Trabajo Fin de Máster
Organización Industrial y Gestión de Empresas I

Planificación de una flota homogénea de drones para cobertura audiovisual de prueba deportiva

Autor: David Sánchez Wells
Tutor Externo: Jose Luis Andrade Pineda
Co-Tutor: Pedro Luis González Rodríguez

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

Sevilla, 2018

Trabajo Fin de Máster
Organización Industrial y Gestión de Empresas I

Planificación de una flota homogénea de drones para cobertura audiovisual de prueba deportiva

Autor:
David Sánchez Wells

Tutor Externo:
Jose Luis Andrade Pineda
Gestor de Proyectos de Transferencia I+D+i en Grupo Robótica, Visión y Control

Co-Tutor:
Pedro Luis González Rodríguez
Profesor Titular de Universidad

Dep. de Organización Industrial y Gestión de Empresas I
Escuela Técnica Superior de Ingeniería
Universidad de Sevilla
Sevilla, 2018

Trabajo Fin de Máster: Planificación de una flota homogénea de drones para cobertura audiovisual de prueba deportiva

Autor: David Sánchez Wells

Tutor Externo: Jose Luis Andrade Pineda

Co-Tutor: Pedro Luis González Rodríguez

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

Presidente:

Vocales:

Secretario:

Acuerdan otorgarle la calificación de:

Sevilla, 2018

El Secretario del Tribunal

A mi familia

A mis maestros

Agradecimientos

Quiero agradecer a Jose Luis Andrade Pineda su sólido apoyo y constancia al dedicar su tiempo a este proyecto, que nunca hubiera existido sin su dirección, y al profesor Pedro Luis González Rodríguez no sólo el habernos puesto en contacto y el estar ahí cuando más lo necesitaba, sino el provocar, hace muchos años, que descubriese mi vocación por la Organización Industrial.

Me gustaría también hacer mención especial a mi madre, Laura, y a mi padre, Jose, ya que sin su esfuerzo nunca habría llegado ni siquiera a soñar con estar realizando este trabajo, habiendo sido figuras clave en momentos muy difíciles en los que todo parecía perdido. Gracias, de todo corazón, por creer en mí.

Debo dedicar unas palabras a la familia que se elige, a mis amigos y personas más cercanas, y en concreto a Jose y Alejandro, por todo el ánimo que han sabido darme y ser el respiro que muchas veces necesitaba. Y, por supuesto, mi más sincero agradecimiento a Ángela, por mucho más de lo que podría escribirse.

Hay personas a las que he dejado de nombrar que también han participado y sido de ayuda, activa o pasivamente, en este trabajo. Ellos saben quiénes son y esta despedida va por ellos.

David Sánchez Wells

Sevilla, 2018

Resumen

Una de las aplicaciones más extendidas en el uso de UAVs (unmanned aerial vehicles) es el seguimiento de objetivos, ya que aquellos permiten tomar imágenes desde posiciones remotas o inalcanzables mediante otros métodos.

En este proyecto, se desarrolla una aplicación parametrizable de seguimiento automatizado con un sistema de UAVs a un conjunto de objetivos que poseen una trayectoria estimada definida por una serie de puntos y tiempos con incertidumbre.

Esta aplicación, mediante el uso de una heurística de asignación secuencial, proporciona dos resultados: una asignación rápida e individualizada de cada UAV a una parte definida de la ruta de cada objetivo y una primera identificación del cruce de trayectorias planificadas como potencial dificultad en llevar a la realidad los planes.

Abstract

One of the most widespread applications in UAVs (unmanned aerial vehicles) usage is objectives tracking, due to their capability of taking images from remote or unreachable positions through other methods.

This project develops a UAV system customizable and automated objective tracking application. The objectives' trajectories are defined by sets of points and timings with uncertainty.

This application, through the usage of a sequential task assignment heuristic, provides two results: a fast and individualized assignment of each UAV to a specific part of each objective's route and a first identification of the planned trajectories crossings as a potential difficulty when it comes to real execution.

Índice

ÍNDICE DE TABLAS

ÍNDICE DE FIGURAS

1 INTRODUCCIÓN Y OBJETIVO DEL PROYECTO

Todo es medida, número y peso.
Sólo es cuestión de exactidud.
- Sabiduría 11:20 (mod. anónima) -

El grado de madurez de la tecnología de UAV, con autonomías considerables en base a electrónicas de bajo coste, hace asequible su utilización en una gran variedad de escenarios civiles. El tipo de UAV (o, como se denomina de ahora en adelante, dron) más extendido en el ámbito civil es el llamado multirrotor que, en base a motorización eléctrica y hélices de fibra de carbono, ofrece hoy día el mejor balance entre maniobrabilidad y autonomía para muchas aplicaciones, incluso extendidas sobre grandes áreas (vigilancia aduanera, monitorización de infraestructuras, etcétera).

En el caso particular de las aplicaciones audiovisuales objeto de este proyecto, típicamente se requiere de una operativa flexible en cuanto a maniobrabilidad, estabilidad de vuelo y facilidad de uso; todas estas capacidades se dan en las soluciones comerciales multirrotor (véase DJI Phantom 4, https://www.dji.com/es/phantom-4-pro) en uso en las flotas de este tipo de empresas. En esta área, se asume que las plataformas en uso tienen niveles de autonomía considerables y equipan sensores de abordo consistentes en cámaras profesionales montadas en unidades gimbal que geo estabilizan y aíslan de las vibraciones de los motores eléctricos y que se usan en la cobertura de puntos/eventos en un área extensa, lo que se hace en modo RPA (remotely piloted aircraft). En este modo, un piloto está al cargo de la supervisión del desempeño de misiones automáticas y/o el manejo en manual del dron. Es más, en la cobertura de eventos, típicamente se necesita una variedad de drones en uso simultáneo para capturar imágenes que un responsable de realización haya de tener a su disposición para cubrir simultáneamente lo que sucede en y/o desde una variedad de puntos. En tal situación, antes de la realización misma de las misiones de vuelo, debe haber una planificación detallada de la coordinación entre los distintos drones de acuerdo a determinados criterios.

En este proyecto, se estudia uno de estos escenarios de planificación. En concreto, se plantea resolver lo siguiente:

"Una corporación organizadora quiere contar con la grabación en video del transcurso de una prueba deportiva, con seguimiento individual para cada uno de los participantes. Para ello utilizará una flota de drones equipados con cámaras de video profesionales, que volando a una misma altura tomen las imágenes de estos participantes en una serie de puntos. El objetivo concreto es diseñar una herramienta de decisión capaz de organizar la flota de drones de manera que se proporcione la máxima cobertura al evento, se minimicen las posibles colisiones entre drones (cuando haya cruces existe ese riesgo) y la distancia total recorrida."

En lo que sigue se asume que un sistema de drones homogéneo es el encargado de servir a un conjunto de objetivos o targets (cada participante) a lo largo de una determinada ruta prevista. La ruta se maneja como un input al problema de la planificación, definiéndose mediante un conjunto de puntos (waypoints) y de tiempos previstos de llegada a los mismos. Además de la suposición de unas capacidades a cada multirrotor en la flota,

se supone la existencia de una infraestructura de apoyo consistente en una serie de estaciones de recarga en unas posiciones conocidas.

Se comienza a abordar el problema con una revisión del estado del arte, que se presenta en el Capítulo 2. El problema de planificación planteado forma parte del tipo VRP, con múltiples vehículos y ventanas temporales. Sin embargo, hay especificidades del problema real que también lo relacionan con problemas de tipo JSP (Job Scheduling Problem), con la existencia de precedencias, recursos limitados, tiempos de inactividad, etcétera.

En el Capítulo 3 se expondrán los requerimientos y particularidades del escenario real de optimización. Aunque se podría plantear la formulación exacta de dicho problema (Song, et al., 2014), para los tamaños considerados en este TFM su solución con un software MILP (Mixed-Integer Linear Programming) no es factible. El objetivo de este TFM es la proposición de una heurística que permita dotar al planificador de una herramienta funcional para obtener buenas soluciones con un tiempo de cómputo limitado.

De estos requerimientos, se expone en el Capítulo 4 el diseño conceptual de la heurística que tienda a cubrir los puntos e identifique el grado de efectividad de los planes propuestos en su operativa en campo. En el Capítulo 5, se presenta la implementación en Python de dicho concepto en forma de metaheurística.

Finalmente, tal y como se presenta en el Capítulo 6, la experimentación realizada indica la utilidad de la herramienta codificada para servir de apoyo a las decisiones de planificación de la corporación audiovisual para la cobertura del evento deportivo.

2 REVISIÓN DE LA LITERATURA

La utilización de robots aéreos (drones) en el ámbito civil ha experimentado en la última década un gran auge, estando principalmente enfocados a la realización de tareas de cierto grado de especialización o que aíslan de ciertos riesgos, ya que gracias a ellos pueden ejecutarse ahora de manera más eficiente. La especialización está directamente relacionada con las capacidades de los sensores que hoy día se pueden equipar a bordo de plataformas dron de coste medio.

En muchos casos, ha surgido toda una nueva problemática derivada de la mejor operación de uno o varios drones en empresas industriales y de servicios. La fácil reconfiguración de los drones a emplear para un caso real concreto, suele ser una estrategia para disponer de una adaptación dinámica a las diferentes necesidades que temporalmente puedan surgir. La configuración dinámica de los recursos del sistema debería permitir, por ejemplo, que una vez una misión concreta se ha realizado, el equipamiento del dron en cuestión se pueda recuperar como para conformar un nuevo dron adecuado para el desarrollo de una misión a ejecutar más tarde. De este modo, la habilidad de reconfiguración de las capacidades instaladas en los UAVs listos para volar dotaría de flexibilidad para poder manejar la diversa naturaleza de los servicios a prestar en los distintos objetivos (posición) y en distintos instantes (tiempo).

Lo anterior, tiene esencialmente que ver con la gestión de los sensores de abordo, que son los elementos más costosos en la plataforma dron. La intercambiabilidad de los sensores con que se equipan los UAVs en una flota de drones ha sido estudiada en (Mufalli, et al., 2012), que aborda simultáneamente la mejor asignación del conjunto de sensores de que se dispone a cada UAV y el mejor enrutamiento de los UAVs así configurados. Básicamente se identificarían distintos objetivos a servir, en un área concreta y unos requerimientos de las tareas a realizar en cada objetivo (posición). En escenarios reales es común que existan diferentes necesidades y diferentes restricciones temporales y de maniobrabilidad como consecuencia de que haya uno u otro equipamiento abordo. Entonces, es prioritario generar un plan de misión que pueda cumplir con todas las restricciones y que optimice un determinado indicador de performance (por ejemplo, minimizar el tiempo de ejecución). Si además se da distinto peso/valor al servicio de los distintos objetivos, surge un problema llamado Team Orienteering Problem (TOP) que es muy combinatorio y que, para garantía de operatividad en condiciones reales, debería de ejecutarse centralizado en un equipo GCS en tierra, mejor que como un algoritmo distribuido (Kaddouh, et al., 2016).

En otras investigaciones, los drones se asumen equipados con capacidad de cómputo y comunicación como para ejecutar de forma distribuida su adaptación a las condiciones dinámicas del entorno (por ejemplo, otros drones en vuelo). Aunque se gana en reactividad individual de cada dron, la robustez de este esquema distribuido necesita de una orquestación/negociación de la visión del escenario que se percibe en cada uno de ellos. Se puede pensar en negociaciones entre los diferentes drones para ganar en la robustez de la aplicabilidad de los planes individualizados que vayan surgiendo de la decisión autónoma de cada dron. En (Oh, et al., 2014) se utiliza una negociación basada la ejecución de una subasta a la que cada dron acude buscando su beneficio propio, y en la aplicación de un posterior procedimiento de consenso para dejar todas las tareas a hacer sobre cada objetivo asignadas (qué dron la hará, y en qué tiempo). El autor afirma que, dado que la búsqueda del beneficio individual de cada dron decidiendo de forma autónoma su apetencia por una u otra tarea se conjuga bien con un objetivo común (makespan), del proceso resultan soluciones útiles para el conjunto.

Sin embargo, en la mayoría de casos en que existe una estructura clara del problema a resolver, se considera mucho más lógico ejecutar un procedimiento asignación de tareas centralizado. La complejidad del problema y las necesidades de cómputo se circunscriben a un ordenador en tierra que típicamente se equipará con facilidad para ejecutar de forma eficiente la planificación de tareas. Un esquema reactivo que se adapta bien a entornos dinámicos como el estudiado se reporta en (Kaddouh, et al., 2016), donde se considera secuenciar tareas con la doble factibilidad del dron (y su equipamiento) con las tareas a realizar en los puntos de interés y de la tarea en si con las precedencias relativas al proceso tecnológico en la línea de producción. Cuestiones de refinamiento como Planificación de Trayectorias, se dejan a un nivel inferior (estas sí apropiadas para ejecutarse en un PC de abordo), en el supuesto de que los drones irán equipados con la sensorización y con capacidad de cómputo para ejecutar algoritmos de optimización de trayectorias (por ejemplo, generación de waypoints y sus timestamps, donde el TAP simplemente definió un "desplazamiento").

Con posterioridad, los mismos autores han aplicado dichas ideas a orquestar el servicio con múltiples puntos de reemplazo en ubicaciones arbitrarias y con drones que comienzan con niveles de batería diferentes (Song, et al., 2016). En realidad, tales ajustes provienen de la consideración de un esquema de planificación de la flota de tipo horizonte rodante que está constantemente a la espera de la llegada de nuevas tareas que aconsejen la reprogramación del sistema de drones para el mejor performance en el servicio prestado. Las tareas tienen unas ventanas temporales para su realización, igual que en el trabajo de (Guerriero, et al., 2014), enfocado a garantizar la cobertura espacio-temporal de un conjunto de puntos de interés dispersos en un área en relación a la cobertura audiovisual de eventos deportivos.

En cuanto a cómo conformar un sistema compuesto por una flota de drones a las necesidades reales de la aplicación concreta y a la gestión dinámica de recursos en el marco del sistema diseñado, referimos al trabajo de (Murray & Karwan, 2010). En dicho artículo se trata la problemática de coordinar drones en misiones militares denominadas como ISR (Inteligencia, Vigilancia y Reconocimiento), donde al igual que en este proyecto los drones equipan cámaras y siguen a unos objetivos determinados. El tipo de problema que implican estas operaciones se conoce como DRM (Gestión Dinámica de Recursos), donde se busca la reasignación de tareas en base a cambios en determinadas condiciones, es decir, en reacción a eventos específicos, de tal manera que se maximice la efectividad total de la misión y se minimicen los cambios respecto al plan inicial. Para Murray y Karwan, las tareas a realizar se clasifican por tipo, recursos necesarios y escenarios de operación, y se caracterizan mediante priorización, duración de servicio, limitaciones en el mínimo y/o máximo número de recursos que pueden realizarlas simultáneamente y otras características temporales, tales como relaciones de precedencia y/o ventanas temporales preferentes. Por último, pero no por ello menos importante, Murray y Karwan mencionan el hecho de que dadas las circunstancias en que se opera un problema DRM, no se cuenta con un lapso de tiempo dilatado en el que calcular la solución óptima, por lo que el modelo utilizado para resolver el problema debe proveer de soluciones rápidas, aunque sean subóptimas.

Para llegar a elaborar el modelo de programación lineal que diese solución a esta problemática, dado que Murray y Karwan enfocan su artículo al desarrollo de un marco de trabajo para el modelado general de este tipo de problemas, los autores tuvieron que analizar tres categorías de problemas con ciertas similitudes respecto al DRM: los TSP (Problema del Viajante), los VRP (Problema de Enrutamiento de Vehículos) y los Problemas de Enrutado de Drones. Como resultado, los autores establecieron que, para caracterizar completamente un problema DRM de drones, es necesario definir cada parte del mismo a partir de un esquema determinado. A través de dicho esquema, se formaliza gran parte de los distintos tipos de problema a los que es lícito denominar como de Enrutado de Drones, con una complejidad matemática y programal suficiente como para permitirnos definir modelos de programación lineal en base a unas directrices.

De vuelta a las necesidades de la aplicación en estudio en este TFM, se necesita el desarrollo de una herramienta de planificación capaz de generar planes de gestión de la flota de múltiples drones para la cobertura audiovisual, que resulten en corto tiempo. Entendemos que una vez se enfrenta nuestro problema de Scheduling, ya se realizaron decisiones estratégicas previas. (Suzuki, et al., 2011) identifican que el dimensionado de la flota y del emplazamiento apropiado para estaciones de reemplazo de baterías son ejemplos de problema de la selección de recursos. Luego, se abordan problemas operativos del tipo de planificar las misiones de los UAVs y su marcha a las estaciones de reemplazo. Por ejemplo, el reparto (asignación) de tareas para la cooperación de los múltiples drones. Éste es un problema bastante complejo cuando se trata de drones heterogéneos, para los que debe manejar diversidad tanto en capacidades como en dinámicas de vuelo. Una primera aproximación a este problema se puede hacer ignorando las ecuaciones cinemáticas del vuelo, y surgen problemas del tipo Task Assignment Problem (TAP).

En este sentido, surgen problemas típicos de la Investigación de Operaciones, como el problema del diseño de rutas para los UAVs manejado como un CVRP con consideración de limitación de autonomía (Olivares, et al., 2015) o de las capacidades/equipamiento del UAV y de los niveles de servicio a los waypoints a servir (Shetty, et al., 2008). En el diseño, tiene mucha relevancia si se decide establecer un único punto abastecimiento/retorno (depot en jerga VRP) como en (Nigam, et al., 2012) o si existe una variedad de ellos, véase (Kim, et al., 2013).

Se podría asumir que la flota de drones que va a servir la misión puede ser heterogénea, caracterizándose cada dron o tipo de dron por su autonomía (o capacidad de fuel), por su carga útil y por un porcentaje de aptitud en base a la realización de una tarea determinada. Cabe destacar que esto supone un avance respecto a estudios anteriores, en los que sólo se consideraba si el recurso era o no capaz de realizar la tarea de manera binaria, no contemplándose la posibilidad de que existiesen distintos grados de aptitud. También se planteaba la existencia de múltiples bases, que eran capaces de servir a determinados tipos de drones.

En (Song, et al., 2014) se usa este enfoque para la conformación de un servicio persistente de un sistema de drones, con la definición del número y ubicación de unas estaciones de reemplazo de baterías para garantizar la continuidad del servicio prestado por una flota de drones en un área concreta y sobre unos targets concretos. Se propone un enfoque Receding Horizon Task Assignment para garantizar un plan de las misiones y, de forma implícita, se planifica cómo unos drones se van dando el relevo a otros (cada dron hace múltiples viajes o misiones) para, en conjunción con las estaciones de reemplazo, garantizar que los targets reciben el servicio comprometido.

De las varias formulaciones del problema de enrutado de vehículos con múltiples viajes (MTVRP) que se listan en (Cattaruzza, et al., 2018), en (Song, et al., 2014) se decantan por usar variables de asignación de cuatro índices. Se trata de una extensión de la formulación VRP clásica de tres índices: i (nodo de salida), j (nodo de destino) y k (vehículo asignado); en que un cuarto índice r representa el ordinal del viaje realizado por el vehículo en cuestión. Esto proporciona un enfoque temporal más específico al problema de configuración persistente del servicio, al facilitar la identificación no sólo del dron que sirve una conexión determinada sino también de la misión en la que se encaja dicho vuelo.

Por último, como en todo proceso de planificación, hay una cierta incertidumbre respecto a cómo podrá llevarse a la práctica dicho plan. En la literatura de aviación comercial existen aproximaciones para dado un plan (slots temporales dados, y ruta) analizar la forma de evitar que vuelos de aviones que cruzan sus trayectorias en determinado punto y con una separación temporal pequeña corran el riesgo de colisionar (Chaimatanan, et al., 2015). En tal caso, se realiza un análisis 4D (espacio + tiempo) de trayectorias, con consideración de restricciones tanto a nivel legal como tecnológico. El planteamiento del método y la formulación propuesta en esta área de conocimiento podrían adaptarse perfectamente al caso de los drones, resultando los conceptos de flight level e incertidumbres especialmente interesantes al caso de uso considerado en este TFM.

Como se planteó en el capítulo anterior, en este TFM asumimos que se necesita manejar múltiples misiones que ocurren al mismo tiempo. Es decir, que hay que controlar la operación persistente de una flota de drones, posiblemente asistidas por una serie de estaciones de servicio distribuidas. La necesaria persistencia de los servicios prestados por la flota está empezando a considerarse en la literatura, con la definición de las trayectorias espacio-temporales de cada dron (Song, et al., 2016). En este trabajo los autores tratan la misión de cada UAV como una sucesión de Jobs o tareas monolíticas a asignar. Al igual que sucede en problemas de Scheduling clásicos, las tareas tienen unas ventanas temporales en que se supone han de realizarse bajo la hipótesis implícita de que en la medida en que se cumplan, las tareas son servidas o no. Dada la simultaneidad de los eventos, el tiempo puede considerarse un recurso costoso con uso intensivo, por lo que debe destacarse que las ventajas en cuanto a flexibilidad que el uso de drones otorga es compatible con mantener la utilización del tiempo lo más alta posible.

En adelante, este documento TFM se concentra en presentar el trabajo realizado para el desarrollo de un planificador eficiente de la cobertura audiovisual a prueba deportiva con una flota de drones.

3 DESCRIPCIÓN DEL PROBLEMA Y REQUISITOS

En este proyecto, un sistema de UAVs homogéneo es el encargado de servir a un conjunto de objetivos (targets) a lo largo de su trayectoria, la cual es definida por un set de puntos (waypoints) y de tiempos. Estos cuentan con el apoyo de una serie de estaciones de recarga en unas posiciones determinadas, en las que son capaces de realizar un cambio de baterías que renueve su autonomía.

Como en cualquier dominio de aplicación, la aplicación de UAVs debe prestar atención a:

- Las tareas a realizar por el UAV
- El entorno en las que se desenvuelve el UAV
- El sistema con que se opera la flota UAV

Tal y como se reporta en (Khosiawan, et al., 2018), un análisis cruzado de estos tres factores lleva a seis cuestiones a detallar para formalizar los requisitos del sistema UAV en cuestión:

1. La naturaleza de las tareas: su tipología y atributos, las acciones prohibidas (por ejemplo, exclusión de determinadas zonas para aterrizaje) y la fijación de restricciones y objetivo perseguido.
2. La clasificación de las distintas áreas (y puesto que el UAV se mueve en 3D, volúmenes) en los que el UAV se desenvuelve.
3. La configuración del sistema de drones en su conjunto: el número de elementos de la flota, el posicionamiento, capacidad y operativa en los centros de recambio de baterías.
4. ¿Existe algún conflicto entre 1 y 2? Es decir, explicítense limitaciones al desempeño de las tareas previstas según 1 derivadas de la naturaleza o del método de gestión identificado en 2.
5. ¿Existe algún conflicto entre 2 y 3? Es decir, ¿puede la infraestructura UAV según 3 funcionar bien en el entorno gestionado en zonas según 2?
6. ¿Existe algún conflicto entre 3 y 1? Es decir, explicítense limitaciones al desempeño de las tareas previstas según 1 derivadas de la infraestructura y gestión del sistema multi-UAV diseñado en 3.

3.1 Naturaleza de las tareas

Las tareas a realizar por el elemento objeto de decisión (UAV/dron) en el espacio-tiempo, las podemos clasificar en dos tipos. Por un lado, las tareas de valor añadido, que son las directamente relacionadas como la configuración del servicio de cobertura audiovisual. Por otro lado, tareas de más bajo nivel, y que tienen que ver con actividades de soporte que es necesario ocurran para que el dron esté en condiciones de prestar el servicio.

Para un esquema simplificado del ciclo de trabajo de cada dron, remitimos a la Figura 3-1:

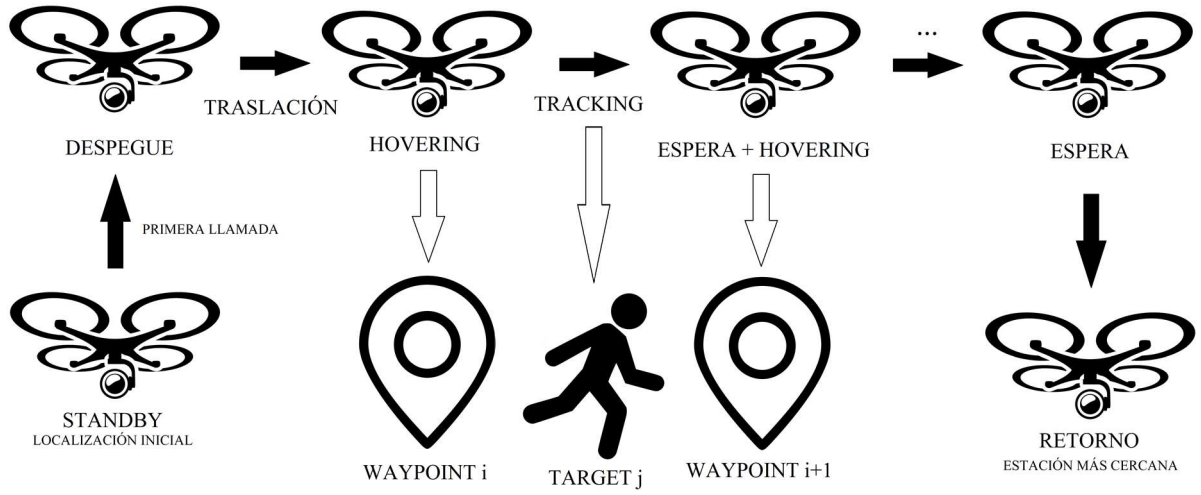

Figura 3-1: Simplificación de ciclo de trabajo de un dron

Las tareas de valor añadido con que el dron sirve a un waypoint se dividen en dos partes bien diferenciadas: *hovering* y *tracking*:

- **Hovering**: Es el acto del dron de permanecer quieto en un lugar determinado o desplazándose muy lentamente alrededor de él durante un tiempo. Cada waypoint a servir puede poseer un tiempo de hovering distinto.
- **Tracking**: Es el acto del dron de desplazarse entre waypoints de la trayectoria de un target a una velocidad crucero. El consumo del dron aumenta al realizar tracking, ya que sus motores deben aportar empuje además de generar sustentación.

El hecho de que un dron sirva a un waypoint implica, pues, que aquel es capaz de satisfacer ambas partes del servicio con el consumo de autonomía que ambas implican y que es capaz de llegar a dicho waypoint antes de un momento determinado, que sería el instante temporal en el que comenzaría el hovering.

Los drones realizan tracking a una velocidad preferente, a la que se denomina *velocidad crucero*, y realizan la mayoría de sus desplazamientos a una altura preferente, a la que se denomina nivel de vuelo o *flight level*.

Nótese cómo se representan unos determinados lapsos en que el dron espera que el target esté en el punto de interés para comenzar su seguimiento. Como puntualización, el anclaje de un dron para la grabación de un objetivo sería una solución intuitiva natural sólo que, dado que la autonomía del dron decae y acabaría por agotarse, sólo se puede acometer de una vez una limitada sucesión de waypoints del mismo objetivo.

En cuanto a las actividades de soporte, éstas en nuestro caso tienen que ver con: despegue, aterrizaje, remplazo de baterías, desplazamiento entre puntos y cambios de rumbo y/o velocidad o de altura de vuelo para evasión de colisiones. Todas éstas son operaciones que posibilitan que las aeronaves puedan acometer de forma efectiva las tareas de valor añadido en esta aplicación: **hovering** a la espera del target y **seguimiento** (tracking) posterior del desplazamiento del mismo.

Además del tipo de tarea según (Murray & Karwan, 2010), también podemos encontrarnos el hecho de que algunas tareas deban satisfacerse en cualquier caso mientras que otras sean opcionales. Por otro lado, alguna de las tareas puede gozar de preferencia en cuanto a ejecutarse en una ventana temporal concreta, y en el caso de las opcionales las tareas pueden gozar de preferencia en cuanto a la realización o no de las mismas.

3.2 Configuración del servicio

Volvemos al primer párrafo de este capítulo: "*Un sistema de drones homogéneo es el encargado de servir a un conjunto de objetivos (targets) a lo largo de su trayectoria, la cual es definida por un set de puntos (waypoints) y de tiempos. Estos drones cuentan con el apoyo de una serie de estaciones de recarga en unas posiciones determinadas, en las que son capaces de realizar un cambio de baterías que renueve su autonomía.*"

Dado que el diseño de las operaciones se pretende realizar de forma dinámica, puede pensarse en la definición de los puntos de recarga, las posiciones iniciales de los drones en el área del servicio y el nivel de batería en esas posiciones como parámetros a definir en el inicio del procedimiento de planificación.

Se entiende que este input a la planificación contiene de forma explícita un tamaño de la flota homogénea y datos relativos a las ventanas temporales de servicio en cada uno de los puntos de interés. Cada punto de interés a servir por la flota pertenece a la trayectoria prevista de algún target, y en el ir y venir de los drones, puede suceder que sobre el papel se detecte riesgo de colisión entre dos drones que cruzan sus trayectorias en una ventana temporal de ancho menor que T.

Sería muy recomendable que desde la etapa de planificación pueda evaluarse la viabilidad operativa de las rutas calculadas. La cuantificación de los mayores o menores riesgos de posibles colisiones (que no tienen por qué llevar a la colisión misma, pero sí a maniobras coordinadas en que varios pilotos RPA deban ocuparse de cambiar de rumbo y/o altura para esquivarse) deberá ser tenida en cuenta a la hora de elegir de entre varias planificaciones posibles.

3.3 El recurso dron: hipótesis de uso

A pesar de suponer la existencia de una flota homogénea de multirrotores para el desempeño de las tareas de cobertura audiovisual, desde un punto de vista práctico, se asume heterogeneidad en cada dron de la flota respecto a su nivel de batería. Tanto este nivel, como la localización inicial de cada dron, serán dependientes del momento en que el usuario decide ejecutar el Scheduler.

Cada dron inicia su misión desde su localización inicial e intenta concentrarse en el seguimiento de un participante concreto, hasta que resulte conveniente su paso por una estación de reemplazo antes de quedar sin batería. Cuando se pasa por una estación de reemplazo, se realiza sobre él una tarea de soporte denominada 'reemplazo de batería', que tiene una duración estándar en todos y cada uno de los puntos de recarga.

Tras abandonar el punto de recarga, el dron queda a la espera de sincronizarse con el que será el nuevo participante a seguir (existe libertad para asignarlo a un participante diferente). La sincronización se debería hacer con determinado tiempo de guarda, para tener en cuenta cuestiones prácticas como la posible existencia de un viento superior al esperado, que pudiese ralentizar los valores nominales de desplazamiento.

En relación a la velocidad de vuelo se considera que, en el espacio entre dos puntos de interés, el multirrotor en uso va a poder volar a una velocidad de crucero (en aplicaciones de este tipo puede estar hablándose de entre 5 y 10 m/s, equivalentes a 18-36km/h). En este caso se supone que los desplazamientos de este tipo ("traslados" en la Figura 3-1) suceden a velocidad constante. La única excepción a esto se da en el caso en que sea necesario tomar en cuenta un cruce de trayectorias de dos drones diferentes en una horquilla temporal pequeña, ya que en tal caso el Scheduler debería corregir alguna de las velocidades de los drones en conflicto para evitar el riesgo de colisión.

En cuanto al servicio sobre cada punto de interés, será posible sólo en el caso en que el dron esté en dicha localización en el inicio de la ventana temporal de cobertura en el entorno del waypoint. Se asume que la posibilidad de manejo del ángulo de la cámara y de las funciones de zoom permiten que la tarea de valor añadido a realizar sobre dicho waypoint sea posible con tan sólo un desplazamiento a baja velocidad en la dirección de desplazamiento del participante. Este tipo de tarea, que hemos identificado como hovering por simplicidad en nuestra exposición, deriva en un consumo de batería diferenciado para la tarea 'hovering', que es menor que el de la tarea de tracking.

3.4 Cómo construir un plan de servicio

La información de la trayectoria de los participantes a los que dar cobertura con la flota de drones, debe proporcionarse como dato de entrada para la programación de un servicio capaz de cubrir los puntos de interés proporcionados. La infraestructura disponible al servicio de la flota de drones también debe quedar explicitada: puntos de despegue, ubicación de las estaciones de reemplazo de baterías y puntos de recogida o finalización a los que pueden ir a parar los drones una vez finalicen su misión.

La definición del plan de servicio consistirá en la asignación de tareas a los drones, cada una de ellas con explicitación del espacio y de la ventana temporal en que ocurre. El plan de servicio será en sí un plan de cobertura de los puntos de interés (véase ejemplo en la Figura 3-2), y debería venir de una herramienta software que, tal y como ya se la ha denominado, podríamos llamar "Scheduler". El Scheduler debería soportar las decisiones que el productor debe tomar, indicando qué evento de los datos de entrada a este software es cubierto en cada instante de tiempo y qué dron miembro de la flota lo hará.

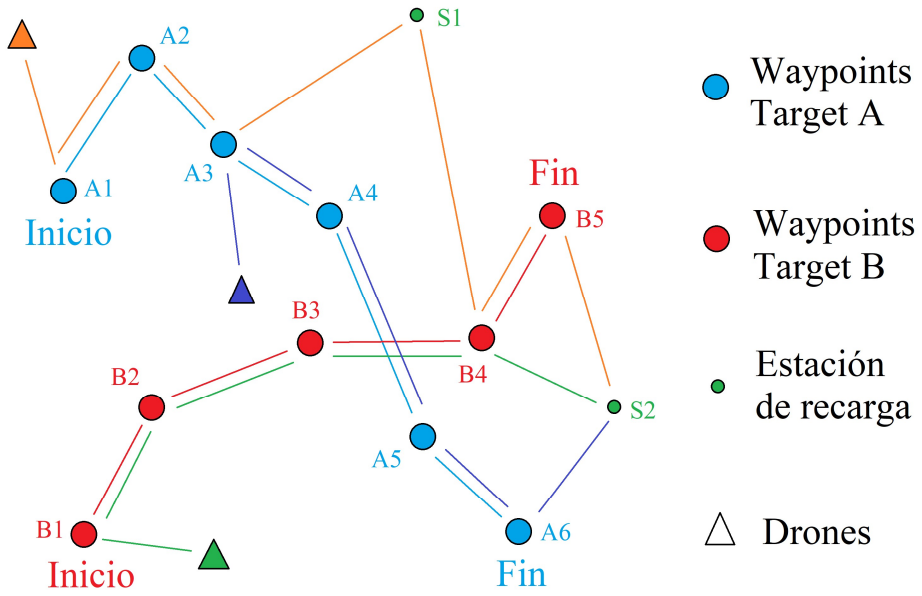

Figura 3-2: Prueba deportiva con 2 participantes a los que seguir en su trayectoria para poder grabar en los waypoints (círculos azules y rojos). Ejemplo de plan de cobertura del evento con 3 drones, partiendo de ubicaciones iniciales distintas (triángulos) y usando 2 estaciones de reemplazo de baterías (S1 y S2).

Dejaremos abierta la posibilidad a que en el plan de cobertura que resulte, algunos puntos puedan no quedar cubiertos. Esto, en la ontología propuesta en (Murray & Karwan, 2010) significaría que nuestro problema de secuenciación de tareas se ocupa de actividades del tipo ONA: opcionales, no interrumpibles (non-preemptive) y que pueden ser ejecutadas en cualquier instante temporal permitido. Esto último significa que los puntos de interés deberán ser cubiertos en la ventana temporal que el fichero de entrada exige.

Consideramos que lo más apropiado es que este Scheduler resulte ser una aplicación **centralizada** (Kaddouh, et al., 2016) y reactiva. Es decir, que en un corto espacio de tiempo pueda producir un plan de cobertura válido. Con ello, se pretende dejar a disposición del usuario de este software la posible actualización de las necesidades del sistema a planificar (un nuevo target que se incluye, un target que abandonó la prueba deportiva, etc.) cuando merece la pena reorganizar el plan de cobertura que rija la adquisición de imágenes audiovisuales de la misma.

Un enfoque **reactivo** parecido al que necesitamos en nuestro caso de estudio es el planteado por (Song, et al., 2016). Allí se propone una heurística cuya medida del performance de las soluciones generadas es una combinación lineal del número de puntos desatendidos y del tiempo total de misión (el lapso entre el primer despegue de un dron y el último de los aterrizajes en ubicaciones de recuperación dispuestas para la flota). Ambas cuestiones, con el añadido de la penalización de los cruces, son criterios lógicos a incluir como métricas de la bondad de las soluciones que surjan del Scheduler objeto del presente TFM.

En la figura 3.2, se presenta una conformación del plan de servicio para un pequeño ejemplo, con dos participantes a los que grabar en un área extensa, haciendo uso de tres drones (en posiciones iniciales graficadas como triángulos rojo, azul y verde), dos estaciones de reemplazo y dos puntos de finalización de la misión, que en este caso coinciden con las estaciones de reemplazo. En este contexto, estas ubicaciones finales son las propias de los puntos de recogida y repliegue de drones. En la práctica, lo natural es que este número de zonas de repliegue sea pequeño.

Más allá de la configuración concreta de la función objetivo que decidamos usar, soluciones lógicas al problema planteado típicamente suponen que una sucesión de waypoints sea asignado al mismo dron, y que los drones se turnen entre ellos para completar el servicio a un determinado target. El atractivo del Scheduler debería estar en generar una diversidad de soluciones según el peso con el que el usuario pondere los tres criterios identificados, y hacerlo en tiempo acotado. Para lograr esto, en el siguiente capítulo presentamos el diseño de una heurística de asignación secuencial de tareas que evoluciona la heurística desarrollada en (Song, et al., 2016) para adecuarse a los requerimientos específicos listados.

4 ANÁLISIS DEL PROBLEMA Y DE SU RESOLUCIÓN

La cobertura audiovisual del evento deportivo debería resultar de la utilización de una herramienta de soporte capaz de generar planes de gestión de la flota de múltiples drones en un corto espacio de tiempo. Cada plan consistirá en la definición de las tareas asignadas a los múltiples drones en la flota, para la consecución de la meta establecida mediante el comportamiento cooperativo de los múltiples drones. La herramienta es, pues, un Scheduler de las distintas tareas de valor añadido a realizar que se explotará para escenarios posibles a valorar por el usuario final.

A continuación, se aborda el modelado del problema de planificación subyacente. Se trata de un problema MILP duro por lo que procedemos a analizar la mejor forma de abordar una resolución aproximada alineada con las necesidades de reactividad que demanda el usuario planificador.

4.1 Descripción del problema

Tal cual está planteado, el Scheduler aborda la mejor resolución de un problema de secuenciación de tareas con un pool de recursos consistente en una variedad de drones elegibles para la realización de todas las tareas (lo que en el campo de programación de tareas se denomina flexibilidad de máquina). Éste problema, planteado como un problema de asignación de tareas y libre del manejo de las ecuaciones de la dinámica de vuelo (un nivel de detalle propios del estudio de los sistemas dron a bajo nivel para la caracterización de la operatividad de las misiones de vuelo) se puede asimilar a un problema tipo FJSP (Chaudhry & Khan, 2016).

Sin embargo, dada la elegibilidad total existente de cada tarea en relación a cada dron, una formulación según un problema VRP es más apropiada. Así pues, asumiendo que la definición a alto nivel de la infraestructura de soporte ya fue resuelta (es decir, el dimensionado de la flota y emplazamiento de estaciones de reemplazo de baterías ya son datos de entrada), nuestro problema es cómo planificar las misiones de los drones y su marcha a las estaciones de reemplazo para completar las tareas especificadas. Además, los puntos de retorno también pueden ser varios, lo que se correspondería a lugares en los que varios drones acaben siendo recogidos para mudar toda la infraestructura de la corporación audiovisual una vez finalizado el evento deportivo.

La existencia de una variedad de puntos de abastecimiento/retorno (depot en jerga VRP) ya aparece en trabajos del área VRP (Kim, et al., 2013), aunque existe mucho mayor producción científica en el caso de un único depot (Nigam, et al., 2012). El problema concreto del diseño de rutas para una flota de drones se ha modelado como un CVRP con consideración de limitación de autonomía en (Olivares, et al., 2015). En (Shetty, et al., 2008), las capacidades/equipamiento del dron y de los niveles de servicio a los waypoints a servir se consideran también para la definición de una solución a la definición del servicio prestado por una flota de drones heterogéneas.

El problema que nos ocupa en este TFM es uno en el que cada uno de los recursos activos del sistema (drones) realiza una multitud de viajes. Para una revisión de las posibilidades de modelado matemático del problema de enrutado de vehículos con múltiples viajes (MTVRP), referimos al interesante trabajo de (Cattaruzza, et al., 2018).

En lo que sigue, presentamos cómo en este TFM nos decantamos por una formulación en cuatro índices. Como se ha mencionado en la revisión de la bibliografía, se trata de una extensión de la formulación VRP clásica de tres índices: i (nodo de salida), j (nodo de destino) y k (vehículo asignado); en que un cuarto índice r representa el ordinal del viaje realizado por el vehículo en cuestión. Con el uso de la asignación según variables de cuatro índices, se saca partido a un enfoque temporal más específico al problema MTVRP (tal y como también hace (Song, et al., 2014)), al facilitar la identificación no sólo del dron que sirve una conexión determinada sino también del ordinal de la misión (en el plan de trabajo de ese dron) en la que se encaja dicho vuelo.

4.2 Formulación matemática

Sea una flota de drones idénticos, que de inicio tienen unos niveles de batería diferentes y se encuentran en posiciones arbitrarias. Éstos datos son asumidos deterministas y conocidos como entrada a la planificación, como también se supone son las trayectorias espacio-temporales de los targets a grabar. Las trayectorias de los targets se asumen discretizadas mediante la división de las mismas en segmentos llamados trabajos[1] ('Jobs'), para acercar nuestra jerga al problema de Scheduling subyacente. Cada trabajo o Job puede ser servido por cualquiera de los drones de la flota, y es el elemento que recibe la asignación del recurso dron.

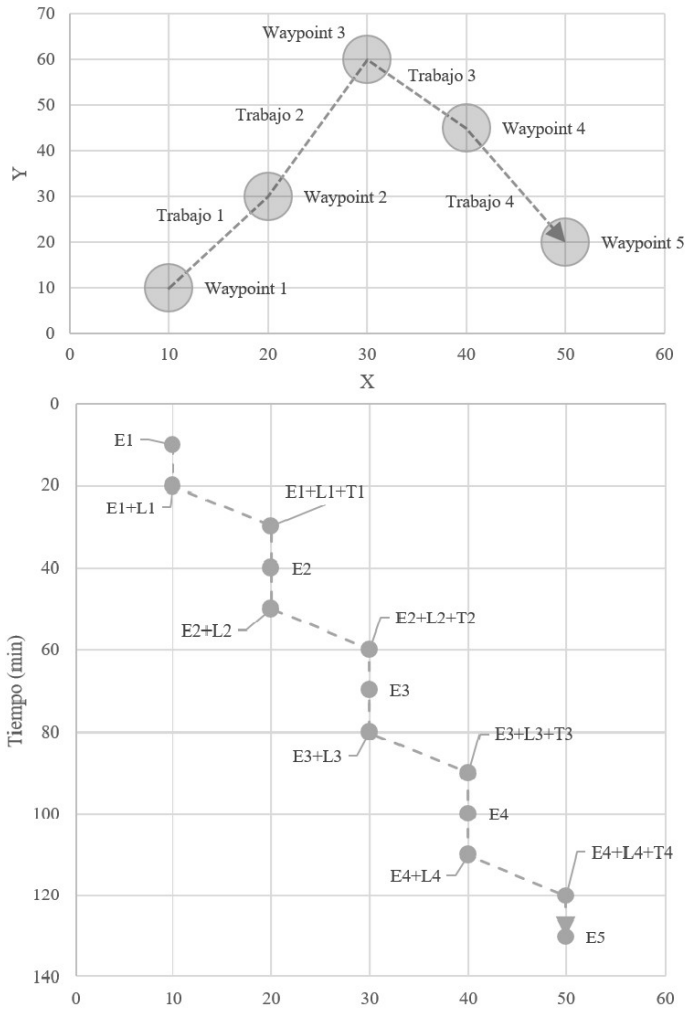

Figura 4-1: Una trayectoria espaciodimensional es dividida en cuatro trabajos

[1] La conversión de trayectoria a trabajos depende del diseñador del sistema – típicamente se utilizan puntos de interés, o se fija un intervalo fijo de tiempo para dividir las trayectorias, como por ejemplo minutos o segundos. Como se ve más adelante en este capítulo, la definición inicial de más trabajos (más puntos de interés o un intervalo de tiempo más pequeño) permitiría una mayor persecución de los objetivos del sistema al coste de una mayor necesidad de computación

Cada trabajo i queda definido por una localización de comienzo (x_{is}, y_{is}) y una localización final (x_{ie}, y_{ie}), teniendo un instante de comienzo estricto E_i, un tiempo de hovering L_i y un tiempo de tracking T_i. El trabajo se caracteriza por un tiempo de procesado $P_i = L_i + T_i$, coincidente con el tiempo que el dron sigue al target en ese segmento de su camino (asumimos que las velocidades durante hovering y tracking son constantes e independientes). La Figura 4-1 muestra un ejemplo de una trayectoria espacio-temporal dividida en cuatro trabajos. La distancia de trabajo D_{ij} desde el punto final del trabajo i (o la estación i) al punto de comienzo del trabajo j (o la estación j) es calculada utilizando la distancia euclídea. Esas distancias forman una red asimétrica, ya que D_{ij} no es necesariamente igual a D_{ji} (los puntos de comienzo y de fin cambian).

Asumimos que los siguientes datos son conocidos y parámetros constantes del problema: puntos de comienzo y fin de los trabajos, tiempos de comienzo y de hovering de los trabajos, localización de las estaciones, localizaciones actuales de los drones y nivel de autonomía de los drones.

4.2.1 Notación

i, j: Índices para los trabajos

s: Índice para las estaciones

k: Índice para los drones

r: Índice del $r_{ésimo}$ vuelo de un dron

N_J: Número de trabajos

N_{UAV}: Número de drones en el sistema

N_{STA}: Número de estaciones de recarga

N_R: Número máximo de vuelos por dron durante el horizonte de tiempo

M: Número positivo grande

D_{ij}: Distancia desde el punto de fin del trabajo i al punto de comienzo del trabajo j

E_i: Instante de comienzo del trabajo i

T_i: Tiempo de tracking del trabajo i

L_i: Tiempo de hovering del trabajo i

P_i: Tiempo de proceso del trabajo i $(T_i + L_i)$

H: Tiempo requerido para la recarga completa de un dron

U: Tiempo de preparación para un proceso de recarga/repostaje

q_k: Tiempo de viaje máximo de cada dron k

$q_{k,ini}$: Nivel inicial de batería/fuel/autonomía de cada dron k

TS_k: Velocidad de viaje del dron k

w_1: Factor de peso entre criterios objetivos, valor entre 0 y 1

w_2: Factor de escala entero positivo del número de trabajos servidos en la función objetivo

Ω_J: Set de trabajos

Ω_{SS}: Set de estaciones de comienzo de drones

Ω_{SE}: Set de estaciones de fin de drones

Ω_A: Set de todos los trabajos y estaciones de recarga

Ω_{INI}: Set de localizaciones iniciales de drones

X_{ijkr}: Variable binaria de decisión. 1 si el dron k procesa el trabajo j o se recarga en la estación j después de procesar el trabajo i o recargarse en la estación i durante su $r_{ésimo}$ vuelo, 0 en caso contrario.

C_{ikr}: Variable de decisión número real, es el instante de comienzo del trabajo i realizado por el dron k durante su $r_{ésimo}$ vuelo o el instante de comienzo de recarga de dicho dron en la estación i, 0 en cualquier otro caso.

Q_{kr}: Variable de decisión número real, consumo total de batería de cada dron k durante su $r_{ésimo}$ vuelo

4.2.2 Recarga

Se asume que los drones se recargan completamente al realizar una visita a una estación. En este estudio, la duración de dicho repostaje es constante, no dependiendo del nivel de batería del dron en el instante de recarga, dado que lo que se realiza es un intercambio de batería usada por una batería llena.

4.2.3 Formulación de la planificación de rutas del programa lineal

En la Figura 4-2 se proporciona el modelo lineal para el sistema planteado.

$$Minimize\ w_1 \sum_{i\in\Omega_A}\sum_{j\in\Omega_A}\sum_{k\in K}\sum_{r\in R} D_{ij}\cdot X_{ijkr} + (1-w_1)\cdot w_2 \cdot \left(|J| - \sum_{i\in\Omega_J}\sum_{\Omega_J\cup\Omega_{SE}}\sum_{k\in K}\sum_{r\in R} X_{ijkr}\right) \quad (11)$$

Subject to

$$\sum_{i\in\Omega_{SS}\cup\Omega_{INI}}\sum_{j\in\Omega_J\cup\Omega_{SE}} X_{ijkr} = 1 \quad (k\in K, r\in R) \quad (12)$$

$$\sum_{i\in\Omega_J\cup\Omega_{SS\cup u}\Omega_{INI}}\sum_{s\in\Omega_{SE}} X_{iskr} = 1 \quad (k\in K, r\in R) \quad (13)$$

$$\sum_{i\in\Omega_J\cup\Omega_{SS}} X_{iskr} = \sum_{i\in\Omega_J\cup\Omega_{SE}} X_{s-1,ikr+1} \quad (k\in K, r=1,\dots N_R-1, s\in\Omega_{SE}) \quad (14)$$

$$\sum_{i\in\Omega_J\cup\Omega_{SS}} X_{iskr} = 0 \quad (k\in K, r\in R, s\in\Omega_{SS}\cup\Omega_{INI}) \quad (15)$$

$$\sum_{k\in K}\sum_{r\in R}\sum_{i\in\Omega_A} X_{ijkr} \le 1 \quad (j\in\Omega_J) \quad (16)$$

$$\sum_{j\in\Omega_A} X_{ijkr} - \sum_{j\in\Omega_A} X_{jikr} = 0 \quad (i\in\Omega_J, k\in K, r\in R) \quad (17)$$

$$C_{skr} = C_{s-1,kr+1} \quad (k\in K, r=1,\dots N_R-1, s\in\Omega_{SE}) \quad (18)$$

$$C_{ikr} + P_i + \frac{D_{ij}}{TS_k} - C_{jkr} \le M\left(1-X_{ijkr}\right) \quad (i\in\Omega_J, j\in\Omega_J\cup\Omega_{SE}, k\in K, r\in R) \quad (19)$$

$$M\cdot\sum_{j\in\Omega_J\cup\Omega_{SE}} X_{ijkr} \ge C_{ikr} \quad (i\in\Omega_J\cup\Omega_{SS}, k\in K, r\in R) \quad (20)$$

$$C_{ikr} + RT_f + \frac{D_{ij}}{TS_k} - C_{jkr} \le M\left(1-X_{ijkr}\right) \quad (i\in\Omega_{SS}, j\in\Omega_J\cup\Omega_{SE}, k\in K, r=2) \quad (21)$$

$$C_{ikr} + RT_r + \frac{D_{ij}}{TS_k} - C_{jkr} \le M\left(1-X_{ijkr}\right) \quad (i\in\Omega_{SS}, j\in\Omega_J\cup\Omega_{SE}, k\in K, r>2) \quad (22)$$

$$\sum_{k\in K}\sum_{r\in R} C_{ikr} = E_i \quad (i\in\Omega_J) \quad (23)$$

$$\sum_{i\in\Omega_A}\sum_{j\in\Omega_A}\frac{D_{ij}}{TS_k}\cdot X_{ijkr} + \sum_{i\in\Omega_J}\sum_{j\in\Omega_A} P_i\cdot X_{ijkr} \le q_{kr} \quad (k\in K, r\in R) \quad (24)$$

$$q_{kr} \le q_{k,ini} \quad (k\in K, r=1) \quad (25)$$

$$q_{kr} \le q_k \quad (k\in K, r\ne 1) \quad (26)$$

$$C_{ikr} \ge 0 \quad (k\in K, r\in R, i\in\Omega_A) \quad (27)$$

$$q_{kr} \ge \quad (k\in K, r\in R) \quad (28)$$

$$X_{ijkr}\in\{0,1\} \quad (i\in\Omega_A, j\in\Omega_A k\in K, r\in R) \quad (29)$$

Figura 4-2: Formulación matemática del modelo lineal de planificación de rutas

La función objetivo (11) minimiza la suma ponderada de la distancia total de viaje y el número de trabajos servidos. El factor de peso w_1 balancea los dos objetivos. La constante w_2 convierte los objetivos a la misma unidad. Grandes valores de w_1 darán como resultado organizaciones de drones energéticamente eficientes reduciendo la distancia de vuelo (las velocidades se asumen constantes en cada trabajo y entre ellos). Aun así, podría reducir la calidad del servicio debido a la presencia de trabajos sin servir. Por otro lado, valores bajos de w_1 garantizarán una calidad de servicio sirviendo tantos trabajos como sea posible.

Las restricciones (12-17) coordinan los caminos de dron. La restricción (12) asegura que cada dron comienza su vuelo desde su localización o estación inicial. Permite a los drones servir trabajos o desplazarse a una estación. La restricción (13) garantiza que cada dron termina su vuelo en una estación. El índice dual para las estaciones es utilizado en la restricción (14).

Cuando el dron k finaliza su $r_{ésimo}$vuelo en la estación $s-1$, su $r+1_{ésimo}$ vuelo comienza en la estación s. La restricción (15) asegura que un dron no puede terminar su vuelo en su estación o localización de comienzo (las estaciones de comienzo y de fin tienen índices distintos). La restricción (16) requiere que los trabajos en Ω_J son servidos por como máximo un dron. Como tal, un trabajo no ha de ser obligatoriamente servido. Los drones no finalizan sus vuelos en un trabajo gracias a la restricción (17).

Las restricciones desde las ecuaciones 18 a la 23 determinan el tiempo de comienzo de los trabajos y de las recargas en las estaciones de servicio. La restricción (18) requiere que el tiempo de comienzo del $r_{ésimo}$ vuelo de un dron y el tiempo de comienzo de su $r+1_{ésimo}$ vuelo sean el mismo. La restricción (19) dicta que el instante de comienzo del trabajo o el tiempo de llegada a una estación (cuando la recarga comienza) iguala al de la siguiente tarea del dron en su $r_{ésimo}$ vuelo. La restricción (20) determina el instante de comienzo de trabajo de los trabajos sin servir. C_{ikr} es puesto a cero si el trabajo i no es asignado al $r_{ésimo}$ vuelo del dron k. Las restricciones (21) y (22) dictan el tiempo de recarga para cada dron y el tiempo de comienzo de un trabajo después de la recarga. Se distinguen basados en el orden de vuelo. Si $r=2$, lo que significa el comienzo del segundo vuelo, las restricciones (21) y la ecuación (1) determinan el nivel de batería del dron k ($q_{k,ini}$). Si $r>2$, la restricción (22) y la ecuación 2 son usadas en su lugar. La restricción (23) fuerza a los drones a proveer servicio en el instante correcto de comienzo del trabajo i en Ω_J.

Las restricciones de autonomía son descritas vía las restricciones de 24 a 26. La restricción (26) asegura que los drones no vuelan más de lo que pueden. Las restricciones 25 y 26 aseguran que las variables de decisión q_{kr} siguen su propio rango.

Finalmente, las restricciones de la 27 a la 29 especifican las variables de decisión real y binaria para el modelo. Requerimos que $N_R > 1$ debido a las constantes (14) y (18). N_R es el máximo número de vuelos permitidos para cada dron, este valor entero no negativo puede ser establecido arbitrariamente.

Este modelo coincide con el presentado en (Song, et al., 2014), y allí queda explicitado un análisis de complejidad que, por reducción a un CVRP, demuestra que es NP-hard.

Dado que el Scheduler que se necesita para la aplicación real abordada en este TFM debe ser reactiva, a continuación presentamos las bases de una resolución aproximada (heurística) capaz de servir al objetivo de alto nivel que se planteó en el arranque de este documento: desarrollar una herramienta flexible capaz de manejar la mayor cantidad de escenarios posibles con el mínimo esfuerzo por la parte del usuario, y al mismo tiempo siendo capaz de ser parametrizada para ajustarse a los criterios y requisitos de aquél para cada situación.

4.3 Estrategia para Construcción de Soluciones Aproximadas

Divisamos un procedimiento de asignación secuencial de tareas que, de forma constructiva, tiende a producir buenas soluciones al problema estudiado.

Supongamos que ordenamos los targets clientes del servicio de grabación según el tiempo de comienzo de su cobertura, desde el más temprano hasta el más tardío. Sea $P = \{1,2,\dots,p\}$ el set de targets y sea P_t el set de trabajos a realizar sobre el cliente t. Por la ordenación previa, sabemos que los elementos de P_t se disponen en orden creciente del instante de comienzo de cada trabajo.

Como hipótesis fundamental, asumamos que cada dron tiene suficiente velocidad como para servir a los trabajos (esto es, que no existe ningún cliente que se mueva más rápido que el dron más lento y que, por tanto, somos libres de asignar cualquier dron a cualquier trabajo).

Sea l el índice de trabajos en P_t, $t \in P$. El procedimiento constructivo que proponemos empezaría con el target $t = 1$ y $l = 1$, sus trabajos son asignados en orden cronológico, comenzando por el primero. Para asignar un dron a un trabajo concreto l, dos valores 'voraces' son calculados para discernir qué dron es el más apropiado para ese trabajo:

- 'Idoneidad de asignación de viaje directo': $V_D(k)$. Este primer valor corresponde a la situación en que el dron k directamente procede a realizar el trabajo l desde el fin de la que fuese su tarea más reciente; y no sólo éste trabajo, sino que este $V_D(k)$ toma en consideración que secuencialmente se sirvan a tantos trabajos del cliente t como se pueda, según su nivel actual de autonomía, y que a partir de ahí vuele a una estación de recarga. Esto es lo que se denomina *vuelo directo*.
- 'Idoneidad de asignación de viaje directo': $V_I(k)$. Este segundo valor corresponde al dron k volando a la estación más cercana al fin de su tarea más reciente, para conseguir una batería llena y desde ahí ir a realizar el trabajo l y sucesivamente servir tantos trabajos del cliente t como pueda y tras esto marchar a una estación de recarga. Esto se denomina *vuelo indirecto*.

El dron que consiga el máximo valor de idoneidad es asignado a ese trabajo. Si dos drones consiguen el mismo valor, se elige uno arbitrariamente. En el caso de empates entre vuelo directo e indirecto, se elige uno arbitrariamente. Si no existen drones para poder abordar dicho trabajo (es decir, ni por vía directa ni indirecta) el trabajo se marcaría como no servido (es decir, existe la posibilidad de dejar sin servir este waypoint). Se continuaría con el proceso constructivo haciendo $l = l + 1$ y se vuelve al procedimiento voraz de evaluación de idoneidad y chequeo de factibilidad descrito.

4.3.1 Variables y Funciones a Computar Pseudocódigo

Según lo anterior, el procedimiento constructivo no negocia trabajo a trabajo, sino que ocupa de la asignación de lotes o subconjuntos de trabajos sucesivos del mismo target, de modo que se haga un uso "voraz" de la autonomía de los drones. Todo sucedería como si los drones fuesen son los que eligen. Es por ello que en adelante se prescinde de los índices distintos al índice de dron k[2]. Esto aplica no sólo a $V_D(k)$ y $V_I(k)$, sino a otras tantas variables auxiliares que nos ayudan a procedimentar la construcción de la solución.

Así, sean $C_l(k)$ y $C_q(k)$ las localizaciones y los niveles de autonomía de cada dron k, respectivamente (después de completarse su última tarea programada), y sea $A(k)$ el instante en el que el dron k completa su última tarea programada y está disponible para dar servicio. Recuérdese que l es nuestro índice para trabajos.

Al respecto de las localizaciones, proponemos la notación $(.)'$ y $(.)''$ para denotar las localizaciones iniciales y finales del trabajo $(.)$, respectivamente, quedando sus índices simbólicos: l' y l''. Sea $s(a,b) \in R^2$ la localización de la estación de recarga que proporciona el valor de distancia total menor cuando el dron k vuela de un punto a a la estación y de la estación al punto b, si el nivel de autonomía $C_q(k)$ es suficiente como para alcanzar la estación y posteriormente llegar a b. Si dicha estación no existe, la función devuelve el punto infinito $x = +\infty, y = +\infty$. Se define:

$$s(a,b) = argmin_{s \in \Omega SE}\{D_{a,s} + D_{s,b} \mid C_q(k) \geq \frac{D_{a,s}}{TS_k}, q(k) \geq \frac{D_{s,b}}{TS_k}\} \quad (1a)$$

$$s(a,b) = (+\infty, +\infty) \quad (1b)$$

Si existen varias estaciones que devuelven un mismo resultado de la primera ecuación, nos basta con elegir una arbitrariamente. Tendremos especial interés en $s(C_l(k), l')$. Defínase (2): $s(a) = argmin\, s \in \Omega_{SE}\{D_{a,s}\}$ para denotar la localización de la estación más cercana al punto a, seleccionada arbitrariamente si hay más de una.

[2] Las variables en uso en el procedimiento que divisamos no necesitan más que el track del dron al que se refieren: el factor temporal y de qué índice de misión se trata son inmediatamente obvias en función de la ubicación de su cálculo en el pseudocódigo del procedimiento.

Sea $N_D(k)$ el máximo número de trabajos que el dron k puede servir secuencialmente para el cliente t, comenzando en l y como vuelo directo, sirviendo como indicador de factibilidad de vuelo directo:

$$N_D(k) = \max\{n \in Z_+ | \frac{D_{C_l(k),l'}}{TS_k} + \sum_{i=l}^{l+n-1} P_i + \frac{D_{(l+n-1)'',s((l+n-1)'')}}{TS_k} \leq C_q(k)\} \qquad (3)$$

Si con vuelo directo no somos capaces de ir y servir ningún punto y después regresar a tiempo a punto de recarga, resultará un $N_D(k) = 0$.

De forma análoga, sea $N_I(k)$ el máximo número de trabajos que el dron k puede servir secuencialmente para el cliente t, comenzando en l y con vuelo indirecto, sirviendo como indicador de factibilidad de vuelo indirecto:

$$N_I(k) = \max\{n \in Z_+ | \frac{D_{C_l(k),s(Cl(k),l')}}{TS_k} \leq C_q(k), \frac{D_{s(C_l(k),l'),l'}}{TS_k} + \sum_{i=l}^{l+n-1} P_i + \frac{D_{(l+n-1)'',s((l+n-1)'')}}{TS_k} \leq q_k\} \qquad (5)$$

Juntos, esos indicadores de factibilidad $N_D(k)$ y $N_I(k)$ serán utilizados para comprobar que una asignación de dron concreto es factible según las restricciones (23-24). Además, refuerzan el rol de las restricciones de marcha a recarga de batería (21-22). Si no existen drones para dicho trabajo, el trabajo no puede ser servido (se dejaría sin servir este waypoint).

Si sí que se puede, los índices voraces asignados al dron k que resultan cuando buscamos asignar el trabajo l son:

$$V_D(k) = \{\propto * \beta(N_D(k)) - (1-\alpha) * Dist\_to\_Add\} \qquad (7)$$

$$V_I(k) = \{\propto * \beta(N_I(k)) - (1-\alpha) * Dist\_to\_Add\} \qquad (8)$$

donde α y β son parámetros para balancear dos términos: uno que computa el máximo número de waypoints a cubrir y otro que considera la distancia de vuelo a añadir por esta asignación de waypoints ($Dist\_to\_Add$). En nuestra implementación hacemos α y β iguales a los parámetros w_1 y w_2. El dron que consiga el mayor resultado para $V_D(k)$ o $V_I(k)$ es seleccionado para ejecutar esa serie de trabajos mediante ese tipo de vuelo. Este procedimiento es repetido hasta que todos los trabajos han sido investigados o asignados. Entonces, pasamos a analizar el siguiente cliente. Después de que cada trabajo en el conjunto de todos los targets haya sido investigado o asignado, todos los drones que no se encuentren en una estación de recarga viajan hasta la más cercana. La heurística ha sido completada.

Hágase l = l+1 y repítase la comprobación de factibilidad.

4.3.2 Pseudocódigo

Según lo anterior, el procedimiento constructivo sigue el siguiente esquema.

Algorithm: Sequential task assignment heuristic

```
1: Let contador be the index for ordering the jobs assignment. Let l1 be the job index for each client.
2: Set contador = 0.
3: For all client t,                                                    \\ {from customer 1 to |P|}
4:     l1 = 0;                                                          \\ {resets job index value}
5:     While l1 < |Pt|,                                                 \\ {finds feasible jobs to be fulfilled}
6:         Calculate ND(k) and NI(k) for all UAV k;
7:         Calculate VD(k) and VI(k) for all UAV k and select the biggest value;
8:         Let k' be the selected UAV;
9:         IF the biggest value is 0 for both, do
10:            l1 = l1 + 1;                                             \\ {waypoint can't be served}
11:        ELSE IF VD(k') is selected or VI(k') = ∞, do                 \\ {direct flight is best option}
12:            Viajes, Distancias, Dist_Total ← D(k');                  \\ {gets values from function}
13:            crear_vuelos(k', contador, directo);                     \\ {assignment of flights and their information}
14:            l1 = l1 + Viajes;                                        \\ {updates index and keeps on searching}
15:        ELSE IF VI(k') is selected or VD(k') = ∞, do                 \\ {indirect flight is best option}
16:            Viajes, Distancia1, Distancia2, Estacion1, Dist_Total ← IND(k');
17:            crear_vuelos(k', contador, indirecto);
18:            l1 = l1 + Viajes;
19:        ELSE, do
20:            pass;                                                    \\ {repeats until every waypoint is served or ignored}
21:    End while;
22:    For all UAV k,
23:        crear_vuelo_final(k);                                        \\ {makes all UAV go to their nearest station}
24:        localizar_intersecciones(k);                                 \\ {looks after potential collisions among UAVs}
25:        resolver_intersecciones(k);                                  \\ {tries to solve potential collisions changing UAVs speeds}
26:    End for;
27: End for
```

5 DISEÑO E IMPLEMENTACIÓN DEL SCHEDULER

Proponemos que el Scheduler sea un desarrollo Python, según un diseño y unos criterios de implementación que se explican a continuación.

5.1 Decisiones de Diseño para la Aplicación Scheduler

5.1.1 Diseño a alto nivel

Definimos como mecánica de trabajo a la manera en que la herramienta a desarrollar interactúa con su entorno para lograr sus objetivos. La mejor acotación de esta mecánica de trabajo es clave para enfocar bien el proceso de desarrollo de una aplicación software a medida capaz de resultar efectiva en el soporte a las decisiones de planificación de la cobertura audiovisual del evento deportivo.

La mecánica de trabajo que planteamos siga el usuario de esta herramienta se ilustra como sigue:

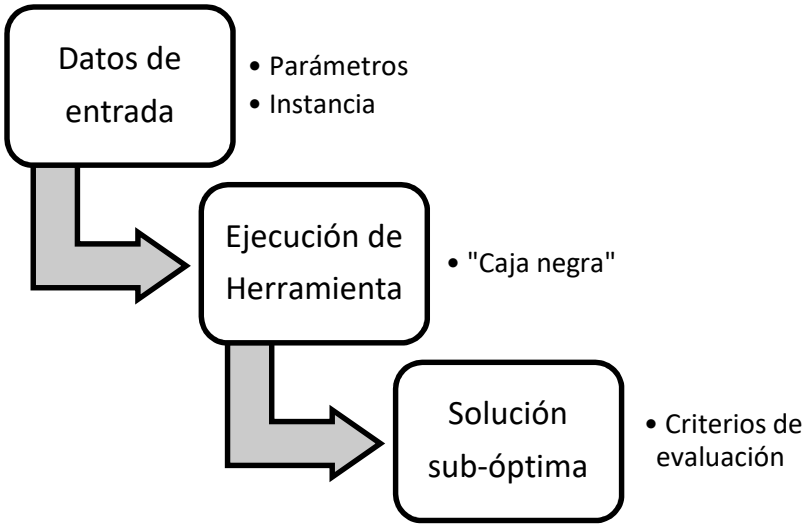

Figura 5-1: Mecánica de trabajo general

Los datos de entrada son tanto los que definen el escenario de la prueba a cubrir, como los parámetros con que se hace variar el balance entre los criterios de optimización del problema MILP subyacente. En cuanto a los

primeros, precisar que se consideran archivos de entrada que codifican información a la manera en que se hace en instancias de benchmarking VRP. En un caso real, sería necesario dar formato idéntico los datos y generar las instancias esta vez con los datos reales del escenario de cobertura audiovisual.

En cuanto a los segundos, dada una instancia determinada (escenario), la variación de los pesos entre los criterios se usará para generar diversidad en las soluciones que proporcione el algoritmo heurístico propuesto. Se obtendrán distintas soluciones sub-óptimas al invocar al núcleo en que se codifica el procedimiento heurístico descrito en 4.3.

Por el momento, tal núcleo puede ser visto simplemente como una caja negra que recibe datos y construye una solución heurística al problema de la planificación coordinada de la cobertura haciendo uso de la flota explicitada. Del estudio comparado que el usuario de la herramienta haga con posterioridad, resultará el plan definitivo para la cobertura audiovisual de la prueba deportiva.

5.1.2 Diseño a medio nivel

En este nivel de detalle, nos centraremos más en cómo necesitamos que funcione la herramienta que en cómo lo hace. Se cuenta con un conjunto de objetivos que recorren una serie de localizaciones en un tiempo determinado, y se tiene la intención de realizarles un seguimiento con drones. Por tanto, un dron que quiera satisfacer las condiciones necesarias para ser capaz de ejecutar dicho seguimiento, debe ser capaz de llegar a un punto determinado antes de un momento determinado. Existen dos tipos de limitaciones para el dron:

- Espaciotemporales, cuando el dron no es capaz de llegar a la localización antes del instante necesario.
- De autonomía, cuando el dron no tiene la autonomía suficiente como para viajar a dicha localización y luego llegar a la estación más cercana.

Para elegir qué drones satisfacen qué waypoints y de qué targets, se siguen las conclusiones de (Song, et al., 2016) donde se demuestra que, para problemas como el del proyecto, una heurística basada en un algoritmo voraz que además favorezca que un mismo dron satisfaga el mayor número de nodos sucesivos posible da como resultado soluciones rápidas y sub-óptimas pero de calidad.

Para flexibilizar al máximo esta heurística, los parámetros que hacen variar los criterios de decisión podrán descomponerse en rangos y ejecutarse en varias posiciones dentro de estos, para así dar una seguridad de su efectividad a la vez que se proporciona una cierta flexibilidad a la hora de elegir, por proveerse de más soluciones, peores que la propuesta inicialmente pero igualmente factibles y de buena eficiencia.

Para dar respuesta a la posibilidad de que se haga necesaria una re planificación, se implementará la posibilidad de que, al ejecutar la herramienta, los drones puedan estar en cualquier lugar e instante, con cualquier valor de autonomía. Esto hace posible que, si se modifican los demás datos de entrada (relativos por ejemplo a trayectorias), sea viable re-ejecutar el código en cualquier instante de manera efectiva.

5.1.3 Diseño a bajo nivel

El siguiente diagrama explicaría el funcionamiento del código que se implementará para generar una diversidad de soluciones al problema objetivo:

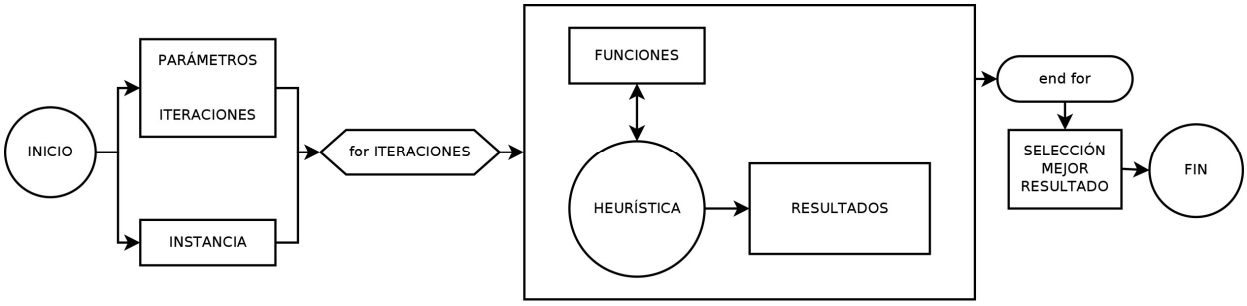

Figura 5-2: Diagrama de flujo de la herramienta

Nótese cómo el núcleo (heurística) no se aplica una sola vez, sino que como parte del caso de uso programado se soporta el lanzamiento de una batería de resoluciones (múltiples iteraciones) en número que el usuario ajusta.

A partir de ahí, el código Python desarrollado es capaz de producir una solución única que refleja la mejor alternativa de entre la variedad de soluciones producidas.

Una descripción como la anterior no supone en sí mismo un detalle a bajo nivel del diseño implementado. Para ello, se considera adecuado presentar diagramas basados en lenguaje de modelado universal (UML) (Booch, et al., 1998). Como es sabido, UML es un lenguaje de modelado gráfico, que utiliza diagramas con símbolos que representan conceptos, líneas que conectan los símbolos y representan relaciones y otras notaciones gráficas para representar restricciones (véase la Figura 5-3).

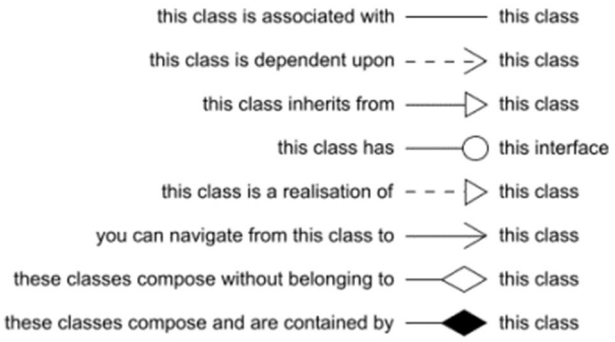

Figura 5-3. Regla que define las flechas de relación de clases en lenguaje de modelado UML (Ivencia, 2018)

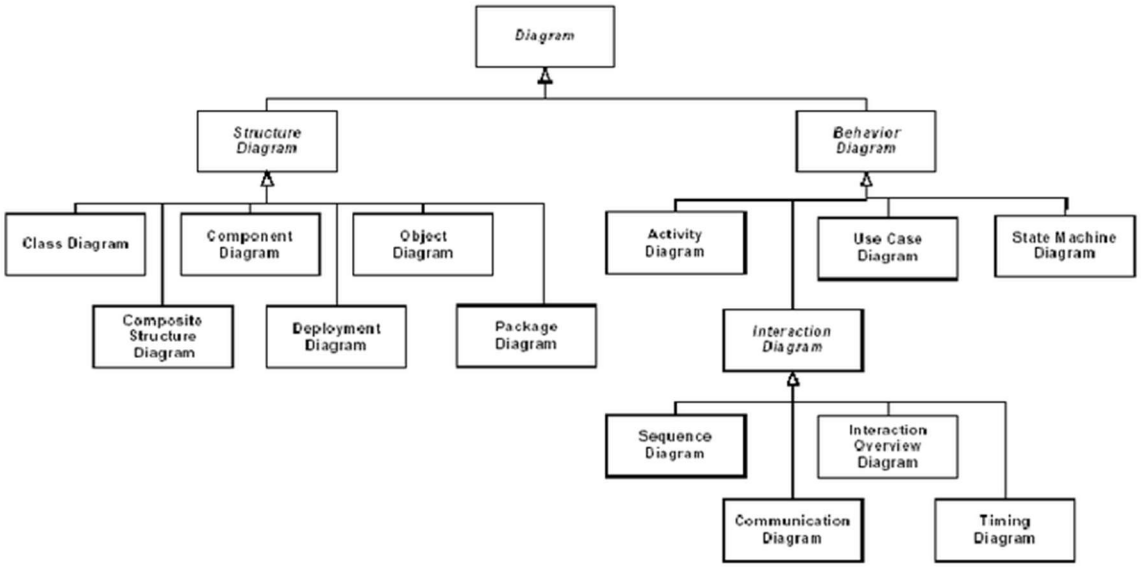

Figura 5-4. Jerarquía de diagramas en UML (Teilans, et al., 2008)

Aclaremos que al usar UML, estamos asumiendo que lo usaremos tal y como se prevé en la herramienta. Es decir, definiendo un conjunto de diagramas que no tiene por qué cubrir toda la jerarquía prevista (ver Figura 5-4), que es tanto como decir que nos vamos a centrar en presentar diagramas que son representaciones gráficas parciales del sistema a diseñar. Este nivel de modelado será suficiente como para que, en apoyo de otra documentación de nuestro diseño, quede suficientemente explicado el diseño funcional de Scheduler.

En nuestro caso, definiremos un modelado conceptual de la herramienta Scheduler haciendo uso de las tres entidades principales del problema: Nodos, Drones y Target.

5.1.3.1 Nodos

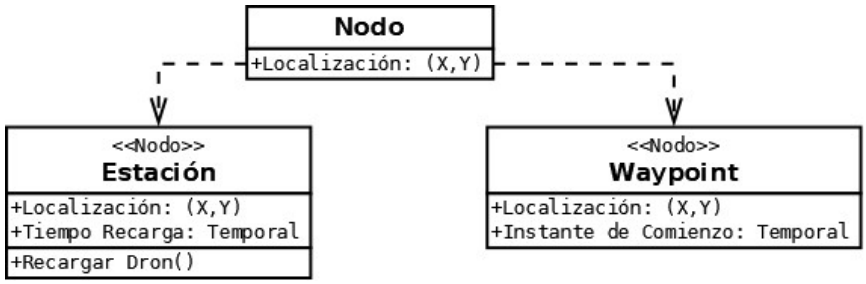

Figura 5-5: Diagrama de clase (Nodo y sus particularizaciones)

Los nodos únicamente poseen localización, son un dato de entrada. Los Waypoints y las Estaciones son casos especiales de Nodo:

- Waypoints: Nodos que forman parte de la ruta de algún Target y a los que un Dron debe satisfacer antes de su Instante de Comienzo.
- Estaciones: Nodos cuya función es recargar la autonomía de un Dron, no formando parte de la ruta de ningún Target.

5.1.3.2 Target

Figura 5-6: Diagrama de clase (Target)

Los Targets, como llamamos a los clientes, representan tan sólo una sucesión ordenada de Waypoints a nivel de dato o estructura. Sin embargo, esto es de gran importancia, pues en la heurística se valora la capacidad de un Dron de servir una serie de Waypoints del mismo Target de manera sucesiva.

5.1.3.3 Dron

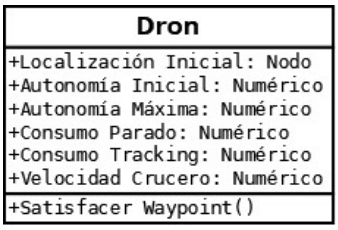

Figura 5-7: Diagrama de clase (Dron)

El Dron es el recurso y entidad principales del sistema. Los drones son los encargados de servir a los Waypoints, lo que logran utilizando su capacidad de traslación a velocidad crucero y su autonomía, la cual son capaces de recargar gracias a las Estaciones. Su autonomía se agota más rápidamente cuando se están desplazando.

5.2 Implementación en Python de la Aplicación Scheduler

El proceso de construcción de solución al problema en estudio presentada en 4.3 se ha implementado en lenguaje Python. Dicho proceso hace uso de ideas presentadas en (Song, et al., 2016), aunque va mucho más allá de la heurística planteada en este artículo primigenio (metaheurística STAH).

Fruto del análisis detallado de la solución planteada en (Song, et al., 2016), se detectaron incongruencias y errores en el planteamiento presentado por estos autores. Además de subsanar esas incongruencias, en el

aplicativo codificado en este TFM incorporamos la detección de los cruces que suponen riesgo de colisión en la traslación a operativa real de las misiones resultantes del procedimiento heurístico.

El problema planteado en el Capítulo 4, por su alta complejidad, requiere de una estructura de datos versátil que dote de la flexibilidad que la herramienta requiere. El tipo de estructura utilizado mayoritariamente en la herramienta desarrollada, es el diccionario Python, por el que se asocia una clave (o nombre) a un contenido determinado (valor).

Por último, precisar que se consideran archivos de entrada que codifican información a la manera en que se hace en instancias de benchmarking VRP. En un caso real, sería necesario dar formato idéntico a los datos y generar las instancias esta vez con los datos reales del escenario de cobertura audiovisual.

5.2.1 Clasificación de datos manejados

- Parametrización de la batería de resoluciones a ejecutar
    - Rango de valores que se desea analizar de los parámetros de peso w1 y w2
    - Iteraciones que se desea realizar variando los parámetros de peso w1 y w2
- Datos de la instancia de entrada
    - Número total de nodos existentes
        - Estos nodos serán divididos más adelante en dos grupos: Waypoints y Estaciones.
    - Localización de cada nodo
        - Expresada en coordenadas X e Y.
    - Número de estaciones
    - Designación de cuáles de las localizaciones son estaciones de reemplazo de baterías
    - Número de targets
    - Ruta determinista de cada target: como una sucesión de waypoints en el plano XY.
    - Instante de comienzo de hovering en cada waypoint
    - Tiempo de hovering en cada nodo
    - Número de drones
    - Autonomía inicial de cada dron
    - Posición inicial de cada dron
    - Velocidad crucero de un dron
    - Velocidad máxima alcanzable por un dron
    - Autonomía máxima de un dron
    - Consumo de un dron al realizar hovering
    - Consumo de un dron al desplazarse
    - Tiempo de cambio de batería de un dron
- Datos de salida de cada resolución
    - Número de waypoints no servidos
    - Número de cruces
        - En realidad, el dato obtenido es el número de cruces que no han conseguido resolverse mediante el método implementado en la herramienta.
    - Distancia total recorrida
- Datos mantenidos para el estudio y consideración de las características de las soluciones generadas:

- Diccionarios que se usan como estructura de datos conteniendo toda la codificación de la solución. Son estructuras que ayudan al usuario avanzado a profundizar en la bondad de la solución que emergió de cada proceso de resolución.
- Traza de todos los valores calculados en los distintos procesos constructivos (variables presentadas en apartado 4.3.1).

5.2.2 Metodología de adaptación de instancia CVRP

Uno de los principales requisitos de la herramienta diseñada es la flexibilidad, lo que no sólo aplica a la resolución sino también a los datos de entrada. Este criterio ha sido tenido en cuenta en el diseño para permitir, en casos en los que no se conozcan todos los datos de entrada del problema, hacerlos constar manualmente o, como en el caso actual, a partir de estancias de problemas CVRP.

Se han codificado una serie de automatizaciones para hacer uso de las instancias de benchmarking en (Xavier, 2014) para crear instancias asimilables a escenarios reales de nuestro problema de estudio. Podría haberse usado un enfoque similar haciendo uso de cualquier otro set de instancias CVRP, a lo que el código escrito se adaptaría fácilmente.

Estas automatizaciones tienen que ver con las siguientes modificaciones respecto a las instancias de partida:

- **Dimensiones**. En primer lugar, hay que hacer notar que las unidades temporales manejadas en el problema son minutos, y las unidades de distancia son kilómetros.
- **Área ocupada por los nodos/waypoints**. El evento deportivo está planteado para cubrir un área de $40km^2$. Al leer las localizaciones de los nodos en instancias VRP genéricas, puede que el área total que comprendan los nodos sea mayor o menor a dicho área, por lo que se aplican reglas de escala para adaptar el área total ocupada por los nodos a esos $40km^2$. Es necesario apuntar que dicha escala puede modificarse fácilmente en el código modificando el valor de la variable entera *Área*.
- **Demanda**. La demanda no es considerada en este problema. Dado que se busca cubrir al máximo el evento, el hecho de no servir a cualquier nodo es igual de desfavorable, por lo que no existen profits/preferencias individuales para cada nodo.
- **Estaciones/depots**. El número de estaciones es una variable entera declarada en el código con el nombre *Num_estaciones*. El código tomará automáticamente, comenzando por el primer nodo almacenado, tantos nodos como estaciones se declaren, y los declarará como depots del problema.
- **Clientes/targets**. El número de targets es otra variable entera declarada en el código, con el nombre *Num_clientes*. Afecta a la hora de crear las rutas, ya que a más targets menos nodos por target habrá.
- **Rutas**. Los nodos que no son estaciones se reparten entre los clientes, creando rutas aleatorias. Estas rutas, por su aleatoriedad, son en realidad ilógicas respecto a una prueba deportiva convencional, pero sirven para probar el funcionamiento de la herramienta bajo condiciones desfavorables.
- **Tiempos de hovering**. Los tiempos de hovering, que en el problema serían dato de entrada, en el código son establecidos manualmente mediante la variable entera *Tiempo_Parado*.
- **Tiempo de comienzo de hovering**. Cada waypoint debe servirse antes de llegar un instante temporal determinado conocido como Tiempo de comienzo de hovering, en el código especificado por el diccionario E_i. De nuevo, este sería un dato de entrada, pero en el código se calcula a partir de los tiempos de hovering, las distancias entre waypoints consecutivos y la velocidad crucero, a la vez que se aporta una cierta aleatoriedad que proporciona más o menos factibilidad según se parametrice.

5.2.3 Metodología de asignación

En la lógica implementada, los clientes son analizados desde el primero hasta el último, y sus correspondientes rutas van tratando de ser satisfechas por todos los drones, desde el primero hasta el último.

Los drones son capaces de satisfacer rutas de targets a través de dos tipos de vuelo distintos, conocidos como vuelo directo y vuelo indirecto:

- Se considera vuelo directo al tipo de vuelo en el que el dron viaja desde su localización actual hacia el primer waypoint que va a satisfacer del target correspondiente, con la posibilidad de seguir satisfaciendo a más waypoints a continuación.
- Por otro lado, se considera vuelo indirecto al tipo de vuelo en el que el dron primero viaja a una estación para realizar un cambio de baterías y luego viaja hacia el waypoint que va a satisfacer de manera similar a como se haría en un vuelo directo.

Cada vez que se ha de elegir qué dron va a satisfacer el siguiente waypoint, se calculan una serie de valores que, en conjunto, determinan dicha elección. Estos valores son:

- Distancia
    - Calcula la distancia entre dos nodos determinados
- Mejor estación para realizar un vuelo indirecto hacia un nodo
    - Determina la estación que minimiza la suma de distancias existentes entre la estación y el nodo actual y entre la estación y un nodo determinado. En caso de empate elige al azar
- Estación más cercana
    - Determina la estación más cercana al nodo actual. En caso de empate elige al azar
- Número máximo de waypoints consecutivos que se podrían satisfacer mediante vuelo directo.
    - Basándose en la autonomía y localización inicial del dron
- Número máximo de waypoints consecutivos que se podrían satisfacer mediante vuelo indirecto.
    - Basándose en la autonomía del dron y en las localizaciones del dron y de la mejor estación para realizar un vuelo indirecto
- Valor de decisión de vuelo directo
    - Basándose en el número máximo de waypoints que se pueden servir, en la distancia total recorrida y en los parámetros w_1 y w_2
- Valor de decisión de vuelo indirecto
    - Ídem del valor anterior

Una vez asignados los drones a las rutas de los targets, se analizan todas las posibles colisiones entre drones. Para esto, en primer lugar, es necesario identificar las intersecciones entre viajes (lo que incluye también los viajes de los drones a las estaciones), para después pasar a analizar el plano temporal. En la lógica implementada, esto se realiza viaje a viaje, en base a lo siguiente:

- Se definen dos tipos de intersección, general y especial:
    - Intersección general: Se da cuando dos segmentos se cortan entre sí sin que ningún punto de cualquiera de los segmentos se encuentre sobre el otro.
    - Intersección especial: Se da cuando un punto de un segmento se encuentra sobre el otro.
- Supónganse dos viajes con las siguientes denominaciones y características:
    - Viaje 1
        - Nodo de partida: p1, con coordenadas XY
        - Nodo destino: q1, con coordenadas XY
    - Viaje 2
        - Nodo de partida: p2, con coordenadas XY
        - Nodo destino: q2, con coordenadas XY

- Defínanse 4 tripletes ordenados de puntos:
    - Triplete 1: (p1, q1, p2)
    - Triplete 2: (p1, q1, q2)
    - Triplete 3: (p2, q2, p1)
    - Triplete 4: (p2, q2, q1)
- Para cada triplete (A, B, C), calcúlese la orientación definida por los dos vectores AB y BC, clasificándola entre:
    - Contraria al sentido de las agujas de un reloj
    - Favorable a dicho sentido
    - Vectores colineales
- Comparar entre sí las orientaciones de los tripletes 1 y 2 y las de los tripletes 3 y 4. Si las orientaciones de los dos primeros o de los dos últimos tripletes son distintas, existe intersección general.
- Si no existe intersección general, para cada triplete, calcúlese si alguno de los puntos de un viaje puede coincidir exactamente en el plano XY sobre el segmento trazable entre los nodos de partida y destino del otro viaje.
- Si se da el hecho de que la orientación de un triplete es colineal y además se cumple la condición del punto anterior, existe intersección especial.
- En el caso de que nada de lo anterior se haya cumplido, no existe intersección.
- A continuación, se comprueba el plano temporal, teniendo en cuenta en qué instante salen los drones de sus nodos de partida, la velocidad crucero y la distancia entre el nodo de partida y el punto de intersección.
- En el caso de que ambos drones pasen por el punto de intersección con menos margen del establecido en los parámetros, se establece que existe riesgo de intersección. En caso contrario, el análisis termina.
- Si existe riesgo de intersección, se calculan las velocidades mínimas a las que podrían viajar los drones, teniendo en cuenta con cuánto tiempo de margen llegan al siguiente nodo. Los drones que viajan a o desde una estación siempre lo hacen a velocidad crucero, por lo que ese es el valor de su velocidad mínima.
- Entonces, se asigna su velocidad mínima a uno de los drones y la máxima al otro para comprobar si en esta situación se puede eliminar el riesgo de intersección. Si el riesgo persiste, se realiza esta misma comprobación, al contrario, asignando su velocidad máxima al primer dron y su mínima al segundo.
- Si ninguna de las dos comprobaciones del punto anterior funciona, la intersección se define como cruce y tiene un peso en la toma de decisiones del problema. Si una de ellas funciona, se hace notar en la consola y la intersección es ignorada en dicha toma de decisiones por ser fácilmente resoluble, si bien es anotada también.

6 EXPERIMENTACIÓN

En este capítulo presentamos la experimentación que demuestra la utilidad de la herramienta desarrollada en la generación de soluciones diversas para la cobertura de escenarios compuestos por una variedad de puntos de interés. Para la disposición de los puntos de interés se acude a instancias de benchmarking de VRP, con las que se consiguen instancias que ofrecen cierta dureza. En particular, a través de la web CVRPLIB hemos experimentado con el set de (Augerat, 1995) y con el de (Uchoa, et al., 2014).

En nuestro caso, acudimos a estas instancias ya contrastadas porque nos procuran una distribución pseudoaleatoria de los nodos. Sobre ellos aplicamos un pre procesado para asociar una sucesión de nodos como waypoints del target en cuestión, intentando evitar agrupaciones según ningún criterio geométrico (la ubicación en el plano XY de los mismos).

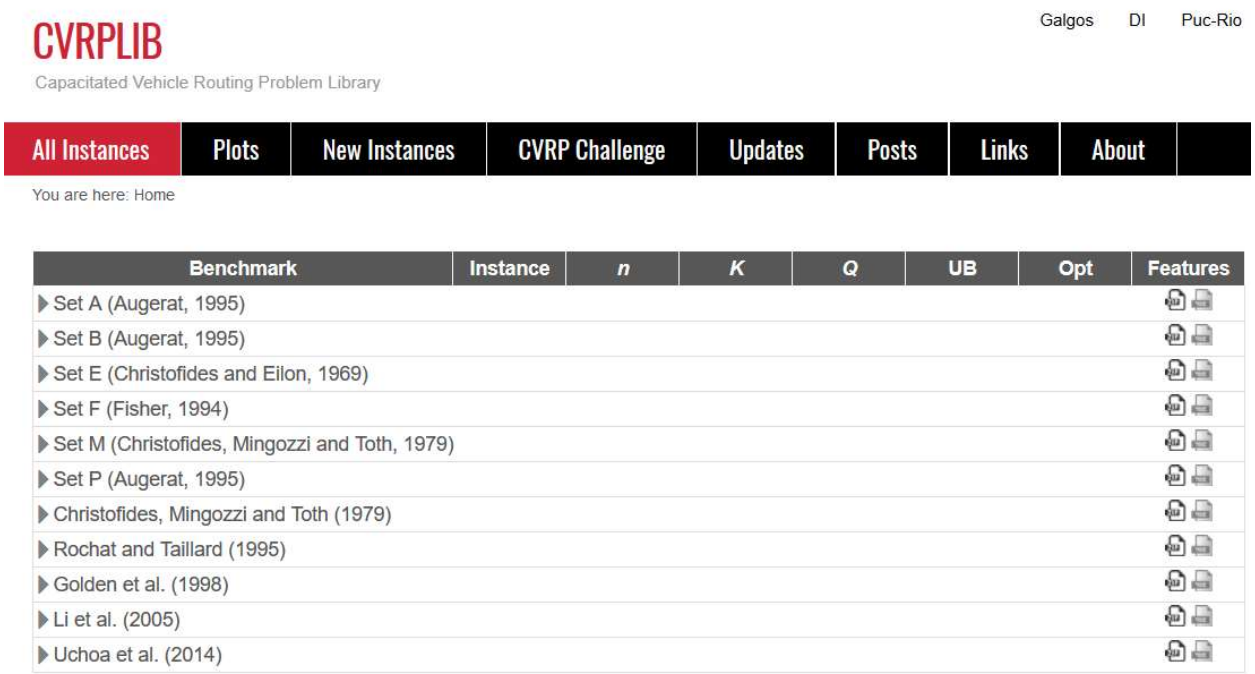

Figura 6-1: Página principal de CVRPLIB (Xavier, 2014)

A continuación, presentamos nuestra experiencia abordando escenarios obtenidos tras el pre procesado de las instancias A-n45-k6 y A-n80-k10 el Set A (Augerat, 1995), y de la instancia X-n101-k25 del set de los mismos creadores de la web CVRPLIB (Uchoa, et al., 2014).

6.1 Experimento 1: Escenario derivado de A-n45-k6

La instancia CVRP A-n45-k6.vrp de (Augerat, 1995) presenta las siguientes características:

- Número de clientes (n): 44
- Número mínimo de vehículos (K): 6
- Capacidad (Q): 100
- Cota superior (UB): 944
- Referencia: (Augerat, 1995)
- Demanda: [0,24]
- Distancia: Euclídea, 2D
- Posicionado de depot: Aleatorio
- Posicionado de clientes: Aleatorio

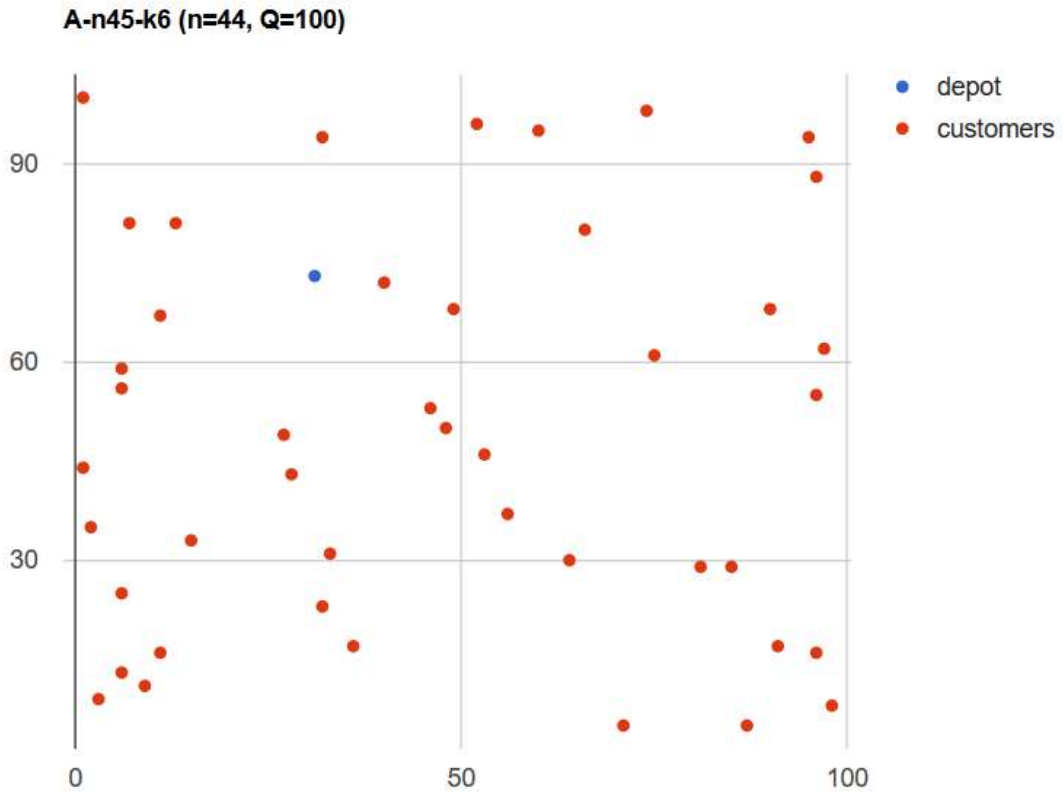

Figura 6-2: Representación de instancia CVRP "A-n45-k6.vrp" (Xavier, 2014)

- Si se considera el kilómetro como unidad de distancia, como puede observarse en la Figura 6-2 , los nodos a visitar ocuparían un área mucho mayor de los 40 kilómetros cuadrados. También hay algunos datos, como la demanda, que no serían útiles para el problema, y otros necesarios, como los tiempos, que no existen en el CVRP. Por ello, sobre dicha instancia se han realizado las siguientes modificaciones:
- Adaptación del área ocupada por los nodos a 40 kilómetros cuadrados.
- Definiciones:
  - Número de estaciones = 5
  - Número de clientes = 5
  - Número de drones = 9
  - Autonomía inicial = 160 minutos
  - Autonomía máxima = 240 minutos
  - Localización inicial de los drones = (6,7,8,9, … ,14)
  - Velocidad crucero = 10m/s = 0.6km/min
  - Velocidad máxima = 15m/s = 0.9km/min
  - Consumo parado = 1min/min
  - Consumo en movimiento = 1.5min/min
  - Tiempo de recarga = 10min

- Rutas de Targets:
  - Target 1: (6, 7, 8, 9, 10, 11, 12, 13)
  - Target 2: (14, 15, 16, 17, 18, 29, 20, 21)
  - Target 3: (22, 23, 24, 25, 26, 27, 28, 29)
  - Target 4: (30, 31, 32, 33, 34, 35, 36, 37)
  - Target 5: (38, 39, 40, 41, 42, 43, 44, 45)
- Representación:

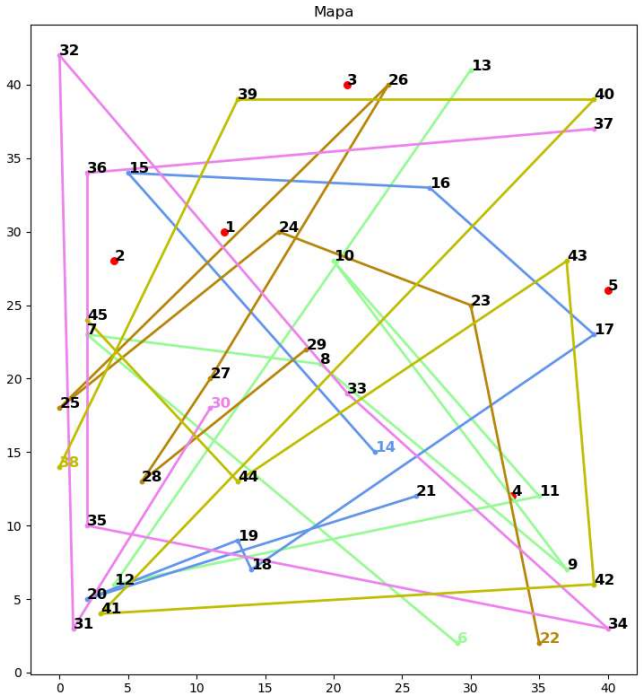

Figura 6-3: Representación de rutas de los targets en la primera experimentación

- Tiempos mínimos para visita de Waypoints:

Target	Waypoint	Comienzo (min)
1	6	0
	7	67
	8	105
	9	153
	10	208
	11	254
	12	316
	13	398
2	14	0
	15	53
	16	99
	17	135
	18	194
	19	207
	20	236
	21	287
3	22	0
	23	49
	24	83
	25	126
	26	190
	27	239
	28	263
	29	298
4	30	0
	31	40
	32	115
	33	176
	34	227
	35	301
	36	351
	37	422
5	38	0
	39	56
	40	109
	41	202
	42	272
	43	318
	44	375
	45	410

Tabla 6-1: Tiempos mínimos para visita de waypoints en primera experimentación

6.1.1 Exp1: Primera iteración

Para la primera iteración (véase Figura 5-2 para el significado de iteración en nuestra mecánica de trabajo), se ha decidido cubrir todo el rango de posibles valores de w_1 y un gran rango de w_2, para tratar de encontrar aquellos intervalos en los que se encuentran buenas soluciones.

- Rango de w_1: de 0 a 1, cubierto en pasos de 0.1
- Rango de w_2: de 0 a 1000, cubierto en pasos de 100
- Iteraciones: 100

Mejor solución:

- $w_1 = 0.2$
- $w_2 = 800$
- 10 nodos sin servir
- 0 cruces
- 1374.93 kilómetros recorridos
- 14.02 segundos de resolución

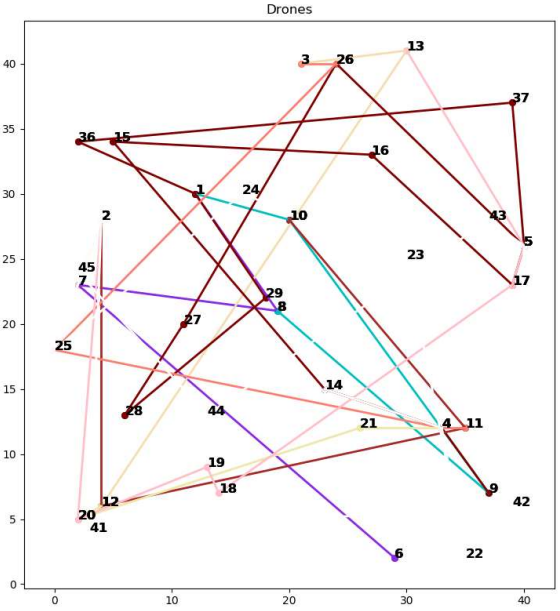

Figura 6-4: Primer experimento: Mapa de rutas de drones en primera iteración

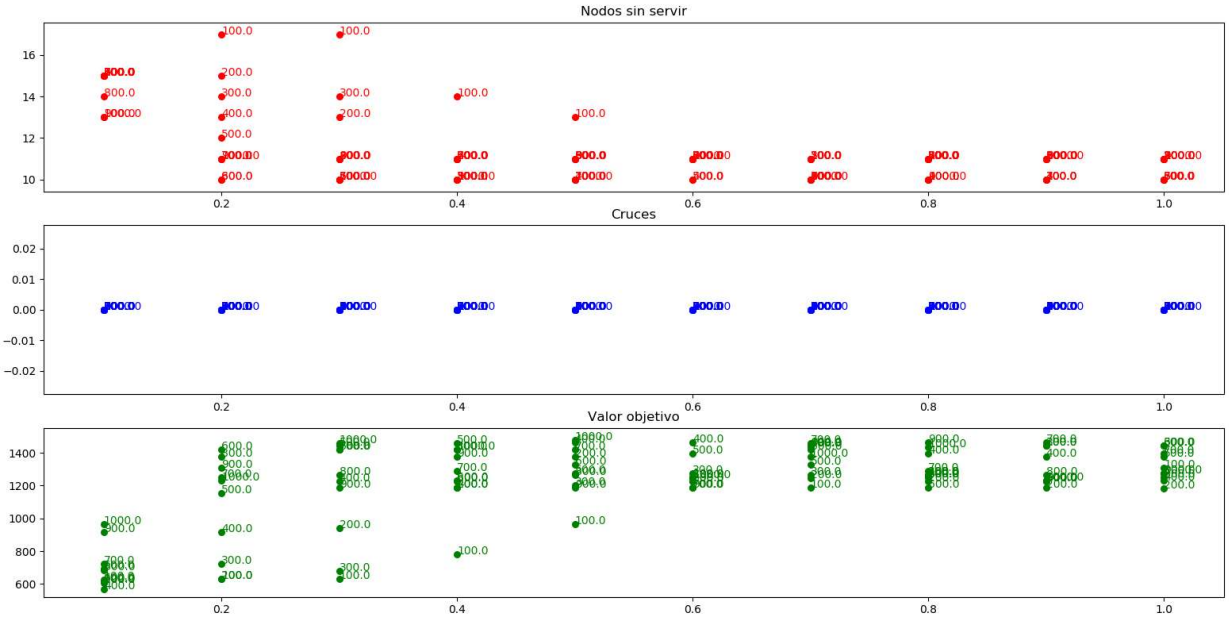

Figura 6-5: Primer experimento: Gráfica de resultados en primera iteración

6.1.2 Exp1: Segunda iteración

Para la segunda iteración, se ha decidido cubrir un rango de posibles valores de w_1 y w_2 más cerrado, para tratar de acotar aquellos intervalos en los que se encuentran las mejores soluciones.

- Rango de w_1: de 0 a 0.5, cubierto en pasos de 0.05
- Rango de w_2: de 0 a 1000, cubierto en pasos de 100
- Iteraciones: 100

La mejor solución es similar a la anterior, de lo que se sacarán conclusiones más adelante:

- $w_1 = 0.15$
- $w_2 = 800$
- 10 nodos sin servir
- 0 cruces
- 1374.93 kilómetros recorridos
- 24.68 segundos de resolución

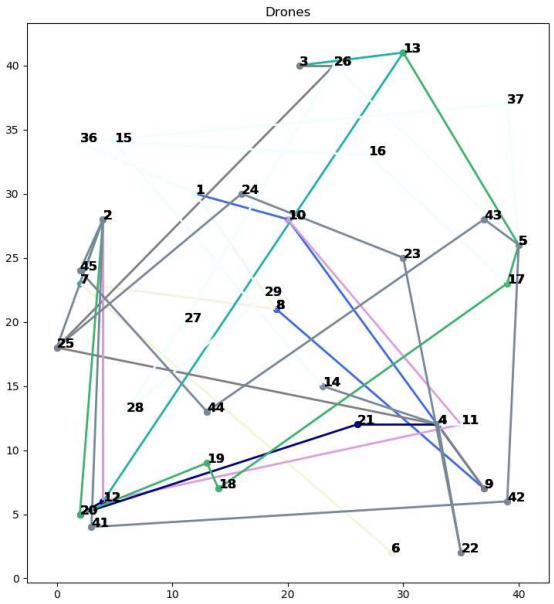

Figura 6-6: Primer experimento: Mapa de rutas de drones en segunda iteración

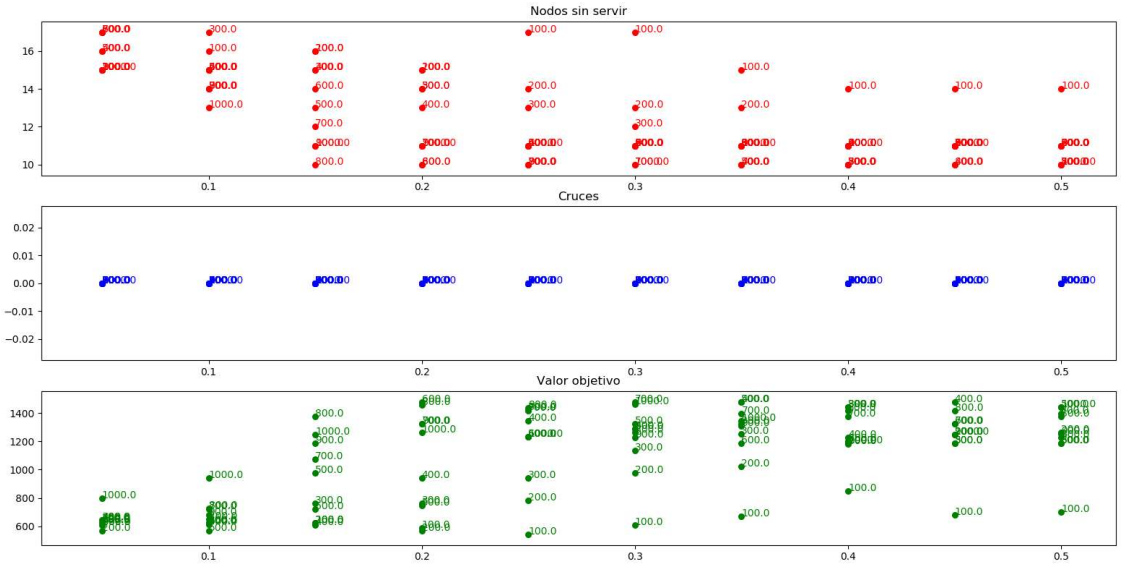

Figura 6-7: Primer experimento: Gráfica de resultados en segunda iteración

De lo anterior se deduce que el número pequeño de waypoints no da la suficiente variabilidad como para propiciar diversidad en las soluciones que el Scheduler genera en las sucesivas iteraciones. En los siguientes experimentos se ilustra como ganamos en diversidad al aumentar el factor de escala de la instancia en uso.

6.2 Experimento 2: Escenario derivado de A-n80-k10

Dado que hemos observado que para un número pequeño de waypoints existe poca variabilidad y alta velocidad en el uso del Scheduler, para llevar a cabo la segunda experimentación se ha utilizado la instancia CVRP A-n80-k10.vrp de (Xavier, 2014) generada por (Augerat, 1995). Esta instancia, según su web, posee las siguientes características:

- Número de clientes (n): 79
- Número mínimo de vehículos (K): 10
- Capacidad (Q): 100
- Cota superior (UB): 1763
- Referencia: (Augerat, 1995)
- Demanda: [0,26]
- Distancia: Euclídea, 2D
- Posicionado de depot: Aleatorio
- Posicionado de clientes: Aleatorio

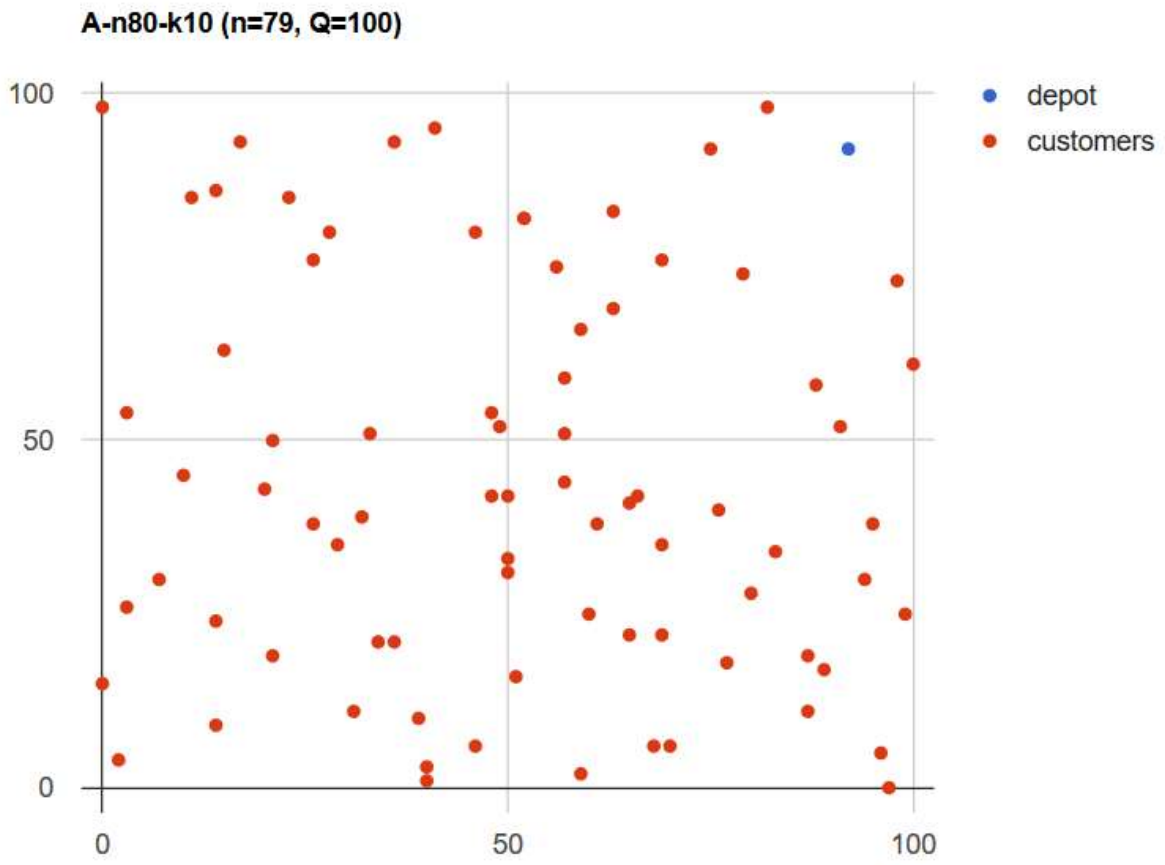

Figura 6-8: Representación de instancia CVRP "A-n80-k10.vrp" (Xavier, 2014)

Se realizan las mismas adaptaciones que con la instancia anterior:

- Adaptación del área ocupada por los nodos a 40 kilómetros cuadrados.
- Definiciones:
  - Número de estaciones = 5
  - Número de clientes = 8
  - Número de drones = 12
  - Autonomía inicial = 160 minutos
  - Autonomía máxima = 240 minutos
  - Localización inicial de los drones = (6,7,8,9, … ,17)
  - Velocidad crucero = 10m/s = 0.6km/min
  - Velocidad máxima = 15m/s = 0.9km/min
  - Consumo parado = 1min/min
  - Consumo en movimiento = 1.5min/min
  - Tiempo de recarga = 10min

- Rutas de Targets:
  - Target 1: (6, 7, 8, 9, 10, 11, 12, 76)
  - Target 2: (13, 14, 15, 16, 17, 18, 19, 77)
  - Target 3: (20, 21, 22, 23, 24, 25, 26, 78)
  - Target 4: (27, 28, 29, 30, 31, 32, 33, 79)
  - Target 5: (34, 35, 36, 37, 38, 39, 40, 80)
  - Target 6: (41, 42, 43, 44, 45, 46, 47)
  - Target 7: (48, 49, 50, 51, 52, 53, 54)
  - Target 8: (55, 56, 57, 58, 59, 60, 61)
  - Target 9: (62, 63, 64, 65, 66, 67, 68)
  - Target 10: (69, 70, 71, 72, 73, 74, 75)

- Representación:

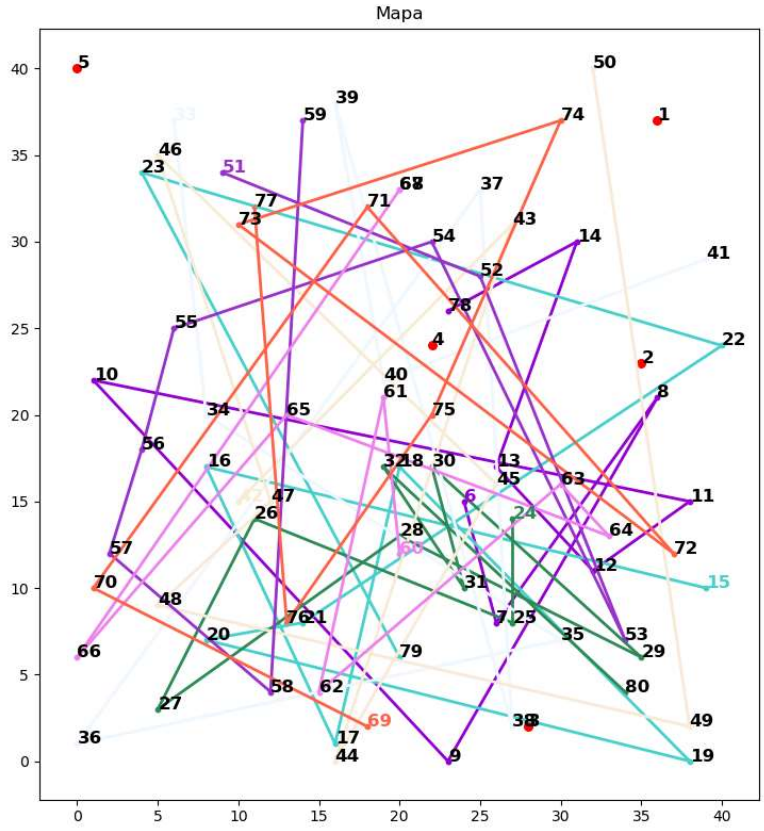

Figura 6-9: Representación de rutas de los targets en la segunda experimentación

- Tiempos mínimos para visita de Waypoints:

Target	Waypoint	Comienzo (min)
1	6	0
	7	22
	8	59
	9	110
	10	171
	11	243
	12	265
	76	307
2	13	0
	14	33
	15	78
	16	140
	17	179
	18	216
	19	267
	77	346
3	20	0
	21	20
	22	80
	23	152
	24	212
	25	232
	26	270
	78	308
4	27	0
	28	40
	29	77
	30	115
	31	137
	32	161
	33	210
	79	276
5	34	0
	35	52
	36	112
	37	189
	38	250
	39	322
	40	359
	80	408
6	41	0
	42	63
	43	111
	44	175
	45	216
	46	273
	47	318
7	48	0
	49	66
	50	140
	51	189
	52	227
	53	275
	54	328
8	55	0
	56	22
	57	42
	58	73
	59	138
	60	190
	61	215
9	62	0
	63	42
	64	59
	65	104
	66	145
	67	211
	68	221
10	69	0
	70	41
	71	97
	72	152
	73	217
	74	261
	75	302

Tabla 6-2: Tiempos mínimos para visita de waypoints en segunda experimentación

6.2.1 Exp2: Primera iteración

Para la primera iteración, se ha decidido cubrir todo el rango de posibles valores de w_1 y un gran rango de w_2, para tratar de encontrar aquellos intervalos en los que se encuentran buenas soluciones.

- Rango de w_1: de 0 a 1, cubierto en pasos de 0.1
- Rango de w_2: de 0 a 1000, cubierto en pasos de 100
- Iteraciones: 100

Mejor solución:

- $w_1 = 0.6$
- $w_2 = 300$
- 26 nodos sin servir
- 1 cruces
- 2221 kilómetros recorridos
- 22.29 segundos de resolución

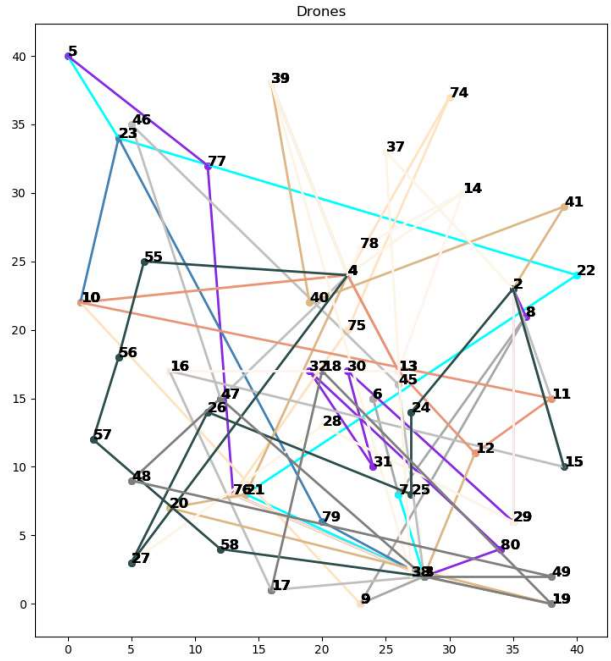

Figura 6-10: Segundo experimento: Mapa de rutas de drones en primera iteración

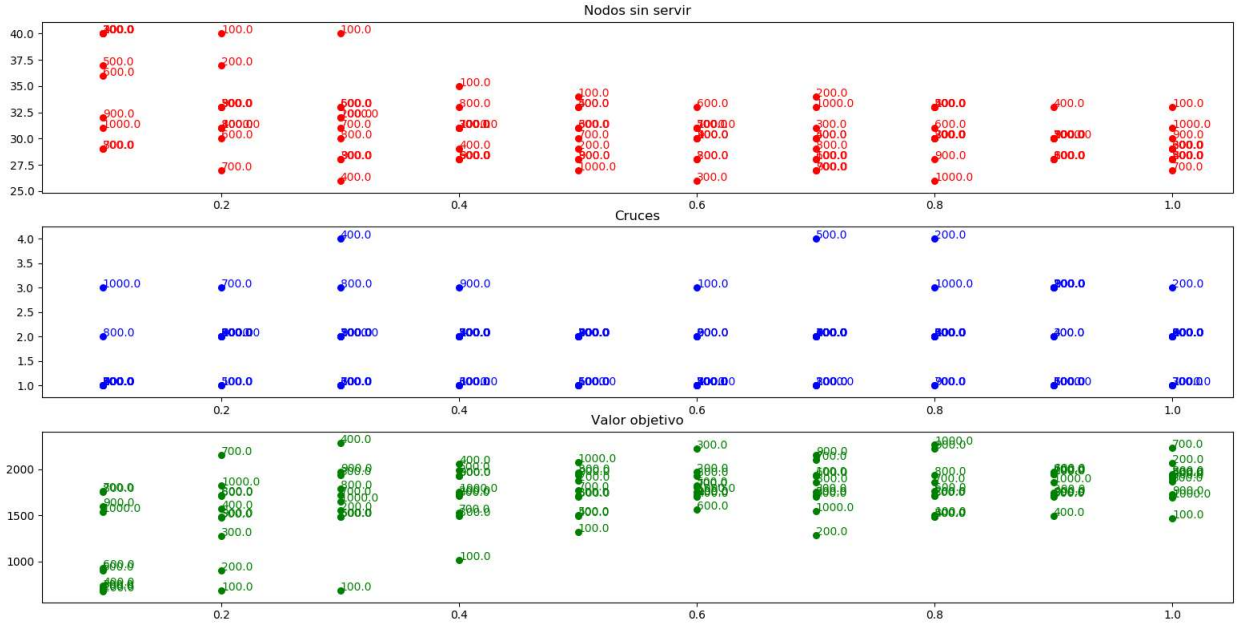

Figura 6-11: Segundo experimento: Gráfica de resultados en primera iteración

6.2.2 Exp2: Segunda iteración

Para la segunda iteración, se ha decidido cubrir un rango de posibles valores de w_1 y w_2 más cerrado, para tratar de acotar aquellos intervalos en los que se encuentran las mejores soluciones.

- Rango de w_1: de 0.1 a 0.9, cubierto en pasos de 0.08
- Rango de w_2: de 0 a 1000, cubierto en pasos de 100
- Iteraciones: 100

Nueva mejor solución:

- $w_1 = 0.66$
- $w_2 = 800$
- 25 nodos sin servir
- 1 cruces
- 2441.92 kilómetros recorridos
- 22.41 segundos de resolución

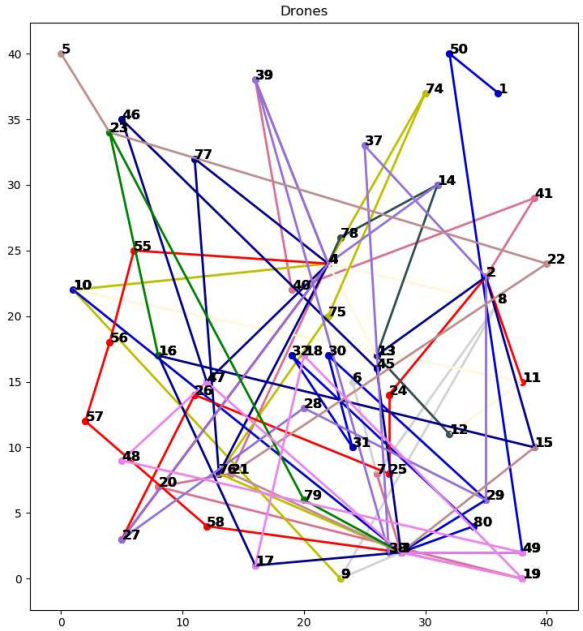

Figura 6-12: Segundo experimento: Mapa de rutas de drones en segunda iteración

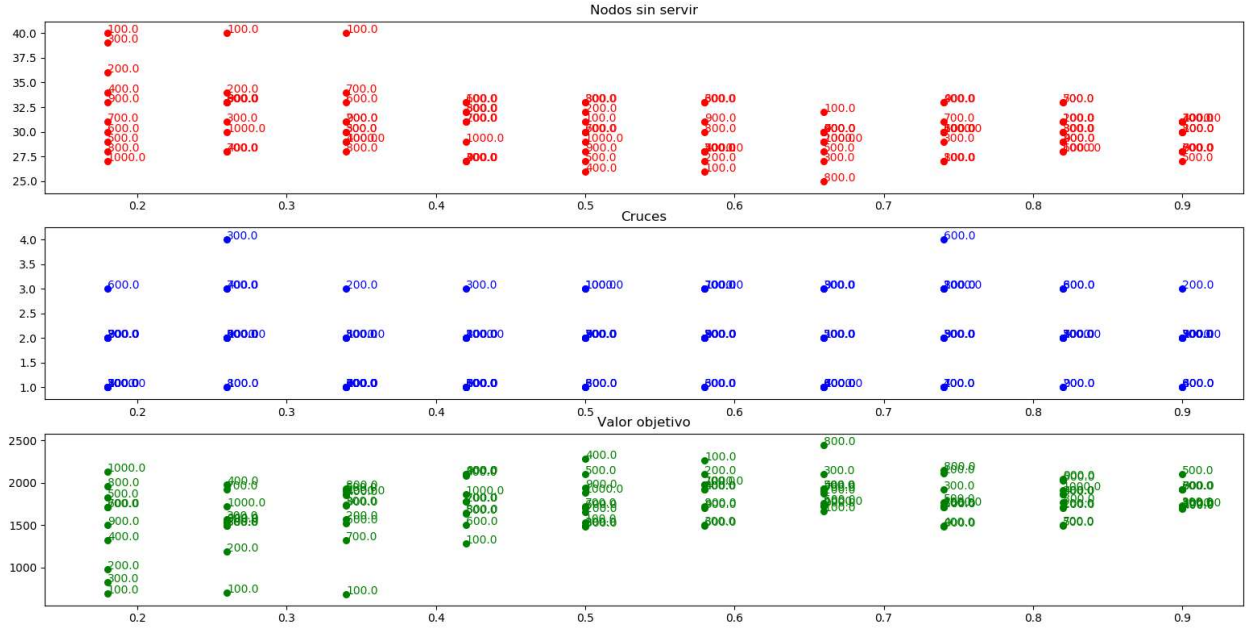

Figura 6-13: Segundo experimento: Gráfica de resultados en segunda iteración

6.3 Experimento 3: Escenario derivado de X-n101-k25

Para llevar a cabo la última experimentación, se ha utilizado la instancia CVRP X-n101-k25.vrp de (Xavier, 2014) generada por (Uchoa, et al., 2014). Esta instancia, según su web, posee las siguientes características:

- Número de clientes (n): 100
- Número mínimo de vehículos (K): 25
- Capacidad (Q): 206
- Cota superior (UB): 27591
- Referencia: (Uchoa, et al., 2014)
- Demanda: [0,100]
- Distancia: Euclídea, 2D
- Posicionado de depot: Aleatorio
- Posicionado de clientes: Aleatorio

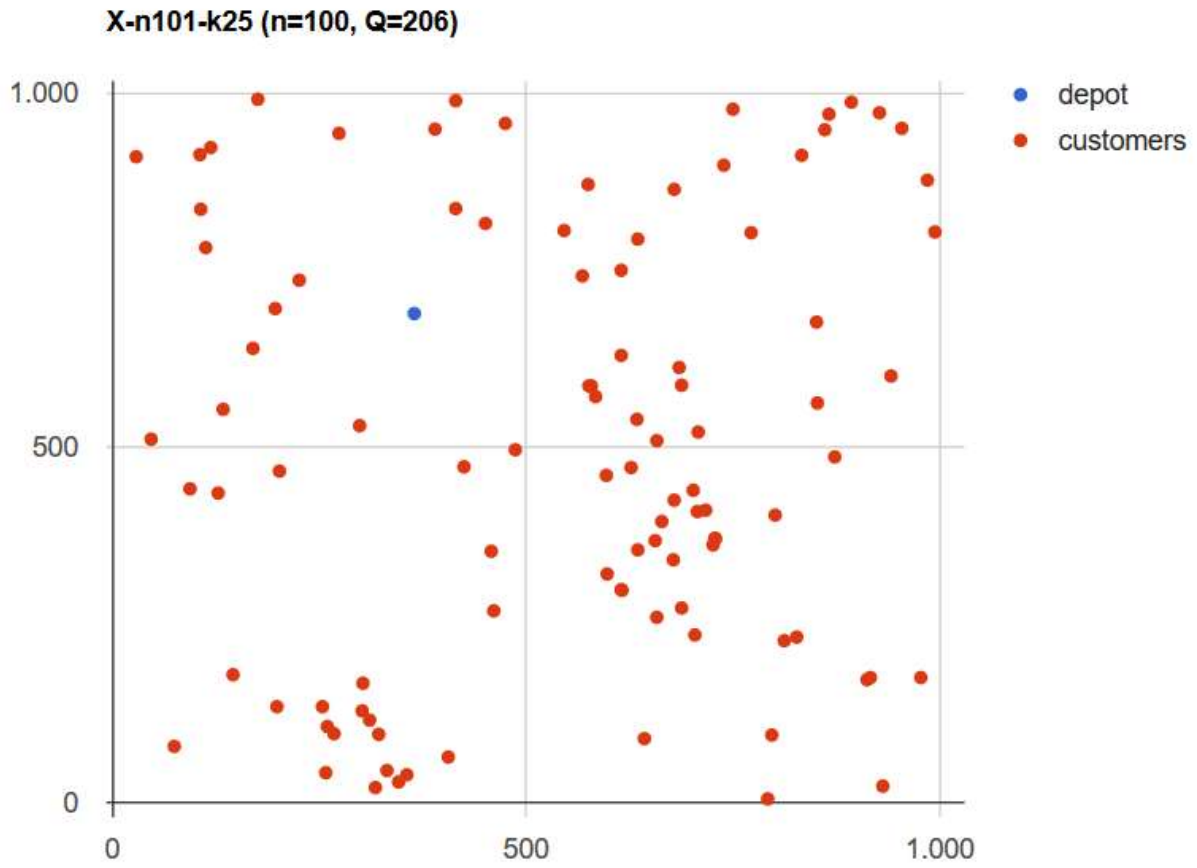

Figura 6-14: Representación de instancia CVRP "X-n101-k25.vrp" (Xavier, 2014)

Se realizan las mismas adaptaciones que con la instancia anterior:

- Adaptación del área ocupada por los nodos a 40 kilómetros cuadrados.
- Definiciones:
  - Número de estaciones = 5
  - Número de clientes = 10
  - Número de drones = 18
  - Autonomía inicial = 160 minutos
  - Autonomía máxima = 240 minutos
  - Localización inicial de los drones = (6,7,8,9, … ,23)
  - Velocidad crucero = 10m/s = 0.6km/min
  - Velocidad máxima = 15m/s = 0.9km/min
  - Consumo parado = 1min/min
  - Consumo en movimiento = 1.5min/min
  - Tiempo de recarga = 10min

- Rutas de Targets:
  - Target 1: (6, 7, 8, 9, 10, 11, 12, 13, 14, 96)
  - Target 2: (15, 16, 17, 18, 19, 20, 21, 22, 23, 97)
  - Target 3: (24, 25, 26, 27, 28, 29, 30, 31, 32, 98)
  - Target 4: (33, 34, 35, 36, 37, 38, 39, 40, 41, 99)
  - Target 5: (42, 43, 44, 45, 46, 47, 48, 49, 50, 100)
  - Target 6: (51, 52, 53, 54, 55, 56, 57, 58, 59, 101)
  - Target 7: (60, 61, 62, 63, 64, 65, 66, 67, 68)
  - Target 8: (69, 70, 71, 72, 73, 74, 75, 76, 77)
  - Target 9: (78, 79, 80, 81, 82, 83, 84, 85, 86)
  - Target 10: (87, 88, 89, 90, 91, 92, 93, 94, 95)

- Representación:

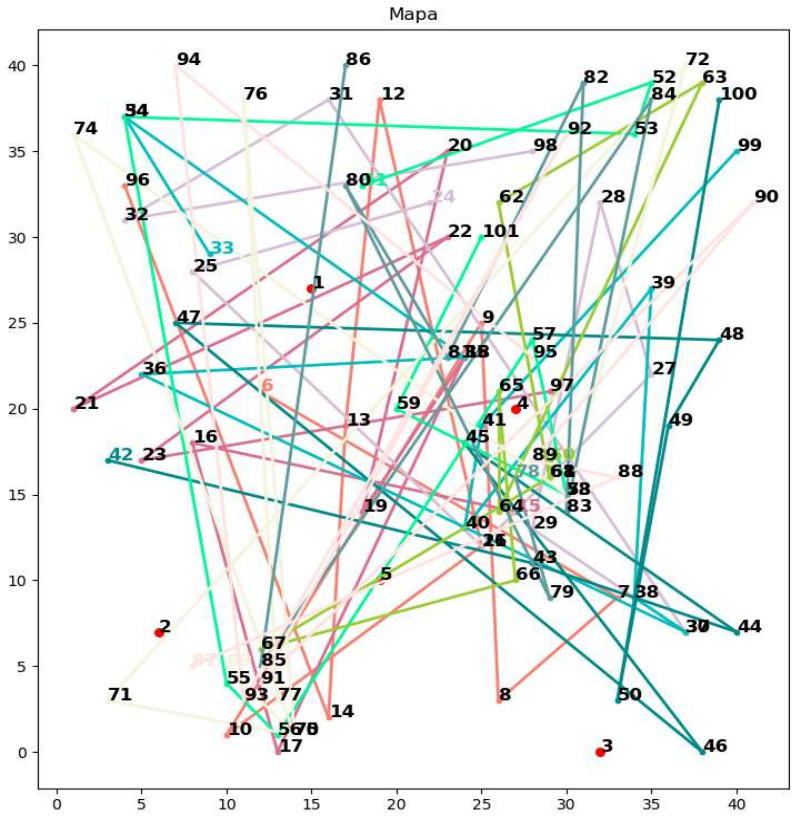

Figura 6-15: Representación de rutas de los targets en la última experimentación

- Tiempos mínimos para visita de Waypoints:

Target	Waypoint	Comienzo (min)
1	6	0
	7	50
	8	75
	9	121
	10	178
	11	219
	12	273
	13	314
	14	352
	96	417
2	15	0
	16	42
	17	83
	18	135
	19	163
	20	208
	21	262
	22	312
	23	359
	97	409
3	24	0
	25	34
	26	82
	27	115
	28	142
	29	184
	30	212
	31	284
	32	317
	98	367
4	33	0
	34	25
	35	75
	36	116
	37	184
	38	200
	39	240
	40	279
	41	299
	99	345
5	42	0
	43	52
	44	83
	45	125
	46	173
	47	249
	48	312
	49	331
	50	368
	100	437

Target	Waypoint	Comienzo (min)
6	51	0
	52	40
	53	55
	54	115
	55	180
	56	197
	57	252
	58	277
	59	305
	101	333
7	60	0
	61	11
	62	48
	63	81
	64	137
	65	158
	66	186
	67	221
	68	263
8	69	0
	70	19
	71	47
	72	140
	73	193
	74	262
	75	334
	76	405
	77	473
9	78	0
	79	22
	80	76
	81	105
	82	144
	83	195
	84	245
	85	322
	86	390
10	87	0
	88	55
	89	73
	90	116
	91	193
	92	264
	93	337
	94	409
	95	464

Tabla 6-3: Tiempos mínimos para visita de waypoints en la última experimentación

6.3.1 Exp3: Primera iteración

Para la primera iteración, se ha decidido cubrir todo el rango de posibles valores de w_1 y un gran rango de w_2, para tratar de encontrar aquellos intervalos en los que se encuentran buenas soluciones.

- Rango de w_1: de 0 a 1, cubierto en pasos de 0.1
- Rango de w_2: de 0 a 1000, cubierto en pasos de 100
- Iteraciones: 100

Como resultado, se obtiene un mapa de rutas de drones. Adicionalmente, se imprime una gráfica que nos habla de las propiedades de las soluciones calculadas, comparando número de nodos sin servir, posibles cruces entre trayectorias de drones y la distancia total recorrida (valor objetivo). La mejor solución corresponde a unos valores de $w_1 = 0.7$ y $w_2 = 800$, con 30 nodos sin servir, 3 cruces y 2329 kilómetros recorridos, tardando el programa 48.69 segundos en resolverse en un Intel I7-7700HQ @ 2,8GHz con 16 GB de RAM:

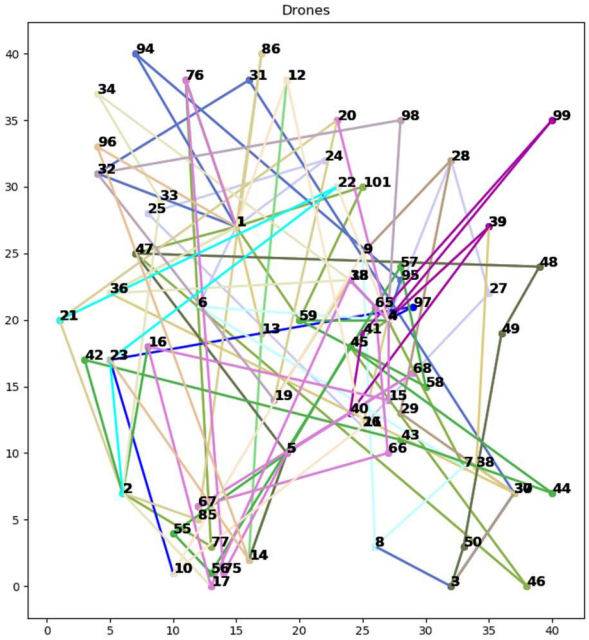

Figura 6-16: Último experimento: Mapa de rutas de drones en primera iteración

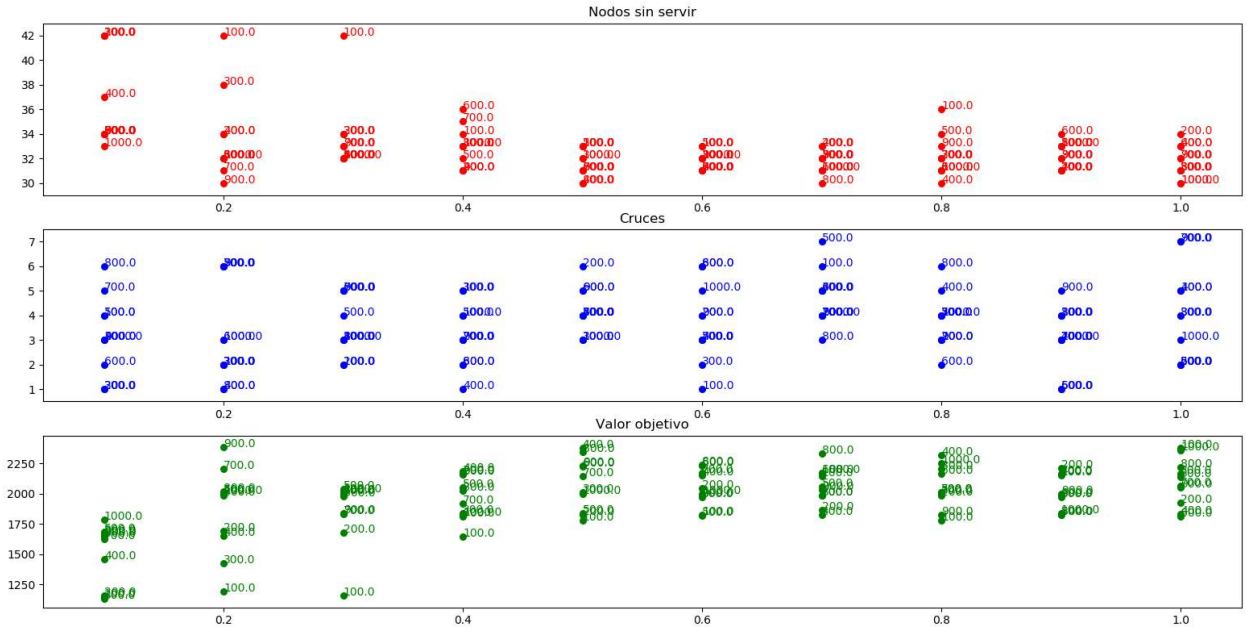

Figura 6-17: Último experimento: Gráfica de resultados en primera iteración

6.3.2 Exp3: Segunda iteración

Para la segunda iteración, se ha decidido cubrir un rango de posibles valores de w_1 y w_2 más cerrado, para tratar de acotar aquellos intervalos en los que se encuentran las mejores soluciones.

- Rango de w_1: de 0.5 a 1, cubierto en pasos de 0.05
- Rango de w_2: de 0 a 1000, cubierto en pasos de 100
- Iteraciones: 100

Se obtiene un nuevo mapa de rutas de drones y una nueva gráfica. La mejor solución corresponde ahora a unos valores de $w_1 = 0{,}75$ y $w_2 = 600$, y tiene 30 nodos sin servir, 2 cruces y 2335 kilómetros recorridos, tardando 2 segundos menos en resolverse:

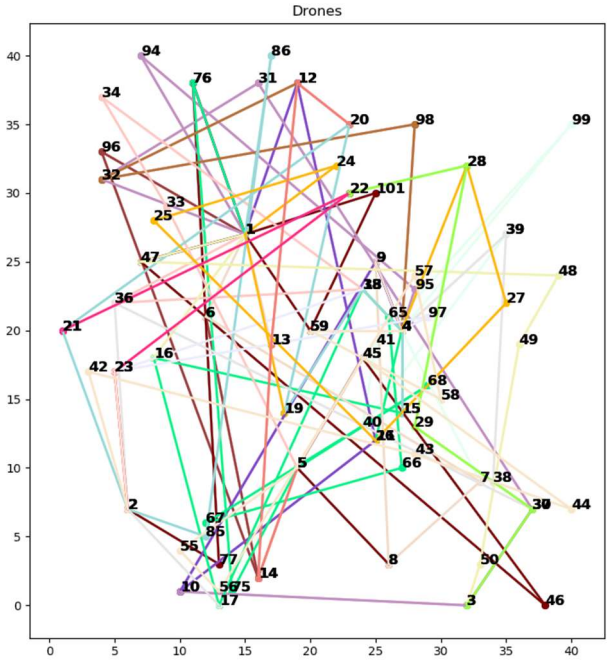

Figura 6-18: Último experimento: Mapa de rutas de drones en segunda iteración

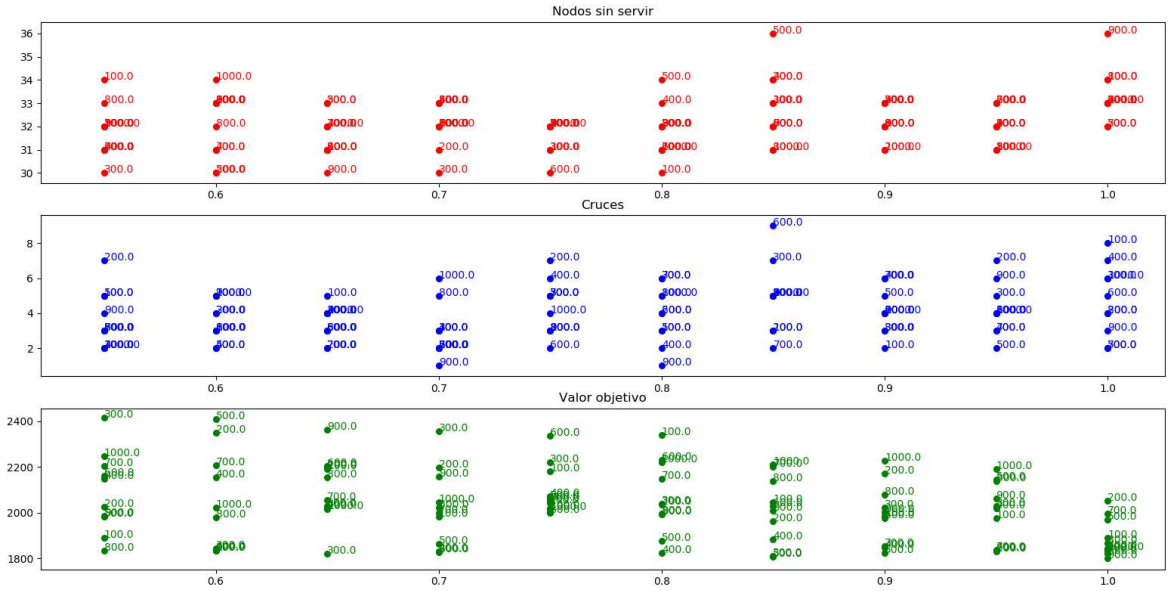

Figura 6-19: Último experimento: Gráfica de resultados en segunda iteración

Del análisis de la Figura 6-7 apreciamos que puede ser interesantes afinar aún más en la búsqueda de soluciones de planificación de la flota de drones en un rango de valores de w_1 y w_2 aún más cerrado, y ahí buscar soluciones haciendo uso de nuevo del Scheduler.

6.3.3 Exp3: Tercera iteración

Realizamos una tercera (y última) iteración caracterizada por:

- Rango de w_1: de 0.5 a 0.8, cubierto en pasos de 0.025
- Rango de w_2: de 0 a 1000, cubierto en pasos de 50
- Iteraciones: 240

Se obtiene un nuevo mapa de rutas de drones y una nueva gráfica. La mejor solución corresponde ahora a unos valores de $w_1 = 0{,}575$ y $w_2 = 600$, y tiene 30 nodos sin servir, 2 cruces y 2330 kilómetros recorridos, tardando 109.39 segundos en resolverse:

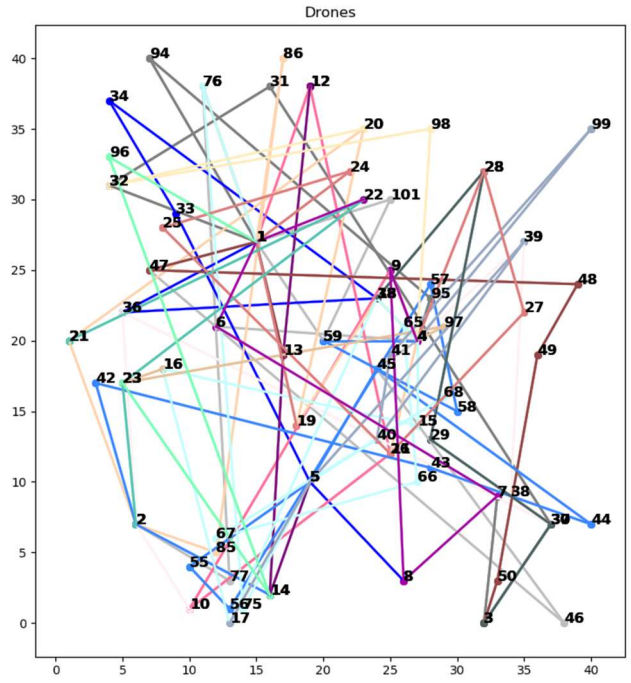

Figura 6-20: Último experimento: Mapa de rutas de drones en tercera y última iteración

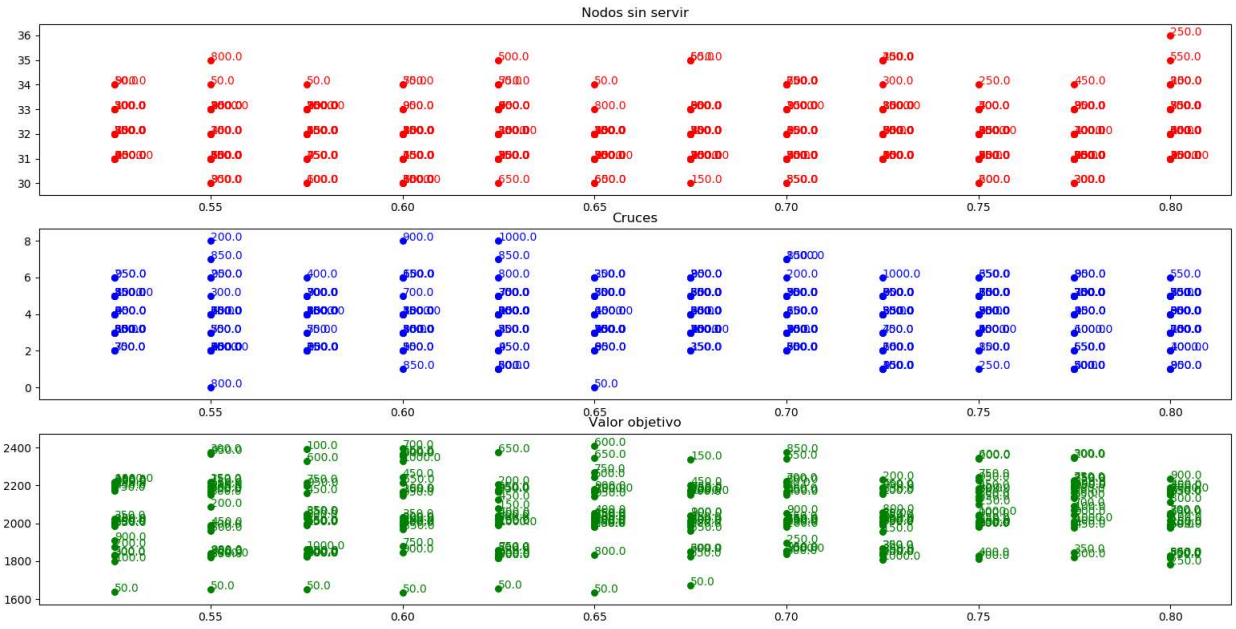

Figura 6-21: Último experimento: Gráfica de resultados en tercera y última iteración

6.4 Análisis de la experimentación

Como puede observarse, para valores de entre 0,5 y 0,8 para w_1 y entre 0 y 1000 para w_2 se consiguen una gran variedad de resultados con implicaciones bien distintas entre ellos pero que, en función de las preferencias del usuario de Scheduler, pueden ser igualmente útiles.

Por otro lado, puede concluirse que un aumento significativo del número de iteraciones no implica necesariamente una mejora en la mejor solución. En el caso del experimento 3, que es en el que más iteraciones se usaron, de la segunda iteración a la tercera y última tan sólo se obtiene mejora de 5 kilómetros menos en la distancia volada por el total de la flota.

Aunque se definió una mecánica de trabajo en la que es posible realizar varias invocaciones del núcleo heurístico desarrollado, para poder ir hacia exploraciones más detalladas de rangos diversos de los parámetros de optimización, nuestra experimentación muestra que puede no ser necesario abordar muchas baterías antes de que emerjan las soluciones más equilibradas al problema de planificación coordinada de la flota de drones. Como se pretendía, en cada uno de los tres experimentos Scheduler ha permitido acotar un rango de valores para los parámetros que modulan el criterio de optimización.

Por último, se demuestra que la herramienta permite que el usuario la pueda invocar una sola vez (facilidad de uso) con un intervalo amplio en los parámetros de ajuste. Sin más interacción con el software, un solo ensayo (iteración) hace perfectamente viable la generación de soluciones de gran calidad.

7 CONCLUSIONES

El grado de madurez de la tecnología de UAV, con autonomías considerables en base a electrónicas de bajo coste, permite que estos equipos cada vez más se incorporen a la producción audiovisual. En este proyecto se ha presentado un planificador para la coordinación de un equipo de drones que trabajan de forma coordinada en la grabación de una serie de objetivos (targets) en sus desplazamientos previstos en un área extensa. El responsable de la definición de la cobertura audiovisual de dichos targets dispone a la entrada de los tiempos de llegada mínimos en que algún dron de la flota debe estar cubriendo cada uno de los puntos en que se discretiza la trayectoria de los targets.

Para el soporte a las decisiones de planificación, se han planteado objetivos específicos que han sido abordados consecutivamente. Primero, se realizó una revisión de la literatura de la que se decide que es necesario abordar la gestión dinámica de los recursos (drones), asumiendo que el problema inherente se relaciona en gran medida con un problema de enrutado de vehículos con múltiples viajes (MTVRP). A continuación, un repaso detallado a los requisitos de nuestro caso real lleva a identificar el problema de programación de tareas en la que los recursos que las sirven son precisamente los drones en la flota.

Dada la dureza del problema de optimización real (hemos presentado un MILP que modela nuestras decisiones, que se puede reducir a un CVRP y que por tanto es NP-hard), se ha desarrollado un algoritmo heurístico capaz de resolver el problema de forma aproximada en un corto espacio de tiempo. Este algoritmo supone una contribución no sólo por dar respuesta a la necesidad planteada (considerando la idoneidad de los planes propuestos también resolviendo, hasta donde es posible, la problemática de los cruces de trayectoria para facilidad de la operativa en campo) sino porque presenta novedad en relación al estado del arte.

Para el mejor apoyo a las decisiones de la cobertura audiovisual objeto de este proyecto, la heurística se ha incrustado en un procedimiento capaz de lanzar baterías de problemas ensayando ajustes de la función objetivo que balancean los diferentes criterios de utilidad que surgieron en la etapa de captura de requisitos: cobertura de puntos de interés, distancia total volada y número de cruces de trayectorias con riesgo de colisión.

La mecánica de trabajo identificada se ha implementado en una aplicación Python. Es una aplicación que genera representación gráfica de soluciones, además de todo el almacenamiento de su traza de ejecución. Esto ha permitido revisar en detalle la corrección del funcionamiento de construcción de soluciones de la heurística. La experimentación ha validado la utilidad de la herramienta para la definición de la estrategia de sincronización de los drones haciendo uso de la infraestructura de despliegue (posiciones iniciales y lugares de recogida) y de los puntos de cambio de batería.

REFERENCIAS

Augerat, P., 1995. *CVRPLIB.* [Online]
Available at: http://vrp.galgos.inf.puc-rio.br/index.php/en/plotted-instances?data=A-n80-k10
[Accessed 1 Noviembre 2018].

Booch, G., Rumbaugh, J. & Jacobson, I., 1998. *Unified Modeling Language User Guide.* Reading: Addison Wesley.

Cattaruzza, D., Absi, N. & Feillet, D., 2018. Vehicle routing problems with multiple trips. *Annals of Operations Research,* Volume 271, pp. 127-159.

Chaimatanan, S., Delahaye, D. & Mongeau, M., 2015. Aircraft 4D trajectories planning under uncertainties. *IEEE Symposium Series on Computational Intelligence.*

Chaudhry, M. A. & Khan, A. A., 2016. A research survey: Review of flexible job shop scheduling techniques. *International Transactions in Operational Research,* 23(3), pp. 551-591.

Guerriero, F., Surace, R., Loscri, V. & Natalizio, E., 2014. A multi-objective approach for unmanned aerial vehicle routing problem with soft time windows constraints. *Applied Mathematical Modelling,* pp. 839-852.

He, X., 2007. *A metamodel for the notation of graphical modeling languages.* Beijing, COMPSAC, pp. 219-224.

Holt, J., 2004. *UML for Systems Engineering: Watching the Wheels IET.* s.l.:Institution of Electrical Engineers.

Ivencia, 2018. *Ivencia.* [Online]
Available at: http://www.ivencia.com/
[Accessed 2018 Mayo 8].

Kaddouh, B. Y., Crowther, W. J. & Hollingsworth, P., 2016. Dynamic Resource Allocation for Efficient Sharing of Services from Heterogeneous Autonomous Vehicles. *Journal of Aerospace Information Systems,* pp. 450-474.

Khosiawan, Y., Khalfay, A. & Nielsen, I., 2018. Scheduling unmanned aerial vehicle and automated guided vehicle operations in an indoor manufacturing environment using differential evolution-fused particle swarm optimization. *International Journal of Advanced Robotic Systems.*

Kim, J., Song, B. D. & Morrison, J. R., 2013. On the Scheduling of Systems of UAVs and Fuel Service Stations for Long-Term Mission Fulfillment. *Journal of Intelligent and Robotic Systems,* pp. 347-359.

Mufalli, F., Batta, R. & Nagi, R., 2012. Simultaneous sensor selection and routing of unmanned aerial vehicles for complex mission plans. *Computers & Operations Research,* pp. 2787-2799.

Murray, C. C. & Karwan, M. . H., 2010. An Extensible Modeling Framework for Dynamic Reassignment and Rerouting in Cooperative Airborne Operations. *Wiley Online Library.*

Nigam, N., Bieniawski, S., Kroo, I. & Vian, J., 2012. Control of Multiple UAVs for Persistent Surveillance: Algorithm and Flight Test Results. *IEEE Transactions on Control Systems Technology,* pp. 1236-1251.

Oh, H. et al., 2014. Behaviour Recognition of Ground Vehicle Using Airborne Monitoring of Unmanned Aerial Vehicles. *International Journal of Systems Science,* pp. 2499-2514.

Olivares, V., Cordova, F., Sepúlveda, J. M. & Derpich, I., 2015. Modeling internal logistics by using drones on the stage of assembly of products. *3rd International Conference on Information Technology and Quantitative Management,* pp. 1240-1249.

Shetty, V. K., Sudit, M. & Nagi, R., 2008. Priority-based assignment and routing of a fleet of unmanned combat aerial vehicles. *Computers and Operations Research,* pp. 1813-1828.

Song, B. D., Kim, J. & Morrison, J. R., 2014. Towards Real Time Scheduling for Persistent UAV Service: A Rolling Horizon MILP Approach, RHTA and the STAH Heuristic. *International Conference on Unmanned Aircraft Systems.*

Song, B. D., Kim, J. & Morrison, J. R., 2016. Rolling Horizon Path Planning of an Autonomous System of UAVs for Persistent Cooperative Service: MILP Formulation and Efficient Heuristics. *Journal of Intelligent and Robotic Systems,* 84(2016), pp. 241-258.

Suzuki, K. A. O., Kemper Filho, P. & Morrison, J. R., 2011. Automatic Battery Replacement System for UAVs: Analysis and Design. *Journal of Intelligent and Robotic Systems.*

Teilans, A., Kleins, A., Merkuryev, Y. & Grinbergs, A., 2008. Design of UML models and their simulation using ARENA. *WSEAS TRANSACTIONS on COMPUTER RESEARCH.*

Uchoa, E. et al., 2014. *New Benchmark Instances for the Capacitated Vehicle Routing Problem.* [Online]
Available at: http://www.optimization-online.org/DB_HTML/2014/10/4597.html
[Accessed 1 Noviembre 2018].

Xavier, I., 2014. *CVRPLIB.* [Online]
Available at: http://vrp.atd-lab.inf.puc-rio.br/index.php/en/
[Accessed November 2018].

Zockoll, G., Scheithauer, A. & Douwe Dekker, M., 2012. *History of Object Oriented Modeling languages,* Delft: s.n.