Diseño de una ruta turística, con inclusión de arcos, ventanas de tiempo y congestión

Lilian Alejandra Pinzón Rodríguez

Escuela Colombiana de Ingeniería Julio Garavito
Decanatura de Ingeniería Industrial
Maestría en Ingeniería Industrial
Bogotá D.C., Colombia
2020

Diseño de una ruta turística, con inclusión de arcos, ventanas de tiempo y congestión

Lilian Alejandra Pinzón Rodríguez

Trabajo de investigación para optar al título de Magíster en Ingeniería Industrial

Director
Angélica Sarmiento Lepesqueur
Magister en Gerencia de Operaciones

Escuela Colombiana de Ingeniería Julio Garavito
Decanatura de Ingeniería Industrial
Maestría en Ingeniería Industrial
Bogotá D.C., Colombia
2020

.

Publicado en 2020 por la Escuela Colombiana de Ingeniería Julio Garavito. Avenida 13 No 205-59
Bogotá. Colombia.
TEL: +57 – 1 668 36 00

Agradecimientos

Primero que todo, gracias a la Escuela Colombiana de Ingeniería por su acompañamiento durante la realización de este trabajo de grado y la oportunidad de poder realizar doble titulación con la Universidad de Tecnología en Troyes - Francia. Igualmente, gracias a Angélica Sarmiento y David Cortés por todo su apoyo y conocimiento que me permitieron tener todas las herramientas para poder desarrollar un trabajo de calidad.

Finalmente, gracias a mi familia, mi novio, mis compañeros y amigos por toda su motivación y soporte durante este tiempo.

Resumen

En esta tesis, el problema de diseño de la ruta turística o TTDP por sus siglas en inglés (Tourist Trip Design Problem) es estudiado, considerando las ventanas de tiempo de los POI (points of interest), los diferentes modos de transporte para ir de un sitio a otro, los momentos de congestión en cada uno de ellos y los arcos turísticos que pueden pertenecer a la ruta, con el objetivo de maximizar la satisfacción del cliente. Se propone un modelo matemático de programación lineal entera-mixta y un método heurístico con un VND para la búsqueda local, con inclusión de un método exacto para selección de tiempos de llegada y un método con múltiples inicios como medida de diversificación. Se realiza una comparación computacional entre estos dos métodos utilizando instancias construidas con información real de POI de la ciudad de Bogotá. Los experimentos computacionales evalúan el rendimiento de ambos métodos en términos de la función objetivo y el tiempo de resolución en segundos, los resultados muestran un buen desempeño del método heurístico, logrando llegar al óptimo para las instancias más pequeñas y con un gap de menos del 1 % para las que no lo hacen. Para instancias más grandes, el método heurístico logra encontrar buenas soluciones en un tiempo de cómputo razonable.

Abstract

In this thesis, the Tourist Trip Design Problem (TTDP) is studied, considering the time windows of the POIs (points of interest), the different modes of transport to go from one POI to another, the moments of congestion in each of them and the tourist arcs that may belong to the route, with the objective of maximizing customer satisfaction. We propose a mathematical model of linear enter-mixed programming and a heuristic method with a VND for local search, including an exact method for selection of arrival times and a method with multiple starts as a measure of diversification. A computational comparison is made between these two methods using instances built with actual POI information from the city of Bogotá. The computational experiments evaluate the performance of both methods in terms of the objective function and the resolution time in seconds. The results show a good performance of the heuristic method, reaching the optimum for the smallest instances and with a gap of less than 1 % for those who do not. For bigger instances, the heuristic method finds good solutions in a reasonable time of computation.

Índice

Índice de cuadros

Índice de figuras

1. Introducción

1.1. Justificación

El turismo es uno de los sectores económicos más importantes a nivel mundial y el cual ha experimentado un gran crecimiento en los últimos años, como lo expresa Rodríguez et al. (2012), por tal razón cada vez son más los trabajos investigativos que se encuentran orientados en este tema.

Xiao et al. (2017) explica que los turistas tienen gran variedad de decisiones a tomar a la hora de hacer un viaje, una de ellas es la elección de la ruta turística a recorrer en una ciudad, la cual en muchas ocasiones es tomada de manera subjetiva y no necesariamente se elige la ruta que genera el máximo beneficio, según las condiciones del viaje.

Rodríguez et al. (2012) menciona que el anterior problema cuenta con características que le generan complejidad, dentro de las cuales se encuentran: la preferencia del viajero por visitar ciertos lugares, la planificación de la ruta para más de un día, las ventanas de tiempo en las cuales están abiertos algunos sitios turísticos, la elección del modo de transporte que se utiliza, la planificación de los tiempos de descanso, la elección del hospedaje, entre otros.

Así mismo, es importante considerar la congestión que se puede presentar en algunas horas del día en ciertos sitios turísticos o puntos de interés (POI, por sus siglas en inglés Points Of Interest), ya que para el viajero no solo es importante conocer los sitios a visitar, sino también el mejor momento del día para hacerlo, como se menciona en el trabajo de Wang et al. (2016), de manera que si para el turista es posible realizar la visita en horas de poca congestión, esto representaría un ahorro en tiempo que puede ser utilizado para realizar otra visita.

De igual forma, la decisión de qué modo de transporte utilizar para realizar el recorrido es de gran importancia, ya que dependiendo de esta decisión, el tiempo de recorrido de un POI a otro y por ende el tiempo total de duración del tour se ven afectados. Adicionalmente, pueden existir condiciones especiales para el trayecto, como por ejemplo, el recorrido de los arcos turísticos, los cuales en la mayoría de los casos se realizan caminando.

Este problema puede ser tratado como una extensión del *Orienteering Problem OP* enfocado a turismo, definido como *Tourist Trip Design Problem* ($TTDP$) el cual ha sido ampliamente estudiado por diferentes autores como Blum et al. (2007) los cuales obtienen aproximaciones que permiten tener resultados en tiempo polinomial, Righini and Salani (2009) que estudian variantes en las que se incluyen factores como ventanas de tiempo, Erdoğan and Laporte (2013) que consideran ganancias variables por visitar un sitio turístico,Gavalas et al. (2014) los cuales incluyen la dependencia del tiempo dentro del modelo, Wang et al. (2016) que considera el factor de la aglomeración, Archetti et al. (2016) que abarcan el problema utilizando arcos turísticos en lugar de nodos, Expósito et al. (2019) que utilizan una clusterización de los sitios en el modelo, entre otras.

En el presente trabajo se estudia el problema de TTDP con ventanas de tiempo, transporte multimodal y momentos de congestión en los sitios. Adicionalmente, se considera de manera simultanea los POI considerados como nodos y arcos. Esto debido a que al tomar los sitios turísticos de una ciudad, es posible identificar que muchos de ellos se encuentran en un punto específico (nodos), como por ejemplo las iglesias o los monumentos; Sin embargo, muchos otros son un camino a recorrer (arcos), como por ejemplo los parques, las avenidas o los pasajes comerciales.

Dirigidos a apoyar esa toma de decisiones, se propone la realización de un modelo matemático de programación lineal y de un método de solución heurístico, que pretenden dar como resultado, la ruta más eficiente para recorrer diferentes POI en una ciudad. El objetivo estará enfocado en maximizar la satisfacción del viajero, respetando ventanas de tiempo de los POI, penalidades por congestión, recorrido de arcos turísticos y una duración máxima del tour.

Esta tesis se encuentra organizada de la siguiente forma. El capítulo 2 presenta una revisión de la literatura. La definición del problema y el modelo matemático propuesto son detallados en el capítulo 3. El capítulo 4 esta dedicado a los resultados computacionales obtenidos en el método exacto. En el capítulo 5 se propone un método heurístico y meta-heurístico de solución. Los resultados obtenidos en el método heurístico son analizados en el capítulo 6. Conclusiones y trabajos futuros son presentados en el capitulo 7.

1.2. Objetivo General

Proponer un modelo matemático de programación lineal entero-mixto y un método de solución, junto a su respectiva evaluación de desempeño, que pretenden dar como resultado, la ruta más eficiente para recorrer diferentes puntos turísticos en una ciudad, de manera que se garantice la satisfacción del viajero al recorrer el mayor número de sitios de su interés, evitando momentos de congestión y utilizando el menor tiempo posible.

1.3. Objetivos Específicos

- Proponer un modelo matemático que resuelva el problema planteado, considerando que algunos puntos turísticos deben ser representados como arcos. El modelo también debe incluir ventanas de tiempo, congestión, ponderación de los sitios y selección del modo de transporte.
- Proponer un método de solución heurístico, que permita obtener buenos resultados en tiempo de cómputo razonables.
- Evaluar y analizar el desempeño del método propuesto.

2. Revisión de la literatura

El problema del diseño de una ruta turística ha sido abordado desde diferentes perspectivas por múltiples autores. Algunos de ellos, enfocándose en la búsqueda de información sobre la oferta turística existente, para poder cruzarla con las preferencias del viajero y así poder generar una ruta de destinos. Colineau and Wan (2001), proponen el sistema "Tiddler" que provee información personalizada a los usuarios para planear su viaje, combinando información estructurada de diferentes bases de datos con información de páginas web. Jakkilinki and Sharda (2007), usa la información de requerimientos del viajero y los cruza con los sitios turísticos ofrecidos en la web. Vansteenwegen and Van Oudheusden (2011), desarrollaron la aplicación *TheCitytripPlanner* para cinco ciudades en Bélgica, que utiliza las preferencias y restricciones de los usuarios para predecir los POI a visitar en diferentes periodos, al cruzar la información con bases de datos que son actualizadas directamente por los sitios turísticos.

Cenamor et al. (2017), toman la información recopilada en la red social de viajes *minube*1 y la información sobre tiempo y distancias disponibles en *GoogleMaps*, para crear un sistema de Guías Turísticas Electrónicas Personalizadas (PETs) utilizando técnicas de agrupación. Diferentes investigaciones sobre el PETs han sido desarrolladas por autores cómo Cheverst et al. (2002), Garcia et al. (2009), Kenteris and Economou (2007), Kenteris and Economou (2010) y Malaka and Zipf (2000).

El problema de diseño de la ruta turística $TTDP$ (por sus siglas en inglés Tourist Trip Design Problem) fue definido por primera vez en 2007 por Vansteenwegen and Oudheusden (2007) como una extensión del *Orienteering Problem* (OP) usando programación entera, cuyo objetivo es seleccionar los puntos de interés (POIs) que corresponden con las preferencias del turista, de manera que se maximice la satisfacción del viajero. Este problema al ser derivado del Orienteering Problem (OP), llamado también TSP selectivo, es considerado como un problema NP-Hard, como se explica en el trabajo de Laporte and Martello (1990).

Distintos autores han centrado sus investigaciones en el TTDP, considerando diferentes parámetros como la distancia entre los sitios, días y horas de apertura, tarifas de entrada, diferentes modos de transporte, tiempo disponible para hacer el recorrido, definición de una única ruta (Single Tour TTDP) o múltiples rutas (Multiple Tour TTDP), entre otras. Muchos de estos estudios han sido recopilados en la revisión de literatura realizada por Gavalas et al. (2014). Dentro de los trabajos presentados se destacan los siguientes.

Bérubé and Potvin (2009), desarrollaron el primer método exacto usando la frontera de Pareto para solucionar el TTDP multiobjetivo, en donde se busca maximizar la satisfacción del viajero y minimizar los costos de viaje. Blum et al. (2007), definen el primer algoritmo de aproximación de factor constante para el *RootedOP* en donde el punto de inicio de la ruta es fijo, pero no lo es el punto de fin, permitiendo que el tour se detenga en cualquier nodo, en este caso el valor del factor constante depende de cómo se define la función objetivo y del tamaño del problema. Los autores logran resolver el problema de forma óptima sobre un grafo no dirigido en tiempo polinomial. Otros métodos heurísticos han sido propuestos por los siguientes autores Bansal et al. (2004), Chekuri and Pál (2012), Chekuri and Kumar (2004), Chekuri and Pal (2005) y Nagarajan and Ravi (2011).

Dentro de las principales variantes para el $TTDP$ se encuentran:

El Orienteering Problem con consideraciones de ventanas de tiempo ($OPTW$) que ha sido es-

tudiado por autores como Righini and Salani (2009), quienes proponen dos algoritmos exactos de programación dinámica para resolver el problema, basados en la relajación del espacio de estado decremental (DSSR), que es un algoritmo iterativo que resuelve de manera óptima el problema relajado. Bansal et al. (2004), realiza una agrupación de nodos basado en sus ventanas de tiempo, luego resuelve el problema para cada grupo ignorando esta consideración y finalmente agrupa los conjuntos de nodos usando programación dinámica y Frederickson and Wittman (2012), propone una variante para el $unrootedOPTW$ que tiene como objetivo visitar el máximo numero de sitios dentro de su ventana de tiempo, considerando que dos POI diferentes pueden compartir la misma ubicación.

El Time Dependent Orienteering Problem ($TDOP$) considera la dependencia del tiempo al calcular el costo de ir de un sitio a otro y es muy utilizado en aplicaciones de turismo para simular el uso multimodal del sistema público de transporte, acorde con lo expuesto por Gavalas et al. (2014). Esta aproximación fue definida por primera vez por los autores Fomin and Lingas (2002). Li et al. (2010) proponen un modelo de programación entera-mixta, y un modelo de etiquetado pre-nodo basado en programación dinámica para resolver el $TDOP$. Abbaspour and Samadzadegan (2011) investigan este problema para las áreas urbanas, haciendo una división entre los nodos asociados a POI y los nodos asociados a paradas de transporte multimodal, los autores plantean un algoritmo genético para resolver el problema.

Otras variantes para el $Single\ Tour\ TTDP$ han sido estudiadas en la literatura. Wang and Wasil (2008), establece que la ponderación o beneficio de un POI se define a partir de factores cómo la belleza natural, el interés cultural, la significancia histórica y el interés educativo. Para esa aproximación, la función objetivo considera cualquier combinación de estos beneficios y los autores utilizan un algoritmo genético para resolver el problema, mientras que Silberholz and Golden (2010) presentan un algoritmo iterativo. Schilde et al. (2009) estudian el multi-objective orienteering problem ($MOOP$) en donde cada POI pertenece a una categoría con un beneficio diferenciado, se propuso una adaptación de colonia de hormigas y una extensión de un VNS como método de solución. Erdoğan and Laporte (2013), investigan el OP con ganancias variables ($OPVP$) en donde la ganancia por visitar un sitio es obtenida después de varias visitas o de un tiempo mínimo de estadía en este, los autores proponen un algoritmo $branch - and - cut$ como método de solución. Rodríguez et al. (2012), plantean un modelo multi-objetivo solucionado a través de una búsqueda tabú, en donde además de las ventanas de tiempo, consideran factores como planeación multiperiodo con determinación de hora de almuerzo y cena, tiempos de descanso, días libres, selección del hotel y ritmo del viaje. Finalmente, Divsalar and Cattrysse (2013b) y Divsalar and Cattrysse (2013a) proponen el Orienteering Problem con selección de hotel ($OPHS$), el cual considera un modelo multi-periodo utilizando un VNS para solucionar el problema.

Los siguientes autores se han concentrado en la inclusión de parámetros estocásticos al modelo. Ilhan and Daskin (2008), consideran el Orienteering Problem con ganancias estocásticas ($OPSP$) las cuales se distribuyen normalmente, se presenta un enfoque de solución exacta para instancias pequeñas y un algoritmo genético bi-objetivo para instancias grandes. Gupta et al. (2012), investigan el Orienteering Problem estocástico (SOP) en donde el beneficio es determinístico y el tiempo de servicio en cada nodo es aleatorio y no es conocido hasta que el nodo es visitado y Campbell and Thomas (2011), estudian el Orienteering Problem con tiempos de viaje y de servicio estocásticos ($OPSTS$) en donde el tiempo de viaje entre nodos y el tiempo de servicio no son parámetros determinísticos, se incluye no solo un beneficio por visitar un POI, sino una penalidad en caso de no hacerlo, se analizan casos en los cuales el problema puede ser resuelto de forma exacta y se planean diferentes métodos heurísticos para instancias grandes.

Otra variante del $TTDP$ es su versión con múltiples toures, como se menciona en Gavalas et al. (2014), llamado también TOP por sus siglas en inglés *Team Orienteering Problem*. Dentro de las principales contribuciones a este problema se encuentran: Archetti et al. (2007), quienes se basan en el VRP con profits para definir el $TTDP$ en donde se debe determinar más de una ruta, usualmente usado para modelos multiperiodo, en donde la recolección de ganancias al visitar un nodo se distribuye en varios vehículos con capacidad limitada. Vansteenwegen (2009), plantean el TOP con ventanas de tiempo establecidas para el servicio en cada POI. Garcia et al. (2012) y Garcia et al. (2013) investigan el $TDTOPTW$ en donde además de las ventanas de tiempo de servicio en los POI, incluyen el tiempo dependiente, los autores proponen dos aproximaciones para resolver el problema aplicados a instancias reales de zonas urbanas. Archetti et al. (2009), se enfocan en el Team Orienteering Problem con capacidad $CTOP$ en donde la demanda de personas que quieren hacer el recorrido en cada día no debe exceder una capacidad definida, se usa el método del árbol de mínima expansión como método de solución y Li and Hu (2011) retoma este trabajo y agrega la consideración de ventanas de tiempo. Ekici and Retharekar (2013), evalúan una variante del TOP en donde el beneficio de cada nodo disminuye a medida que el tiempo pasa, así el objetivo es maximizar la diferencia entre el beneficio recolectado y el costo de viaje. Finalmente, Garcia et al. (2009) retoma el TOP con ventanas de tiempo y agrega atributos a cada POI, como tarifas de entrada a los sitios turísticos lo cuales no deben exceder el presupuesto máximo, se propone un ILS como método de solución.

Trabajos posteriores a la revisión de literatura presentada por Gavalas et al. (2014) han sido desarrollados. Dentro de las principales contribuciones se encuentran:

Wang et al. (2016), proponen el sistema $PersCT$ para recomendar viajes personalizados que evitan los momentos más concurridos de los POI, utilizando datos reales de la ciudad de Melbourne a partir de fotos etiquetadas con $Flickrgeo$ y 6 meses de datos de tráfico peatonal, se utiliza una extensión del algoritmo de colonia de hormigas para fusionar los intereses de los usuarios, con la popularidad de POI y los datos de hacinamiento para poder realizar la recomendación de los viajes.

Migliorini et al. (2018), desarrollan un sistema de viajes personalizado, que considera los picos de congestión de los diferentes POI para darle una sugerencia al usuario y realiza una verificación de la ocupación de cada sitio en tiempo real, para evitar que debido a las sugerencia hecha se generen nuevos picos de congestión, de manera que se equilibra a los usuarios entre los diferentes POI. Se define como un problema multi-objetivo que es abordado usando recocido simulado.

Expósito et al. (2019), se enfocan en el TTDP con POI clusterizados, en donde los sitios turísticos son agrupados según el tipo de atracción y se restringe el máximo/mínimo número de POI que pertenecen a cada uno de los clústeres visitados en la misma ruta. El problema es atacado utilizando un método GRASP.

Archetti et al. (2016), investigan el Orienteering Arc Routing Problem ($OARP$) que a diferencia del TTDP basado en el OP, los POI no están ubicados en los nodos, sino en los arcos, el objetivo es encontrar una ruta de servicio a los clientes que maximice el beneficio total recaudado mientras se cumple un límite de tiempo determinado para el tour. Se propone un modelo de programación lineal y un algoritmo de ramificación y corte como método de solución.

Kotiloglu et al. (2017), consideran algunos POI opcionales que pueden maximizar la satisfacción

del turista, mientras que hay algunos puntos obligatorios preseleccionados que se deben visitar, teniendo en cuenta las disponibilidades diarias, horarios de apertura, limitaciones en la duración del recorrido y presupuestos. El problema es resuelto mediante una búsqueda Tabú.

Zheng and Liao (2019), analizan el diseño de rutas turísticas personalizadas para grupos con intereses heterogéneos, el problema es modelado con una adaptación de la optimización de Pareto y luego diseña un enfoque heurístico con el algoritmo de colonias de hormigas.

Después de realizar una revisión sobre los trabajos realizados para el $TTDP$, no se evidencia ninguna investigación que trabaje de manera simultánea los POI considerados nodos y arcos. Así mismo, las aproximaciones en el tema de congestión no incluyen una propuesta de un método exacto de resolución. Por lo anterior, se decide incorporar estos aspectos en el presente trabajo, así como las consideraciones de ventanas de tiempo por ser una característica común dentro de los POI y la selección del modo de transporte para ir de un POI a otro, debido al impacto que estas decisiones tienen en la duración del tour y por las restricciones que se pueden generar para algunos POI, especialmente el recorrido de los arcos.

3. Definición del problema y modelo matemático propuesto

El *single tour TTDP* estudiado en este trabajo se define en un grafo ponderado y no dirigido $G = (V, A)$ y se asume que la matriz de tiempos de recorrido es simétrica, siendo V un conjunto de nodos compuesto por un subconjunto i y j de m sitios turísticos, donde el nodo 1 y n representan el hotel. A corresponde al conjunto de arcos que conecta el punto de origen (hotel) con los sitios turísticos y los sitios turísticos entre ellos. Los nodos pueden ser recorridos en diferentes modos de transporte k y t_{ijk} define el tiempo de traslado entre los nodos usando el modo k.

Existen parejas de nodos que al visitarse, el arco que los une debe ser recorrido obligatoriamente, ya que son considerados arcos turísticos. Para el presente trabajo, un arco turístico se representa a través de los dos nodos que lo conforman y estas parejas de nodos están definidas en $arcosT_{i,j}$.

Los sitios turísticos cuentan con una ponderación w_i que mide la preferencia del turista por visitar este sitio, así como un tiempo de estadía s_i, una hora de apertura o_i y una hora de cierre l_i. La máxima duración del tour deseada por el turista es definida en DT.

Cada sitio turístico puede tener diferentes momentos de congestión m_i. Las ventanas de congestión inician en a_{m_i} y finalizan en b_{m_i}. Una penalización Q es aplicada si se llega en congestión.

Las siguientes variables de decisión son consideradas:

$y_i = 1$ si el sitio turístico i es visitado, 0 si no

$x_{ijk} = 1$ si voy del sitio i al sitio j usando el medio de transporte k, 0 si no

$e_{im_i} = 1$ si se llega al sitio i en el momento de congestión m_i, 0 si no

Así mismo, se definen las siguientes variables auxiliares que ayudan a conocer el tiempo de llegada a cada sitio y a determinar si se llega en momento de congestión, lo cual sucede cuando c_{m_i} y $d_{m_i} =$ toman valores de 1 simultáneamente :

$c_{m_i} = 1$ si se llega después del inicio del momento de congestión m_i, 0 si no

$d_{m_i} = 1$ si se llega antes del fin del momento de congestión m_i, 0 si no

$P_i =$ Tiempo en el que se llega al sitio i

En este estudio, la función objetivo busca maximizar la satisfacción del cliente a través de tres factores que son de interés para el turista: 1). visitar los sitios que son de su preferencia, 2). evitar los momentos de congestión en cada visita, de manera que el tiempo que se usa de más cuando existe una gran afluencia de personas, se pueda usar en visitar otro sitio y 3). realizar el recorrido en un tiempo que sea razonable para el viajero, al considerar la hora de regreso al hotel.

La función objetivo se encuentra medida en puntos de satisfacción, por lo que un mayor puntaje será mejor para el turista. Para cada uno de los factores que componen la función objetivo, las

unidades se logran de la siguiente manera: 1. La ponderación ya está expresada en puntos. 2. La congestión es llevada a esta unidad al multiplicar la variable e_{im_i} por el factor Q, el cual me indica los puntos que se restarían al entrar en congestión a un POI y 3. la hora de regreso al hotel, es llevada a puntos de congestión al multiplicarse por el factor de conversión α, el cual representa los puntos a los que equivale la utilización de un minuto. Estos factores Q y α varían de acuerdo a la percepción que tenga cada turista sobre estos aspectos, por lo cual para el presente trabajo se tomaron valores de 60 y 1 respectivamente. Sin embargo, estos valores pueden ser modificados por el usuario.

$$\text{máx}\, z = \sum_i y_i w_i - \sum_i \sum_m e_{im_i} Q - \alpha P_n$$

Para el modelo matemático propuesto, las siguiente restricciones fueron consideradas:

(1) $$\sum_j \sum_k x_{1jk} \leq 1$$

(2) $$\sum_i \sum_k x_{ink} \leq 1$$

(3) $$\forall_{i\neq 1, i\neq n} \quad \sum_j \sum_k x_{ijk} = y_i$$

(4) $$\forall_{i\neq 1, i\neq n} \quad \sum_{j,i\neq j} \sum_k x_{ijk} = \sum_{j,i\neq j} \sum_k x_{jik}$$

(5) $$\sum_j \sum_k x_{njk} + \sum_i \sum_k x_{i1k} + \sum_i \sum_k x_{iik} = 0$$

(6) $$\forall_{i,i\neq 1, i\neq n} \quad o_i \leq P_i$$

(7) $$\forall_{i,i\neq 1, i\neq n} \quad P_i + s_i \leq l_i$$

(8) $$P_n - P_1 - \sum_k t_{01k} \leq DT$$

(9) $$\forall_{ijk, i\neq n} \quad P_i + s_i + t_{ijk} \leq P_j + M(1 - x_{ijk})$$

(10)

$$\forall_{ij,arcos(i,j)=1} \quad y_i = x_{ij1} + x_{ji1}$$

$$\forall_{ij} \quad \sum_k x_{ijk} \leq 1 \tag{11}$$

$$\forall_{m_i} \quad P_i \leq a_{m_i} + Mc_{m_i} \tag{12}$$

$$\forall_{m_i} \quad P_i + Md_{m_i} \geq b_{m_i} \tag{13}$$

$$\forall_{m_i} \quad e_{im_i} \geq c_{m_i} + d_{m_i} - 1 \tag{14}$$

$$\forall_m \quad e_{im_i}, c_{m_i}, d_{m_i} \in [0,1] \tag{15}$$

$$\forall_i \quad P_i \geq 0 \tag{16}$$

$$\forall_{ijk} \quad x_{ijk}, y_i \in [0,1] \tag{17}$$

Restricciones (1) y (2) garantizan que el inicio y final de la ruta es el hotel. Restricciones (3) y (4) verifican que cada sitio turístico es visitado solo una vez y que cada sitio tiene un único predecesor y sucesor en la ruta. La eliminación de arcos prohibidos es definida en la restricción (5). Restricciones (6) y (7) establecen el cumplimiento de las ventanas de tiempo. La duración máxima del tour es garantizada en la restricción (8), la cual toma la hora de llegada al hotel y le resta la hora en la que salió. Restricción (9) garantiza el cumplimiento de la cronología en el tiempo y además permite la eliminación de sub-toures. La definición de los arcos turísticos y su recorrido con el modo de transporte a pie ($k = 1$) se establece en la restricción (10). Restricción (11) establece que solo un modo de transporte puede ser usado para ir de un sitio a otro. Restricciones (12), (13) y (14) definen si se llega en momento de congestión o no. Finalmente, restricciones (15) definen la naturaleza binaria de las variables de decisión, mientras que (16) y (17) establecen la no negatividad.

La $figura1$ presenta un ejemplo para una instancia de 6 POI. La duración máxima del tour es 1260, la penalidad por entrar en congestión es 60 y los POI 2 y 3 pertenecen a un arco turístico. La información relevante de cada POI se muestra en el $cuadro1$.

* Las unidades indicadas para la apertura, cierre y rango de congestión están dadas en el minuto en el que se da el evento.

Cuadro 1: Información POI - instancia de prueba

Nodo	Ponderación (puntos de congestión)	Apertura*	Cierre*	Estadía (min)	Rango Congestión*
Hotel	0	0	1440	0	na
POI 2	500	0	1440	0	[1140-1260]
POI 3	0	0	1440	0	[5-10]
POI 4	750	480	1200	180	[480-540],[720-780],[1080-1140]
POI 5	300	540	1020	60	[540-600],[1020-1080]
POI 6	1000	600	840	30	[720-780]
POI 7	200	600	1320	60	[1080-1140]

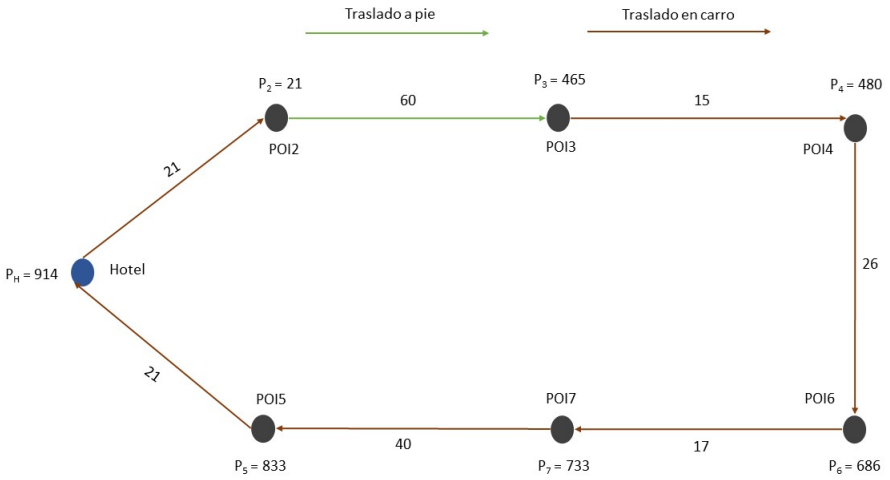

Figura 1: Resultados instancia de prueba

En el anterior ejemplo los POI 2 y 3 pertenecen a un arco turístico, por ende son recorridos de forma consecutiva, en el modo de transporte a pie. Para este caso se utilizó un tiempo de servicio de 0 ya que el tiempo de estadía en el arco corresponde al tiempo de traslado, sin embargo si el turista desea detenerse durante el recorrido, este tiempo puede ser considerado en la estadía.

Se puede verificar que el flujo en el tiempo se mantiene y que decide esperar en el $POI3$ durante 405 unidades de tiempo antes de dirigirse al $POI4$, de manera que cumpla la restricción de ventanas de tiempo, ya que éste nodo abre a las 408. Es importante resaltar, que a pesar que el tiempo de espera no esta restringido por ningún valor, el modelo buscará que sea lo más pequeño posible debido a la naturaleza de la Función Objetivo.

De igual forma, se puede observar que se llega al $POI4$ durante el periodo de congestión

$[480 - 540]$, permitiendo que la variable e_4 toma el valor de 1. La función objetivo arrojó un valor de 1776 unidades de tiempo, correspondientes a 2750 de beneficio al visitar los POI, menos 60 de penalidad por llegar al $POI4$ en congestión, menos 914 de puntos de satisfacción, los cuales son obtenido de multiplicar la hora de llegada al hotel por el factor alfa.

4. Experimentos computacionales - método exacto

La formulación matemática descrita en la sección anterior es resuelta usando CPLEX 12.9. Los siguientes experimentos fueron ejecutados en un computador Intel® Xeon® X5560 @ 2.80 GHz 2.79 GHz y 12 GB RAM con un tiempo máximo de corrida de 2 horas, el cual se considera pertinente para una decisión que puede ser tomada por el turista el mismo día que desea hacer el recorrido.

Las instancias probadas fueron construidas utilizando data real de los POI de la ciudad de Bogotá - Colombia. La información relativa a distancias con diferentes modos de transporte, horas de apertura y cierre y momentos de congestión, fue obtenida de la plataforma Google Maps. Finalmente la ponderación de los POI es obtenida de forma aleatoria, ya que este valor depende de los gustos y preferencias del turista.

Basados en la información anterior, se crearon instancias de 10, 20, 40 y 50 POI, para cada una de ellas se tomaron 5 instancias diferentes, las cuales incluyen al menos un arco turístico.

La sección 4.1 presenta los resultados para las instancias pequeñas (10 y 20 POI) y la sección 4.2 se concentra en las instancias grandes.

4.1. Instancias pequeñas

Para el grupo de instancias con 10 POI los resultados obtenidos fueron los siguientes:

Cuadro 2: Instancias 10 POI

Instancia	Valor FO (puntos de congestión)	Cota sup (puntos de congestión)	Tiempo (s)	Duración Tour (minutos)
Instancia10-0	11795	11795	112.13	1145.95
Instancia10-1	10434.4	10434.4	11.35	1214.87
Instancia10-2	10069.1	10069.1	336.73	1166.49
Instancia10-3	9428.01	9428.01	236.44	1149.99
Instancia10-4	9205.97	9205.97	72.851	1238.03

La anterior tabla muestra el valor de la función objetivo, la cota superior calculada directamente por CPLEX, el tiempo en el que logró encontrar la solución y el valor de la duración del tour.

Para este primer grupo de instancias, el método exacto propuesto encuentra una solución óptima dentro del tiempo máximo establecido de 2 horas, el cual se considera adecuado para una decisión que debe tomar el turista sobre su recorrido diario.

La instancia 10-0 incluye los 10 POI dentro de su ruta óptima y es la que obtiene un valor mayor de la función objetivo al considerar la ponderación de todos los POI de la instancia, mientras que el resto del grupo solo incluyen 8 de los 10 POI.

Es posible observar que el tiempo de solución varia considerablemente de una configuración a otra, siendo la instancia 10-1 la que encuentra el óptimo más rápido en un tiempo de 11.35 segundos,

respecto a la instancia 10-2 que lo hace en 336.73 segundos.

En cuanto a la congestión, todas las instancias logran encontrar una configuración que evite entrar en momentos de congestión y por ende no ser penalizados en la función objetivo.

Finalmente, la instancia que esta teniendo una mayor duración del tour es la 10-4, que a su vez es la que obtiene un menor valor de la función objetivo, por lo que se puede observar el impacto de este factor en la determinación de la solución.

Para el segundo grupo de instancias, las cuales contemplan 20 POI, los resultados fueron los siguientes:

Cuadro 3: Instancias 20 POI

Instancia	F.O.Solución Factible Encontrada (puntos de congestión)	Cota sup (puntos de congestión)	Gap	Tiempo (s)	Criterio de parada
Instancia20-0	17685.4	25032	29 %	3147.49	Memoria
Instancia20-1	15636.3	23560	34 %	7200.65	Tiempo máximo
Instancia20-2	11089.6	21433	48 %	5722.72	Memoria
Instancia20-3	16101.2	24695	35 %	7200.4	Tiempo máximo
Instancia20-4	17564.2	29716	41 %	7200.9	Tiempo máximo

Para este grupo de instancias ninguna pudo encontrar el óptimo, en el caso de las instancias 20-0 y 20-2 fue por memoria y para el resto del grupo por que no pudo hacerlo en las dos horas que se establecieron como criterio de parada.

En este caso, el método exacto tuvo un mejor desempeño con la instancia 20-0, la cual encontró una solución con un gap de 29 % respecto a la cota superior, mientras que la instancia 20-2 fue la que más alejada estuvo de la cota con un 48 %, ambas pararon por memoria, por lo que el criterio de parada no tiene ninguna relación con el desempeño o el valor de la función objetivo.

Las rutas obtenidas para este grupo de instancias incluyen desde 10 a 13 POI de los 20 posibles, siendo la 20-0 aquella con más POI y la 20-2 con menos, lo cual coincide con los valores máximos y mínimos obtenidos para la función objetivo.

En cuanto a la congestión, solo la instancia 20-1 entra durante congestión en uno de los POI que visita y por ende recibe una penalización en la función objetivo, lo cual tiene un gran impacto, ya que a pesar de ser una de las instancias con más POI en la ruta (12 sobre 20) es la segunda con la función objetivo más baja.

Finalmente, la instancia con la menor duración de tour es la 20-4 con 1156.8 segundos y la mayor son 20-0 y 20-3 con 1260 segundos. En este caso, la duración del tour no esta teniendo un impacto diferenciador, ya que dentro de estas instancias se encuentran los valores más altos para la función objetivo.

4.2. Instancias grandes

El método exacto fue probado para las instancias grandes y no pudo encontrar soluciones factibles antes de terminar por alguno de los dos criterios de parada: memoria y tiempo máximo de corrida de 2 horas. Por lo anterior, se evidencia la necesidad de generar un método heurístico que permita solucionar el TTDP con inclusión de arcos y congestión para la construcción de rutas turísticas que incluyan una oferta grande de POI.

5. Método heurístico propuesto

En la sección 4.2 se evidenció la necesidad de proponer un método heurístico que genere buenas soluciones para instancias grandes. En ese sentido, este capítulo es dedicado a presentar una propuesta de una heurística constructiva con movimientos de intensificación y diversificación que pretenden dar solución al problema.

5.1. Heurística constructiva

El procedimiento para construir la solución inicial se describe en el *Algoritmo*1, en donde la primera línea declara las variables a incluir en la solución de la ruta. Las líneas 2-3 inician la solución definiendo al hotel como inicio y fin de la ruta. Las líneas 4-7 inicializan la función objetivo y los tiempos de llegada, entrada y salida en cero. La línea 8 llama al procedimiento *Lista_ordenada* el cual toma todo el conjunto de *POI* y lo ordena de mayor a menor, según la ponderación establecida por el turista. La línea 9 llama al procedimiento *Inserción_nodo*, que es mostrado en el *Algoritmo*2 y que se detalla más adelante. La línea 10 llama a *Actualización_lista*, en donde se va eliminando de la lista ordenada los nodos que ya han sido evaluados en el procedimiento anterior, en caso de ser un arco elimina los dos nodos correspondientes. La línea 11 llama a *Insertar_nodo*, la cual va poblando la ruta turística según *Decisión_nodo*, que es calculado en *Inserción_nodo* y que puede tomar el valor de 0 cuando el nodo no se incluye en la ruta y se retira de la lista ordenada y 1 cuando el nodo se incluye en la ruta, en caso de ser un arco, incluye simultáneamente los dos nodos. La línea 12 imprime la solución del ruteo. La línea 13 determina la hora óptima de llegada a cada nodo y es explicado en la sección 5.1.1. Finalmente, la solución completa es mostrada en la línea 14.

Algoritmo 1: Heurística Constructiva

```
1  Int NodoNuevo, DecisionNodo, ModoLlegada, ModoSalida
2  Inicio_ruta = Hotel
3  Fin_ruta = Hotel
4  TiempoLlegada(Hotel) = 0
5  TiempoEntrada(Hotel) = 0
6  TiempoSalida(Hotel) = 0
7  FO = 0
8  Call Lista_ordenada
9  Call Inserción_nodo
10 Call Actualización_lista
11 Call Insertar_nodo
12 Imprimir solución
13 Call TiemposOptimos
14 Imprimir solución final
```

El procedimiento Inserción_nodo mostrado en el *Algoritmo*2, inicia declarando las variables en las líneas 1-2. Las líneas 3-5, inicializan las variables *NodoNuevo* como el nodo actual de evaluación y *PenPosicion* como el último *POI* incluido en la solución, que a su vez es el *NodoAnterior*.

Las líneas 6-14, establecen el modo de transporte con el tiempo mínimo de recorrido para ir entre

el $NodoAnterior$ y el $NodoNuevo$.

Líneas 15-19, calculan la hora de llegada y el tiempo de espera para entrar al POI.

Líneas 20-21, calculan la hora de entrada y de salida del POI.

Líneas 22-27, incluyen un condicional para los nodos que pertenecen a un arco turístico, para esto se crea la matriz $Arco$ en donde cada nodo tiene asociado un valor, en caso de pertenecer a un arco ese valor será su compañero, sino el valor será -1. Para el segundo nodo del arco, se calcula de nuevo la hora de entrada y salida y se garantiza que el modo de recorrido del arco sea a pie. Con el nuevo cálculo de estas variables, las verificaciones posteriores se realizan directamente con el segundo nodo, garantizando que es posible incluir todo el arco a la ruta.

Líneas 28-30, verifican que la salida del POI sea menor a su hora de cierre, de no ser así el nodo se excluye de la ruta, el procedimiento termina y asigna un valor de 0 a $Decisión\_nodo$.

Líneas 31-39, calculan el modo de transporte con el tiempo mínimo de recorrido para ir entre el $NodoNuevo$ y el $Hotel$.

Líneas 40-43, determinan la hora de llegada al hotel, la cual debe ser menor a la duración del tour, de no ser así el nodo no es ingresado a la ruta, el procedimiento termina y asigna un valor de 0 a $Decisión\_nodo$.

Finalmente, si el nodo cumple con las verificaciones realizadas y llega a la línea 44, este es agregado a la solución. Este procedimiento se repite hasta que se haya recorrido en su totalidad la lista ordenada de POI y no haya otro sitio candidato a incluir en la ruta.

Algoritmo 2: Inserción_nodo

```
1  Int NodoAnterior, ModoMinimo, ModoMinHotel, PenPosicion, NodoNuevo, NodoTemp
2  Double TiempoMinimo, TiempoMinHotel, HoraLlegada, Tespera, HoraEntrada, HoraSalida,
     LlegadaHotel
3  for nodo=1; nodos = #nodos do
4      nodoNuevo = nodo
5      penPosicion = ruta.size() - 2
6      nodoAnterior = ruta[penPosicion]
7      TiempoMinimo = Traslado(NodoAnterior, NodoNuevo, 0)
8      ModoMinimo = 0
9      for i=1; i≤ no_modos do
10         if Traslado(NodoAnterior, NodoNuevo, i) ≤ TiempoMinimo then
11             TiempoMinimo = Traslado (NodoAnterior, NodoNuevo, i)
12             ModoMinimo = i
13         end
14     end
15     ModoLlegada = ModoMinimo
16     HoraLlegada = Tsalida[penPosicion] + TiempoMinimo
17     Tespera = Apertura(NodoNuevo) - HoraLlegada
18     if Tespera < 0 then
19         Tespera = 0
20     end
21     HoraEntrada = HoraLlegada + Tespera
22     HoraSalida = HoraEntrada + Tservicio(nodoNuevo)
23     if Arco(NodoNuevo) ≠ −1 then
24         NodoTemp = NodoNuevo
25         NodoNuevo = Arco(NodoTemp)
26         HoraEntrada = HoraSalida + Traslado(NodoTemp, NodoNuevo, 0)
27         HoraSalida = HoraEntrada + Tservicio(NodoNuevo)
28     end
29     if HoraSalida ≥ Cierre(NodoNuevo) then
30         Decisión_nodo ← 0
31     end
32     TiempoMinHotel = Traslado(NodoNuevo, 0, 0)
33     ModoMinHotel = 0
34     for i=1; i≤ no_modos do
35         if Traslado(NodoNuevo, 0, i) ≤ TiempoMinHotel then
36             TiempoMinHotel = Traslado (NodoNuevo, 0, i)
37             ModoMinHotel = i
38         end
39     end
40     ModoSalida = ModoMinHotel
41     LlegadaHotel = HoraSalida + TiempoMinHotel
42     if LlegadaHotel ≥ Duración then
43         Decisión_nodo ← 0
44     end
45     Decisión_nodo ← 1
46 end
47
```

5.1.1. Asignación de las horas de llegada

La heurística constructiva explicada en el Algoritmo 1, en sus líneas 1 a 12, crea una solución factible y su respectiva ruta turística. Sin embargo, aún no se ha determinado si se ingresa o no en momento de congestión y por ende, no se ha realizado el cálculo de la función objetivo.

Para realizar este cálculo, es importante notar que una vez la ruta es definida, dependiendo de como se asignen los tiempos de llegada, se obtendrá un valor diferente para la función objetivo. Para ilustrar ese comportamiento se presenta el siguiente ejemplo:

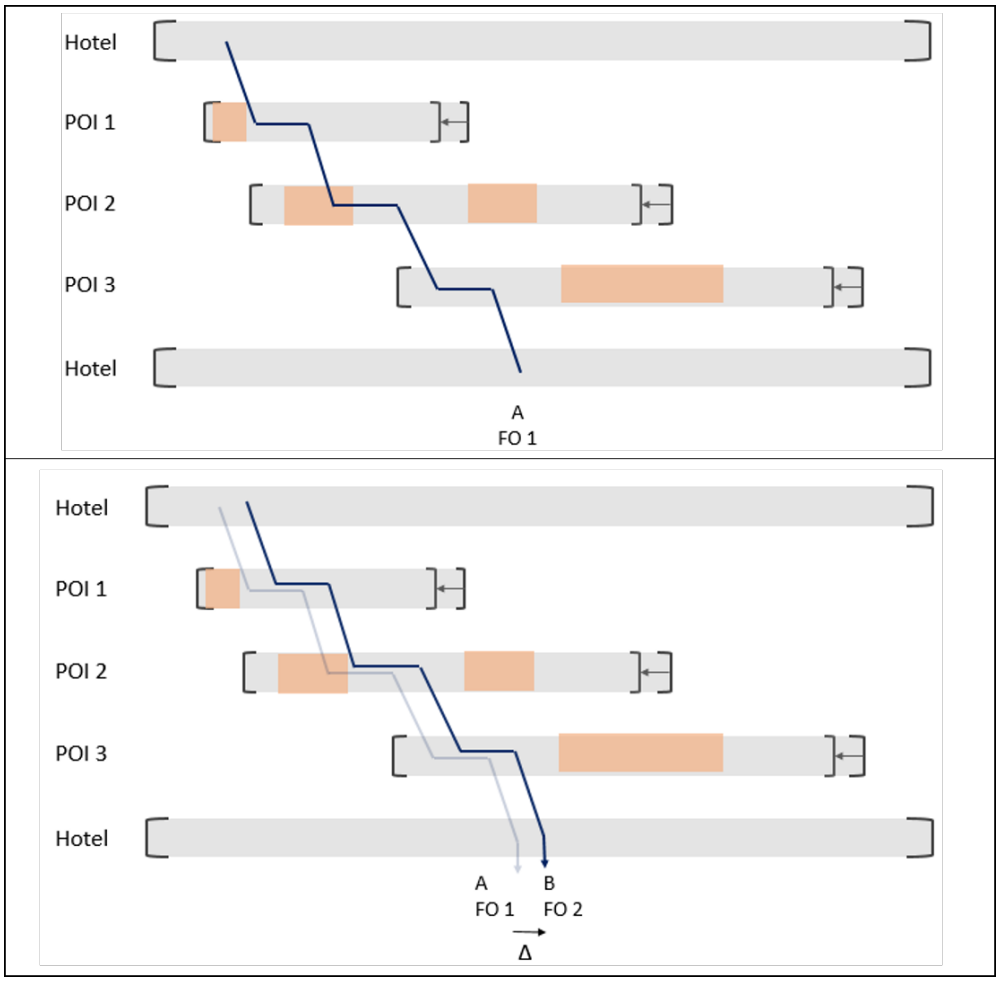

Figura 2: Asignación de tiempos, ejemplo 1

Las barras en gris determinan la ventana de tiempo de cada nodo, para los POI la ventana de cierre en realidad es más corta ya que se debe garantizar que el turista sale antes del cierre del POI, por lo tanto se debe contemplar el tiempo de estadía y esta reducción es representada por la flecha que va de derecha a izquierda. Las barras en rosado indican los momentos de congestión y finalmente la línea en azul indica el recorrido del turista.

La figura 2 muestra un ejemplo con dos elecciones diferentes de tiempos de llegada dada una ruta turística ya definida. En la primera parte se observa que se llega al $POI2$ durante uno de sus momentos de congestión, llegando al hotel en un tiempo A, con una función objetivo FO1. En la segunda parte se muestra un desplazamiento hacia la derecha, lo que significa que se decide esperar un poco en el hotel antes de salir, esta decisión genera que se evite la congestión del $POI2$, generando una hora de llegada al hotel B y una función objetivo FO2. En este ejemplo se evidencia que si bien la segunda opción evita la congestión y con ello una penalidad en la función objetivo, también genera una llegada más tardía al hotel, la cual también tiene un impacto. En este caso, la llegada B es mayor que A, pero no necesariamente la FO2 es mejor que la FO1.

La figura 3 muestra otro escenario posible. En la primera parte se presenta una solución en la cual para los $POI1$ y $POI2$ se llega durante congestión, se espera para entrar al $POI3$ y se llega al hotel en el tiempo A, con una función objetivo FO1. La segunda parte propone un cambio a la primera solución, en donde la espera ya no se realiza antes de ingresar al $POI3$ sino antes de salir del hotel, lo que genera que se evite la congestión en los POI anteriores, en este caso la llegada al hotel se mantiene en el tiempo A, pero se genera un nuevo valor de función objetivo FO2 el cual será mejor que FO1.

Finalmente, en la tercera parte de la figura 3, se muestra una región sombreada, en esta zona no importa la elección que se haga, se llegará al hotel en el tiempo A y se obtendrá la función objetivo FO2, en este caso estamos en el escenario de múltiples soluciones.

Los ejemplos anteriores nos permiten evidenciar la complejidad de este problema de asignación de horas de llegada, ya que tiene en cuenta los siguientes aspectos:

- Se trata de un subproblema que esta embebido en el problema principal de determinación de la ruta turística, debido a sus características de ventanas de tiempo y momentos de congestión.
- La llegada en congestión es una decisión dinámica, la cual depende de las decisiones que se hayan tomado en nodos anteriores.
- Para evaluar si una solución es mejor que otra, se debe realizar una comparación nodo a nodo en donde se determine el impacto en la función objetivo y sus cambios debido a entradas en congestión y llegadas al hotel.
- Se puede tener una muy buena solución para la secuencia de POI a recorrer, pero de no hacerse una buena asignación de los tiempos de llegada, la función objetivo puede no ser la mejor, inclusive en un escenario de secuenciación óptima.

Teniendo en cuenta lo anterior, se determina que este subproblema se encuentra por fuera del alcance de la tesis de maestría y que es de interés para investigaciones futuras. Para el presente trabajo, se propone utilizar un modelo de programación lineal para resolver el subproblema, el cual será ejecutado una vez se hayan completado las primeras 12 lineas del algoritmo 1, en donde se define la heurística constructiva para determinar la ruta.

Es importante resaltar, que por las características del problema mismo, si se esta trabajando con instancia grandes, el turista no podrá visitar más de cierto número de POI al día, lo que genera una reducción considerable en los nodos a incluir en el método exacto para la asignación de tiempos y que pueda correr en tiempos de computo razonables para el viajero.

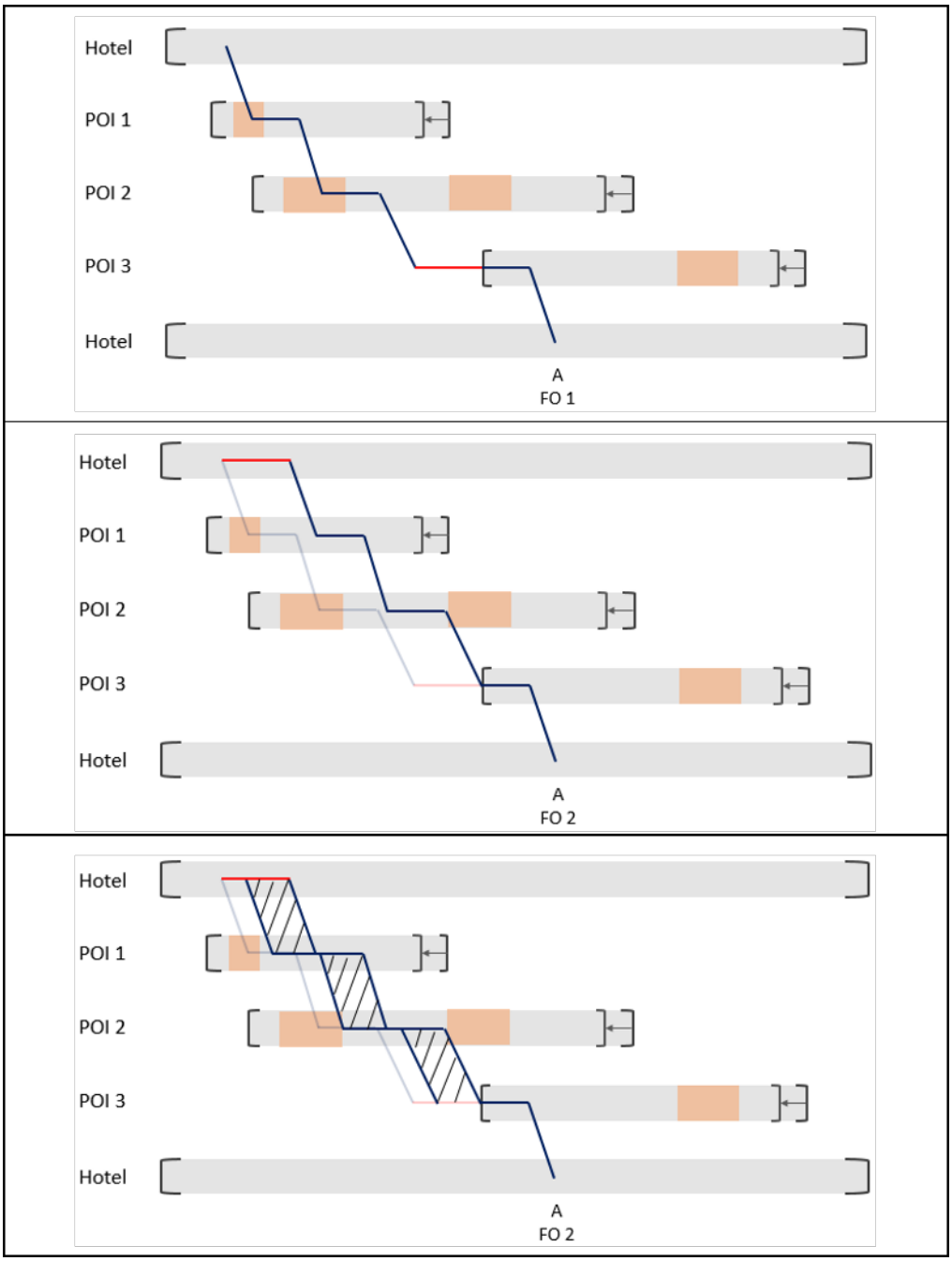

Figura 3: Asignación de tiempos, ejemplo 2

Para el método exacto que determina la asignación óptima de los tiempos de llegada, se parte del modelo presentado en la sección 3, con las siguientes modificaciones:

- La ruta es conocida, por lo que las variables y_i, x_{ijk} no son necesarias.
- La ponderación por visitar los nodos es conocida una vez se determina la ruta. Por lo tanto la función objetivo se reemplaza por una función de minimización que contempla únicamente la penalidad por congestión y la llegada al hotel.
- Las restricciones 1 a 5, 10 y 11 son eliminadas.
- La restricción 9 se replantea de la siguiente forma, en donde t_i representa el tiempo de traslado para llegar al POI_i:

$$\forall_{i,i\neq n} \qquad P_i \geq P_{i-1} + s_{i-1} + t_i$$

5.2. Búsqueda local

Una vez se ha obtenido la solución inicial utilizando la heurística constructiva presentada en la Sección 5.1, se procede a realizar la búsqueda local con ayuda de movimientos de intensificación, los cuales son seleccionados de acuerdo a la mejora de una función la cual tiene tres componentes: i) el cambio en la ponderación para los casos en los que intervienen nodos que no se encontraban en la solución inicial, es decir que estaban en la lista de no visitados, ii) el impacto en el tiempo de estadía y iii) el cambio en el tiempo de recorrido generado por los cambios en los POI.

Es importante resaltar que durante la ejecución de los movimientos se tiene en cuenta que los nuevos arcos resultantes de la ejecución del movimiento son evaluados para determinar el modo de transporte que genere un menor tiempo de trayecto, que no se deben romper arcos turísticos y en caso de moverlos se deben mover los dos nodos pertenecientes a éste.

Al evaluar los movimientos se determina si el cambio propuesto es prometedor o no, al generar una mejora en la función descrita anteriormente. Sin embargo es necesario evaluar si el movimiento es factible y determinar cuáles serían los tiempos de llegada a cada POI con este cambio y así evaluar si produce una mejora en la función objetivo o no, para ello se soluciona el modelo lineal presentado con las modificaciones descritas en las sección 5.1.1.

En la búsqueda local se sigue una estrategia de First Improvement, en donde el primer movimiento que genera mejora en la FO es ejecutado.

A continuación se presentan los movimientos realizados con sus características principales:

- Relocate
  En este movimiento se busca reubicar un POI de una posición de la ruta turística a otra. La determinación de si un movimiento es prometedor o no se realiza evaluando el delta del tiempo de recorrido, en este caso no se tiene en cuenta ni ponderación ni tiempo de estadía, ya que no se esta modificando los POI de la ruta y por ende no hay modificación en estos parámetros.

El cálculo del delta se realiza de la siguiente manera:

$$Delta\_t.recorrido = t.recorrido\_positivo - t.recorrido\_negativo$$

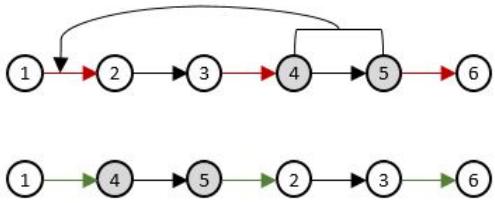

Figura 4: Ejemplo de relocate de un arco turístico

En la figura 4 los arcos en verde corresponden a los tiempos de recorrido positivos y los rojos a los negativos. En este caso, se considera que un movimiento es prometedor cuando el delta toma valores negativos.

- Swap
  En este movimiento se busca intercambiar dos *POI* pertenecientes a la ruta, es decir a la lista de nodos visitados. Al igual que en relocate, la determinación de si un movimiento es prometedor o no, se realiza evaluando únicamente el delta del tiempo de recorrido.

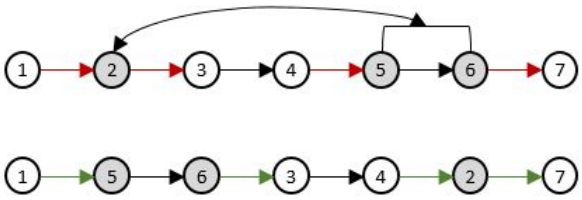

Figura 5: Ejemplo de swap entre un POI y un arco turístico

- Swap NoVis - Vis
  A diferencia del swap definido anteriormente, en este movimiento el intercambio se da entre un POI perteneciente a la ruta y uno de la lista de no visitados. Para este caso, además del delta del tiempo de recorrido, se tiene en cuenta el impacto de la ponderación y el tiempo de estadía, estas evaluaciones se hacen de la siguiente manera:

$$Delta\_tiempo = (t.recorrido\_positivo + t.servicio\_positivo) - (t.recorrido\_negativo + t.servicio\_negativo)$$

$$Delta\_ponderación = ponderación\_nodo\_a\_entrar - ponderación\_nodo\_a\_salir$$

Para este caso, los impactos positivos estarán asociados al POI que entra a la ruta y los negativos al que sale y un movimiento prometedor sera aquel que tenga un delta de ponderación mayor a cero y un delta de tiempo menor a cero.

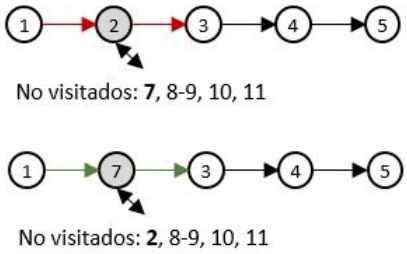

Figura 6: Ejemplo de swap entre un POI de la lista de no visitados y un POI de la ruta

- Insertar No Vis
  En este movimiento se busca agregar a la ruta un POI/arco turístico de la lista de no visitados. Para ello se tiene en cuenta la ponderación del POI/arco a ingresar y el delta de tiempo de recorrido como se ha calculado en Swap No Vis - Vis. Un movimiento prometedor sera aquel que tenga un delta de tiempo menor a cero.

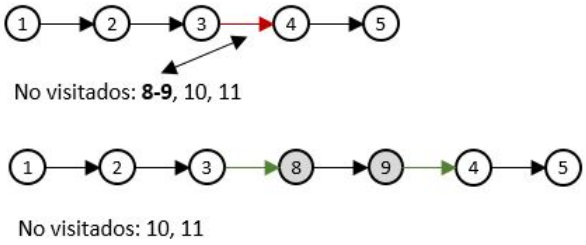

Figura 7: Ejemplo de inserción de un arco turístico a la ruta

- Sentido arco
  En este movimiento se busca cambiar el sentido del arco turístico, de manera que genere una mejora en la función objetivo. Para este caso no hay diferencia en ponderación ni tiempo de servicio, por tanto la evaluación del movimiento estará dado únicamente por el delta de tiempo de recorrido.

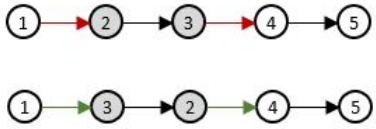

Figura 8: Ejemplo de cambio de sentido de un arco turístico perteneciente a la ruta

Una vez se tienen los movimientos definidos para la intensificación, la búsqueda local es realizada con la meta-heurística VND (Variable Neighborhood Descent), la cual cambia de movimiento solo cuando no encuentra mejora en el que esta recorriendo y en caso de hacerlo, regresa de nuevo al primer movimiento, que en este caso es el relocate. El cambio de vecindario lo realiza de la siguiente forma:

Algoritmo 3: VND

```
1  Stop = falso
2  Do
3  if no Relocate() then
4  |  if no Swap() then
5  |  |  if no Swap No Vis-Vis() then
6  |  |  |  if no Insertar No Vis() then
7  |  |  |  |  if no sentido arco then
8  |  |  |  |  |  stop = Verdadero
9  |  |  |  |  end
10 |  |  |  end
11 |  |  end
12 |  end
13 end
14 while (stop = falso);
```

5.3. Meta-heurística de múltiples inicios

Finalmente, se propone un método con múltiples inicios, el cual consiste en generar diferentes soluciones iniciales con ayuda de un factor aleatorio que es introducido dentro de la construcción de la solución inicial, de esta manera, se genera un grupo de soluciones a las cuales se les aplica la búsqueda local con el VND y se toma aquella que haya tenido mejor desempeño.

El factor aleatorio es generado en la línea 44 del Algoritmo 2: Inserción nodo, en esta línea se definía que el POI turístico era candidato a ingresar a la ruta ya que cumplía con las validaciones de ventanas de tiempo y llegada al hotel. Para aplicar la meta-heurística de múltiples inicios, esa línea de código es reemplazada por:

Algoritmo 4: Múltiples inicios

```
Generación de aleatorio (0,1)
if aleatorio≤ 0,7 then
    Decisión_nodo ← 1
end
```

Para cada instancia se utiliza un factor aleatorio de 0.7, el cual es obtenido después de hacer diferentes pruebas y lograr el factor de aleatoriedad deseado que generará una buena diversificación del modelo. Para el número de inicios se toma un valor de 20 soluciones iniciales a las cuales se les aplica el VND y se toma aquella con mejor desempeño en la función objetivo, este valor es tomado después de evidenciar que con valores más altos, el modelo requería más tiempo de resolución al tener que correr el método exacto en diferentes momentos, sin que la calidad de la solución mejorará significativamente. Finalmente se hacen 10 iteraciones para cada instancia.

El modelo propuesto presenta un algoritmo heurístico que contiene la interacción de una meta-heurística (VDN y Múltiples inicios) con técnicas de programación matemática (MIP para asignación de tiempos), por lo tanto se considera como una *Mateheurística*, la cual es definida por Boschetti et al. (2009).

6. Resultados obtenidos método propuesto - Mateheurística

Los resultados obtenidos para cada grupo de instancias, así como la medición del desempeño del método propuesto son mostrados a continuación.

6.1. Instancias pequeñas

Para las instancias de 10 POI, el método propuesto fue capaz de encontrar el óptimo para 3 de las 5 instancias del grupo y lo hizo en un tiempo menor al del método exacto. La columna Gap muestra el porcentaje por encima o por debajo, que estuvo el método heurístico respecto al método exacto. El gap es calculado de la siguiente manera:

$$Gap = (\frac{Resultado.heurístico}{Resultado.método.exacto}) - 1$$

Para las instancias 10-1 y 10-4 en las que no llego al óptimo, la solución del método propuesto estuvo un 0.061 % y 0.925 % por debajo del óptimo respectivamente, llegando a esta solución en un tiempo menor.

Al revisar el promedio obtenido en las corridas de cada instancia, se puede ver que la 10-0 y la 10-2 llegaron al óptimo en todas las corridas, con un tiempo promedio menor al obtenido con el método lineal.

Cuadro 4: Resultados instancias 10 POI

Instancia		Método Heurístico		Método Exacto-óptimo		GAP %	
Grupo	Corrida	FO (ptos. cong)	Tiempo (s)	FO (ptos. cong)	Tiempo (s)	FO (ptos. cong)	Tiempo (s)
10-0	0	11795.0	40.425	11795.0	130.513	0.000	-0.690
10-1	2	10428.0	12.075	10434.4	18.863	-0.061	-0.360
10-2	0	10069.1	13.759	10069.1	434.73	0.000	-0.968
10-3	4	9428.0	62.328	9428.0	313.15	0.000	-0.801
10-4	1	9120.8	60.656	9206.0	95.408	-0.925	-0.364

Cuadro 5: Instancias 10 POI - promedio corridas

Instancia	Promedio FO (ptos. cong)	Promedio Tiempo (s)
10-0	11795	72.523
10-1	10310	13.510
10-2	10069	15.204
10-3	9406	70.234
10-4	9121	67.337

Para el grupo de instancias de 20 POI, el método exacto pudo encontrar una solución factible, pero no logró comprobar optimalidad, sea por el criterio de parada de tiempo o por memoria. Para el método heurístico propuesto, se obtuvo soluciones con mejor función objetivo respecto a la mejor solución encontrada por el método exacto y en un tiempo de resolución menor.

Cuadro 6: Resultados instancias 20 POI

Instancia		Método Heurístico		Método Exacto-mejor solución		GAP %	
Grupo	Corrida	FO (ptos. cong)	Tiempo (s)	FO (ptos. cong)	Tiempo (s)	FO (ptos. cong)	Tiempo (s)
20-0	0	18389.4	74.162	17685.4	3307.92	3.981	-0.978
20-1	0	15830.3	173.308	15636.3	7200.65	1.241	-0.976
20-2	4	11931.0	93.278	11089.6	5722.72	7.587	-0.984
20-3	1	16123.4	129.466	16101.2	10741.4	0.138	-0.988
20-4	0	18382.1	52.533	17564.2	11135.9	4.657	-0.995

En cuanto a las diferentes corridas del grupo de instancias de 20 POI, todas encontraron una mejor solución que la arrojada por el método exacto, excepto en la 20-3 en donde solo 2/10 corridas superaron el valor.

Cuadro 7: Instancias 20 POI - promedio corridas

Instancia	Promedio FO (ptos. cong.)	Promedio Tiempo (s)
20-0	18377	123.769
20-1	15830	242.062
20-2	11540	100.270
20-3	15620	165.062
20-4	17973	86.452

Con estos resultados es posible decir que el método exacto tiene un buen desempeño en la resolución del problema estudiado para las instancias pequeñas, ya que en la mayoría de las instancias de 10 POI, el método logró encontrar el óptimo y en aquellas en las que no lo hizo, el gap estuvo por debajo del 1 %. En el caso de las instancias de 20 POI, el método logro superar la mejor solución encontrada al solucionarlo de forma exacta, solo para 8 corridas de la instancia 20-3 no logró superar el valor.

Si bien este análisis demuestra que el método propuesto obtiene buenos resultados para el problema estudiado, es importante resaltar que en la búsqueda local no hay forma de evaluar el impacto de la congestión cuando se considera un movimiento como prometedor y se decide ingresarlo a la ruta, este factor solo es considerado dentro del método exacto que se soluciona para seleccionar los tiempos de llegada cuando los POI de la ruta ya están definidos, por esta razón es posible que el método propuesto no encuentre el óptimo en todas las corridas de las instancias probadas.

6.2. Instancias grandes

En el capitulo 4 se detalla el por qué el método exacto no es probado en instancias grandes, es decir aquellas que se construyeron con 40 y 50 POI. Para el caso del método heurístico propuesto, estas instancias pudieron ser probadas, obteniendo los siguientes resultados:

Para este grupo de instancias no es posible realizar una comparación de resultados con el método exacto, ya que cómo se explicó anteriormente, estas no pudieron ser probadas. Sin embargo, el método propuesto logra encontrar soluciones factibles y además lo hace en un tiempo de computo razonable, lo cual le permitiría al turista hacer la selección de la ruta cuando tiene un grupo grande de posibles POI a visitar.

Cuadro 8: Instancias grandes - resultados método heurístico

Instancia	Mejor solución FO (ptos cong.)	Promedio FO (ptos cong.)	Mejor Tiempo (s)	Promedio Tiempo (s)
40-0	27740	27740	415,1	1285,1
40-1	25339	24664	371,3	732,9
40-2	26715	26426	375,9	893,5
40-3	27344	26247	147,0	240,6
40-4	25986	25981	365,2	885,6
50-0	26147	26142	469,1	1014,6
50-1	26405	26405	377,3	778,0
50-2	29081	28800	606,3	2508,5
50-3	27866	27747	378,9	813,8
50-4	28326	27674	490,8	1326,8

7. Conclusiones y trabajos futuros

- Se presenta un modelo de programación lineal entera mixta, para abordar el problema de selección de la ruta turística, con consideraciones de congestión, selección de modo de transporte, ventanas de tiempo e inclusión de arcos turísticos, el cual no ha sido estudiado en la literatura.

- Una Mateheurística es propuesta para solucionar el problema, de manera que el turista pueda elegir la ruta a seguir en un tiempo razonable, para un conjunto grande de posibles POI a visitar.

- Basados en información real, se construyen los cuatro grupos de instancias evaluados en este trabajo, mostrando un caso de aplicación del modelo propuesto sobre una selección de ruta turística de la ciudad de Bogotá - Colombia.

- El método heurístico presentado demostró tener un buen desempeño en la evaluación del problema, llegando al óptimo para la mayoría de las instancias de 10 POI y encontrando mejores soluciones que las arrojadas en el método exacto para las instancias de 20 POI.

- A pesar de encontrar buenos resultados, el método heurístico no puede evaluar el impacto de la congestión a la hora de considerar posibles movimientos en la búsqueda local, por lo cual este puede ser un factor a evaluar para mejorar el desempeño del método.

- En la búsqueda local se consideraron movimientos de inserción e intercambio entre nodos no visitados y visitados. Sin embargo, al no tener movimientos de retirar un POI de la ruta, se puede estar afectando el desempeño del método, ya que con la propuesta actual, la única forma de retirar un POI es reemplazarlo con uno que no este dentro de la ruta.

- El modelo heurístico presentado puede ser adaptado y utilizado para otras aplicaciones de ruteo y debido a la eficiencia obtenida en el tiempo de resolución, se propone como trabajo futuro, poder explorar el modelo para diferentes sectores productivos.

- Se propone como trabajos futuros adicionales desarrollar un método heurístico que solucione el problema de asignación de tiempos de llegada, el cual está embebido en el $TTDP$. Así mismo, las investigaciones futuras pueden enfocarse en variantes del modelo, como la inclusión de múltiples periodos, asignación de tiempos de descanso, decisiones de presupuesto, entre otros.

Referencias

R. A. Abbaspour and F. Samadzadegan. Time-dependent personal tour planning and scheduling in metropolises. *Expert Systems with Applications*, 38-10:12439–12452, 2011.

C. Archetti, A. Hertz, and M. G. Speranza. Metaheuristics for the team orienteering problem. *Journal of Heuristics*, 13(1):49–76, 2007.

C. Archetti, D. Feillet, A. Hertz, and M. G. Speranza. The capacitated team orienteering and profitable tour problems. *Journal of the Operational Research Society*, 60(6):831–842, 2009.

C. Archetti, Á. Corberán, I. Plana, J. M. Sanchis, and M. G. Speranza. A branch-and-cut algorithm for the orienteering arc routing problem. *Computers & Operations Research*, 66:95–104, 2016.

N. Bansal, A. Blum, S. Chawla, and A. Meyerson. Approximation algorithms for deadline-tsp and vehicle routing with time-windows. In *Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*, pages 166–174, 2004.

A. Blum, S. Chawla, D. R. Karger, T. Lane, A. Meyerson, and M. Minkoff. Approximation algorithms for orienteering and discounted-reward tsp. *SIAM Journal on Computing*, 37(2):653–670, 2007.

M. A. Boschetti, V. Maniezzo, M. Roffilli, and A. B. Röhler. Matheuristics: Optimization, simulation and control. In *International Workshop on Hybrid Metaheuristics*, pages 171–177. Springer, 2009.

G. M. Bérubé, J. F. and J. Y. Potvin. An exact -constraint method for bi-objective combinatorial optimization problems: Application to the traveling salesman problem with profits. *European journal of operational research*, 194-1:39–50, 2009.

G. M. Campbell, A. M. and B. W. Thomas. The orienteering problem with stochastic travel and service times. *Annals of Operations Research*, 186-1:61–81, 2011.

I. Cenamor, T. de la Rosa, S. Núñez, and D. Borrajo. Planning for tourism routes using social networks. *Expert Systems with Applications*, 69:1–9, 2017.

C. Chekuri and A. Kumar. Maximum coverage problem with group budget constraints and applications. *Approximation, Randomization, and Combinatorial Optimization*, pages 72–83, 2004.

C. Chekuri and M. Pal. A recursive greedy algorithm for walks in directed graphs. *46th Annual IEEE Symposium on Foundations of Computer Science*, pages 245–253, 2005.

K. N. Chekuri, C. and M. Pál. Improved algorithms for orienteering and related problems. *ACM Transactions on Algorithms*, 8-3:1–27, 2012.

K. Cheverst, K. Mitchell, and N. Davies. The role of adaptive hypermedia in a context-aware tourist guide. *Communications of the ACM*, 45(5):47–51, 2002.

N. Colineau and S. Wan. Mobile delivery of customised information using natural language generation. *Monitor*, 26-3:27–31, 2001.

V. P. Divsalar, A. and D. Cattrysse. A variable neighborhood search method for the orienteering problem with hotel selection. *International Journal of Production Economics*, 145-1:150–160, 2013a.

V. P. Divsalar, A. and D. Cattrysse. A memetic algorithm for the orienteering problem with intermediate facilities. *ORBEL*, 2013b.

A. Ekici and A. Retharekar. Multiple agents maximum collection problem with time dependent rewards. *Computers & Industrial Engineering*, 64(4):1009–1018, 2013.

G. Erdoğan and G. Laporte. The orienteering problem with variable profits. *Networks*, 61-2:104–116, 2013.

A. Expósito, S. Mancini, J. Brito, and J. A. Moreno. A fuzzy grasp for the tourist trip design with clustered pois. *Expert Systems with Applications*, 127:210–227, 2019.

F. V. Fomin and A. Lingas. Approximation algorithms for time-dependent orienteering. *Information Processing Letters*, 83-2:57–62, 2002.

G. N. Frederickson and B. Wittman. Approximation algorithms for the traveling repairman and speeding deliveryman problems. *Algorithmica*, 62:1198–1221, 2012.

A. Garcia, M. T. Linaza, O. Arbelaitz, and P. Vansteenwegen. Intelligent routing system for a personalised electronic tourist guide. In *ENTER*, pages 185–197, 2009.

A. Garcia, M. T. Linaza, O. Arbelaitz, M. FUCHS, F. RICCI, and L. CANTONI. Evaluation of intelligent routes for personalised electronic tourist guides. In *ENTER*, pages 284–295, 2012.

A. Garcia, P. Vansteenwegen, O. Arbelaitz, W. Souffriau, and M. T. Linaza. Integrating public transportation in personalised electronic tourist guides. *Computers & Operations Research*, 40(3): 758–774, 2013.

D. Gavalas, C. Konstantopoulos, K. Mastakas, and G. Pantziou. A survey on algorithmic approaches for solving tourist trip design problems. *Journal of Heuristics*, 20(3):291–328, 2014.

A. Gupta, R. Krishnaswamy, V. Nagarajan, and R. Ravi. Approximation algorithms for stochastic orienteering. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 1522–1538. SIAM, 2012.

I. S. M. Ilhan, T. and M. S. Daskin. The orienteering problem with stochastic profits. *Iie Transactions*, 40-4:406–421, 2008.

G. M. Jakkilinki, R. and N. Sharda. Connecting destinations with ontology-based e-tourism planner. *Information and Communication technologies in tourism*, pages 21–32, 2007.

G. D. Kenteris, M. and D. Economou. An innovative mobile electronic tourist guide application. *Pers.Ubiquit*, 13:103–118, 2007.

G. D. Kenteris, M. and D. Economou. Electronic mobile guides: a survey. *Pers.Ubiquit*, 15:97–111, 2010.

S. Kotiloglu, T. Lappas, K. Pelechrinis, and P. Repoussis. Personalized multi-period tour recommendations. *Tourism Management*, 62:76–88, 2017.

G. Laporte and S. Martello. The selective travelling salesman problem. *Discrete applied mathematics*, 26:193–207, 1990.

J. Li, Q. Wu, X. Li, and D. Zhu. Study on the time-dependent orienteering problem. In *2010 International Conference on E-Product E-Service and E-Entertainment*, pages 1–4. IEEE, 2010.

Z. Li and X. Hu. The team orienteering problem with capacity constraint and time window. In *The Tenth International Symposium on Operations Research and Its Applications (ISORA 2011)*, pages 157–163, 2011.

R. Malaka and A. Zipf. Deep map - challenging it research in the framework of a tourist information system. *Proceedings of the International Conference on Information and Communication Technologies in Tourism*, pages 15–27, 2000.

S. Migliorini, D. Carra, and A. Belussi. Adaptive trip recommendation system: Balancing travelers among pois with mapreduce. In *2018 IEEE International Congress on Big Data (BigData Congress)*, pages 255–259. IEEE, 2018.

V. Nagarajan and R. Ravi. The directed orienteering problem. *Algorithmica*, 60-4:1017–1030, 2011.

G. Righini and M. Salani. Decremental state space relaxation strategies and initialization heuristics for solving the orienteering problem with time windows with dynamic programming. *Computers operations research*, 36-4:1191–1203, 2009.

B. Rodríguez, J. Molina, F. Pérez, and R. Caballero. Interactive design of personalised tourism routes. *Tourism Management*, 33(4):926–940, 2012.

M. Schilde, K. F. Doerner, R. F. Hartl, and G. Kiechle. Metaheuristics for the bi-objective orienteering problem. *Swarm Intelligence*, 3(3):179–201, 2009.

J. Silberholz and B. Golden. The effective application of a new approach to the generalized orienteering problem. *Journal of Heuristics*, 16-3:393–415, 2010.

Vansteenwegen and Oudheusden. The mobile tourist guide: An or opportunity. *OR insight*, 20-3: 21–27, 2007.

P. Vansteenwegen. *Planning in tourism and public transportation.* PhD thesis, Springer, 2009.

S. W. V. B. G. Vansteenwegen, P. and D. Van Oudheusden. The city trip planner: An expert system for tourists. *Expert Systems with Applications*, 38-6:6540–6546, 2011.

G. B. L. Wang, X. and E. A. Wasil. Using a genetic algorithm to solve the generalized orienteering problem. *The vehicle routing problem: latest advances and new challenges*, pages 263–274, 2008.

X. Wang, C. Leckie, J. Chan, K. H. Lim, and T. Vaithianathan. Improving personalized trip recommendation by avoiding crowds. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 25–34, 2016.

Z. Xiao, L. Sen, F. Yunfei, L. Bin, Z. Boyuan, and L. Bang. Tourism route decision support based on neural net buffer analysis. *Procedia Computer Science*, 107:243–247, 2017.

W. Zheng and Z. Liao. Using a heuristic approach to design personalized tour routes for heterogeneous tourist groups. *Tourism Management*, 72:313–325, 2019.