablemente. En la primera tabla se muestran los resultados para simulaciones con 50 clientes y el algoritmo heurístico ejecutado en paralelo.

Tabla 21. Heurística en paralelo: Comparativa para distintas iteraciones para problemas de 50 clientes

ID PROBLEMA	VALOR OBJETIVO [SEC] (1 ITER)	VALOR OBJETIVO [SEC] (10 ITER)	VALOR OBJETIVO [SEC] (100 ITER)
20170606T114000833192	12806,86	12806,86	12806,86
20170606T114145593946	12726,22	12638,89	12638,89
20170606T114330252507	12515,7	12515,7	12515,70
20170606T114511221132	10954,97	10925,85	10925,85
20170606T114654882472	12678,58	12629,29	12629,29

En segundo lugar, se muestran la comparativa para problemas de 100 clientes. En esta ocasión se han incluido también los tiempos de ejecución para que se pueda observar el incremento que existe entre el tiempo empleado en 1 iteración y en 100 iteraciones, mientras que la mejora del mínimo local es muy reducida.

Tabla 22. Heurística en paralelo: Comparativa para distintas iteraciones para problemas de 100 clientes

ID PROBLEMA	TIEMPO EJECUCIÓN PARALELO [SEC] (1 ITER)	VALOR OBJETIVO PARALELO [SEC] (1 ITER)	TIEMPO EJECUCIÓN PARALELO [SEC] (100 ITER)	VALOR OBJETIVO PARALELO [SEC] (100 ITER)
20170606T114000833192	82,64	19335,41	5582,218285	19028,57153
20170606T114145593946	57,99	19144,83	6597,481354	19105,787
20170606T114330252507	58,74	20037,26	5537,596732	19512,16707
20170606T114511221132	74,15	19030,22	8516,834135	18930,75996
20170606T114654882472	60,59	18689,6	6810,592544	18493,22983

6 CONCLUSIONES

En este capítulo se expondrán las conclusiones de este trabajo una vez visto los resultados obtenidos de las simulaciones realizadas. Además, también se realizará una breve recopilación sobre las posibles mejoras y ampliaciones del trabajo desarrollado.

6.1 Conclusiones

Para poder comentar las conclusiones del trabajo, es necesario primero recordar cuál era el objetivo de este trabajo. En este proyecto se buscaba mejorar el uso de recursos computacionales por parte de los algoritmos heurísticos que se aplican a la resolución de problemas de planificación de rutas de reparto. En estas rutas trabajan cooperativamente distintos tipos de vehículos (UAVs y camiones). Para conseguir este objetivo, se decidió tomar un algoritmo heurístico existente y modificarlo para ser capaz de ejecutarlo en paralelo.

Además de simular este algoritmo, se han simulado también el algoritmo heurístico original y un algoritmo exacto en las mismas condiciones computacionales, permitiendo comparar los tres algoritmos en función de distintos criterios y parámetros.

La comparativa más importante, y a partir de la que se obtiene más información, es la comparativa en función del número de clientes del problema. Conforme se aumenta el número de clientes, el problema se vuelve más complejo, y por lo tanto, los tiempos de ejecución crecen exponencialmente. Es tal el efecto de este incremento, que no se han podido realizar simulaciones del algoritmo exacto para problemas con más de 8 clientes. Para esta cantidad de clientes, las diferencias se hacen muy acusadas entre los algoritmos heurísticos y el algoritmo exacto, mientras que los heurísticos tienen un tiempo de ejecución de algunas centésimas de segundo, el algoritmo exacto tiene una duración media de unos 10 segundos.

Para valores bajos de clientes, las diferencias entre los dos algoritmos heurísticos son prácticamente nulas, pero conforme se aumenta la complejidad, se comienza a observar las diferencias. Para problemas cuya fase 1 tiene un coste computacional muy elevado respecto al resto de fases, el ahorro de tiempo es bajo (en torno al **5%** o menos). Sin embargo, para problemas que tienen un tiempo similar para la fase 1 y para las fases 2 y 3, el beneficio se encuentra en torno al **15%.** Esto se debe a que el tiempo de la fase 1 se reduce drásticamente, mientras que el tiempo de las fases 2 y 3 aumenta ligeramente debido a que los valores calculados por la fase 1 son peores en el algoritmo paralelo que en el exacto. Esto surge de que la comunicación entre los hilos tiene retraso, por lo que el hilo de la fase 1 realiza cálculos sin tener la información actualizada, por lo que calcula soluciones más alejadas del óptimo, siendo necesario un mayor esfuerzo computacional por parte de las fases 2 y 3.

Otro aspecto a tener en cuenta es la tipología de los UAVs. A partir de los resultados se desprende que el uso de la ejecución en paralelo no tiene un efecto directo sobre las distintas tipologías, obteniéndose los mismos ahorros de tiempo independientemente de la tipología de los UAVs. En cambio, cuando se comparan entre sí los distintos tipos de UAVs, se observa que los UAVs de alta velocidad tienen tiempos de ejecución bajos cuando los problemas tienen una complejidad baja, mientras que una autonomía baja permite reducir el tiempo de ejecución para problemas con una complejidad alta, ya que se descartan más subrutas desde un inicio, y por lo tanto se reduce la complejidad inicial del problema.

El número de UAVs y el número de iteraciones empleadas en las simulaciones tienen el mismo efecto tanto en la heurística secuencial como en la paralela, por lo que no tienen un gran impacto, ni positivo ni negativo sobre

la ejecución en paralelo. Aunque un aspecto que se puede destacar es que mientras que aumentar el número de UAVs permite reducir el valor objetivo a costa de aumentar el tiempo de ejecución, aumentar el número de iteraciones conlleva un aumenta del tiempo de ejecución pero no conlleva una mejora del valor objetivo.

Como se ha comentado en el capítulo 5, esto puede deberse tanto que la heurística implementada no consiga acercarse al óptimo global incluso cuando se aumenta el número de iteraciones, o por el contrario, que el valor objetivo calculado por la heurística se encuentre muy cercano al valor óptimo, y por lo tanto, por mucho que se aumente el número de iteraciones, este valor objetivo no variará en grandes cantidades. Para conocer cuál de estas dos posibilidades es la más cercana a la realidad, sería necesario calcular mediante algoritmos exactos soluciones a los problemas de alta complejidad (100 clientes), por lo que sería necesario obtener mejores recursos computacionales para conseguir dicho objetivo.

Por último, y aunque se trate de un efecto secundario, el valor objetivo ha sido evaluado en todas las simulaciones y representado en los resultados del capítulo 5. Esto permite concluir que el uso del algoritmo paralelo no tiene ningún efecto sobre dicho valor objetivo, o en el peor de los casos, empeora los valores en menos de un **1%.** Esto significa que a pesar de no ser un objetivo principal, el valor objetivo no se ha visto distorsionado excesivamente por el algoritmo paralelo.

6.2 Mejoras y futuros trabajos

Finalmente, algunas de las mejoras o posibles ampliaciones que se pueden incluir son:

- Estudiar problemas de 100 clientes o más para todos los parámetros descritos en este trabajo, ya que únicamente han podido ser evaluados los problemas de 100 clientes para 1 UAV y 1 iteración debido al coste computacional.
- Ser capaz de conocer la solución óptima para problemas de complejidad media-alta (50-100 clientes) para conocer la cercanía de los valores objetivos de la heurística con los valores óptimos. Esto se podría hacer tanto mediante el uso de algoritmos exactos, como mediante otro tipo de heurísticas o incluso una combinación de varias heurísticas.
- Emplear otra estrategia para implementar el algoritmo de ejecución en paralelo distinto al empleado en este trabajo, ya sea empleando otra heurística distinta que pueda ser más favorable a este tipo de implementaciones, o empleando esta misma heurística pero partiendo desde otro punto de inicio.
- En este trabajo, a pesar de ser un objetivo secundario, siempre se ha tratado de que el efecto del algoritmo paralelo sea mínimo sobre el valor objetivo, por lo que otra posible mejora en cuanto al tiempo de ejecución es emplear otros mecanismos distintos a los empleados en este trabajo, que descuiden en mayor medida el valor objetivo a cambio de reducir más el tiempo objetivo.

REFERENCIAS

[1] M Roca-Riu, M Menendez, «Logistic deliveries with drones: State of the art of practice and research,» *19th Swiss Transport Research Conference,* 2019.

[2] A. Adwibowo, «Assessments of heavy lift UAV quadcopter drone to support COVID 19 vaccine cold chain delivery for indigenous people in remote areas in South East Asia,» 2021.

[3] B Rabta, C Wankmüller, G Reiner, «A drone fleet model for last-mile distribution in disaster relief operations.,» *International Journal of Disaster Risk Reduction,* vol. 28, pp. 107-112, 2018.

[4] A. S. C. Weinstein, «UAV scheduling via the vehicle routing problem,» *AIAA Infotech@ Aerospace 2007 Conference and Exhibit,* 2007.

[5] CC Murray, AG Chu, «The flying sidekick traveling salesman problem: Optimization of drone-assisted parcel delivery,» *Transportation Research Part C: Emerging Technologies,* vol. 54, pp. 86-109, 2015.

[6] Thibbotuwawa, Amila, et al, «Planning deliveries with UAV routing under weather forecast and energy consumption constraints,» *IFAC-PapersOnLine,,* vol. 52, nº 13, pp. 820-825, 2019.

[7] Chase C. Murray and Ritwik Raj, «The multiple flying sidekicks traveling salesman,» *ransportation Research Part C:,* vol. 110, pp. 368-398, 2020.

[8] A. M. Ham, «Integrated scheduling of m-truck, m-drone, and m-depot constrained by time-window, drop-pickup, and m-visit using constraint programming,» *Transportation Research Part C: Emerging Technologies,* vol. 91, pp. 1-14, 2018.

[9] S Poikonen, B Golden, «Multi-visit drone routing problem,» *Computers & Operations Research,* vol. 113, p. 104802, 2020.

[10] G. B. Dantzig, J. H. Ramser, «The truck dispatching problem. Management science,» vol. 6, nº 1, pp. 80-91, 1959.

[11] M. Dorigo, L. M. Gambardella, «Ant Colonies for the Traveling Salesman Problem,» *Biosystems,* vol. 43, nº 2, pp. 73-81, 1997.