# Inteligencia Artificial

## Informe Final: Resolviendo Team Orienteering Problem mediante Greedy y Simulated Annealing

José Pinto Muñoz

November 20, 2023

## Evaluación

| | |
|---|---|
| Código Fuente (20%): | ___________ |
| Representación (15%): | ___________ |
| Descripción del algoritmo (20%): | ___________ |
| Experimentos (10%): | ___________ |
| Resultados (10%): | ___________ |
| Conclusiones (20%): | ___________ |
| Bibliografía (5%): | ___________ |
| **Nota Final (100)**: | ___________ |

**Abstract**

En el Team Orienteering Problem (TOP), un equipo de participantes busca maximizar la puntuación total al visitar ubicaciones con puntajes asociados, dentro de límites de tiempo predefinidos. Cada ubicación visitada agrega su puntaje a la puntuación acumulada. Sin embargo, solo uno de los participantes obtiene el puntaje de cada ubicación. El desafío radica en determinar rutas eficientes que permitan a los participantes colaborar para recolectar la máxima puntuación posible, minimizando el solapamiento de ubicaciones visitadas. Este artículo presenta un enfoque investigativo que emplea una solución inicial Greedy y la mejora posteriormente utilizando Simulated Annealing, explorando así estrategias para abordar eficazmente el TOP en diversas situaciones prácticas.

# 1 Introducción

El Team Orienteering Problem (TOP) es un desafío de optimización que surge de la fascinante actividad al aire libre del orienteering, donde los competidores combinan habilidades de navegación y resistencia física para visitar puntos de control dentro de un límite de tiempo. Cada punto de control tiene un puntaje asociado, y el objetivo es maximizar la puntuación total mientras se cumple con la restricción de tiempo. El problema que se ha convertido en un área de interés crucial en la optimización y la toma de decisiones en diversas aplicaciones en el mundo real. Se utiliza en la planificación de recorridos turísticos para maximizar la experiencia del viajero, en la optimización de rutas de entrega para la gestión de la cadena de suministro, y en la programación de visitas técnicas a clientes en empresas de servicios. Además, el TOP desempeña un papel importante en la recopilación de datos ambientales para investigaciones científicas y en la asignación eficiente de agentes de servicio en el sector de atención al cliente. Estas aplicaciones demuestran la versatilidad del TOP en la resolución de desafíos de enrutamiento y asignación con restricciones de tiempo y recursos en una amplia variedad de campos.

Uno de los objetivos de este informe es revisar el Estado del Arte en el campo del Team Orienteering Problem. Este proceso tiene como propósito identificar y analizar las investigaciones previas y los enfoques existentes que han sido aplicados para abordar este problema de optimización. Además de este análisis, se busca destacar las limitaciones y áreas de mejora identificadas en la literatura relacionada con el TOP. Este enfoque tiene como finalidad contribuir a la resolución del problema, proponiendo nuevas perspectivas y enfoques mejorados.

A lo largo de este informe, se abordará la revisión del Estado del Arte, se definirá un modelo matemático y una representación del problema, y se explorarán algoritmos de solución. Este enfoque se diseñará con el propósito de avanzar significativamente en la comprensión y resolución del TOP, contribuyendo así al avance general en este campo de investigación.

Este informe se estructura en secciones clave que abordan distintos aspectos del Team Orienteering Problem. En la Definición del Problema, se proporciona una comprensión sólida de sus fundamentos y contexto. La sección Estado del Arte revisa investigaciones previas, destacando enfoques y soluciones existentes. En el Modelo Matemático, se presenta un modelo de optimización, definiendo parámetros, variables y restricciones. Además, se detalla la Representación adoptada para abordar el TOP, seguida por la Descripción del Algoritmo, que se enfocará inicialmente en un enfoque Greedy y se mejorará con Simulated Annealing. Los Experimentos describirán los escenarios y conjuntos de datos utilizados, mientras que los Resultados presentarán métricas y hallazgos. Finalmente, en Conclusiones, se resumirán los puntos clave y se destacarán las contribuciones, proporcionando una visión general del impacto y las futuras direcciones de resolución del TOP.

## 2 Definición del Problema

El orienteering es un deporte al aire libre que combina habilidades de navegación y resistencia física. Su origen se remonta a fines del siglo XIX y principios del XX en Escandinavia, donde se utilizaba principalmente para fines militares. En este deporte, los competidores navegan a través de un terreno desconocido utilizando un mapa y una brújula. Deben visitar una serie de puntos de control marcados en el terreno en el menor tiempo posible. Cada punto de control tiene un puntaje asociado, y el objetivo es acumular la mayor cantidad de puntos dentro de un límite de tiempo.

En contraste, el Orienteering Problem (OP) es un modelo matemático basado en el deporte de orienteering. En el OP, se considera un conjunto de puntos de control con puntajes asociados. El objetivo es encontrar una ruta óptima que visite algunos de estos puntos de control dentro de un límite de tiempo para maximizar la suma de los puntajes. A diferencia del deporte real, en el OP, no es necesario visitar todos los puntos de control, lo que lo convierte en un problema de optimización combinatoria. El OP ha sido estudiado en la investigación de operaciones y la teoría de la optimización. Se podría decir que uno de los primeros en estudiar este tipo de problemas, fueron S. E. Butt y T. M. Cavalier, introduciendo un problema de recaudación máxima de múltiples recorridos [1]. Luego está el concepto del TOP que es una extensión del Orienteering Problem que introduce el concepto de trabajo en equipo. En el TOP, varios competidores o miembros de un equipo deben visitar puntos de control con puntajes asociados. Sin embargo, solo uno de los participantes obtiene el puntaje del punto de control al visitarlo, lo que significa que las rutas de los participantes deben superponerse lo menos posible. El objetivo es maximizar la suma total de los puntajes obtenidos por el equipo en un límite de tiempo. Esta variante se aplica en situaciones en las que múltiples agentes colaboran para maximizar la eficiencia. Los principales investigadores con un enfoque a un problema de TOP y no de recaudación máxima de múltiples recorridos fueron I. M. Chao, B. L. Golden y E. A. Wasil en 1996 [2], nombrando el problema por como un Problema de Orienteering en Equipos y denominándolo una optimización combinatoria NP-hard, lo cual significa que, no se podría encontrar una solución eficiente (polinómica) para ello, hasta esa fecha. Lo cual implicaría un gran inconveniente y grandes problemas al momento de querer encontrar soluciones, ya que el número de posibles soluciones crece exponencialmente con el tamaño del problema. Esto hace que la búsqueda de una solución óptima sea extremadamente costosa en términos computacionales.

En resumen, el orienteering como deporte se centra en la navegación y la resistencia física, mientras que el OP y el TOP son problemas de optimización matemática basados en el deporte. En el orienteering deportivo, el objetivo es completar un curso en el menor tiempo posible, mientras que, en el OP y el TOP, el objetivo es encontrar rutas que maximicen la suma de los puntajes. El TOP agrega la dinámica de trabajo en equipo, donde los participantes deben coordinar sus rutas para maximizar la puntuación total del equipo sin superponerse en la visita de puntos de control. Una opción podría ser buscar soluciones con variaciones de TSP Selectivo (Problema del Viajero o Viajante selectivo), el cual tiene como objetivo seleccionar y conectar de manera eficiente un subconjunto de ciudades o clientes, maximizando la eficiencia al minimizar la distancia recorrida. Esto lo hace útil en situaciones donde no es necesario visitar todas las ubicaciones, pero aún se busca optimizar la ruta entre un conjunto específico de puntos de control [8]. Es importante destacar que, aunque el Problema del Viajante Selectivo y el TOP pueden parecer similares en ciertos aspectos, no se pueden considerar soluciones intercambiables debido a las diferencias fundamentales en la naturaleza de estos problemas. El *Selective TSP* permite seleccionar un subconjunto de ciudades para visitar y encontrar la ruta más corta que pase por ellas, minimizando la distancia total. Por otro lado, el TOP involucra equipos de competidores que deben coordinar sus rutas para maximizar la puntuación total, sin superponerse en las visitas a los puntos de control. Estas diferencias en los objetivos y restricciones hacen que las soluciones eficientes para uno de los problemas no sean aplicables al otro, ya que abordan desafíos de optimización completamente diferentes.

Además existen diferentes variantes del problema, a continuación mencionaremos algunas referenciando artículos donde se trabajan en ellas:

- Team Orienteering Problem with Time Windows (TOPTW): Esta variante agrega ventanas de tiempo a cada ubicación que se puede visitar. Los participantes deben cumplir con estas ventanas de tiempo durante sus recorridos para recolectar puntajes mientras respetan los límites temporales [16].
- Team Orienteering Problem with Time Dependent Profits (TOPTDP): En TOPTDP, los puntajes asociados a cada ubicación varían con el tiempo. Esto significa que los puntajes no son constantes y pueden cambiar según la hora del día o el momento en que se visite cada ubicación [6].

- Orienteering Problem with Neighborhoods (OPN): Esta variante reemplaza las ubicaciones individuales por vecindarios. Los participantes deben visitar todos los vecindarios en lugar de ubicaciones específicas, lo que puede requerir una estrategia de recorrido diferente para abordar conjuntos de ubicaciones en lugar de puntos individuales [10].
- Multi-Objective Team Orienteering Problem (MOTOP): En MOTOP, se buscan soluciones que optimicen múltiples objetivos simultáneamente. Esto implica maximizar el puntaje total recopilado mientras se minimiza la distancia recorrida o se satisfacen otros criterios competitivos al mismo tiempo [5].

Cada una de estas variantes introduce desafíos adicionales y condiciones específicas que deben considerarse al diseñar algoritmos y estrategias para resolver el problema. Dependiendo de la aplicación específica y los requisitos del problema, una u otra variante podría ser más adecuada para la optimización.

En este escenario, abordaremos una variante de TOP con una serie de parámetros clave. Estos parámetros incluyen el número de rutas a ser planificadas, el número de vértices o clientes que deben ser visitados, el límite de tiempo disponible para cada ruta, los puntajes o recompensas asociados a cada vértice, y las distancias o tiempos requeridos para viajar entre los vértices. Podemos definir los vertices como nodos que contienen un valor y coordenadas, datos que nos sirven para poder calcular la distancia o tiempo entre vértices y la puntuación de cada ruta. Para resolver este problema, introducimos una variable para cada vértice, que indica si el vértice en cuestión es visitado o no por cada equipo. La función objetivo en este contexto busca maximizar la suma de los puntajes acumulados por todos los equipos, cumpliendo con ciertas restricciones. Estas restricciones se dividen en varios aspectos clave: cada ruta debe comenzar en un vértice inicial específico y terminar en un vértice determinado. Además, en ninguna ruta se permite visitar el mismo vértice más de una vez, lo que garantiza la variedad de clientes atendidos. Asimismo, se impone la condición de que cada vértice solo puede ser visitado y premiado una vez, evitando la duplicación de recompensas. Por último, se exige que el tiempo total de recorrido de cada ruta individual no supere el límite de tiempo establecido.

Por lo tanto, el desafío consiste en encontrar las rutas óptimas para cada equipo de manera que maximicen la recompensa total acumulada, garantizando que cada participante seleccione cuidadosamente los clientes que generen la mayor recompensa, sin exceder el límite de tiempo asignado. Este problema es de gran relevancia en aplicaciones logísticas y de planificación, donde múltiples equipos deben coordinar sus recorridos para obtener el máximo beneficio en términos de recompensas. Resolver los problemas de TOP puede presentar diversas dificultades, especialmente en la coordinación eficiente de múltiples equipos. La tarea de asignar rutas de manera óptima para maximizar la recompensa total, considerando las limitaciones de tiempo y las preferencias individuales de cada participante, agrega complejidad al proceso. La gestión adecuada de la interacción entre los miembros del equipo y la selección estratégica de clientes para maximizar la recompensa, al tiempo que se cumplen con las restricciones temporales, son tareas al final de cuenta lo desafiante.

# 3 Estado del Arte

El Problema de Orientación en Equipo (TOP) ha evolucionado a lo largo del tiempo, y su estudio se ha beneficiado de contribuciones significativas en términos de métodos y técnicas de resolución. Los siguientes aspectos destacan las contribuciones clave en el estado del arte de TOP.

El problema, inicialmente denominado Problema de Recolección Máxima de Recorridos Múltiples (MTMCP), fue pioneramente explorado por Butt y Cavalier como ya mencionamos antes. Este problema consiste en encontrar los recorridos óptimos con limitaciones de tiempo, donde cada recorrido visita un conjunto de nodos ponderados en un grafo con el objetivo de maximizar la suma de los pesos de los nodos visitados. En el artículo de Butt y Cavalier [1], se presenta una heurística especialmente diseñada para abordar el MTMCP, destacando tres características sobresalientes. En primer lugar, la heurística garantiza la generación de soluciones factibles siempre que estas existan. En segundo lugar, demuestra una capacidad notable para producir soluciones de alta calidad, ya que en la mayoría de los casos donde se conoce la solución óptima, la heurística la encuentra. Por último, su eficiencia se refleja en su capacidad para resolver eficazmente problemas de gran envergadura en tiempos computacionales muy reducidos. Butt y Cavalier desarrollaron una heurística conocida como MAXIMP (Maximizar Importancia) para abordar el Problema de Recolección Máxima de Recorridos Múltiples (MTMCP). En MAXIMP, se asignan pesos a los pares de nodos en función de su importancia relativa, lo que define una lista de prioridades. Luego, se construyen recorridos parciales siguiendo esta lista, comenzando en el punto de partida y avanzando hacia los siguientes nodos. Los recorridos se seleccionan considerando la importancia y el tiempo disponible para cada uno, y finalmente se combinan para formar una solución completa al MTMCP. Esta heurística, basada en la asignación estratégica de pesos y prioridades, ha demostrado ser efectiva para encontrar soluciones factibles de alta calidad en el MTMCP y ha influido en el desarrollo de enfoques similares para problemas como el TOP. Es importante notar que el MTMCP, aunque comparte algunas similitudes con TOP, no es idéntico a este último. Él se enfoca en recolectar la mayor cantidad de peso posible dentro de un límite de tiempo, mientras que el TOP implica un equipo de competidores que visitan puntos de control con puntajes, maximizando la puntuación total. La diferencia clave radica en que en el MTMCP, un nodo puede ser visitado más de una vez, lo que no es válido en el TOP, donde cada punto de control debe ser visitado a lo sumo una vez por un único competidor del equipo. Por lo que, aunque comparten ciertas características, el MTMCP y el TOP abordan problemas con naturalezas ligeramente distintas. Pero si es un buen punto de partida para poder ir armando el camino de cómo abordar el problema en sí.

Más tarde, se popularizó bajo el nombre de TOP, que se utiliza comúnmente en la literatura para describir este problema de optimización combinatoria. Fue en el campo de la investigación operativa donde el problema adquirió una formulación matemática más rigurosa. I. Chao, B. Golden y E. Wasil dieron un paso importante en la evolución del problema cuando presentaron el artículo titulado *The Team Orienteering Problem* [2]. En este artículo, los autores formalizaron el problema al introducir puntos de inicio y finalización, junto con otros lugares que tienen puntajes asociados. El objetivo principal era determinar múltiples rutas (una para cada miembro del equipo) que maximizaran el puntaje total dentro de un límite de tiempo fijo para cada miembro. Este enfoque cambió significativamente la naturaleza del problema, ya que pasó de ser un desafío deportivo a una cuestión de optimización combinatoria. Además, Chao, Golden y Wasil reconocieron que el TOP, tal como lo definieron, pertenece a la categoría de problemas NP-hard, lo que significa que encontrar una solución óptima en un tiempo razonable es un desafío computacionalmente difícil. Comparado con las formulaciones anteriores del problema, como el Problema de Recolección Máxima de Recorridos Múltiples (MTMCP) y el enfoque heurístico denominado MAXIMP (Maximizar Importancia), el TOP ofrece una ventaja significativa. En primer lugar, el TOP introduce una formalización más clara y realista al considerar puntos de inicio y finalización, reflejando mejor las restricciones del mundo real. Además, la formulación del TOP ha estimulado una mayor investigación académica y ha llevado al desarrollo de algoritmos más avanzados y eficientes para resolver el problema. El artículo de Chao, Golden y Wasil presentó una heurística que utiliza diferentes herramientas, como el intercambio de dos puntos entre dos caminos y el movimiento de un punto de un camino a otro para aumentar la puntuación recopilada. Además, se emplea la Búsqueda Local (algoritmo de 2 opciones) para disminuir el tiempo en un recorrido, lo que puede permitir incluir nuevos puntos y, por lo tanto, aumentar la puntuación recopilada. Estos enfoques y el reconocimiento de la dificultad NP-hard del problema marcaron un hito importante en la evolución del TOP desde sus raíces deportivas y heurísticas iniciales.

A lo largo de los años, se han desarrollado varios intentos para la resolución del TOP. Estos incluyen enfoques meta-heurísticos, que representan un nivel superior de abstracción en la búsqueda de soluciones de alta calidad en problemas de este tipo. Entre las metaheurísticas aplicadas al TOP se incluyen la optimización de Optimización de Colonias de Hormigas (ACO) [7]. Que es una metaheurística inspirada en el comportamiento de las hormigas reales. En el contexto del TOP, cada "hormiga virtual" construye una solución factible al asignar rutas a los vehículos, intentando encontrar un equilibrio entre la distancia recorrida y la carga de trabajo de los vehículos. Estas soluciones se mejoran a través de un procedimiento de búsqueda local que ajusta las rutas de manera iterativa. El algoritmo ACO también emplea feromonas virtuales para comunicar información entre las hormigas, lo que fomenta la exploración y explotación de soluciones. Otra meta-heurística es la Búsqueda Local Guiada (GLS) [12] y Búsqueda de Vecindario Variable Sesgada (SVNS) [13]. Tanto GLS como SVNS se centran en la mejora de las soluciones de manera iterativa. GLS utiliza procedimientos de intensificación y diversificación para ajustar las rutas de los vehículos. SVNS, por otro lado, se destaca por su capacidad para explorar soluciones cercanas y lejanas, empleando procedimientos de intensificación y diversificación para mejorar la calidad de las soluciones. Ambos enfoques son conocidos por su eficacia en la búsqueda local. También, existe el Procedimiento Codicioso de Búsqueda Adaptativa Aleatoria con Reenlace de Ruta (GRASP) [11]. GRASP se basa en un enfoque constructivo y busca crear soluciones iniciales de alta calidad mediante un procedimiento codicioso. En el contexto del TOP, construye rutas de vehículos paso a paso, considerando la relación entre la avidez y la aleatoriedad. Luego, aplica procedimientos de búsqueda local para refinar estas soluciones, alternando entre la reducción de la duración del viaje y el aumento de la puntuación total de la solución. GRASP también utiliza técnicas de reenlace de ruta para mejorar la diversidad de soluciones exploradas.

En 2011 se llevó a cabo una investigación que comparó distintas estrategias para abordar el problema de TOP, como se describe en "The orienteering problem: A survey" [14]. Al contrastar los resultados de la Búsqueda de Vecindario Variable Sesgado (SVNS) con el Procedimiento Codicioso de Búsqueda Adaptativa Aleatoria con Reenlace de Ruta (GRASP) para enfrentar el TOP, ambos enfoques destacan por su eficacia. SVNS logra una brecha promedio del 0.97% con un tiempo de CPU de 3.8 segundos, convirtiéndolo en una opción rápida y altamente precisa. Por otro lado, GRASP, en su variante lenta (FPR), alcanza una brecha promedio ligeramente más baja del 0.39%, aproximándose aún más a la solución óptima, aunque requiere un tiempo de CPU de 5.0 segundos. La variante rápida de GRASP (ESP) es más lenta, con 212.4 segundos, pero ofrece una brecha promedio excepcionalmente baja del 0.04%. En términos generales, si se busca un equilibrio entre la calidad de la solución y la eficiencia computacional, la variante lenta de GRASP podría ser preferible, mientras que la elección entre estos enfoques dependerá de las restricciones de tiempo y la importancia de obtener soluciones óptimas o casi óptimas en un contexto dado.

Después de ese trabajo, se ha producido un despertar de investigaciones enfocadas en la mejora de los modelos y algoritmos para abordar TOP y sus variantes. Estos esfuerzos han resultado en una amplia gama de enfoques y técnicas que buscan optimizar la asignación de rutas y tiempos de servicio bajo diversas restricciones. Por ejemplo, la Optimización de Enjambre de Partículas (PSOiA) [3]. En este enfoque, se utilizó la optimización de enjambre de partículas para abordar la versión clásica del TOP. La principal contribución fue su capacidad para explorar rápidamente una gran cantidad de vecindarios, lo que resultó en un error relativo extremadamente bajo del 0,0005%, superando significativamente a los métodos existentes y detectando la mayoría de las soluciones más conocidas. Luego está el Algoritmo Híbrido de Búsqueda Local e Inicio Aleatorio Codicioso [15]. Para abordar el MC-TOP-MTW, se presentó un algoritmo híbrido que combina la búsqueda local iterada con un procedimiento de búsqueda adaptativa aleatoria codiciosa. Este enfoque demostró su eficacia al lograr una diferencia de puntuación promedio del 5,19% en comparación con soluciones conocidas de alta calidad, y todo esto en un tiempo computacional muy breve, utilizando solo 1,5 segundos en promedio. Y luego apareció la Revisión Integral de las Variantes del OP [4]. La cual realizó un estudio exhaustivo de las variantes recientes del Problema de Orientación (OP), que incluyen el OP estocástico, el OP generalizado y el OP multiagente. Además de analizar estos problemas, se proporcionó un análisis en profundidad de los enfoques de solución correspondientes. También se destacaron las diversas aplicaciones prácticas del OP, como el diseño de viajes turísticos y el crowdsourcing móvil. Esta revisión identificó áreas prometedoras para futuras investigaciones y contribuyó al avance continuo en el campo del OP y sus aplicaciones.

# 4 Modelo Matemático

El modelo matemático que se presenta a continuación se basa en el modelo básico del TOP y toma como referencia el trabajo realizado por Vansteenwegen, Souffriau y Van Oudheusden en su encuesta sobre el problema de orientación [14].

El modelo se define mediante una serie de parámetros, variables de decisión y restricciones que capturan las características esenciales del TOP. Los parámetros incluyen el número de rutas, el número de vértices, el tiempo máximo permitido, los puntajes asociados a cada vértice y las coordenadas de los vértices en el plano. Las variables de decisión determinan si se visita un vértice en una ruta específica y en qué orden se visitan los vértices. La función objetivo busca maximizar la puntuación total recopilada por el equipo de orientadores.

Las restricciones del modelo garantizan que cada ruta comienza y termina en el primer y último vértice, que cada vértice se visita a lo sumo una vez, que se mantenga la conectividad en cada ruta y que se cumpla el límite de tiempo para cada ruta individual. Además, se implementan restricciones para evitar subgiros y asegurar que las rutas sean coherentes.Este modelo proporciona una base sólida para abordar el TOP y se basa en investigaciones previas que han contribuido a definir y resolver este desafío de optimización.

## Parámetros:

$m$ : Número de rutas
$n$ : Número de vértices
$t_{\max}$ : Tiempo límite
$S_i$ : Puntaje asociado al vértice $i$
$C_i(x_i, y_i)$ : Coordenada en el plano del vértice $i$
$t_{ij} = \sqrt{(C(x_i) - C(x_j))^2 + (C(y_i) - C(y_j))^2}$ : Tiempo entre los vértice $i$ y $j$

## Variables de Decisión:

$$x_{ijp} = \begin{cases} 1 & \text{si, en ruta p , una visita al vértice } i \text{ es seguida por una visita al vértice } j \\ 0 & \text{en otro caso} \end{cases} \tag{1}$$

$$y_{ip} = \begin{cases} 1 & \text{si el vértice } i \text{ es visitado en la ruta } p \\ 0 & \text{en otro caso} \end{cases} \tag{2}$$

## Función Objectivo:

Maximizar la puntuación total recopilada:

$$\text{Maximizar } Z = \sum_{p=1}^{m} \sum_{i=2}^{n-1} S_i \cdot y_{ip} \tag{3}$$

## Dominio:

$$x_{ijp}, y_{ip} \in \{1, 0\}$$
$$\forall i, j = 1, \ldots, n$$
$$\forall p = 1, \ldots, m$$

## Restricciones:

- Cada ruta comienza en el primer vértice y termina en el vértice n-ésimo:

$$\sum_{p=1}^{m} \sum_{j=2}^{n} x_{1jp} = \sum_{p=1}^{m} \sum_{j=1}^{n-1} x_{inp} = m \tag{4}$$

- Se garantizan que cada vértice sea visitado como máximo una vez:

$$\sum_{p=1}^{m} y_{kp} \leq 1, \quad \forall k = 2, \ldots, n-1 \tag{5}$$

- Se garantizan la conectividad de cada camino:

$$\sum_{i=1}^{n-1} x_{ikp} = \sum_{j=2}^{n} y_{kp}, \quad \forall k = 2, \ldots, n-1, \quad \forall p = 1, \ldots, m \tag{6}$$

- El tiempo de recorrer cada ruta por separado no excede $t_{max}$:

$$\sum_{i=1}^{n-1}\sum_{j=2}^{n} t_{ij} \cdot x_{ijp} \leq t_{\max}, \quad \forall p = 1, \ldots, m \tag{7}$$

# 5 Representación

La representación de la solución para abordar el Team Orienteering Problem se estructura mediante el uso de nodos, rutas y una solución. Cada nodo, que denota una ubicación en el plano, se caracteriza por un identificador único, coordenadas y un puntaje asociado. Esta representación proporciona información detallada sobre cada punto de interés, fundamental para la formulación de las rutas. La representación de las rutas implica un conjunto de nodos ordenados, reflejando el recorrido de un equipo. Cada ruta se define con un identificador único, el conjunto de nodos que la compone, la distancia total o tiempo necesario para cubrirla, y el puntaje acumulado al visitar los nodos específicos.

Esta estructura permite capturar la esencia del desafío, considerando tanto las restricciones de tiempo como el objetivo de maximizar la puntuación total recopilada. La solución del problema se representa como un vector que contiene todas las rutas generadas para los diferentes equipos participantes. Este vector se compone de instancias de la estructura de rutas previamente definida. Esta representación facilita la implementación de algoritmos de mejora, como Simulated Annealing, al proporcionar una estructura clara y modular para manipular y actualizar las rutas. La flexibilidad de esta representación permite adaptarse a diferentes contextos y optimizar la eficiencia de los algoritmos de resolución del TOP.

En la Tabla 1, se presenta la instancia de ejemplo con información detallada sobre los nodos, cada uno identificado por un ID único. Cada nodo tiene coordenadas $(X, Y)$ que representan su posición en el plano, y se asocia un Puntaje específico. Este conjunto de datos es esencial para la formulación del problema.

En la Tabla 2, se muestra la representación del problema, donde se presentan las rutas generadas para la instancia en cuestión. Las rutas están claramente separadas, indicando el recorrido de cada equipo. último nodo, cumpliendo con las restricciones del problema. Los nodos en las rutas se identifican por su ID, y se incluye información adicional sobre sus coordenadas y puntajes asociados.

La estructura de la Tabla 2 permite visualizar cómo se han asignado los nodos a diferentes equipos, respetando las restricciones de tiempo y maximizando la puntuación total acumulada. Cada sección de la tabla representa una ruta específica, destacando la solución propuesta para la instancia de ejemplo del TOP.

Es importante destacar que la representación basada en el artículo de Shih-Wei Lin [9] presentada en la Tabla 2 no garantiza la factibilidad de las soluciones, ya que no considera las restricciones de tiempo. Existe la posibilidad de que, al sumar los tiempos de visita de los nodos en una ruta, se supere el límite de tiempo asignado. Este desafío específico relacionado con la factibilidad de las soluciones se abordará y discutirá detalladamente en la sección correspondiente del algoritmo, donde se describirán las estrategias implementadas para asegurar que las soluciones generadas cumplan con todas las restricciones establecidas, especialmente aquellas relacionadas con el límite de tiempo.

| ID | X | Y | Puntaje |
|---|---|---|---|
| 1 | 4.600 | 7.100 | 0 |
| 2 | 5.700 | 11.400 | 20 |
| 3 | 4.400 | 12.300 | 20 |
| 4 | 2.800 | 14.300 | 30 |
| 5 | 3.200 | 10.300 | 15 |
| 6 | 3.500 | 9.800 | 15 |
| 7 | 4.400 | 8.400 | 10 |
| 8 | 7.800 | 11.000 | 0 |

Table 1: Datos instancia 1 de ejemplo

| ID | X | Y | Puntaje |
|---|---|---|---|
| 1 | 4.600 | 7.100 | 0 |
| 7 | 4.400 | 8.400 | 10 |
| 2 | 5.700 | 11.400 | 20 |
| 8 | 7.800 | 11.000 | 0 |
| 1 | 4.600 | 7.100 | 0 |
| 6 | 3.500 | 9.800 | 15 |
| 8 | 7.800 | 11.000 | 0 |

Table 2: Posible solución a instancia 1

# 6 Descripción del algoritmo

El algoritmo propuesto para abordar el Team Orienteering Problem se estructura en cuatro etapas fundamentales. La primera fase involucra la lectura de los datos de la instancia y su posterior organización en una representación coherente que facilite el procesamiento de la información. En segunda instancia, se desarrolla una solución inicial mediante un enfoque constructivo basado en un algoritmo Greedy, que establece la base para las mejoras posteriores. La tercera etapa se centra en mejorar la solución inicial empleando un algoritmo reparador basado en Simulated Annealing (SA). Sin embargo, se aplican tres configuraciones distintas de temperatura inicial, adaptadas según la cantidad de nodos, equipos y el tiempo máximo disponible. Esta variabilidad en los parámetros permite explorar un amplio espectro de posibles soluciones y evaluar su efectividad en diferentes contextos de problemas. La fase final del algoritmo se enfoca en seleccionar la mejor solución entre las tres alternativas generadas con distintos valores de temperatura inicial. Este enfoque de comparación permite establecer criterios de mejora y analizar la influencia directa de la temperatura inicial en la calidad de las soluciones obtenidas en cada instancia del problema.

## 6.1 Lectura de Datos y estructuras

La primera etapa del algoritmo se inicia con la adquisición de los datos de entrada correspondientes a la instancia del problema. Estos datos contienen información crucial: el número total de nodos o vértices, representado por $n$; el número de equipos o rutas que intervienen en el desafío, representado por $m$; y el límite de tiempo, $tmax$, que cada equipo tiene para recorrer los nodos asignados. A partir de estos datos iniciales, se extraen las coordenadas espaciales y el puntaje asociado a cada nodo. Mediante una función de lectura, se genera una instancia que almacena estos valores, estableciendo así los fundamentos para el proceso de resolución del problema. Se procede a inicializar un vector denominado *nodos iniciales*, que contiene la información detallada de cada nodo, considerando sus coordenadas y puntaje asociado. Paralelamente, se crea una estructura de datos denominada *ruta*, la cual consiste en un conjunto de nodos que forman una ruta específica, incluyendo el puntaje total obtenido al recorrer dichos nodos. La construcción de estas estructuras de datos, tanto de nodos como de rutas, sienta las bases para la formación de una solución. Esta solución se representa mediante un vector que contiene las rutas generadas, estableciendo así un marco inicial sobre el cual se desarrollarán las fases subsiguientes del algoritmo de resolución del problema.

## 6.2 Solución inicial

Después de leer los datos y formar el conjunto de nodos, la segunda fase del algoritmo se basa en construir una solución inicial empleando un enfoque Greedy. Este proceso se realiza sin utilizar recursión, sino a través de ciclos iterativos simples (for loops), haciendo uso exclusivo de vectores de nodos y una función de evaluación para asegurar el cumplimiento de las restricciones y la factibilidad. El algoritmo comienza generando rutas individuales para cada equipo, todas partiendo desde el nodo inicial. A continuación, se establece un conjunto de nodos disponibles, excluyendo el nodo inicial y el final. En cada iteración, se evalúa la posibilidad de agregar cada nodo disponible a las rutas existentes, considerando su contribución al puntaje total y las restricciones de tiempo. Es crucial mencionar que la evaluación de cada opción se basa en la representación mediante vectores de nodos y en la función de evaluación, que toma en cuenta tanto el puntaje como las penalizaciones por exceder los límites temporales. El objetivo es optimizar localmente, eligiendo en cada paso la mejor opción para maximizar el puntaje acumulado. Este enfoque asegura el cumplimiento de las restricciones de tiempo y la factibilidad de la solución inicial. Este proceso continúa hasta que se hayan considerado todos los nodos disponibles, luego de esto aseguramos agregar el nodos final (el de llegada del probelema) a cada ruta de esta solución incial.

```
funcion Evaluacion(rutas, inst):
    total_score = 0
    Para cada ruta en rutas:
        total_score += CalcularPuntajeTotal(ruta.waypoints)
    Para cada ruta en rutas:
        Si CalcularTiempoTotal(ruta.waypoints) > inst.tmax:
            total_score −= 1000000
    Devolver total_score
```

Listing 1: Función de evaluación para calcular el puntaje total y penalizar si se excede el tiempo límite.

## 6.3 Simulated Annealing

En la tercera etapa, se implementa el algoritmo de mejora de la solución inicial mediante el método de Simulated Annealing, un enfoque que incorpora características aleatorias y no deterministas. Esto implica que al aplicarlo en múltiples ocasiones a la misma instancia, se esperan resultados diferentes. El SA busca mejorar la solución actual permitiendo transiciones a soluciones que puedan empeorar la función de evaluación, lo cual facilita escapar de los óptimos locales. Este método admite movimientos que degradan la calidad de la función de evaluación según una probabilidad asociada a la temperatura del sistema. Al inicio, la temperatura es alta, lo que permite transiciones entre estados y aceptación de soluciones que empeoran la evaluación con mayor probabilidad. A medida que la temperatura disminuye, las soluciones que empeoran la evaluación tienen menos probabilidad de ser aceptadas. Sin embargo, las soluciones que mejoran siempre son aceptadas.

En la aplicación del algoritmo SA a este problema, se utiliza la solución inicial generada con el algoritmo Greedy, ya que el SA se enfoca en mejorar y no en construir soluciones. Se inicializan variables para las iteraciones: una variable de temperatura (*temp*), un *delta* que refleja la diferencia de puntaje de la función de evaluación entre iteraciones, y un número de iteraciones *n inter*). Además, se crea un vector de rutas *best ruta* que almacena la mejor solución obtenida hasta el momento, y un vector de rutas *rutas nuevas*. En cada iteración, se genera una nueva posible solución (*rutas nuevas*) mediante un movimiento que combina el swap y la inserción de nodos no utilizados. Este movimiento modifica una ruta seleccionada al azar, cambiando un nodo aleatorio de esta ruta por uno de los nodos no utilizados. Este proceso se repite hasta obtener una solución factible que no exceda el límite de tiempo establecido. Posteriormente, se evalúa esta nueva solución y se calcula el delta entre la solución actual y la anterior. Se determina la probabilidad de aceptación mediante una fórmula basada en la diferencia de puntajes.

$$P(.) = e(delta/temperatura)$$

Esta fórmula matemática asegura que la probabilidad sea mayor cuando la nueva solución es mejor que la anterior. Se realiza una comparación entre la nueva solución y la mejor solución obtenida hasta el momento (*best rutas*). Si la nueva solución es mejor, se actualiza *best rutas*. Luego de completar las iteraciones, se devuelve la última solución generada (*rutas nuevas*) y la mejor solución (*best rutas*).

```
funcion SimulatedAnnealing(instancia, nodos_iniciales, rutas, temp0, n_iter):
    temp = temp0
    puntaje_actual = Evaluacion(rutas, instancia)
    mejor_solucion = rutas
    Para i desde 1 hasta n_iter:
        rutas_nuevas = RealizarSwap(instancia, nodos_iniciales, rutas)
        puntaje_nuevo = Evaluacion(rutas_nuevas, instancia)
        delta = puntaje_nuevo − puntaje_actual
        Si puntaje_actual > 0:
            Si Aceptar(delta, temp):
                rutas = rutas_nuevas
                puntaje_actual = puntaje_nuevo
                Si puntaje_actual > Evaluacion(mejor_solucion, instancia):
                    mejor_solucion = rutas
        temp = ReducirTemperatura(temp)
    Devolver mejor_solucion, rutas_nuevas
```

Listing 2: Función que ejecuta el algoritmo de Simulated Annealing.

## 6.4 Soluciones según Temperaturas

En la cuarta fase, implementamos el algoritmo de Simulated Annealing tres veces, cada una con una temperatura inicial distinta. Estas temperaturas iniciales se ajustan considerando datos específicos de la instancia del problema. La primera temperatura inicial se determina en función de la cantidad de nodos iniciales en la instancia. La segunda temperatura inicial se establece en relación con la cantidad de rutas o equipos presentes en la instancia. Por último, la tercera temperatura inicial se configura según el tiempo límite definido en la instancia.

$$temp0inst = n * 10$$
$$temp0rutas = m * 10$$
$$temp0tmax = tmax * 10$$

*En este caso multiplicamos los parámetros por* 10 *considerando que realizaremos* 1000 *iteraciones.*

Cada ejecución del algoritmo SA se inicia con la solución inicial generada por el algoritmo Greedy. Las tres ejecuciones con diferentes temperaturas iniciales proporcionan tres soluciones mejoradas distintas. De estas tres soluciones, se elige la mejor como solución final. A pesar de que los resultados de los tres algoritmos SA se imprimen en el código, únicamente se selecciona la mejor solución de entre ellos para ser presentada como la solución final en el archivo correspondiente. Esta etapa final del algoritmo garantiza la selección de un rango de temperaturas iniciales adecuado para cada instancia. Esto evita valores que podrían resultar insuficientes para reducir la temperatura a lo largo del proceso. De esta manera, se asegura una exploración y explotación personalizadas para cada instancia, lo que optimiza la búsqueda de soluciones óptimas.

# 7 Experimentos

Greedy al ser determinista no requiere experimentación repetida, ya que són dadas las mismas condiciones iniciales. Sin embargo, en el caso del Simulated Annealing, al ser un algoritmo estocástico, es crucial realizar múltiples pruebas para observar la variabilidad en los resultados debido a su naturaleza aleatoria.

Los objetivos de los experimentos se centraron en evaluar el rendimiento del algoritmo bajo diversas instancias, abarcando un amplio espectro de parámetros. Se buscaba poner a prueba el algoritmo en condiciones que representaran situaciones variadas, desde instancias pequeñas y simples hasta aquellas de mayor complejidad y tamaño. Este análisis permitiría comprender el comportamiento del algoritmo frente a diferentes configuraciones de nodos, rutas y límites de tiempo.

Se llevaron a cabo un total de 6 pruebas, divididas en dos aspectos distintos. El primero consistió en comparar tres instancias completamente diferentes con parámetros dispares. Las instancias *p2.2.a.txt* con $n = 21$, $m = 2$ y $t_{\max} = 7.5$, *p6.2.d.txt* con $n = 64$, $m = 2$ y $t_{\max} = 15.0$, y *p7.2.c.txt* con $n = 102$, $m = 2$ y $t_{\max} = 30.0$, fueron utilizadas para evaluar el algoritmo bajo condiciones extremas.

El segundo aspecto se enfocó en comparar instancias con un número fijo de nodos, pero variando otros parámetros. Se utilizaron los archivos p2.2.k.txt con $n = 21$, $m = 2$ y $t_{\max} = 22.5$, p2.3.a.txt con $n = 21$, $m = 3$ y $t_{\max} = 5.0$, y p2.4.k.txt con $n = 21$, $m = 4$ y $t_{\max} = 11.2$, para observar cómo el algoritmo se comporta ante cambios mínimos en parámetros específicos.

Los experimentos se ejecutaron en un MacBook Pro con procesador Apple M2 y 8 GB de RAM, utilizando macOS Sonoma (14.0). Esta elección de hardware se realizó para garantizar la ejecución sin inconvenientes, aprovechando la potencia del procesador y la capacidad de memoria. El tiempo de ejecución no se consideró como criterio de evaluación, centrándose el análisis en la calidad de las soluciones respecto al número de iteraciones.

Además, se utilizaron semillas de números aleatorios basadas en el tiempo, asegurando la aleatoriedad en cada ejecución. Los indicadores utilizados para la evaluación fueron el número de evaluaciones y los puntajes finales obtenidos en cada solución.

# 8 Resultados

Los resultados obtenidos muestran una mejora significativa en los puntajes mediante la aplicación del algoritmo Simulated Annealing sobre las soluciones generadas por el algoritmo Greedy. En la comparación de instancias diferentes, se observa un aumento notable en los puntajes de las soluciones mejoradas por SA en todos los casos. Por ejemplo, en el archivo p2.2.a.txt, el puntaje mejorado pasó de 70 a 80 puntos, en p6.2.d.txt de 12 a 36 puntos, y en p7.2.c.txt de 46 a 64 puntos. Estos resultados sugieren que SA logra mejorar consistentemente las soluciones iniciales generadas por Greedy en una variedad de instancias.

Al comparar instancias similares, se observa que, aunque las soluciones iniciales generadas por Greedy muestran puntajes similares antes de la mejora, las mejoras logradas por SA difieren entre las instancias. Por ejemplo, en p2.2.k.txt, el puntaje mejorado se mantuvo en 90 puntos, igualando el puntaje Greedy, mientras que en p2.3.a.txt y p2.4.k.txt, los puntajes mejorados permanecieron iguales a los puntajes Greedy. Estos resultados resaltan la sensibilidad de SA a las características específicas de cada instancia, lo que indica que para ciertos casos, las soluciones generadas por Greedy podrían ser óptimas o no mejorable con las configuraciones de temperatura inicial empleadas.

| Archivo | Instancia | Greedy | SA | N de Evaluaciones |
|---|---|---|---|---|
| p2.2.a.txt | $n = 21$, $m = 2$, $t_{\max} = 7.5$ | 70 pts | 80 pts | 28,551 |
| p6.2.d.txt | $n = 64$, $m = 2$, $t_{\max} = 15.0$ | 12 pts | 36 pts | 23,442 |
| p7.2.c.txt | $n = 102$, $m = 2$, $t_{\max} = 30.0$ | 46 pts | 64 pts | 119,012 |

Table 3: Comparación de Instancias Diferentes

| Archivo | Instancia | Greedy | SA | N de Evaluaciones |
|---|---|---|---|---|
| p2.2.k.txt | $n = 21$, $m = 2$, $t_{\max} = 22.5$ | 80 pts | 90 pts | 8,589 |
| p2.3.a.txt | $n = 21$, $m = 3$, $t_{\max} = 5.0$ | 60 pts | 60 pts | 14,807 |
| p2.4.k.txt | $n = 21$, $m = 4$, $t_{\max} = 11.2$ | 160 pts | 160 pts | 160,069 |

Table 4: Comparación de Instancias Similares

Los resultados obtenidos muestran una variación significativa en el número de evaluaciones entre las diferentes instancias, lo que coincide con la complejidad de cada problema. Es evidente que las instancias con un mayor número de nodos tienden a requerir más evaluaciones debido a la mayor cantidad de decisiones y posibles configuraciones que el algoritmo debe explorar. Sin embargo, se observaron diferencias notables en el número de evaluaciones incluso cuando se mantenía constante la cantidad de nodos.

Por ejemplo, al comparar las instancias p2.2.a.txt y p6.2.d.txt, ambas con 21 nodos pero con tiempos máximos de 7.5 y 15.0 respectivamente, se encontró una diferencia de aproximadamente 5000 evaluaciones, lo que sugiere una influencia significativa del tiempo máximo en el número de iteraciones necesarias para mejorar la solución inicial. Similarmente, al comparar p2.2.a.txt con p2.4.k.txt, ambas con el mismo número de nodos pero con 2 y 4 rutas respectivamente, se encontró una diferencia notable de más de 130000 evaluaciones.

Es esencial destacar que se mantuvo un límite de 1000 iteraciones para el algoritmo SA en todos los casos. Estas diferencias en el número de evaluaciones, incluso con instancias de dimensiones similares pero parámetros distintos, resaltan la influencia que el tiempo máximo y la cantidad de rutas tienen en la complejidad del problema y en la necesidad de iteraciones adicionales para lograr mejoras significativas en la solución.

Es importante mencionar que los experimentos se realizaron sin errores ni bucles infinitos. Se llevó a cabo una revisión de los resultados donde el algoritmo SA no logró mejorar la solución generada por Greedy. En estos casos, se repitió la prueba de instancia para descartar posibles coincidencias de números aleatorios, asegurando la consistencia de los resultados y confirmar que la solución Greedy no pudo mejorarse con las configuraciones utilizadas para SA. Este método entrega confiabilidad de las pruebas realizadas, así como la variabilidad en la capacidad de SA para mejorar soluciones Greedy según las particularidades de cada instancia.

# 9 Conclusiones

Las conclusiones relevantes obtenidas de este estudio revelan una mejora significativa en los puntajes al aplicar el algoritmo Simulated Annealing (SA) sobre las soluciones generadas por el algoritmo Greedy. La comparación entre instancias diferentes muestra un aumento notable en los puntajes mejorados por SA en todos los casos analizados. En instancias como p2.2.a.txt, p6.2.d.txt y p7.2.c.txt, se observaron incrementos significativos, pasando de 70 a 80 puntos, de 12 a 36 puntos, y de 46 a 64 puntos respectivamente. Estos resultados sugieren que SA logra mejorar consistentemente las soluciones iniciales generadas por Greedy en una variedad de instancias.

Sin embargo, al comparar instancias similares, se evidenció que las mejoras logradas por SA difieren entre las instancias, a pesar de que las soluciones iniciales generadas por Greedy mostraban puntajes similares antes de la mejora. Por ejemplo, en p2.2.k.txt, el puntaje mejorado se mantuvo en 90 puntos, igualando el puntaje Greedy, mientras que en p2.3.a.txt y p2.4.k.txt, los puntajes mejorados permanecieron iguales a los puntajes Greedy. Estos resultados resaltan la sensibilidad de SA a las características específicas de cada instancia, indicando que para ciertos casos, las soluciones generadas por Greedy podrían considerarse óptimas o no mejorables con las configuraciones de temperatura inicial empleadas.

Los experimentos también revelaron una variación significativa en el número de evaluaciones entre diferentes instancias. A pesar de mantener un límite de 1000 iteraciones para el algoritmo SA en todos los casos, se observaron diferencias notables en el número de evaluaciones incluso en instancias de dimensiones similares pero con parámetros distintos. Por ejemplo, entre p2.2.a.txt y p6.2.d.txt, con el mismo número de nodos pero con tiempos máximos diferentes, se encontró una diferencia de aproximadamente 5000 evaluaciones. Esto sugiere una influencia significativa del tiempo máximo en la cantidad de iteraciones necesarias para mejorar la solución inicial.

A lo largo de este informe, se ha trazado un camino organizado para abordar el problema del Team Orienteering. Este enfoque ha cubierto desde la definición del problema hasta la exploración de distintas estrategias para resolverlo. Se han presentado modelos matemáticos y se ha detallado cómo se representan estos problemas. Estos métodos incluyen desde un enfoque inicial directo hasta una mejora utilizando Simulated Annealing. Los experimentos realizados implicaron describir los escenarios y los datos utilizados, presentando luego los resultados obtenidos a través de pruebas intensivas. Lo relevante aquí es que, al utilizar Simulated Annealing para mejorar las soluciones iniciales generadas por el algoritmo Greedy, se lograron mejoras notables en los puntajes en diferentes situaciones. Esto destaca la capacidad de Simulated Annealing para adaptarse a las particularidades de cada problema y mejorar las soluciones iniciales.

El algoritmo Simulated Annealing presenta ventajas significativas en la resolución de problemas de optimización, especialmente cuando se trata de encontrar soluciones cercanas al óptimo global en entornos no lineales o con múltiples óptimos locales. Su capacidad para explorar soluciones subóptimas inicialmente y aceptar movimientos que empeoran la función objetivo, con la probabilidad adecuada, le permite escapar de mínimos locales y realizar una búsqueda más exhaustiva del espacio de soluciones. Sin embargo, su efectividad depende en gran medida de la configuración de parámetros, como la temperatura inicial y la tasa de enfriamiento. Además, la implementación adecuada de SA puede ser compleja y la búsqueda de buenas configuraciones de parámetros puede requerir tiempo y recursos computacionales considerables. Además, su rendimiento puede ser sensible a la elección de la función de enfriamiento y a la calidad de la función de evaluación del problema en cuestión.

En cuanto al trabajo futuro, estos resultados sugieren la necesidad de explorar configuraciones adicionales de temperatura inicial y parámetros de SA para evaluar su impacto en la mejora de las soluciones Greedy en diferentes escenarios. Los factores que no se evaluaron en las pruebas son la temperatura variable respecto al número de iteraciones. Otro factor no revisado fue la prueba de otro tipo de movimientos, como por ejemplo Swap entre rutas de la solición o inserción de nodos sin eliminar nodos de la solución. Además, investigar otras técnicas de mejora que puedan complementar o potenciar el rendimiento de SA en situaciones donde su efectividad pueda ser limitada.

# 10 Bibliografía

## References

[1] Steven E. Butt and Tom M. Cavalier. A heuristic for the multiple tour maximum collection problem. *Computers Operations Research*, 21(1):101–111, 1994.

[2] I-Ming Chao, Bruce L. Golden, and Edward A. Wasil. The team orienteering problem. *European Journal of Operational Research*, 88(3):464–474, 1996.

[3] Duc-Cuong Dang, Rym Nesrine Guibadj, and Aziz Moukrim. An effective pso-inspired algorithm for the team orienteering problem. *European Journal of Operational Research*, 229(2):332–344, 2013.

[4] Aldy Gunawan, Hoong Chuin Lau, and Pieter Vansteenwegen. Orienteering problem: A survey of recent variants, solution approaches and applications. *European Journal of Operational Research*, 255(2):315–332, 2016.

[5] Wanzhe Hu, Mahdi Fathi, and Panos M. Pardalos. A multi-objective evolutionary algorithm based on decomposition and constraint programming for the multi-objective team orienteering problem with time windows. *Applied Soft Computing*, 73:383–393, 2018.

[6] Hossein Jandaghi, Ali Divsalar, and Saeed Emami. The categorized orienteering problem with count-dependent profits. *Applied Soft Computing*, 113:107962, 2021.

[7] Liangjun Ke, Claudia Archetti, and Zuren Feng. Ants can solve the team orienteering problem. *Computers Industrial Engineering*, 54(3):648–665, 2008.

[8] Gilbert Laporte and Silvano Martello. The selective travelling salesman problem. *Discrete Applied Mathematics*, 26(2):193–207, 1990.

[9] Shih-Wei Lin. Solving the team orienteering problem using effective multi-start simulated annealing. *Applied Soft Computing*, 13(2):1064–1073, 2013.

[10] Robert PÄniÄka, Jan Faigl, and Martin Saska. Variable neighborhood search for the set orienteering problem and its application to other orienteering problem variants. *European Journal of Operational Research*, 276(3):816–825, 2019.

[11] Wouter Souffriau, Pieter Vansteenwegen, Greet Vanden Berghe, and Dirk Van Oudheusden. A path relinking approach for the team orienteering problem. *Computers & operations research*, 37(11):1853–1859, 2010.

[12] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. A guided local search metaheuristic for the team orienteering problem. *European Journal of Operational Research*, 196(1):118–127, 2009.

[13] Pieter Vansteenwegen, Wouter Souffriau, Greet Vanden Berghe, and Dirk Van Oudheusden. *Metaheuristics for Tourist Trip Planning*, pages 15–31. Springer Berlin Heidelberg, Berlin, Heidelberg, 2009.

[14] Pieter Vansteenwegen, Wouter Souffriau, and Dirk Van Oudheusden. The orienteering problem: A survey. *European Journal of Operational Research*, 209(1):1–10, 2011.

[15] Greet Vanden Berghe Dirk Van Oudheusden Wouter Souffriau, Pieter Vansteenwegen. The multiconstraint team orienteering problem with multiple time windows. *Transportation Science 47(1):53-63*, (2011).

[16] Vincent F. Yu, Nabila Yuraisyah Salsabila, Shih-Wei Lin, and Aldy Gunawan. Simulated annealing with reinforcement learning for the set team orienteering problem with time windows. *Expert Systems with Applications*, 238:121996, 2024.