Sección de Matemáticas
Universidad de La Laguna

Laura Hernández Bethencourt

Diseño de rutas turísticas con intermodalidad

Tourist trip design with intermodal transportation

Trabajo Fin de Grado
Grado en Matemáticas
La Laguna, Julio de 2023

Dirigido por
Julio Antonio Brito Santana

Julio Antonio Brito Santana
Ingeniería Informática y de Sistemas
Universidad de La Laguna
38200 La Laguna, Tenerife

Agradecimientos

A todos esos compañeros de viaje que me han acompañado por el camino.

Laura Hernández Bethencourt
La Laguna, 9 de julio de 2023

Resumen · Abstract

Resumen

La resolución de problemas de rutas turísticas tiene un impacto significativo en el sector turístico ya que ayuda a proporcionar itinerarios atractivos y completamente personalizados a los turistas. Mediante el uso de herramientas matemáticas y técnicas de computación se pueden ofrecer soluciones que maximicen la satisfacción de los turistas. En este proyecto se propone un modelo matemático con sus correspondientes parámetros, variables, objetivos y restricciones. Además se diseña e implementa una metaheurística Ant Colony Optimization para encontrar soluciones a dicho problema. Para la experimentación se utilizan datos reales y se muestran algunos resultados obtenidos con un breve análisis de los mismos.

Palabras clave: *Rutas turísticas – Optimización – Colonia de hormigas – ACO – Problema de orientación – OP – Intermodalidad.*

Abstract

Solving tourist route problems significantly impacts the tourism sector as it helps to provide attractive and fully customized itineraries to tourists. Using mathematical tools within computer science makes it possible to offer solutions that maximize tourist satisfaction. This paper proposes a mathematical model with its corresponding variables, objectives and constraints, and an ACO type heuristic that allows finding solutions to this problem. Real data is used for the experimentation, and some results obtained are shown along with a brief analysis of them.

Keywords: *Touristic route – Optimization – Ant Colony – ACO – Orienteering Problem – OP – Intermodality.*

Contenido

Introducción

El turismo es uno de los sectores que más riqueza aporta a la economía española y, en particular, es uno de los pilares fundamentales de la economía en las Islas Canarias. Su rico patrimonio cultural, sus variados paisajes y su clima favorable son, entre otros muchos factores, los motivos por los que las islas se han convertido en uno de los destinos turísticos más populares a nivel mundial.

En 2020 con el comienzo de la pandemia del COVID-19 y debido a las restricciones a la movilidad impuestas por los gobiernos de los diferentes países del mundo, el sector se llevó un duro golpe. Datos previos a la pandemia situaban al turismo canario como motor clave del desarrollo económico, representando alrededor del 35 % del Producto Interior Bruto (PIB) y empleando al 40 % de su población activa. No obstante, tras el fin de las restricciones y el paulatino retorno a la "nueva normalidad"se ha observado un crecimiento del turismo que, si bien no ha alcanzado aún los niveles pre-pandemia, resulta significativo en la recuperación del sector.

El uso de nuevas tecnologías para ofrecer información y asistencia a los turistas ya se aplicaba con anterioridad a la crisis sanitaria. La necesidad de mejorar los servicios ofrecidos y plantear nuevas ofertas turísticas de calidad es un objetivo que no es novedoso. Sin embargo, el confinamiento no solo tuvo un impacto significativo a nivel económico sino también a nivel social. Cabe destacar que uno de los efectos que ha tenido sobre la población ha sido la concienciación sobre el medio ambiente y la gran huella ambiental que deja el turismo tras de sí. Muchos medios de comunicación e incluso las redes sociales se han hecho eco de estudios que muestran la disminución de la contaminación en grandes ciudades, el retorno de ciertos animales a hábitats actualmente ocupados por humanos, etc. Es por ello que muchas personas han decidido, en la medida de lo posible, tratar de paliar el daño ocasionado al medio ambiente y reducir lo máximo posible su huella ambiental.

Por lo tanto, podemos afirmar que las necesidades de los turistas han evolucionado y, por ende, es imprescindible aplicar una nueva perspectiva al sector turístico. Es aquí donde resulta más relevante que nunca la programación combinatoria y el uso de técnicas informáticas para resolver los denominados problemas de optimización de rutas turísticas o *Tourist Trip Design Problems* (TTDP). Estos problemas se centran en diseñar itinerarios óptimos que maximicen la satisfacción del turista y que cumplan con ciertas restricciones como limitaciones de tiempo, de costos, etc. La resolución efectiva de estos problemas permite la creación y mejora de recomendadores o guías turísticas, webs y aplicaciones para dispositivos portátiles que puedan ser utilizados tanto por los turistas que planifican sus propios viajes como a las agencias de viaje que desean ofrecer itinerarios atractivos; además de poder contribuir en el aumento de la eficiencia de recursos turísticos y promover un desarrollo turístico más sostenible.

En este proyecto plantea un problema de optimización que refleje los intereses de los turistas, identificando diferentes variables de los modelos conocidos como *Orienteering Problems*. También implementaremos un algoritmo para la resolución dicho problema. Además, se considerará la intermodalidad para realizar viajes entre las distintas islas, es decir, se tomará en cuenta el deseo de visitar más de una isla en el tiempo de estancia disponible. De manera resumida, se pretende seleccionar un subconjunto de puntos de interés de un conjunto dado, que responda lo máximo posible a las preferencias del turista.

Para ello, en el capítulo 1 se explica la relevancia que tiene el problema de diseño de rutas y, en particular, el problema de diseño de rutas turísticas. Se describirán los modelos usuales que se utilizan para facilitar la resolución de dichos problemas dependiendo de los objetivos y restricciones concretas. Además, se formalizará el problema utilizando los modelos elegidos para abordar el caso de estudio, para lo que se describirá de manera explícita tanto la función objetivo como las restricciones pertinentes.

En el capítulo 2, se describirán brevemente los métodos habituales de resolución de estos problemas: exactos, heurísticos y metaheurísticos; y se describirá el algoritmo heurístico empleado para la resolución del modelo.

En el capítulo 3 se mostrarán algunos resultados de la experimentación desarrollada para la resolución de un caso particular, utilizando los modelos propuestos. Así mismo, se mostrarán los resultados obtenidos y se analizarán brevemente.

Finalizaremos con las conclusiones y posibles extensiones a este trabajo de cara al futuro.

1

Planificación de rutas turísticas

La optimización combinatoria es una rama de las matemáticas con aplicaciones en ingeniería informática que se encarga de resolver problemas aportando la mejor solución posible entre un conjunto finito de estas. Sus inicios se pueden atribuir a problemas económicos como la planificación y gestión de operaciones o el uso eficiente de recursos, cuya resolución suponía una mejora considerable en eficiencia, productividad, etc. A partir de estos inicios y aplicaciones se formulan otros problemas de diferentes ámbitos de la vida utilizando las herramientas de la optimización combinatoria. Hoy en día podemos ver aplicaciones en diversos campos como la bioinformática, donde se analizan y comparan secuencias genéticas; en telecomunicaciones, donde se trata de mejorar el diseño de redes; en la ingeniería logística, donde se trata de mejorar los procesos de almacenamiento y distribución de bienes, etc.

La aplicación de los modelos de optimización combinatoria desempeña un papel muy importante en el diseño de rutas, ya que para planificarlas necesitamos considerar diversos elementos como la ubicación de los lugares a visitar, así como restricciones como la distancia que hay entre dichos lugares o el tiempo del que se dispone para realizar dicha ruta. Es por ello que para su resolución, se debe formular como un problema de optimización que refleje la realidad, teniendo en cuenta los siguientes elementos: las variables de decisión, que serán los factores a determinar; la función objetivo, que permite medir la efectividad según las variables de decisión utilizando criterios de maximización o minimización; y unas restricciones que se aplican a las variables de decisión para acotarlas en torno a valores factibles.

Algunos problemas relevantes en los que se trata de diseñar y planificar rutas son los siguientes:

- El Problema del Viajante o *Travelling Salesman Problem* (TSP), donde un vendedor ambulante debe visitar un conjunto finito de ciudades para poder vender su producto. Este problema parte de un punto inicial que también debe ser el punto de finalización de la ruta y trata de establecer la ruta más corta que se puede hacer visitando todas las ciudades.

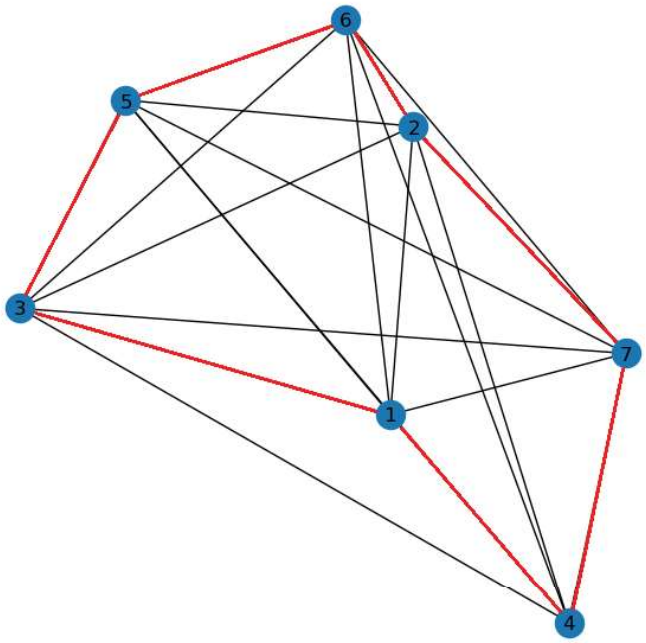


Figura 1.1. Representación de una solución del TSP en un grafo con 7 nodos.

- El Problema de Rutas de Vehículos o *Vehicle Routing Problem* (VRP), donde contamos con un número finito de vehículos y necesitamos trazar diferentes rutas para satisfacer la demanda de un conjunto finito de clientes. Este problema es una extensión del TSP pues las condiciones iniciales se mantienen, es decir, se inicia y finaliza en el mismo punto y se debe pasar por todos los puntos, pero se añade la posibilidad de realizar varias rutas.

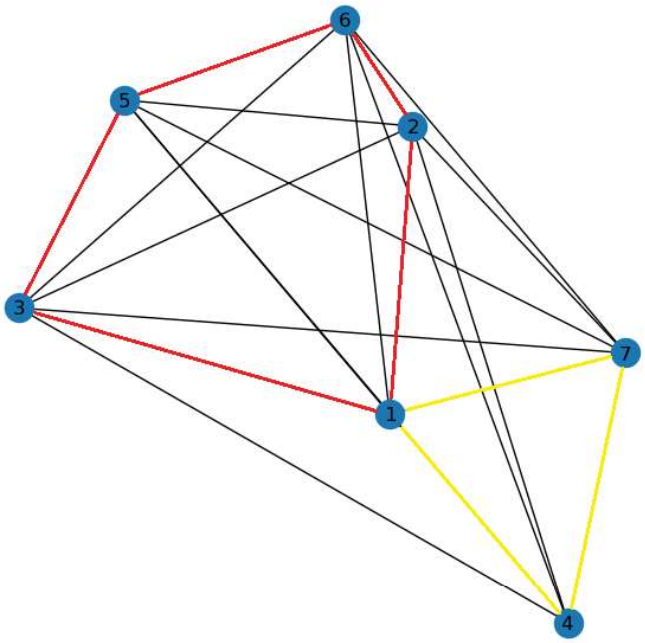


Figura 1.2. Representación de una posible solución del VRP en un grafo con 7 nodos.

Sin embargo, en general es imposible visitar absolutamente todos los lugares de interés cuando se realiza un viaje turístico y, por tanto, se debe hacer una selección de estos. Es por ello que muchos de los problemas de diseño y planificación de rutas turísticas se identifican y modelan como los conocidos como el *Orienteering Problem* (OP) y el *Team Orienteering Problem* (TOP). En estos problemas se determina un recorrido óptimo seleccionando un conjunto de puntos que tienen diferentes puntuaciones y el orden en que se deben visitar. Además, se fija un tiempo determinado para la realización del recorrido. Por lo tanto, las mejores soluciones para este problema serán aquellas rutas que obtengan la mayor puntuación en el tiempo determinado. Observamos pues que el OP es una versión del TSP en el que se selecciona un subconjunto de puntos de interés a visitar para un único día de ruta, mientras que el TOP corresponde con el VRP donde la planificación de rutas con los distintos vehículos pasa a considerarse las rutas de varios días de estancia. Estos modelos cuentan, además, con variantes que incorporan distintas restricciones como las que consideran intervalos temporales (o *Time Windows*) para los puntos de interés, es decir, intervalos de tiempo específicos en los que el lugar es visitable. Estos son el *Orienteering Problem with Time Windows* (OPTW) o *TeamvOrienteering Problem with Time Windows* (TOPTW). En nuestro caso de estudio para modelar el problema de diseño y planificación de rutas turística se utiliza el OP como problema a

formalizar y resolver.

Para la resolución de este tipo de problemas se pueden utilizar métodos exactos o métodos aproximados (heurísticos y metaheurísticos). Uno de los métodos exactos más utilizados es el *Branch and Bound* (o Ramificación y Acotamiento), este algoritmo permite linealizar un modelo de Programación Entera y generar cotas de manera recursiva para la obtención de valores enteros para las variables de decisión. Por otro lado, y debido a la complejidad de este tipo de problemas, muchos investigadores proponen algoritmos aproximados que permitan obtener soluciones suficientemente buenas.

Marco Dorigo propone una metaheuristica a principios de los años 90 y que recoge posteriormente en su libro *Ant Colony Optimization* [2] como herramienta para la solución de problemas de optimización combinatoria complejos. El procedimiento propuesto, que comparte nombre con el libro, está inspirado en el comportamiento real de las hormigas. Otro algoritmos también evocan el comportamiento de elementos observables de la naturaleza como el *Particle Swarm Optimization* (PSO).

1.1. Formalización del problema

Como ya se mencionó en la sección anterior, los problemas de diseño de rutas turísticas pueden ser modelados como el OP. Este problema puede ser formulado matemáticamente como un problema de optimización combinatoria. Por lo tanto, necesitamos definir una función objetivo junto con sus variables de decisión, así como delimitar el problema mediante el uso de restricciones. La información relevante es aquella que está estrechamente relacionada con los Puntos de Interés o *Point Of Interest* (POI), que consistirán en atracciones y servicios turísticos que puedan generar interés en el turista. Tal y como se recoge en [3], para modelar el OP tenemos que conocer la cantidad total de POIs que son posible objeto de visita así como el tiempo total disponible para realizar la ruta. También necesitaremos los tiempos de desplazamiento entre los POIs, el tiempo que se tarda en visitar cada uno, es decir, el tiempo de permanencia en ellos y la puntuación asociada a las preferencia del turista, que se obtiene al visitarlos.

La formulación propuesta considera los siguientes parámetros:

- $n \in \mathbb{N}$ será el número de POIs.
- $P = \{1, ..., n\} \in \mathbb{R}^+$ será el conjunto de POIs.
- $w_i \in \mathbb{R}^+, \forall i \in P$ será la puntuación que tiene visitar el POI i.
- $t_{i,j} \in \mathbb{R}^+, \forall i, j \in P$ será el tiempo de desplazamiento entre los POIs i y j.

- $t^i \in \mathbb{R}^+, \forall i \in P$ será el tiempo invertido en visitar el POI i.
- $T_{max} \in \mathbb{R}^+$ será el tiempo total disponible para realizar la ruta turística.

Por otro lado, necesitamos definir las variables de decisión que harán posible la toma de decisiones para nuestro problema. Estas variables permitirán especificar los itinerarios y valorar tanto su factibilidad como su optimalidad.

Las variables de decisión que tendremos en cuenta son las siguientes:

- $v_i \in \{0, 1\}$ variable binaria que adquiere el valor 1 si se visita el POI i y el valor 0 en otro caso.
- $x_{i,j} \in \{0, 1\}$ variable binaria que adquiere el valor 1 si se visitan consecutivamente el POI i y el POI j y el valor 0 en otro caso.
- $p_i \in \{0, 1, ..., n\}$ variable de posición que indicará en qué orden va a ser visitado el POI i dentro de la ruta.

La función objetivo refleja el grado de satisfacción alcanzado por el turista. Lo habitual es evaluar el *score* obtenido al sumar las diferentes puntuaciones obtenidas de los POI que están incluidos en la ruta, por lo que la función a maximizar se define como:

$$f(v, x, p) = \sum_{i=1}^{n} v_i w_i \tag{1.1}$$

El modelo además considera una serie de restricciones, las cuales garantizan la factibilidad de las posibles rutas a obtener. En nuestro modelo tenemos en cuenta las siguientes:

- El punto de partida y el punto de finalización de la ruta deberá ser el mismo.
- No se considerarán rutas que excedan del tiempo máximo disponible.
- Cada POI podrá ser visitado como máximo una única vez, a excepción del punto de partida.
- Cada ruta deberá estar correctamente conectada.

En definitiva, en el OP partimos de un conjunto de n puntos de interés dado en el cual cada uno de los puntos tiene un *score* w_i. El punto de inicio y de finalización de la ruta es el mismo, y se conoce el tiempo $t_{i,j}$ que se tarda en recorrer la distancia entre el POI i y el POI j. Además, tenemos un tiempo máximo T_{max} que impide visitar la totalidad de los POI y se entiende que una vez visitado el POI i, este no se vuelve a visitar.

Por lo tanto, debemos determinar una ruta obteniendo la máximo puntuación posible en el tiempo determinado. Y así, valiéndonos de la notación utilizada anteriormente, podemos escribir el OP como un problema de programación combinatoria de la siguiente manera:

$$\max \quad \sum_{i=1}^{n} v_i w_i \tag{1.2}$$

sujeto a:

$$\sum_{j=2}^{n} x_{1,j} = \sum_{i=2}^{n} x_{i,1} = 1 \tag{1.3}$$

$$\sum_{i=1}^{n} x_{i,k} = \sum_{j=1}^{n} x_{k,j} \leq 1, \quad k = 2, 3, ..., n \tag{1.4}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j} t_{i,j} + \sum_{i=1}^{n} v_i t^i \leq T_{max} \tag{1.5}$$

$$2 \leq p_i \leq n; \quad \forall i = 2, 3, ..., n \tag{1.6}$$

$$p_i - p_j + 1 \leq (n-1)(1 - x_{i,j}); \quad \forall i = 2, 3, ..., n \tag{1.7}$$

La función objetivo (1.2) maximiza la puntuación total acumulada. La primera restricción (1.3) nos asegura que salimos del punto de partida y volvemos a llegar a él; la segunda, (1.4), nos asegura que los POI estén conectados y que solo puedan ser visitados una única vez; la tercera restricción, (1.5), imposibilita que se supere el tiempo máximo disponible para la ruta; y las dos últimas, (1.6) y (1.8), nos garantizan que no se creen subciclos dentro de las posibles soluciones y están formuladas acorde con [8].

Como la estancia de un turista suele incluir la pernocta por varios días, debemos planificar una ruta por cada día de estancia. En estos casos es necesario pasar del OP generalizado al *Team Orienteering Problem* (TOP) que, como ya se mencionó anteriormente, se puede considerar una extensión del primero que trata de optimizar la planificación para varios miembros de un mismo equipo.

En nuestro caso particular, resolveremos el OP para cada día de estancia obligando a que cada ruta nueva dependa de las anteriores. Además, en este trabajo abordamos la posibilidad de realizar rutas entre isla, así la ruta en un día puede realizar una parte de la ruta en una isla y cambiar de isla finalizando en esta. En este caso no podemos utilizar el OP. Uno de los motivos es que el nuevo modelo tiene que considerar que el punto final de la ruta no puede ser el mismo que el inicio. La otra es que tenemos que considerar incorporar los

aeropuertos o puertos como POI obligatorios de la ruta. Una posibilidad en este caso es darle una puntuación muy alta, pero en este caso en espacio de soluciones encontraremos rutas que realizan viajes de ida y vuelta a una isla en el mismo día. Por ello debemos plantear un segundo modelo que nos permita empezar y acabar en puntos diferentes y que cumpla la condición obligatoria de pasar por un aeropuerto o puerto para realizar el cambio de isla.

Asñi hemos considerado un OP abierto, es decir, que los puntos de inicio y fin sean diferentes. La formulación de este nuevo problema es la misma que la del OP propuesto pero quitando la restricción correspondiente a que el inicio y el fin es el mismo. El nuevo modelo es formulado como sigue:

$$\max \quad \sum_{i=1}^{n} v_i w_i \tag{1.8}$$

sujeto a:

$$\sum_{j=2}^{n} x_{1,j} = \sum_{i=1}^{n-1} x_{i,n} = 1 \tag{1.9}$$

$$\sum_{i=2}^{n-1} x_{i,k} = \sum_{j=2}^{n-1} x_{k,j} \leq 1, \quad k = 2, 3, ..., n-1 \tag{1.10}$$

$$\sum_{i=1}^{n} \sum_{j=1}^{n} x_{i,j} t_{i,j} + \sum_{i=1}^{n} v_i t^i \leq T_{max} \tag{1.11}$$

$$2 \leq p_i \leq n; \quad \forall i = 2, 3, ..., n \tag{1.12}$$

$$p_i - p_j + 1 \leq (n-1)(1 - x_{i,j}); \quad \forall i = 2, 3, ..., n \tag{1.13}$$

La función objetivo (1.8) sigue siendo la misma que para el OP. La primera restricción (1.9) nos asegura que salimos del punto de partida (POI 1) y llegamos al punto final (POI n), que en este caso no es el mismo; la segunda, (1.10), nos asegura que el resto de los POI estén conectados y que solo puedan ser visitados una única vez; la tercera restricción, (1.11), imposibilita que se supere el tiempo máximo disponible para la ruta; y las dos últimas, (1.12) y (1.13), nos garantizan que no se creen subciclos dentro de las posibles soluciones.

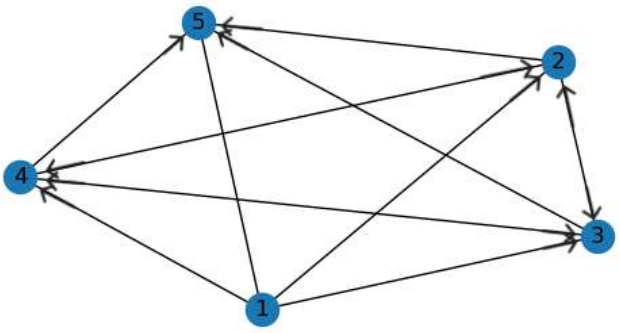


Figura 1.3. Representación del OP abierto en un grafo con 5 POI.

La figura 1.3 representa un ejemplo del OP abierto con 5 POI, es decir, $n = 5$. El POI 1 representa el punto de partida y el POI 5 el punto final. Por lo tanto, el POI 1 tiene 4 aristas o recorridos de salida y ninguno de entrada mientras que el POI 5 tiene 4 recorridos de entrada y ninguno de salida. Los POI 2, 3 y 4 tienen 3 recorridos de entrada y 3 recorridos de salida respectivamente. La figura 1.4 representa un proceso de construcción de una ruta. Se comienza desde el POI 1 y el POI 4 es seleccionado como siguiente POI a visitar. En la siguiente iteración se observa que el resto de recorridos que se podían hacer desde el POI 1 desaparecen y se selecciona el POI 2 como siguiente destino. Se observa que nuevamente los recorridos que se podían hacer desde y hasta el POI 4 desaparecen. Finalmente, el último recorrido se realiza hasta el POI 5, que es el punto final, y por tanto se termina la ruta y se obtiene el camino POI 1 $\rightarrow$ POI 4 $\rightarrow$ POI 2 $\rightarrow$ POI 5.

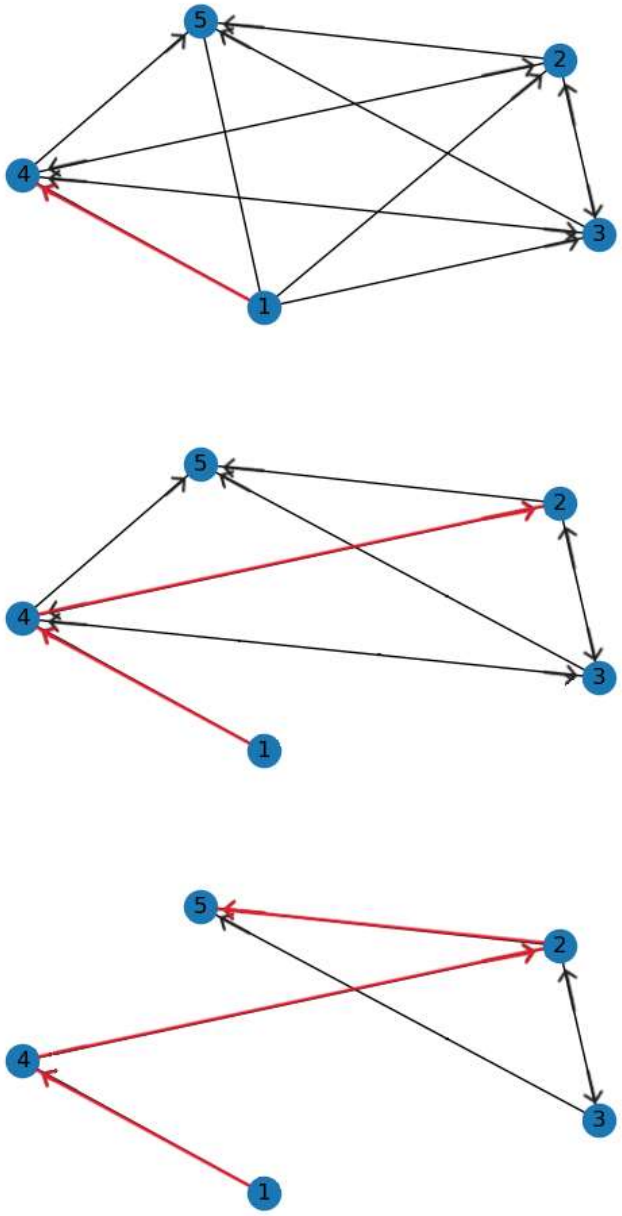


Figura 1.4. Proceso de construcción de la ruta.

2

Métodos de resolución para problemas de rutas turísticas

Como ya se mencionó brevemente en el capítulo 1, los procedimientos para resolver problemas de optimización combinatoria son muy variados pero se pueden agrupar en tres categorías: métodos exactos, métodos heurísticos y métodos metaheurísticos.

En la categoría de métodos exactos se agrupan los algoritmos que tienen como base el uso de técnicas analíticas que aseguran la convergencia a una solución óptima, si esta existe. El diseño de estos métodos se basa en teoremas matemáticos que garantizan la optimalidad de la solución y se realizan bajo supuestos y características específicas como podrían ser la linealidad, continuidad, etc.

Sin embargo, estos métodos no siempre son la respuesta adecuada ya que presentan inconvenientes que impiden su uso en una gran cantidad de problemas reales aplicados. Una de las principales desventajas que tienen los métodos exactos es lo específico que resultan para resolver problemas y las dificultades para encontrar la solución exacta en tiempos razonables, Esto hace que, en cierto modo, no se puedan considerar como algoritmos robustos debido a que no se pueden aplicar a una gran diversidad de problemas y asegurar que mantienen su eficiencia en el procedimiento de encontrar optimalidad en la solución.

Esta problemática suele estar causada por diferentes factores como por ejemplo que el tamaño del espacio de soluciones sea excesivamente grande y no se pueda hacer una búsqueda exhaustiva, que la complejidad del problema obligue a usar modelos simplificados que no aporten una solución real o que las posibles soluciones tengan unas restricciones tan fuertes que poder construir una solución, ni siquiera óptima, sea muy difícil.

Algunos de los métodos exactos más utilizados son el método de *Branch and Bound* (Ramificación y Acotamiento) o el *Branch and Cut* (Ramificación y Corte) que son algoritmos que permiten linealizar un problema de programación

entera. El *Branch and Bound* genera cotas de forma recursiva que favorecen la obtención de valores enteros para las variables de decisión mientras que el *Branch and Cut* parte del algoritmo anterior y se le aplica también el método de Planos de Corte para eliminar partes de la solución no entera y reducir el espacio total de soluciones.

En contraposición, los métodos heurísticos y metaheurísticos son métodos de búsqueda aproximado. La heurística hace referencia a la disciplina o ciencia del descubrimiento. Su objetivo es determinar una solución lo suficientemente buena para resolver un problema dado, en un intervalo de tiempo razonable. Esta solución no tiene que ser necesariamente la mejor de todas pero se aproxime mucho a la solución exacta. Se puede decir entonces que un buen algoritmo heurístico debe ser bueno, eficiente y robusto.

Las metaheurísticas son familias de algoritmos heurísticos que se aplican a problemas cuya resolución exacta es imposible de implementar o carecen de algoritmos o heurísticas específicas para su resolución. Es decir, las metaheurísticas son estrategias inteligentes para diseñar o mejorar procedimientos heurísticos muy generales con un alto rendimiento. La gran mayoría de metaheurísticas se aplican en problemas de optimización combinatoria. Por lo tanto, las metaheurísticas proponen algoritmos de propósito general, lo que las hace aplicables a innumerable tipos de problemas. Además, suelen tener una fácil implementación y, lo más importante de todo, su aplicación tiene un gran éxito a nivel práctico. Sin embargo, por definición son algoritmos aproximados (no exactos) y habitualmente no deterministas, es decir, bajo las mismas entradas o condiciones iniciales no tienen por qué producir los mismos resultados. También cabe mencionar que no siempre existe una base teórica firme que los sustente.

En este proyecto encontramos soluciones al problema propuesto a través de algoritmos metaheurísiticos. En particular, utilizaremos un algoritmo de optimización de colonia de hormigas o *Ant Colony Optimization* (ACO).

2.1. Ant Colony Optimization

Este algoritmo está inspirado en el comportamiento que muestran las hormigas en la naturaleza. Cuando una hormiga busca alimento realiza una ruta de manera aleatoria alrededor del hormiguero hasta dar con una fuente de comida. Tras esto, regresa al hormiguero con una pequeña cantidad y por el camino de vuelta secreta un rastro de feromonas. Estas feromonas atraen al resto de hormigas de la colonia, lo que propiciará que un mayor número de estas sigan el mismo camino hacia la fuente de alimentos si esta fuera considerada de calidad.

Sin embargo, debido al tamaño de las poblaciones de hormigas, lo habitual es que haya varias hormigas que realicen rutas alternativas de diferentes distancias entre la comida y el hormiguero. Y aquí es donde toma relevancia la volatilidad de las feromonas, ya que a mayor tiempo pasa desde que se secretan, menor es la intensidad de las mismas. Esto causará que los recorridos de mayor longitud presenten un rastro menos intenso que los de menor longitud, por lo que el rastro de feromonas de las rutas más cortas resultarán más atractivas y, por ende, se recorrerán un mayor número de veces.

Esta metaheurística se ha aplicado en problemas reales relacionados con planificación de rutas y resulta muy útil para encontrar soluciones a estos problemas. En nuestro caso particular este procedimiento nos permite combinar dos elementos de información. El primero, el tiempo de desplazamiento entre los POIs (ya que mientras menos se tarde en llegar al siguiente POI, más probabilidad habrá de dirigirse a él); el segundo, la feromona, es decir, la información obtenida de soluciones alternativas exploradas con anterioridad. Por tanto, cuando nos encontremos en el POI i, los POI más cercanos serán más atractivos, sin embargo, en soluciones anteriores ir del POI i al POI j, $\forall j \neq i$ habrá dado lugar a mejores o peores soluciones, lo que permitirá combinar de forma equilibrada ambos elementos.

La idea clave de este procedimiento es que las hormigas construyen de manera iterativa distintas soluciones al problema e intercambian información sobre estas para construir soluciones mejores.

Esta metaheurística se aplica en primera instancia para el TSP [2], donde se observa su efectividad en instancias reducidas y posterioriormente se desarrollan variaciones del algoritmo que permiten resolver instancias más grandes de estos problemas y de otros problemas de rutas.

Esta metaheurística se utilizará como base para resolver el OP planteado en este trabajo. Para construir una solución las hormigas elegirán sucesivamente los POI que se añadirán al recorrido inicial. Si la suma total de tiempo utilizado hasta llegar al POI sobrepasa la restricción de tiempo total T_{max} se finaliza el recorrido y se reinicia la búsqueda. Para la selección de los POI las hormigas utilizarán información heurística $\eta_{i,j}$ específica para el problema así como la intensidad del rastro de feromona $\tau_{i,j}$ de la arista (i, j). La heurística se definirá como $\eta_{i,j} = \frac{w_j}{t_{i,j}}$, donde w_j representa la puntuación que tiene visitar el POI j y $t_{i,j}$ representa el tiempo que se tarda en ir del POI i al POI j. Este primer parámetro es un indicador de lo buena que puede ser la elección del siguiente POI mientras que el segundo es un indicador de lo buena que ha sido la elección

del POI actual hasta el momento.

Para poder mantener un balance entre la exploración de soluciones óptimas y la del espacio de búsqueda, se propone una regla de paso para la hormiga k-ésima que viene dada por la siguiente probabilidad:

$$p_{i,j}^{k} = \begin{cases} \frac{(\tau_{i,j})^{\alpha}(\eta_{i,j})^{\beta}}{\sum_{r \in P_k^i}(\tau_{i,r})^{\alpha}(\eta_{i,r})^{\beta}}, \text{ si} \quad j \in P_k^i \\ 0, \qquad\qquad\qquad \text{en otro caso} \end{cases} \tag{2.1}$$

Donde P_k^i es el conjunto de POI que quedan por visitar y α y β son dos parámetros que controlan la importancia relativa del rastro de feromona frente a la información heurística.

Tras cada iteración se actualizará el rastro de feromonas que dejan la hormiga k-ésima de la siguiente manera:

$$\tau_{i,j}^{k} = \begin{cases} \frac{Q}{L_k}, \text{ si la arista (i,j) pertenece a la solución} \\ 0, \quad \text{en otro caso} \end{cases} \tag{2.2}$$

Donde Q es una constante fija y L_k es el tiempo invertido por la hormiga k-ésima en construir su propio camino.

Por lo tanto, podemos resumir los pasos a seguir como los siguientes:

1. Se inicializa el rastro de feromonas. Se localiza a las hormigas en el POI de partida.
2. Cada una de las hormigas construye un recorrido eligiendo el siguiente punto a visitar aleatoriamente según la función de probabilidad.
3. Obtener el tiempo total para el circuito construido por cada hormiga.
4. Actualizar el rastro de feromonas.
5. Repetir el proceso hasta satisfacer el criterio de parada.
6. Evaluar las soluciones y proponer la mejor como solución al problema.

2.1.1. Algoritmo implementado

En esta subsección se expondrán algunas de las partes más relevantes del algoritmo. Este ha sido implementado utilizando la versión Python 3.10.9 de

Anaconda sobre el entorno de desarrollo Visual Studio Code.

- Importación de librerías y lectura de los datos expuestos en las tablas 3.1, 3.2 y 3.9 del siguiente capítulo.

```
import numpy as np
import pandas as pd

DATA_FOLDER = "C:\\Users\\TFG\\ALGORITMO\\DistanceMatrixes"
FROM_ID = 'FROM_ID'
TO_ID = 'TO_ID'
DURATION = 'DURATION'
POI_N = 'POI_N'
BENEFIT = 'BENEFIT'

TENERIFE_DATA_FRAME = pd.DataFrame(pd.read_csv(DATA_FOLDER + "\\TimeMatrixFromTenerife.csv"), columns=[FROM_ID, TO_ID, DURATION])
HIERRO_DATA_FRAME = pd.DataFrame(pd.read_csv(DATA_FOLDER + "\\TimeMatrixFromElHierro.csv"), columns=[FROM_ID, TO_ID, DURATION])
FULL_DATA_FRAME = pd.DataFrame(pd.read_csv(DATA_FOLDER + "\\TimeMatrix.csv"), columns=[FROM_ID, TO_ID, DURATION])

POI_TENERIFE = pd.DataFrame(pd.read_csv(DATA_FOLDER + "\\PointOfInterestFromTenerife.csv"), columns=[POI_N, BENEFIT, DURATION])
POI_HIERRO = pd.DataFrame(pd.read_csv(DATA_FOLDER + "\\PointOfInterestFromElHierro.csv"), columns=[POI_N, BENEFIT, DURATION])
POI = pd.DataFrame(pd.read_csv(DATA_FOLDER + "\\POI.csv"), columns=[POI_N, BENEFIT, DURATION])
```

- Definición de funciones locales que permitan obtener información relevante para el algoritmo tales como la función que calcula la matriz de probabilidad (*calculateProbabilityMatrix*) o la función que actualiza la feromona de los recorridos visitados (*pheromoneUpdate*).

```
def benefit(dataframe, point):
    return dataframe.loc[dataframe.POI_N == point, BENEFIT].values[0]

def calculateNextPoint(currentPoint, unvisitedPoints, probabilities):
    probability = []
    for unvisitedPoint in unvisitedPoints:
        currentProbability = getMatrixValue(probabilities, currentPoint, unvisitedPoint)
        probability.append(currentProbability)

    return np.random.choice(unvisitedPoints, p=probability)

def calculateProbabilityMatrix(currentPoint, unvisitedPoints, visitedPoints, n_points, pheromone, alpha, beta):
    probabilities = np.zeros((n_points, n_points))
    probabilityUnvisitedPointsSum = 0

    for unvisitedPoint in unvisitedPoints:
        if unvisitedPoint != currentPoint and unvisitedPoint not in visitedPoints:
            pheromoneAtThisPoint = getMatrixValue(pheromone, currentPoint, unvisitedPoint)
            pheromoneRelevance = pheromoneAtThisPoint**alpha
            heuristic = benefit(POI, unvisitedPoint) / distance(FULL_DATA_FRAME, currentPoint, unvisitedPoint)
            heuristicRelevance = heuristic**beta

            probability = pheromoneRelevance * heuristicRelevance

            updateMatrix(probabilities, currentPoint, unvisitedPoint, (probability))
            probabilityUnvisitedPointsSum += probability

        else:
            updateMatrix(probabilities, currentPoint, unvisitedPoint, 0)

    for unvisitedPoint in unvisitedPoints:
        currentProbability = getMatrixValue(probabilities, currentPoint, unvisitedPoint)
        updateMatrix(probabilities, currentPoint, unvisitedPoint, currentProbability / probabilityUnvisitedPointsSum)

    return probabilities

def pheromoneUpdate(pheromone, evaporation_rate, paths, path_lengths, Q):
    pheromone *= evaporation_rate

    for path, path_length in zip(paths, path_lengths):
        for i in range(len(path) - 1):
            pheromone[path[i], path[i+1]] += Q/path_length
```

- Definición de la función principal que calcula los recorridos de las hormigas y que tras finalizar muestra por pantalla el mejor recorrido junto con el tiempo que se tarda en realizarlo.

```
def ant_colony_optimization(points, n_ants, n_iterations, alpha, beta, evaporation_rate, Q, Tmax):
    n_points = len(points)
    pheromone = np.ones((n_points, n_points))
    best_path = []
    best_path_length = np.inf

    for iteration in range(n_iterations):
        paths = []
        path_lengths = []

        for ant in range(n_ants):
            unvisitedPoints = np.copy(points)
            visitedPoints = []
            currentPoint = unvisitedPoints[0]
            path = [currentPoint]
            pathLength = 0

            while pathLength <= Tmax:
                probabilities = calculateProbabilityMatrix(currentPoint, unvisitedPoints, visitedPoints, n_points, pheromone, alpha, beta)

                nextPoint = calculateNextPoint(currentPoint, unvisitedPoints, probabilities)
                path.append(nextPoint)
                pathLength += distance(FULL_DATA_FRAME, currentPoint, nextPoint) + duration(POI, nextPoint)

                visitedPoints.append(currentPoint)
                unvisitedPoints = removeCurrentPointFromUnvisited(unvisitedPoints, currentPoint)

                currentPoint = nextPoint

            paths.append(path)
            path lengths.append(pathLength)

            print("iteration :", iteration, ", ant n: ", ant)
            print("paths: ", paths)
            print("pathlength: ", path_lengths)

            if pathLength < best_path_length:
                best_path.clear()
                best_path.append(path)
                best_path_length = pathLength

        pheromoneUpdate(pheromone, evaporation_rate, paths, path_lengths, n_points, Q)

    print("Best path found: ", best_path)
    print("Best path length: ", best_path_length)
```

3

Experimentación

En este capítulo se experimentará con un caso real en las islas de Tenerife y El Hierro. Para comenzar se describirá la información necesaria para poder aplicar el algoritmo y cuál ha sido su método de obtención. A continuación, se mostrarán los resultados obtenidos y se analizarán brevemente.

3.1. Información

En primer lugar, se han elegido 20 puntos de interés en la isla de Tenerife y 15 en la de El Hierro. Se ha realizado esta selección de POI teniendo en cuenta la frecuencia con la que se visitan dichas atracciones turísticas. Para los lugares de alojamiento del turista se han elegido los hoteles DWO Nopal en la isla de Tenerife y el Parador de El Hierro en la isla de El Hierro. El tiempo máximo para cada ruta se fijará en 6, 7 y 8 horas para poder observar las diferencias entre los recorridos obtenidos.

En las siguientes tablas se muestran los datos establecidos para los diferentes POI seleccionados en el orden siguiente:

- Nº asociado al POI.
- Nombre de los POI que se pueden visitar.
- Categoría de cada POI.
- Puntuación que tiene visitar cada POI.
- Duración de la visita de cada POI.

Para los tres últimos parámetros se han tenido en cuenta lo siguiente: la categoría se ha utilizado para equilibrar la selección de POI y que no acabaran siendo todos del mismo tipo, aunque solo tiene carácter informativo; la puntuación (w_i) se ha establecido de manera aleatoria utilizando la función *randbetween* de Google Sheets; y la duración (t^i) se ha establecido acorde a las horas que se invierten en la visita de ese POI.

Toda la información referente a distancias y tiempos entre POIs se ha recopilado utilizando el plug-in *Open Route Service* en el QGIS (*Geographic Information System*), el cual es un conjunto de herramientas que permiten a los usuarios crear consultas interactivas, analizar la información espacial y crear mapas, entre otros. También se ha utilizado para realizar las imágenes 3.1 y 3.2.

POI Nº	POI	CATEGORÍA	PUNTUACIÓN	DURACIÓN
0	Hotel DWO Nopal	Hotel	0	0,0
1	Benijo	Playa	23	3,0
2	El Bollullo	Playa	26	3,0
3	Las Vistas	Playa	21	3,4
4	La Tejita	Playa	44	3,2
5	Siam Park	Parque temático	39	5,0
6	Loro Parque	Parque temático	46	4,3
7	Mariposario del Drago	Parque temático	41	1,5
8	Forestal Park	Parque temático	26	2,5
9	Aqualand	Parque temático	30	4,5
10	Museo de Naturaleza y Arqueología	Museo	24	1,5
11	Museo de La Ciencia y El Cosmos	Museo	37	1,5
12	San Cristobal de La Laguna	Patrimonio de la Humanidad	39	2,0
13	El Teide	Patrimonio de la Humanidad	38	3,5
14	Barranco de Masca	Monumento natural	28	2,5
15	Cueva del viento	Monumento natural	20	2,0
16	Roque de Garachico	Monumento natural	48	1,5
17	Montaña Pelada	Monumento natural	36	1,5
18	Barranco de Herques	Monumento natural	36	2,0
19	Acantilado de Los Gigantes	Monumento natural	35	2,0
20	Aeropuerto Tenerife Norte	Aeropuerto	100	1,5
21	Los Cristianos	Puerto	100	1,5

Tabla 3.1. POI de Tenerife

POI Nº	POI	CATEGORÍA	PUNTUACIÓN	DURACIÓN
0	Parador de El Hierro	Hotel	0	0,0
1	Tacorón	Playa	24	3,0
2	Charco Azul	Playa	47	3,0
3	Tamaduste	Playa	48	3,4
4	El Verodal	Playa	41	3,2
5	Santuario Nuestra Señora de los Reyes	Iglesia	34	1,0
6	Campanario de Joapira	Iglesia	27	0,8
7	Ermita de la Peña	Iglesia	31	0,5
8	Ecomuseo de Guinea y Lagartario	Museo	49	2,0
9	Parque Cultural de El Julan	Museo	25	2,0
10	Centro de Interpretación Geológica	Museo	44	1,5
11	Arbol Garoé	Monumento natural	20	1,3
12	Faro de Orchilla	Monumento natural	48	2,0
13	El Sabinar	Monumento natural	22	2,0
14	Roque de la Bonanza	Monumento natural	28	1,5
15	Aeropuerto de El Hierro	Aeropuerto	100	1,5
16	Puerto de la Estaca	Puerto	100	1,5

Tabla 3.2. POI de El Hierro

Tanto en la tabla 3.1 como en la 3.2 se han incluido dos POI adicionales que representan el aeropuerto y puerto que unen ambas islas. Esta información será necesaria únicamente para el OP abierto, por lo que los POI no se considerarán en las rutas exclusivas para cada isla.

Las imágenes 3.1 y 3.2 representan la ubicación de los POI elegidos para cada una de las islas.

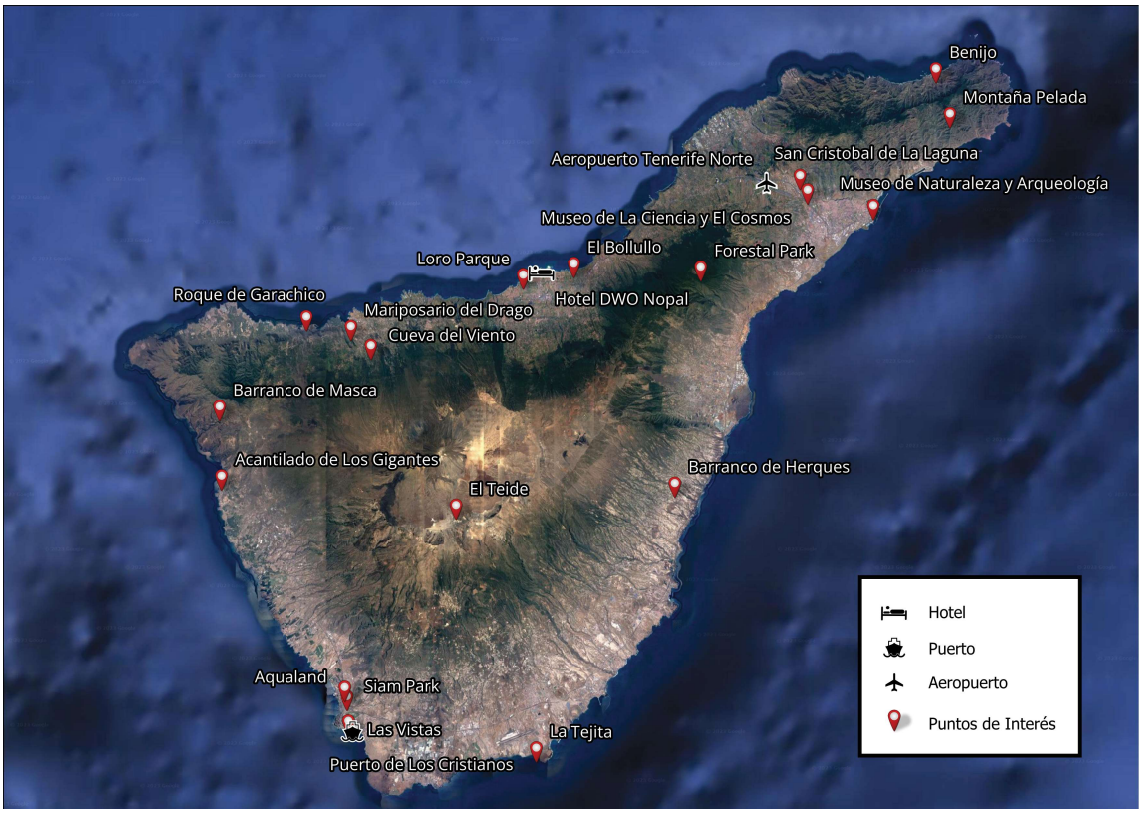


Figura 3.1. Ubicación de los puntos de interés de la isla de Tenerife.

Figura 3.2. Ubicación de los puntos de interés de la isla de El Hierro.

Las tablas 3.3 y 3.4 recogen el tiempo en horas que se tarda en desplazarse en cada isla desde el POI i hasta el resto. La tabla 3.5 recoge el tiempo en horas de desplazamiento entre islas, donde se les asigna un número elevado de horas a las combinaciones que son imposibles, como partir de un aeropuerto y acabar en un puerto.

POI Nº	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21
0	0,000	1,113	0,247	1,262	1,131	1,205	0,142	0,634	0,657	1,225	0,567	0,458	0,520	1,004	1,084	0,635	0,722	0,897	0,958	1,276	0,453	1,288
1	1,079	0,000	1,029	1,413	1,282	1,356	1,097	1,467	1,003	1,376	0,590	0,722	0,692	1,728	1,917	1,467	1,554	0,447	1,109	1,826	0,793	1,439
2	0,248	1,040	0,000	1,190	1,059	1,132	0,266	0,636	0,585	1,152	0,495	0,386	0,448	0,919	1,086	0,636	0,723	0,825	0,886	1,278	0,381	1,215
3	1,241	1,391	1,191	0,000	0,373	0,156	1,259	1,006	1,030	0,176	0,840	0,884	0,950	0,897	0,753	1,033	1,058	1,175	0,647	0,626	0,955	0,041
4	1,114	1,264	1,064	0,404	0,000	0,347	1,132	1,196	0,903	0,367	0,714	0,757	0,823	0,933	0,944	1,223	1,248	1,048	0,520	0,816	0,829	0,429
5	1,216	1,366	1,166	0,153	0,348	0,000	1,235	0,941	1,005	0,094	0,816	0,860	0,926	0,901	0,688	0,968	0,993	1,151	0,622	0,561	0,931	0,179
6	0,142	1,129	0,278	1,278	1,147	1,221	0,000	0,563	0,673	1,241	0,584	0,475	0,536	0,968	1,013	0,564	0,651	0,913	0,974	1,206	0,470	1,304
7	0,635	1,485	0,634	0,988	1,164	0,944	0,582	0,000	1,029	0,906	0,940	0,830	0,892	1,001	0,583	0,183	0,157	1,269	1,330	0,776	0,825	1,014
8	0,646	0,992	0,596	1,045	0,914	0,987	0,664	1,034	0,000	1,007	0,447	0,314	0,376	0,725	1,484	1,035	1,122	0,777	0,740	1,457	0,395	1,070
9	1,224	1,373	1,174	0,180	0,355	0,064	1,242	0,931	1,013	0,000	0,823	0,867	0,933	0,908	0,679	0,958	0,983	1,158	0,629	0,551	0,938	0,206
10	0,555	0,658	0,505	0,889	0,758	0,832	0,573	0,943	0,480	0,852	0,000	0,198	0,264	1,205	1,393	0,943	1,030	0,443	0,585	1,301	0,269	0,915
11	0,452	0,718	0,402	0,885	0,754	0,828	0,470	0,840	0,354	0,848	0,191	0,000	0,096	1,079	1,289	0,840	0,927	0,521	0,581	1,298	0,166	0,911
12	0,414	0,715	0,364	0,898	0,767	0,840	0,433	0,802	0,311	0,860	0,203	0,047	0,000	1,036	1,252	0,803	0,890	0,533	0,594	1,310	0,129	0,923
13	0,987	1,720	0,919	0,892	0,927	0,845	0,955	1,024	0,727	0,865	1,174	1,042	1,103	0,000	0,919	1,034	1,113	1,504	1,224	0,907	1,123	0,918
14	1,076	1,926	1,075	0,748	0,923	0,704	1,024	0,595	1,471	0,666	1,381	1,272	1,333	0,934	0,000	0,622	0,629	1,711	1,197	0,536	1,267	0,774
15	0,635	1,485	0,634	1,027	1,202	0,983	0,582	0,191	1,029	0,945	0,940	0,831	0,892	1,022	0,622	0,000	0,325	1,269	1,330	0,815	0,826	1,053
16	0,721	1,571	0,720	1,050	1,225	1,005	0,668	0,148	1,115	0,967	1,026	0,916	0,978	1,115	0,631	0,318	0,000	1,355	1,416	0,837	0,911	1,075
17	0,863	0,447	0,813	1,198	1,067	1,140	0,882	1,251	0,788	1,160	0,375	0,506	0,573	1,513	1,701	1,252	1,339	0,000	0,894	1,610	0,578	1,223
18	0,958	1,108	0,908	0,650	0,519	0,592	0,976	1,346	0,747	0,612	0,558	0,601	0,667	1,202	1,190	1,346	1,434	0,892	0,000	1,062	0,672	0,675
19	1,295	1,795	1,294	0,602	0,777	0,557	1,242	0,814	1,434	0,519	1,245	1,288	1,355	0,914	0,562	0,841	0,866	1,579	1,051	0,000	1,360	0,627
20	0,428	0,781	0,378	0,930	0,799	0,873	0,446	0,816	0,367	0,893	0,236	0,127	0,188	1,092	1,266	0,817	0,904	0,565	0,626	1,343	0,000	0,956
21	1,263	1,413	1,213	0,044	0,395	0,178	1,281	1,028	1,052	0,198	0,862	0,906	0,972	0,919	0,775	1,055	1,080	1,197	0,669	0,648	0,977	0,000

Tabla 3.3. Distancia entre los POI i y j de Tenerife.

POI Nº	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
0	0,000	0,693	0,712	0,338	1,034	0,883	0,724	0,466	0,620	0,657	0,457	0,433	1,112	1,001	0,046	0,340	0,213
1	0,689	0,000	0,772	0,776	1,067	0,764	0,685	0,534	0,727	0,537	0,239	0,501	0,993	0,882	0,643	0,778	0,651
2	0,705	0,776	0,000	0,558	0,355	0,603	0,158	0,349	0,139	0,691	0,539	0,444	0,661	0,722	0,658	0,561	0,535
3	0,346	0,788	0,573	0,000	0,895	0,917	0,585	0,383	0,481	0,751	0,551	0,424	1,146	1,036	0,299	0,117	0,176
4	1,026	1,067	0,354	0,879	0,000	0,325	0,464	0,670	0,460	0,665	0,845	0,765	0,383	0,444	0,979	0,882	0,856
5	0,879	0,764	0,602	0,901	0,325	0,000	0,695	0,623	0,707	0,361	0,549	0,610	0,250	0,119	0,833	0,904	0,841
6	0,715	0,688	0,157	0,569	0,464	0,695	0,000	0,360	0,117	0,603	0,452	0,455	0,770	0,814	0,669	0,571	0,546
7	0,462	0,540	0,349	0,363	0,671	0,623	0,360	0,000	0,256	0,503	0,303	0,158	0,852	0,742	0,415	0,365	0,340
8	0,611	0,733	0,139	0,465	0,461	0,708	0,117	0,256	0,000	0,689	0,497	0,351	0,767	0,827	0,565	0,467	0,442
9	0,656	0,537	0,692	0,742	0,665	0,361	0,604	0,501	0,690	0,000	0,322	0,468	0,590	0,480	0,610	0,745	0,618
10	0,450	0,239	0,533	0,537	0,840	0,546	0,445	0,295	0,488	0,320	0,000	0,262	0,776	0,665	0,404	0,539	0,412
11	0,429	0,507	0,443	0,405	0,764	0,609	0,454	0,158	0,350	0,470	0,270	0,000	0,838	0,728	0,382	0,407	0,382
12	1,109	0,993	0,660	1,131	0,383	0,250	0,770	0,852	0,766	0,590	0,778	0,839	0,000	0,369	1,062	1,133	1,071
13	0,998	0,882	0,721	1,020	0,444	0,119	0,814	0,742	0,826	0,480	0,668	0,729	0,369	0,000	0,952	1,022	0,960
14	0,046	0,647	0,665	0,291	0,987	0,836	0,677	0,420	0,573	0,610	0,411	0,387	1,065	0,955	0,000	0,293	0,167
15	0,326	0,769	0,554	0,096	0,876	0,898	0,566	0,364	0,462	0,732	0,532	0,405	1,127	1,017	0,280	0,000	0,157
16	0,215	0,657	0,540	0,166	0,862	0,846	0,552	0,350	0,448	0,620	0,420	0,391	1,075	0,965	0,168	0,168	0,000

Tabla 3.4. Distancia entre los POI i y j de El Hierro.

POI	Tenerife Norte	Los Cristianos	Apto. de El Hierro	Puerto de La Estaca
Tenerife Norte	0	100	0,667	100
Los Cristianos	100	0	100	2,5
Apto. de El Hierro	0,667	100	0	100
Puerto de La Estaca	100	2,5	100	0

Tabla 3.5. Tiempos de desplazamiento entre Tenerife y El Hierro.

3.2. Resultados obtenidos

En esta sección se detallarán los resultados obtenidos por el algoritmo, haciendo una pequeña comparativa entre las distintas soluciones dependiendo del tiempo máximo permitido. Se presentará una ruta solución para los POI de Tenerife, una ruta que nos permita viajar de Tenerife a El Hierro y una última ruta para los POI de El Hierro.

Todas los resultados se han obtenido utilizando algunos parámetros de entrada fijados:

- Número de hormigas: 10
- Número de iteraciones: 30
- Parámetro de control a la feromona: $\alpha = 2$
- Parámetro de control de la información heurística: $\beta = 1$
- Tasa de evaporación: 0'5
- Constante: Q = 1

Tras ejecutar el programa obtenemos una salida que nos indica el número de iteración, el número de hormiga, el camino encontrado por dicha hormiga y el tiempo que ha tardado en recorrerlo. Además, una vez las hormigas terminan los recorridos, se actualiza el mejor camino y se muestra el orden en que se debe recorrer junto con el tiempo que se tarda en recorrerlo.

```
iteration : 29 , ant n:  0
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  1
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  2
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  3
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  4
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  5
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  6
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  7
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  8
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
iteration : 29 , ant n:  9
paths:  [0, 20, 37, 25]
pathlength:  7.61609444444445
Best path found: [[0, 20, 37, 29, 32, 22]]
Best path length:  7.34623611111111
```

Figura 3.3. Ejemplo de output para la última iteración de una ejecución del algoritmo.

3.2.1. Ruta en Tenerife

Se han obtenido los siguientes resultados para un tiempo máximo de 6, 7 y 8 horas respectivamente:

```
Best path found: [[0, 12, 11, 0]]
Best path length: 4.518730555555557
Best path found: [[0, 6, 15, 0]]
Best path length: 7.005825
Best path found: [[0, 5, 0]]
Best path length: 7.62106666666666
```

Figura 3.4. POI visitados en la isla de Tenerife.

Esto es, que el resultado para cada tiempo máximo es el siguiente:

RUTA (6 HORAS)	
POI Nº	**POI**
0	Hotel DWO Nopal
12	San Cristóbal de La Laguna
11	Museo de La Ciencia y El Cosmos
0	Hotel DWO Nopal

Tabla 3.6. Ruta con tiempo máximo de 6h.

Figura 3.5. Representación geográfica de la ruta con tiempo máximo de 6h.

El tiempo total necesario para realizar la ruta es de **4'518730555555557 horas (4 horas y 31 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 12'6 segundos.

RUTA (7 HORAS)	
POI Nº	**POI**
0	Hotel DWO Nopal
6	Loro Parque
15	Cueva del Viento
0	Hotel DWO Nopal

Tabla 3.7. Ruta con tiempo máximo de 7h.

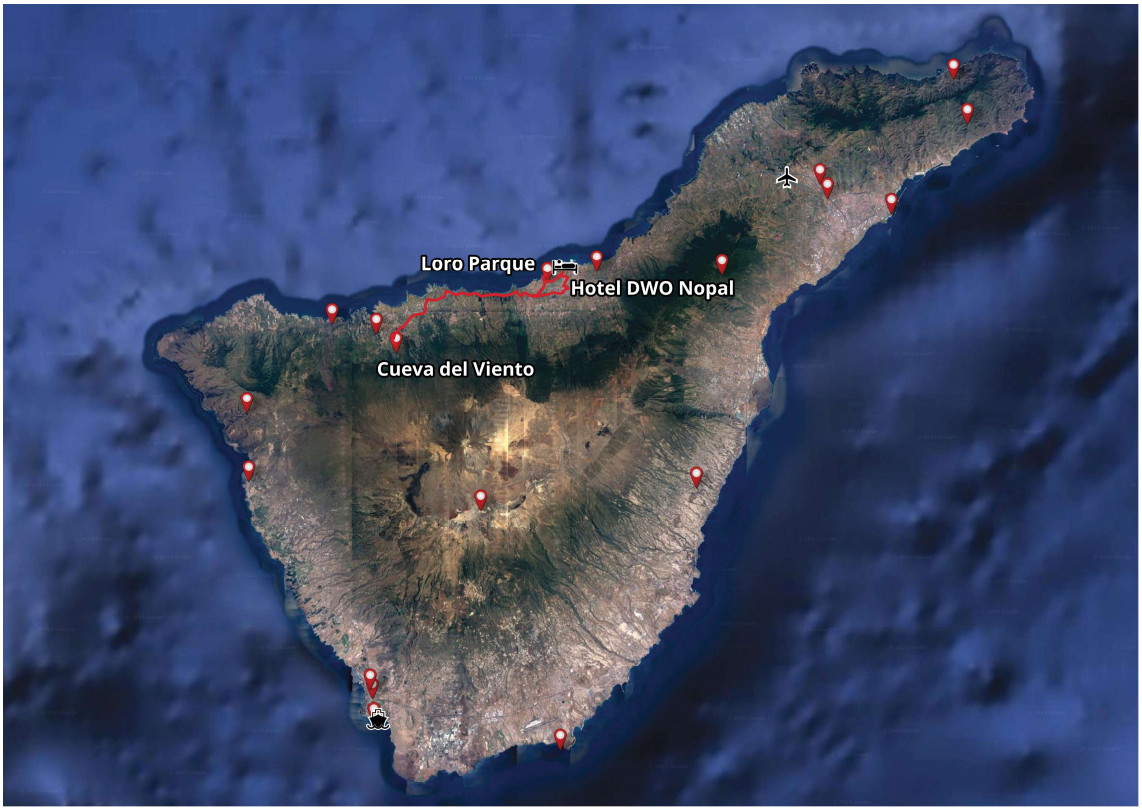


Figura 3.6. Representación geográfica de la ruta con tiempo máximo de 7h.

El tiempo total necesario para realizar la ruta es de **7'005825 horas (7 horas)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 12'4 segundos.

RUTA (8 HORAS)	
POI Nº	**POI**
0	Hotel DWO Nopal
5	Siam Park
0	Hotel DWO Nopal

Tabla 3.8. Ruta con tiempo máximo de 8h.

Figura 3.7. Representación geográfica de la ruta con tiempo máximo de 8h.

El tiempo total necesario para realizar la ruta es de **7'62106666666666 horas (7 horas y 37 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 17'3 segundos.

3.2.2. Ruta de Tenerife a El Hierro

Para ejecutar el algoritmo y viajar de una isla a otra necesitamos considerar los POIs de ambas islas al mismo tiempo. Para ello se ha unificado la información de ambas islas y se ha vuelto a numerar los POI como se recoge en la siguiente tabla:

POI Nº	POI
0	Hotel DWO Nopal
1	Benijo
2	El Bollullo
3	Las Vistas
4	La Tejita
5	Siam Park
6	Loro Parque
7	Mariposario del Drago
8	Forestal Park
9	Aqualand
10	Museo de Naturaleza y Arqueología
11	Museo de La Ciencia y El Cosmos
12	San Cristobal de La Laguna
13	El Teide
14	Barranco de Masca
15	Cueva del viento
16	Roque de Garachico
17	Montaña Pelada
18	Barranco de Herques
19	Acantilado de Los Gigantes
20	Aeropuerto Tenerife Norte
21	Los Cristianos

POI Nº	POI
22	Parador de El Hierro
23	Tacorón
24	Charco Azul
25	Tamaduste
26	El Verodal
27	Santuario Nuestra Señora de los Reyes
28	Campanario de Joapira
29	Ermita de la Peña
30	Ecomuseo de Guinea y Lagartario
31	Parque Cultural de El Julán
32	Centro de Interpretación Geológica
33	Arbol Garoé
34	Faro de Orchilla
35	El Sabinar
36	Roque de la Bonanza
37	Aeropuerto de El Hierro
38	Puerto de la Estaca

Tabla 3.9. Numeración conjunta de los POIs de Tenerife (izquierda) y El Hierro (derecha).

Para esta contexto de planificación se han obtenido los siguientes resultados para 6, 7 y 8 horas respectivamente:

```
Best path found: [[0, 20, 37, 33, 22]]
Best path length: 5.825241666666667
Best path found: [[0, 20, 37, 32, 22]]
Best path length: 6.60245000000003
Best path found: [[0, 20, 37, 28, 30, 22]]
Best path length: 8.087269444444441
```

Figura 3.8. POI visitados en el cambio de isla.

Los resultados obtenidos para cada tiempo máximo de la ruta son el siguiente:

RUTA (6 HORAS)	
POI Nº	**POI**
0	Hotel DWO Nopal
20	Aeropuerto Tenerife Norte
37	Aeropuerto de El Hierro
33	Árbol Garoé
22	Parador de El Hierro

Tabla 3.10. Ruta con tiempo máximo de 6h.

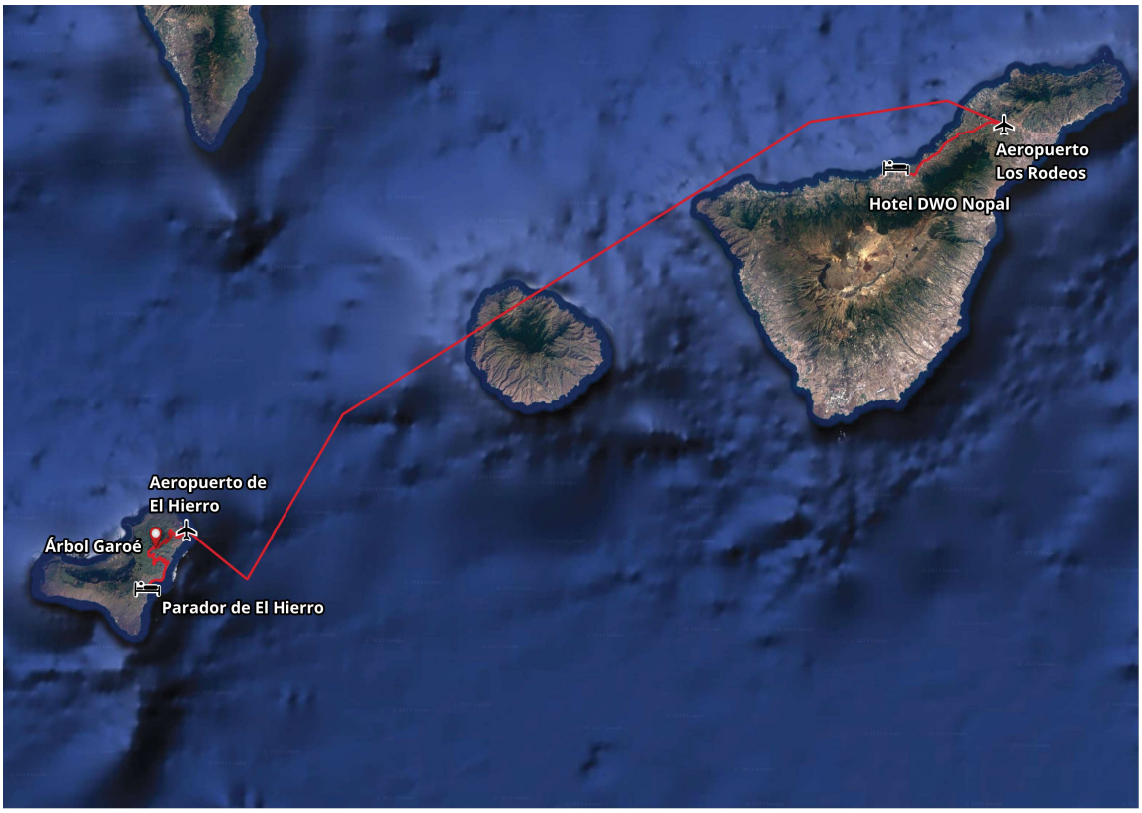


Figura 3.9. Representación geográfica de la ruta con tiempo máximo de 6h.

El tiempo total necesario para realizar la ruta es de **5'825241666666667 horas (5 horas y 50 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 32'7 segundos.

RUTA (7 HORAS)	
POI Nº	**POI**
0	Hotel DWO Nopal
20	Aeropuerto Tenerife Norte
37	Aeropuerto de El Hierro
32	Centro de Interpretación Geológica
22	Parador de El Hierro

Tabla 3.11. Ruta con tiempo máximo de 7h.

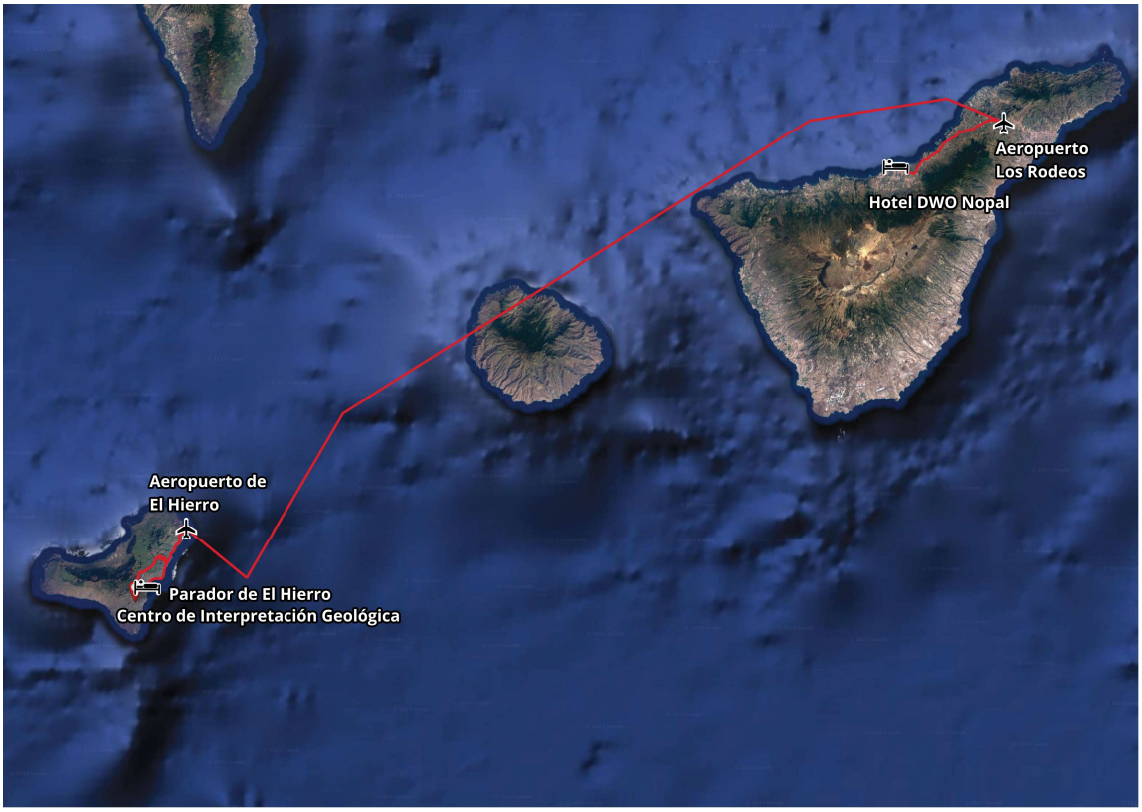


Figura 3.10. Representación geográfica de la ruta con tiempo máximo de 7h.

El tiempo total necesario para realizar la ruta es de **6'60245000000003 horas (6 horas y 36 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 35'7 segundos.

RUTA (8 HORAS)	
POI Nº	**POI**
0	Hotel DWO Nopal
20	Aeropuerto Tenerife Norte
37	Aeropuerto de El Hierro
28	Campanario de Joapira
30	Ecomuseo de Guinea y Lagartario
22	Parador de El Hierro

Tabla 3.12. Ruta con tiempo máximo de 8h.

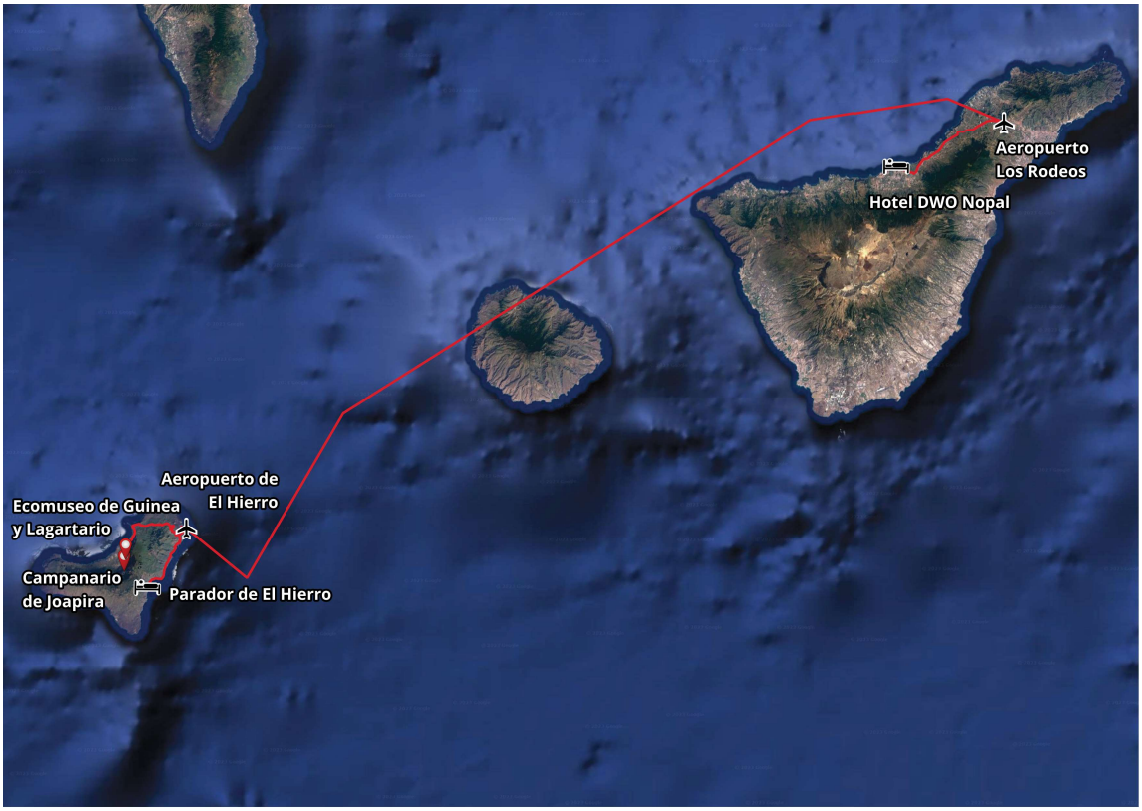


Figura 3.11. Representación geográfica de la ruta con tiempo máximo de 8h.

El tiempo total necesario para realizar la ruta es de **8'087269444444441 horas (8 horas y 5 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 51.1 segundos.

3.2.3. Ruta en El Hierro

Se han obtenido los siguientes resultados para un tiempo máximo de 6, 7 y 8 horas respectivamente:

```
Best path found: [[0, 5, 12, 0]]
Best path length: 5.241344
Best path found: [[0, 7, 11, 14, 10, 0]]
Best path length: 6.86693055555553
Best path found: [[0, 2, 6, 8, 0]]
Best path length: 7.397810555533
```

Figura 3.12. POI visitados en la isla de El Hierro.

Así, los resultados obtenidos para cada tiempo máximo son los siguientes:

RUTA (6 HORAS)	
POI Nº	**POI**
0	Parador de El Hierro
5	Ermita Virgen de los Reyes
12	Faro de Orchilla
0	Parador de El Hierro

Tabla 3.13. Ruta con tiempo máximo de 6h.

Figura 3.13. Representación geográfica de la ruta con tiempo máximo de 6h.

El tiempo total necesario para realizar la ruta es de **5'241344 horas (5 horas y 14 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 14'0 segundos.

RUTA (7 HORAS)	
POI Nº	**POI**
0	Parador de El Hierro
7	Ermita de la Peña
11	Arbol Garoé
14	Roque de la Bonanza
10	Centro de Interpretación Geológica
0	Parador de El Hierro

Tabla 3.14. Ruta con tiempo máximo de 7h.

Figura 3.14. Representación geográfica de la ruta con tiempo máximo de 7h.

El tiempo total necesario para realizar la ruta es de **6'86693055555553 horas (6 horas y 52 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 16'2 segundos.

RUTA (8 HORAS)	
POI Nº	**POI**
0	Parador de El Hierro
2	Charco Azul
6	Campanario de Joapira
8	Ecomuseo de Guinea y Lagartario
0	Parador de El Hierro

Tabla 3.15. Ruta con tiempo máximo de 8h.

Figura 3.15. Representación geográfica de la ruta con tiempo máximo de 8h.

El tiempo total necesario para realizar la ruta es de **7'397810555533 horas (7 horas y 24 minutos)**. El tiempo utilizado por el algoritmo para encontrar una solución ha sido de 17'3 segundos.

4

Conclusiones

En este trabajo hemos podido comprobar que la combinación de técnicas de programación matemática y computación nos permite desarrollar procedimientos para obtener soluciones al problema de planificación de rutas turísticas. El modelo y la metodología propuesta en este proyecto son una potente herramienta para generar de manera personalizada diferentes itinerarios según el tiempo del que se disponga y las preferencias asignadas a cada POI. Además, cabe destacar que dichos itinerarios pueden incluir desplazamientos sustanciales en los destinos turísticos. Esto mejora la toma de decisiones para la organización de las vacaciones de los turistas que desean múltiples destinos, lo que se traduce en una mayor satisfacción y una mejor experiencia de viaje. Podemos decir entonces que el modelo propuesto puede resultar un instrumento útil para el sector turístico y que, junto a información complementaria como estudios de mercado o encuestas de satisfacción, se pueden realizar propuestas realmente personalizadas para los turistas.

Aún así, existen propuestas de mejora que ampliarían la aplicación del modelo y que ajustándose a la realidad aún más. En primer lugar, establecer ventanas de tiempo o *time windows* para los puntos de interés, ya que lugares como museos o parques temáticos suelen tener unos horarios fijos de apertura. En segundo lugar, considerar intermodalidad entre los distintos POI de una misma área, ya que no todos los turistas desean alquilar vehículo para sus vacaciones. En tercer y último lugar, considerar los recorridos con menor impacto en el medio ambiente (menor huella de carbono); este punto está muy relacionado con el segundo, y es que cada vez hay una mayor demanda de que el turismo sea sostenible.

En definitiva, en este proyecto se ha desarrollado un instrumento que mejora la personalización de itinerarios con múltiples destinos en el sector turístico y se plantean posibles mejoras para perfeccionar el modelo de manera que las soluciones obtenidas se ajusten aún más a la realidad.

Lista de Figuras

Lista de Tablas

Bibliografía

[1] Blum, C. (2005). Ant colony optimization: Introduction and recent trends *Physics of Life Reviews*, vol. 2(4), pp. 353–373. https://www.sciencedirect.com/science/article/abs/pii/S1571064505000333?via%3Dihub

[2] DORIGO, M. y STÜTZLE, T. *Ant Colony Optimization.* Reading: A Bradford Book, 2004.

[3] Gunawana, A., Lau, H.C., Vansteenwegenb, P. (2016). Orienteering Problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, vol. 3(3), pp. 315–332. https://www.sciencedirect.com/science/article/abs/pii/S037722171630296X

[4] Gavalas, D., Konstantopoulos, C., Mastakas, K., Pantziou, G., Vathis, N. (2015). Heuristics for the time dependent team orienteering problem: Application to tourist route planning. *Computers & Operations Research*, vol. 62, pp. 36–50. https://www.sciencedirect.com/science/article/abs/pii/S0305054815000817

[5] Lee, H., Nee, A.Y.C., Zhou, R. (2008). Applying Ant Colony Optimisation (ACO) algorithm to dynamic job shop scheduling problems. *International Journal of Manufacturing Research*, vol. 3(3), pp. 301–320. https://www.researchgate.net/publication/220650606_Applying_Ant_Colony_Optimisation_ACO_algorithm_to_dynamic_job_shop_scheduling_problems

[6] Liang, Y.C., Smith, A. (2006) An ant colony approach to the orienteering problem. *Journal of the Chinese Institute of Industrial Engineers*, vol. 10(5), pp. 403–414. https://www.researchgate.net/publication/251808664_An_ant_colony_approach_to_the_orienteering_problem

[7] Matplotlib [Fecha de consulta: 24-04-2022]. Disponible en: https://matplotlib.org/.

[8] Miller, C., Tucker, A., Zemlin, R. (1960) Integer programming formulations and travelling salesman problems. *Journal of the AMC*, vol. 7, pp. 326–329.

[9] Numpy [Fecha de consulta: 24-04-2022]. Disponible en: https://numpy.org/.

[10] Pandas [Fecha de consulta: 24-04-2022]. Disponible en: https://pandas.pydata.org/.

[11] Keshtkaran, M. Sohrabi, S., Ziarati, K. (2021). ACS-OPHS: Ant Colony System for the Orienteering Problem with hotel selection. *EURO Journal on Transportation and Logistics*, vol. 10, 100036. https://www.sciencedirect.com/science/article/pii/S219243762100008X?via%3Dihub

Tourist trip design with intermodal transportation

Laura Hernández Bethencourt
Facultad de Ciencias • Sección de Matemáticas
Universidad de La Laguna
alu0100821475@ull.edu.es

Abstract

Solving the problems of tourist routes has a significant impact on the tourism sector, as it helps to offer attractive and fully personalised itineraries for tourists. Using mathematical and computer tools makes it possible to provide solutions that maximise tourist satisfaction. We propose a combinatorial optimisation model to maximise tourists' preferences, considering specific constraints in the itinerary. An ACO metaheuristic allows for finding solutions to this problem. For the experimentation, real data from points of interest of some islands are used, and some results are shown together with a brief analysis of them.

1. Introduction

Planning tourist routes that are engaging and charming for tourists is one of the main goals for companies that base their services on tourism. These problems focus on designing optimal itineraries that maximize tourist satisfaction and comply with certain constraints such as time limitations, costs, etc. The effective resolution of these problems allows the creation and improvement of recommenders or tourist guides, websites and applications for portable devices that can be used by tourists who plan their own trips and travel agencies that wish to offer attractive itineraries. In this project, we will propose the OP model and an open OP and solve them using metaheuristic methods.

2. Optimization of tourist routes

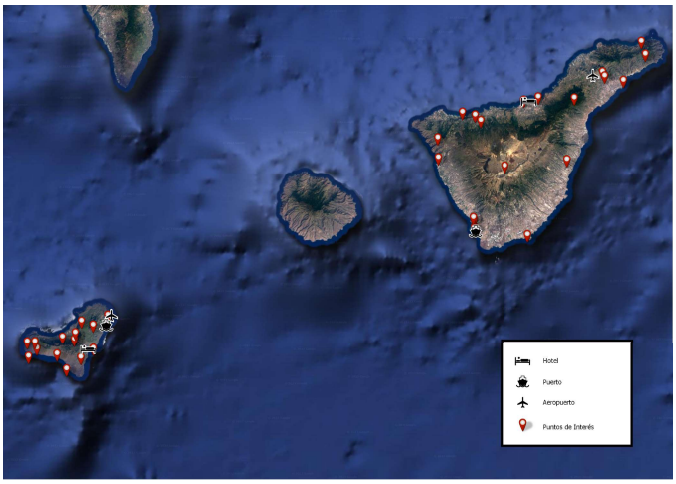
The tourist route optimization problem can be modelled with what is known as Orientation Problem (OP). This problem involves a set of checkpoints with an assigned score. The goal is to go through a subset of the checkpoints while getting the highest possible score in a predetermined time set beforehand. In this work, we also use the Open Orientation Problem (OOP), similar to OP, with the difference that OP starts and ends at the same control point, while in the OOP, the start and end are different.

The problem is adequately formulated. So, let's take a set P of n interest points and an available time of T_{max}. Let w_i be the score of each point, the duration of the visit to each point be t^i and the time needed to go from point i to point j be $t_{i,j}$. Lastly, let the decision variables be $x_{i,j}$ that indicates if the route goes from point i to point j, v_i that indicates if the point i has been visited and p_i that indicates the position of the point i in the route. Both $x_{i,j}$ and v_i are binary variables that take value 1 while true and 0 otherwise, while p_i takes values from $1, 2, ..., n \in P$.

The OP and the OOP will be solved using the Ant Colony Optimization (ACO) metaheuristic. This is based on the behaviour of ants in the natural world. Ants wander randomly, trying to find a food source, and upon finding it, they return to the colony while laying down pheromone trails. Other ants of the colony that find the same path are likely to follow the trail and stop travelling at random, thus reinforcing the trace if they eventually find the food.

3. Experimentation

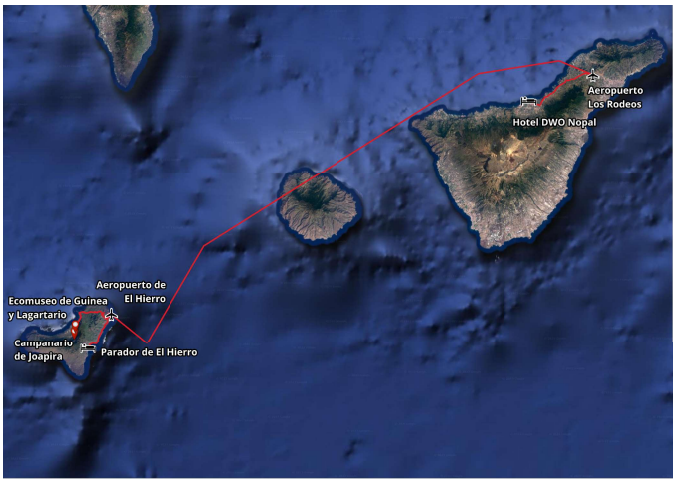
The data for the experimentation was obtained from the ORS plug-in in QGIS. A set of points of interest (POIs) were taken from the islands of Tenerife and El Hierro.



The main goal is to find good routes within the two islands, solving the OP on each island separately and finding a route that allows us to move from one island to the other, solving the OOP with the entirety of the dataset. Both the OP and the OOP were solved using an ACO heuristic implemented on Python.

Table 1: A solution given by the algorithm.

POI Nº	POI
0	Hotel DWO Nopal
20	Aeropuerto Tenerife Norte
37	Aeropuerto de El Hierro
28	Campanario de Joapira
30	Ecomuseo de Guinea y Lagartario
22	Parador de El Hierro



References

[1] Gunawana, A., Lau, H.C., Vansteenwegenb, P. (2016). Orienteering Problem: A survey of recent variants, solution approaches and applications. European Journal of Operational Research, vol. 3(3), pp. 315–332.

[2] Liang, Y.C., Smith, A. (2006) An ant colony approach to the orienteering problem. Journal of the Chinese Institute of Industrial Engineers, vol. 10(5), pp. 403–414.