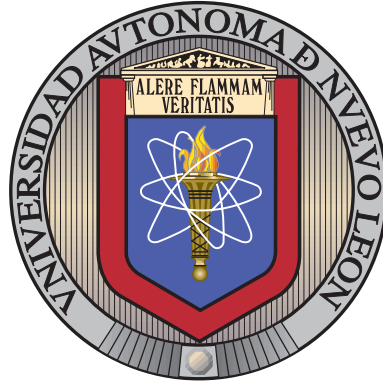


UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

SUBDIRECCIÓN DE ESTUDIOS DE POSGRADO



PROBLEMA DE RUTEO CON MÁXIMA COBERTURA
Y TIEMPO LÍMITE

POR

BRENDA YANETH SOTELO BENÍTEZ

EN OPCIÓN AL GRADO DE

MAESTRÍA EN CIENCIAS DE LA INGENIERÍA

CON ORIENTACIÓN EN SISTEMAS

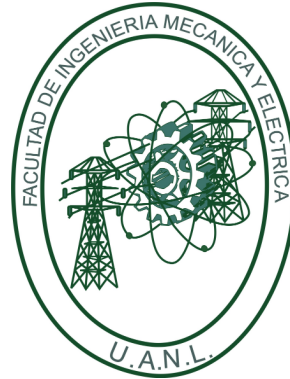
SAN NICOLÁS DE LOS GARZA, NUEVO LEÓN

JULIO 2020

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

SUBDIRECCIÓN DE ESTUDIOS DE POSGRADO



PROBLEMA DE RUTEO CON MÁXIMA COBERTURA
Y TIEMPO LÍMITE

POR

BRENDA YANETH SOTELO BENÍTEZ

EN OPCIÓN AL GRADO DE

MAESTRÍA EN CIENCIAS DE LA INGENIERÍA

CON ORIENTACIÓN EN SISTEMAS

SAN NICOLÁS DE LOS GARZA, NUEVO LEÓN

JULIO 2020



UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA
SUBDIRECCIÓN DE ESTUDIOS DE POSGRADO

Los miembros del Comité de Tesis recomendamos que la Tesis «Problema de ruteo con máxima cobertura y tiempo límite», realizada por el alumno Brenda Yaneth Sotelo Benítez, con número de matrícula 1565705, sea aceptada para su defensa como requisito para obtener el grado de Maestría en Ciencias de la Ingeniería con Orientación en Sistemas.

El Comité de Tesis

Dra. María Angélica Salazar Aguilar
Directora

Dr. Vincent André Lionel Boyer
Revisor

Jania Astrid Saucedo

Dra. Jania Astrid Saucedo Martínez
Revisor

Vo. Bo.

Dr. Simón Martínez Martínez
Subdirector de Estudios de Posgrado



FIME
038

San Nicolás de los Garza, Nuevo León, julio 2020

A Dios, por estar en cada paso que doy, cuidándome y dándome fortaleza para continuar y permitirme llegar hasta este momento tan importante de mi formación profesional y, sobre todo, hacer que este proyecto se hiciera realidad.

A mis padres, Norma e Isaias, por su incondicional apoyo y motivarme constantemente para alcanzar mis anhelos; muchos de mis logros se los debo a ustedes entre los que se incluye este.

A mis hermanos, Kelly y Brandon, por brindarme ayuda, apoyo y compañía cuando lo necesitaba.

A Mario Gutiérrez, el amor de mi vida, quien ha estado a mi lado incluso en los momentos y situaciones más difíciles, siempre apoyándome, alentándome y diciéndome “Todo va a estar bien” y así fue.

*“El éxito no es un accidente.
Es trabajo duro,
perseverancia, aprendizaje,
estudio, sacrificio y sobre todo,
amar lo que estás haciendo.”*

-Pelé

ÍNDICE GENERAL

Agradecimientos	XIII
Resumen	XVI
1. Introducción	1
1.1. Descripción del problema	2
1.2. Objetivo	4
1.2.1. Objetivos específicos	4
1.3. Motivación y justificación	5
1.4. Hipótesis	6
1.5. Metodología de investigación	6
1.6. Estructura de la tesis	7
2. Revisión de literatura	9
2.1. El problema del agente viajero (TSP) y sus variantes	9
2.2. El problema de ruteo de vehículos (VRP) y sus variantes	14
2.3. El problema de cobertura y sus variantes	17

2.4. Problema de ruteo con máxima cobertura y tiempo límite (TCMCRP)	20
3. Descripción y modelación del problema	22
3.1. Formulación matemática	23
3.1.1. Descripción formal del problema	23
3.1.2. Modelo matemático	23
4. Metodología de solución	27
4.1. Adaptive Large Neighborhood Search	28
4.1.1. Solución inicial	31
4.1.2. Operadores de destrucción	35
4.1.3. Operadores de reparación	38
4.1.4. Evaluación de las soluciones	40
5. Experimentación y resultados	41
5.1. Descripción de las instancias	42
5.2. Entorno experimental	45
5.3. Parámetros utilizados en el ALNS	45
5.4. Análisis de las soluciones obtenidas mediante CPLEX	54
5.5. Desempeño del algoritmo de solución	56
5.5.1. Rendimiento de los operadores de destrucción y reparación	60
5.6. Análisis de las soluciones obtenidas utilizando MIP Start	67

5.7. Comparación con los algoritmos heurísticos propuestos en la literatura	72
6. Conclusiones y trabajo futuro	75
6.1. Conclusiones	75
6.2. Trabajo futuro	77
A. Tablas de resultados	81
A.1. Resultados del modelo TCMCRP	81
A.2. Comparación de resultados del modelo TCMCRP	92
A.3. Comparación del ALNS con los algoritmos heurísticos propuestos por Amiri y Salari (2019)	104

ÍNDICE DE FIGURAS

4.1. Descripción general del ALNS.	29
4.2. Proceso del algoritmo constructivo.	35
4.3. Ejemplo ilustrativo con 9 instalaciones, 31 clientes y 2 vehículos.	36
4.4. Ejemplo de ruta con los respectivos tiempos de visita.	38
5.1. Ejemplo de topografía de una instancia con 29 instalaciones y 70 clientes.	43
5.2. Ejemplo de topografía de una instancia con 99 instalaciones y 100 clientes.	44
5.3. Ejemplo de topografía de una instancia con 217 instalaciones y 506 clientes.	44
5.4. Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias pequeñas.	49
5.5. Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias medianas.	50
5.6. Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias grandes.	51

5.7. Rendimiento individual de cada uno de los operadores en el conjunto de instancias pequeñas.	61
5.8. Rendimiento individual de cada uno de los operadores en el conjunto de instancias medianas.	62
5.9. Rendimiento individual de cada uno de los operadores en el conjunto de instancias grandes.	62
5.10. Rendimiento por parejas de cada uno de los operadores en el conjunto de instancias pequeñas.	64
5.11. Rendimiento por parejas de cada uno de los operadores en el conjunto de instancias medianas.	65
5.12. Rendimiento por parejas de cada uno de los operadores en el conjunto de instancias grandes.	66
5.13. Valor de la función objetivo en instancias pequeñas.	71
5.14. Valor de la función objetivo en instancias medianas.	71
5.15. Valor de la función objetivo en instancias grandes.	72
6.1. Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias pequeñas.	78
6.2. Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias medianas.	79
6.3. Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias grandes.	80

ÍNDICE DE TABLAS

2.1. Estado del arte relacionado al TCMCRP	21
4.1. Pesos asignados a los operadores	40
5.1. Descripción de las características de los conjuntos de instancias de prueba.	43
5.2. Niveles para el factor 1.	47
5.3. Niveles para el factor 2.	47
5.4. Mejores configuraciones obtenidas por pruebas no paramétricas para el conjunto de instancias pequeñas.	52
5.5. Mejores configuraciones obtenidas por pruebas no paramétricas para el conjunto de instancias medianas.	52
5.6. Mejores configuraciones obtenidas por pruebas no paramétricas para el conjunto de instancias grandes.	53
5.7. Resultados reportados por CPLEX para el modelo propuesto por Amiri y Salari (2019).	55
5.8. Tiempos de ejecución totales, en segundos, requeridos por el algoritmo ALNS y el optimizador CPLEX.	57

5.9. Resultados del ALNS versus CPLEX para el modelo propuesto por Amiri y Salari (2019).	58
5.10. Porcentajes de mejora de las soluciones obtenidas por ALNS con respecto al constructivo.	60
5.11. Resultados de MIPStart versus CPLEX para el modelo propuesto por Amiri y Salari (2019).	68
5.12. Resultados del ALNS versus MIPStart para el modelo propuesto por Amiri y Salari (2019).	69
5.13. Porcentajes de mejora de las soluciones obtenidas por CPLEX con respecto al MIPStart.	70
5.14. Comparación del rendimiento del ALNS versus ILS, TS y VNS para el conjunto de instancias grandes.	74
A.1. Resultados obtenidos por CPLEX, ALNS y MIPStart para el conjunto de instancias pequeñas.	83
A.1. Continuación.	84
A.1. Continuación.	85
A.2. Resultados obtenidos por CPLEX, ALNS y MIPStart para el conjunto de instancias medianas.	86
A.2. Continuación.	87
A.3. Resultados obtenidos por CPLEX, ALNS y MIPStart para el conjunto de instancias grandes.	88
A.3. Continuación.	89
A.3. Continuación.	90

A.3. Continuación.	91
A.4. Ejemplo de inconsistencia en el tiempo límite de los vehículos.	93
A.5. Ejemplo de inconsistencia en valor óptimo reportado.	94
A.6. Detalles de resultados obtenidos por Amiri y Salari (2019) y Sinnl (2019) para el conjunto de instancias pequeñas.	95
A.6. Continuación.	96
A.6. Continuación.	97
A.7. Detalles de resultados obtenidos por Amiri y Salari (2019) y Sinnl (2019) para el conjunto de instancias medianas.	98
A.7. Continuación.	99
A.8. Detalles de resultados obtenidos por Amiri y Salari (2019) y Sinnl (2019) para el conjunto de instancias grandes.	100
A.8. Continuación.	101
A.8. Continuación.	102
A.8. Continuación.	103
A.9. Comparación de los algoritmos heurísticos propuestos para el TCMCRP en el conjunto de instancias grandes.	105
A.9. Continuación.	106
A.9. Continuación.	107
A.9. Continuación.	108

AGRADECIMIENTOS

Deseo agradecer a la Universidad Autónoma de Nuevo León (UANL) la oportunidad que me ha brindado de realizar mis estudios de posgrado. A la Facultad de Ingeniería Mecánica y Eléctrica (FIME) por el apoyo brindado durante mis estudios de maestría. Al Consejo Nacional de Ciencia y Tecnología (CONACYT) por el apoyo económico otorgado mediante una beca de estudios de tiempo completo.

Por su parte, también agradezco al Posgrado de Ingeniería de Sistemas (PISIS) por darme la oportunidad de realizar mis estudios de maestría, a los profesores y personal administrativo. Gracias por compartir su conocimiento, paciencia, tiempo y dedicación conmigo.

Quiero agradecer a mi directora de tesis, la Dra. María Angélica Salazar por apoyarme desde que empecé a ser su asesora y confiar en mí como estudiante. Como asesora, agradezco su paciencia, tiempo, comentarios y consejos que fueron de gran ayuda para mejorar la calidad de este trabajo y culminarlo. Como maestra, fue una excelente profesora y agradezco las enseñanzas y aprendizajes que me dejó y me permitieron ampliar mis conocimientos. Como persona, le doy gracias por confiar y ver potencial en mí, incluso cuando yo no lo veía, por motivarme siempre a enfrentar retos cada vez mayores y superar mis miedos, por compartirme sus experiencias y escucharme ante alguna situación académica o personal y brindarme algunos consejos.

Doy gracias al Dr. Vincent Boyer y la Dra. Jania Saucedo, por formar parte del comité de revisión de esta tesis y haber dedicado parte de su tiempo a la lectura

de este trabajo y brindar aportes para su mejora.

Quiero agradecer a mis compañeros de generación por todas las veces que recibí ayuda de parte de ellos, por su paciencia y consejos, por todas las experiencias que compartimos y por la amistad que forjamos. También agradezco a mis amigos, que a pesar de no vernos seguido, me mostraron siempre su apoyo.

Agradezco a mi familia, especialmente a mis padres Norma e Isaias, por el apoyo infinito e incondicional que siempre he recibido de su parte en cada aspecto o decisión que he tomado, por motivarme a nunca rendirme, por sus consejos y por siempre preocuparse por mí. Papá, gracias por tu apoyo, la orientación y educación que me has dado, por iluminar mi camino y darme la pauta para poder realizarme en mis estudios y mi vida. Agradezco los consejos sabios que en el momento exacto has sabido darme para no dejarme caer y enfrentar los momentos difíciles, por ayudarme a tomar las decisiones que me ayuden a balancear mi vida, por las veces que me llevabas y pasabas por mí a la escuela cuando era tarde y, sobre todo, gracias por el amor tan grande que me das. Mami, tu eres la persona que siempre me ha levantado los ánimos tanto en los momentos difíciles de mi vida estudiantil como personal. Gracias por tu paciencia y esas sabias palabras que siempre tienes para mis enojos, mis tristezas y mis momentos felices, por tus atenciones, por ser mi amiga, y ayudarme a cumplir mis sueños, te quiero mucho.

A mis hermanos Kelly y Brandon, gracias porque también me han apoyado y han aportado su granito de arena ayudándome de una u otra forma a lo largo de estos años.

Doy gracias a Mario Gutiérrez por convertirse en mi mejor amigo desde que lo conocí en la licenciatura y recibir desde entonces su apoyo y consejos, por escuchar mis presentaciones antes de cada congreso, seminario o clase y ayudarme a calmar mis nervios, por prestarme atención cada vez que me emociono por algo nuevo que aprendí o entendí, por estar presente con su apoyo, ánimos, felicitaciones y abrazos cuando logro superar algunos de mis miedos o retos que se me presentan y las cosas

salen mejor de lo que esperaba, y cuando no, por motivarme a levantarme para no rendirme y seguir intentándolo, por ayudarme a resolver mis problemas técnicos y dudas y, sobre todo, por amarme. Por todo eso y más, gracias.

RESUMEN

Brenda Yaneth Sotelo Benítez.

Candidato para obtener el grado de Maestría en Ciencias de la Ingeniería con Orientación en Sistemas.

Universidad Autónoma de Nuevo León.

Facultad de Ingeniería Mecánica y Eléctrica.

Título del estudio: PROBLEMA DE RUTEO CON MÁXIMA COBERTURA Y TIEMPO LÍMITE.

Número de páginas: 117.

OBJETIVOS Y MÉTODO DE ESTUDIO: En esta tesis se estudia el problema de ruteo de vehículos con máxima cobertura y tiempo límite, conocido como *Time Constrained Maximal Covering Routing Problem* (TCMCRP). En el TCMCRP se tiene un depósito central, un conjunto de vehículos y un conjunto de instalaciones, las cuales pueden cubrir a clientes que se encuentran dentro de un radio de cobertura dado. El objetivo del TCMCRP es determinar cuáles son las instalaciones a visitar y cuál será el orden de las visitas que deben realizar los vehículos disponibles, de tal forma que la duración total de cada ruta no exceda un tiempo límite y se cubra la mayor cantidad de clientes con las instalaciones visitadas.

El presente estudio propone una metaheurística para la solución del problema

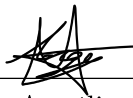
planteado, basado en un esquema ALNS (Adaptive Large Neighborhood Search) y se evalúa el desempeño del mismo a través de experimentos computacionales sobre un gran conjunto de instancias tomadas de la literatura. Los resultados obtenidos son comparados contra las soluciones óptimas obtenidas con el modelo matemático propuesto por Amiri y Salari (2019), haciendo uso del optimizador CPLEX 12.9. Además, se realiza la comparación con los resultados reportados en la literatura, específicamente con los reportados por Amiri y Salari (2019) y Sinnl (2019), quienes han realizado estudios de este problema proponiendo algoritmos heurísticos y exactos para la resolución del mismo.

Asimismo, se realiza el análisis de los resultados obtenidos utilizando MIPStart para CPLEX, tomando como punto de partida las soluciones obtenidas por el ALNS.

CONTRIBUCIONES Y CONCLUSIONES: Se propone una metaheurística ALNS para resolver el *Time Constrained Maximal Covering Routing Problem*, la cual es nueva en la literatura para este problema.

El ALNS logra obtener soluciones de buena calidad en tiempos de cómputo aceptables, resolviendo a optimalidad 198 de las 203 instancias reportadas por el optimizador. Por otro lado, al inicializar CPLEX con la solución que encuentra el ALNS, se alcanzaron 258 óptimos de las 268 instancias, es decir, un 20 % más que los alcanzados por CPLEX sin solución inicial, y de esta manera se validó que el ALNS propuesto resolvió a optimalidad 250 instancias de las 258 reportadas por el optimizador utilizando MIPStart.

Firma del asesor:



Dra. María Angélica Salazar Aguilar

CAPÍTULO 1

INTRODUCCIÓN

Actualmente la planificación de rutas representa una actividad importante dentro de la logística de servicio y distribución y la logística humanitaria.

En esta tesis se estudia el problema de ruteo con máxima cobertura y tiempo límite, introducido recientemente en la literatura como *Time Constrained Maximal Covering Routing Problem* (TCMCRP), en el cual se tiene un depósito central, un conjunto de vehículos y un conjunto de instalaciones, las cuales pueden cubrir a clientes que se encuentran dentro de un radio de cobertura dado; el objetivo es maximizar la cantidad de clientes cubiertos con las instalaciones visitadas, tomando en cuenta que la duración total de cada ruta no debe exceder un tiempo límite.

Se propone un algoritmo basado en un esquema ALNS (*Adaptive Large Neighborhood Search*) cuya eficiencia computacional es evaluada sobre tres conjuntos de instancias de prueba.

A lo largo de este trabajo, se analizan trabajos previos en donde se aborda el problema de cobertura, así como los elementos que consideraron para su análisis, el tipo de solución empleada, los resultados obtenidos y la contribución. También se describirán cuáles son las áreas de oportunidad en estos trabajos y posteriormente se utilizarán herramientas propias de la investigación de operaciones para abordar de manera eficiente el problema de ruteo de vehículos con máxima cobertura y tiempo

límite.

1.1 DESCRIPCIÓN DEL PROBLEMA

La importancia de tener un buen diseño y planificación correcta de las rutas de transporte en ocasiones pasa desapercibida. La optimización de rutas consiste en definir la planificación óptima de una flota de vehículos, en la cual se determine, para cada vehículo, el conjunto de nodos (clientes, instalaciones, zonas, etc.) a visitar y el orden en que deben ser visitados. Esto es de gran importancia en diferentes áreas como la industria, la salud, logística humanitaria e incluso cuando suceden situaciones imprevistas en las que es necesario contar con un plan de acción para gestionar la escasez de recursos de la manera más eficiente posible. Para el caso particular de las rutas, se suele disponer de un número finito de vehículos, trabajadores, horas laborales, capacidad, tiempo, presupuesto, entre otros, a los que se desea sacar el mayor rendimiento posible.

El antecedente histórico de las primeras investigaciones teóricas y aplicadas sobre la optimización de rutas, data en el año de 1959, donde Dantzing y Ramser propusieron una formulación matemática como una generalización del problema del agente viajero (TSP), planteado por primera vez por Flood (1956), y un algoritmo de aproximación para dar una solución al problema de suministro de gasolina en las estaciones de servicio. El problema del agente viajero consiste en determinar una ruta para un agente que quiere visitar n ciudades, partiendo y regresando de su ciudad origen y pasando una y solo una vez por cada una de ellas, de tal manera que la distancia total recorrida sea mínima. Mientras que en el TSP suponemos que hay un único vehículo con capacidad suficiente para visitar una vez a cada cliente, si se añaden otras condiciones, este problema se generaliza al problema de ruteo de vehículos (VRP), donde se cuenta con una flota de vehículos que parten de un nodo central (almacén, centro de distribución) hasta un número de clientes dispersos geográficamente. Cada ruta inicia y termina en el depósito, cada cliente debe ser

visitado exactamente una vez por un solo vehículo para satisfacer su demanda y la carga de cada vehículo no debe exceder su capacidad. El objetivo clásico es minimizar el costo total de transportación.

En este contexto podemos formalizar el problema de estudio, para ello consideremos un depósito central, un conjunto de vehículos y un conjunto de instalaciones, las cuales pueden cubrir a clientes que se encuentran dentro de un radio de cobertura dado. El problema consiste en determinar cuáles son las instalaciones a visitar y cuál será el orden de las visitas que deben realizar los vehículos disponibles, de tal forma que la duración total de cada ruta no exceda un tiempo límite y se cubra la mayor cantidad de clientes con las instalaciones visitadas. Este problema fue introducido por Amiri y Salari (2019) y lo denominan como *Time Constrained Maximal Covering Routing Problem*. Sus principales aplicaciones se pueden encontrar en el contexto de la logística humanitaria, en los sistemas de prestación de atención médica rural para proporcionar servicios de atención general (como vacunación, servicios de emergencia o actividades para la prevención de epidemias) y en la asignación de algunas ubicaciones para establecer hospitales de campaña en las áreas potenciales. En lo anterior, maximizar la demanda cubierta sujeta a la disponibilidad de recursos limitados es el problema principal del TCMCRP.

El TCMCRP combina dos importantes problemas de optimización combinatoria: el problema del agente viajero con cobertura (CSP) y el problema del agente viajero selectivo múltiple (TOP). Ambos son extensiones del problema de agente viajero (TSP) y consideran un conjunto de clientes que son visitados a lo más una vez. La diferencia entre ambos radica en el número de agentes o vehículos y en el criterio de desempeño que se considera en cada caso. En el problema del vendedor con cobertura se busca la construcción de una ruta de costo mínimo sobre un subconjunto de n clientes determinados, de manera que cada cliente no visitado se encuentre dentro de un radio de cobertura de al menos uno de los clientes que estén en la ruta. Por otra parte, en el problema del agente viajero selectivo múltiple se tiene un conjunto de agentes o vehículos y un conjunto de clientes, cada uno de ellos

asociado a un puntaje que es recolectado únicamente si es visitado por uno de los vehículos. El TOP consiste en determinar las rutas que seguirán los vehículos, con el objetivo de maximizar la puntuación colectada y que la duración de cada una de las rutas no exceda un tiempo límite.

Debido a que el problema que se estudia en esta tesis es clasificado como NP-difícil (Amiri y Salari, 2019), este trabajo se dedica al desarrollo de un algoritmo metaheurístico capaz de ofrecer soluciones de calidad en un tiempo razonable de cómputo.

1.2 OBJETIVO

Proponer e implementar una metodología ALNS para encontrar soluciones eficientes en tiempo y calidad al problema de ruteo de vehículos con máxima cobertura y tiempo límite, el cual consiste en determinar cuáles son las instalaciones a visitar y cuál será el orden de las visitas que deben realizar los vehículos disponibles, de tal forma que la duración total de cada ruta no exceda un tiempo límite y se cubra la mayor cantidad de clientes con las instalaciones visitadas.

1.2.1 OBJETIVOS ESPECÍFICOS

- Validar la formulación propuesta por Amiri y Salari (2019).
- Diseñar e implementar computacionalmente un algoritmo de solución para el problema en estudio basado en estrategias metaheurísticas, de manera que permita obtener soluciones eficientes en tiempo y que sean iguales o mejores que las reportadas hasta el momento en la literatura.
- Utilizar la solución obtenida por la metaheurística como solución inicial en CPLEX.

- Analizar los resultados obtenidos en la experimentación computacional, resaltando los beneficios obtenidos y las áreas de oportunidad.

1.3 MOTIVACIÓN Y JUSTIFICACIÓN

La relevancia de este trabajo está dada principalmente por la investigación de un problema recientemente introducido y poco tratado en la literatura científica. Hasta el momento y hasta donde se tiene conocimiento, no se ha implementado un método de solución basado en un esquema ALNS para resolverlo. El ALNS es una metaheurística propuesta por Ropke y Pisinger (2006), que se ha seleccionado para este estudio por su buen desempeño en problemas combinatorios, ya que tiene como principal ventaja un proceso adaptativo de búsqueda. Con esta tesis se tiene una contribución al avance del estado del arte en las áreas de ruteo de vehículos, logística y transporte, así como en la incorporación de nuevos resultados.

El problema de estudio resulta interesante e importante desde el punto de vista de la optimización y de las diversas aplicaciones reales en las que no es viable visitar a todos los clientes debido a las limitaciones de recursos o problemas de eficiencia. Ejemplos de estas aplicaciones son los problemas de localización de instalaciones de servicio y emergencia, planificación de los servicios de transporte de personal o transporte escolar los cuáles cuentan con un conjunto de paradas ya establecidas pero que no todas pueden ser visitadas debido a los costos en dinero y tiempo; empresas de transporte de público que quieren cambiar, eliminar o crear nuevas rutas conservando y ofreciendo el servicio a la mayor cantidad de usuarios; empresas turísticas o que proporcionan asistencia técnica, así como en el aspecto de emergencia o logística humanitaria cuando ocurren desastres naturales o desastres provocados por el hombre y es necesaria la distribución inmediata de recursos tales como, agua, vacunas, medicamentos y víveres. Bajo esta situación, el problema incluye la asignación de algunas ubicaciones para establecer hospitales de campaña entre las áreas

potenciales, donde dichas ubicaciones deben elegirse con el objetivo de maximizar el número de personas que podrán ser atendidas y cuya distancia recorrida desde los centros principales a estas ubicaciones sea mínima. O bien, en la atención de servicios de salud en zonas rurales para los cuales es necesario brindar servicios de atención general como vacunación, servicios de emergencia o actividades para la prevención de epidemias, donde los equipos deben trasladarse a diversas áreas en un tiempo limitado y al tener una cantidad limitada de recursos se tienen que visitar algunas áreas de manera que el máximo número de pacientes reciba tratamientos médicos para prevenir los brotes de enfermedades.

De esta manera, el contar con una metodología para encontrar soluciones de manera rápida y de calidad contribuye a que el proceso de toma de decisiones sea más ágil, y se tenga la certeza de contar con una planificación de rutas eficiente, acorde a los intereses que se requieran y el contexto bajo el que se esté trabajando.

1.4 HIPÓTESIS

- La implementación de una metaheurística basada en ALNS permitirá encontrar soluciones de buena calidad para el problema de estudio en tiempos computacionales aceptables.

1.5 METODOLOGÍA DE INVESTIGACIÓN

Para el desarrollo del tema de investigación, se trabajará bajo la siguiente metodología:

- Definición del problema. Delimitación de las características del problema que se estudiará en este trabajo.

- Revisión de antecedentes del problema. Recabar información sobre el problema de ruteo de vehículos y su variante tratada en este trabajo, los diversos estudios, aplicaciones o casos de estudio.
- Validación del modelo propuesto por Amiri y Salari (2018). Solución de las instancias pequeñas, medianas y grandes proporcionadas por Amiri y Salari (2019), haciendo uso del optimizador CPLEX 12.9, con la finalidad de revisar el alcance del modelo matemático.
- Implementación del algoritmo metaheurístico tipo ALNS (Búsqueda adaptativa de grandes vecindarios).
- Experimentación. Realizar experimentación computacional sobre las instancias previamente mencionadas, con el propósito de determinar la calidad del algoritmo propuesto y además, inicializar el modelo en CPLEX con la solución reportada por el ALNS, con la finalidad de analizar si estos inicios ayudan al optimizador a converger más rápido al óptimo.
- Análisis de resultados. Con los resultados obtenidos en la experimentación computacional se podrá evaluar la eficiencia de la metodología propuesta y verificar el cumplimiento de los objetivos propuestos, de manera que se puedan realizar modificaciones o analizar otros factores que pueden ser considerados en el análisis.

1.6 ESTRUCTURA DE LA TESIS

La estructura de este documento se presenta en capítulos. En este Capítulo, se describen los conceptos necesarios para el estudio del problema, así como el objetivo, motivación, justificación y metodología bajo las que se realizó este trabajo.

En el Capítulo 2 se presenta la revisión de la literatura correspondiente al *Time Constrained Maximal Covering Routing Problem* y otros problemas de ruteo

de vehículos relacionados con el problema de estudio, además, de revisar los métodos de solución propuestos.

En el Capítulo 3, se presenta formalmente el problema, formulación matemática, los diferentes supuestos y restricciones que abarcan al mismo.

En el Capítulo 4, se describe a detalle el diseño de una metaheurística tipo ALNS como metodología de solución.

Posteriormente, en el Capítulo 5 se muestran los resultados de los experimentos computacionales realizados para la validación del modelo y el desempeño del ALNS. En este último, se incluyen las experimentaciones hechas para la elección adecuada de los parámetros y el rendimiento de los operadores implementados, así como los resultados obtenidos al utilizar la solución del ALNS como punto de partida para CPLEX (MIPStart).

Finalmente, en el Capítulo 6 se discuten las conclusiones y el trabajo futuro.

CAPÍTULO 2

REVISIÓN DE LITERATURA

El *Time Constrained Maximal Covering Routing Problem* (TCMCRP) es un problema recientemente introducido en la literatura y por tal razón ha sido poco estudiado hasta el momento, sin embargo, desde que fue introducido se han propuesto diferentes métodos para su solución, tanto heurísticos como exactos, diferentes formulaciones y algunas aplicaciones que pueden ser estudiadas a través de él. No obstante, no se conoce algún trabajo que haya implementado el método solución que se propone en esta tesis.

En este capítulo se presenta el estado del arte del TCMCRP y otros trabajos existentes en la literatura que comparten características con el problema de estudio, comenzando con el clásico problema del agente viajero (TSP), sus variantes y generalizaciones.

2.1 EL PROBLEMA DEL AGENTE VIAJERO (TSP) Y SUS VARIANTES

El problema del agente viajero, conocido también por sus siglas en inglés como TSP (Flood, 1956), dispone de un conjunto de nodos (ciudades, clientes, localidades, etc.) de los que se conoce la distancia entre cada par de ellos, de tal forma que un

solo agente o vehículo debe visitarlos a todos en una ruta. El objetivo es encontrar la ruta de menor costo que visite cada nodo exactamente una vez y regrese al nodo origen. Este problema pertenece a la clase de problemas NP-difícil (Woeginger, 2003) y es uno de los más clásicos y estudiados dentro de los problemas de optimización combinatoria.

Hasta el día de hoy, el TSP ha sido utilizado para representar una gran variedad de problemas dentro de los campos de la logística, industria, organización de datos, robótica, entre otros. Algunos ejemplos de estas aplicaciones son las siguientes:

Programación de tareas en una máquina. González Velarde y Ríos Mercado (1999) plantean el problema de una máquina en la cual se pueden procesar diferentes tareas, una a la vez. Para realizar el procesamiento de cada una de las tareas, la máquina necesita de cierta configuración característica de la tarea, de manera que una vez terminada es necesario preparar la máquina para procesar una nueva tarea. Es importante encontrar el orden en que se deben procesar las tareas con la finalidad de reducir el tiempo de preparación de la máquina. En este caso, se puede formular como un TSP suponiendo que cada tarea representa cada uno de los nodos a visitar, y el tiempo necesario para realizar la configuración de la máquina corresponde a la distancia que hay entre cada nodo. De manera que encontrar el orden de las tareas que minimice el tiempo total de preparación es equivalente a diseñar la ruta que minimice la distancia total recorrida.

Distribución de mercancía. Generalmente las empresas que distribuyen artículos perecederos necesitan hacerlo en un tiempo corto, para ello disponen de un almacén central en donde se concentran los artículos y una flotilla de unidades de transporte que se encarga de visitar a los clientes para hacer entrega de la mercancía. Tomando el caso particular en el cual la empresa dispone de una sola unidad nos lleva al problema del TSP, dado que se debe determinar la ruta que debe seguir la unidad para entregar la mercancía en el menor tiempo posible (González Velarde y Ríos Mercado, 1999).

Planificación de rutas laborales y/o escolares. Determinar una ruta escolar óptima, fue una de las primeras aplicaciones del TSP (Applegate *et al.*, 2006). Actualmente existen empresas encargadas del transporte de personas, que hacen uso de algún software que resuelve el problema del agente viajero con el objetivo de reducir sus gastos de manera significativa.

Turismo y agencia de viajes. Generalmente las agencias de viajes no tienen un conocimiento explícito o no suelen buscar en la literatura sobre el problema del agente viajero, ya que a menudo utilizan algún software para el diseño de la ruta, que incluye algún algoritmo que resuelve el TSP (Applegate *et al.*, 2006).

Organización de datos en clústers. Un problema básico en el análisis de datos es la agrupación u organización de datos en grupos. Jain y Maheswari (2012) realizaron una revisión literaria sobre la importancia de la agrupación de datos donde mencionan algunas aplicaciones en reconocimiento de objetos, recuperación de información y minería de datos. Dada una buena medida de similitud entre cada par de datos y utilizando esta medida como distancia de tal forma que un camino hamiltoniano de costo máximo sitúe las observaciones más semejantes unas cercas de otras, se podrían entonces utilizar intervalos del camino como clústers. Dado que el problema del agente viajero busca encontrar una ruta de costo mínimo bastaría con multiplicar las distancias por (-1) que sería equivalente al camino de máximo costo. Este enfoque TSP se utilizó por primera vez en 1972 por McCormick Jr *et al.*, pero la relación es más explícita en el trabajo de Lenstra (1974).

Así como estas aplicaciones, existen muchas más, tales como el envío de postales, reparto de bienes o servicios, perforación de placas de los circuitos impresos (Grötschel y Holland, 1991), etc. Además, el TSP ha aparecido en muchos otros escenarios, como por ejemplo, en un problema de programación de una máquina (Gilmore y Gomory, 1964), en un modelo para minimizar el desperdicio de un papel tapiz (Garfinkel, 1977), problema de recolección de artículos en un almacén rectangular (Ratliff y Rosenthal, 1983), planificación de la producción en la industria del

vidrio (Madsen, 1988), organización de lista de reproducción en dispositivos portátiles (Pohle *et al.*, 2007), otros trabajos con modelos y aplicaciones que usan el TSP se pueden encontrar en los trabajos de Reinelt (1994), Lenstra y Kan (1975) y Davendra (2010).

Existen varias variantes del TSP, a continuación se presentan las más relevantes:

TSP con múltiples visitas (*Travelling Salesman Problem with Multiple visits*, TSPM). De la misma forma que el TSP clásico, este problema consiste en visitar cada nodo al menos una vez, partiendo y regresando al nodo inicial, de tal manera que se minimice la distancia total recorrida (Gutin y Punnen, 2007).

TSP múltiple (*Multiple Travelling Salesman Problem*, MTSP). El problema del agente viajero con múltiples viajantes consiste en encontrar rutas para todos los m -agentes viajeros que empiezan y terminan en el origen, de modo que cada ciudad debe ser visitada una sola vez por un solo vendedor y se minimice el costo total de visitar todas las ciudades (Bektas, 2006). Una de las principales aplicaciones del MTSP surge en la programación de una imprenta (Gorenstein, 1970; Carter y Ragsdale, 2002), mientras que Angel *et al.* (1972) investigaron un problema de ruteo de vehículos de transporte escolar como una variación del MTSP con algunas restricciones adicionales.

TSP periódico (*Periodic Travelling Salesman Problem*, PTSP). Este problema cumple con las mismas características básicas del TSP con la diferencia de que un cliente no puede ser visitado más de una vez en el mismo periodo manteniendo el objetivo de minimizar costos. Vega Ocaña (2008) describe este problema como “un comerciante que debe visitar a diferentes clientes un número determinado de veces, en un intervalo de tiempo preestablecido, por ejemplo, un mes”.

TSP con ventanas de tiempo (*Traveling Salesman Problem with Time Windows*, TSPTW). Consiste en buscar la ruta que minimice costos, empezando y terminando en el nodo origen y visitando a todos los clientes una sola vez, de acuerdo

con un rango de tiempo predefinido que estos hayan establecido, conocida como ventana de tiempo. No se permite visitar al cliente después de este intervalo de tiempo, pero es posible llegar antes y esperar hasta poder empezar el servicio.

TSP selectivo (*Orienteering Problem*, OP). Es un problema perteneciente a la clase de problemas de ruteo de vehículos con beneficios, donde se tiene un conjunto de ubicaciones o clientes asociados a una puntuación por ser visitadas y se tiene un solo vehículo. El objetivo es maximizar la puntuación colectada, sujeto a que la duración de la ruta no puede exceder de un límite específico. El OP fue introducido por Tsiligirides (1984) y en la literatura también es conocido como *Selective Traveling Salesman Problem* (Laporte y Martello, 1990), *Maximum Collection Problem* (Kataoka y Morito, 1988) y *Bank Robber Problem* (Arkin *et al.*, 1998). El término *Orienteering Problem*, fue introducido por Golden *et al.* (1987).

TSP selectivo múltiple (*Team Orienteering Problem*-TOP). El TOP es una extensión del OP realizada por primera vez por Butt y Cavalier (1994) con el nombre de *Multiple Tour Maximum Collection Problem* (MTMCP) que se obtiene al disponer de múltiples vehículos para visitar a los clientes. El objetivo es maximizar la puntuación colectada por todas las rutas, sujeto a que la duración de cada una de estas no exceda de un tiempo límite. Más tarde Chao *et al.* (1996) se refirieron a este problema con el nombre de *Team Orienteering Problem*, por primera vez.

Varias extensiones del OP y TOP se encuentran en la literatura, incluyendo el *Orienteering Problem with Time Windows* (OPTW)(Righini y Salani, 2009; Kantor y Rosenwein, 1992), y el *Team Orienteering Problem with Time Windows* (TOPTW) (Vansteenwegen *et al.*, 2009; Montemanni y Gambardella, 2009).

Al considerar restricciones de capacidad en los vehículos, el OP se extiende al *Capacitated Orienteering Problem* (COP) (Bock y Sanità, 2015) y el TOP al *Capacitated Team Orienteering Problem* (CTOP), propuesto por Archetti *et al.* (2009). Una revisión completa de las variantes recientes y los enfoques de solución del OP y TOP se pueden encontrar en los artículos de Gunawan *et al.* (2016) y Vansteenwegen

et al. (2011).

Como se ha visto, las aplicaciones del TSP y sus variantes van mucho más allá de resolver un problema de planificación de rutas para un vendedor ambulante y abarcan varias áreas de conocimiento como las matemáticas, ciencias de la computación, investigación de operaciones, ingeniería y electrónica (Gutin y Punnen, 2007).

2.2 EL PROBLEMA DE RUTEO DE VEHÍCULOS (VRP) Y SUS VARIANTES

El problema de ruteo de vehículos conocido como VRP por sus siglas en inglés es una generalización del TSP que surge por primera vez en 1959, por Dantzig y Ramser (1959) quienes representaron una aplicación real relacionada con el problema de distribución de combustible a las estaciones de servicio y propusieron la formulación matemática a este problema. En 1964 Clarke y Wright propusieron el primer algoritmo que dio solución a este problema, conocido como el algoritmo de los ahorros. Desde entonces es motivo de considerables investigaciones que han dado lugar a diversos modelos y búsqueda de métodos de solución que sean cada vez más eficientes.

El VRP consiste en, dado un conjunto de clientes, un depósito central y una flotilla de vehículos, determinar un conjunto de rutas que empiezan y terminan en un determinado depósito, de modo que satisfagan todas las demandas de los clientes dispersos geográficamente sin exceder la capacidad máxima de cada vehículo y que cada cliente sea visitado por un solo vehículo. El problema puede estar sujeto a diversas restricciones, tales como la capacidad del vehículo, longitud de la ruta, ventanas de tiempo, características de los clientes, el depósito, etc. Por lo general, en este tipo de problemas se suele buscar minimizar costos de transporte, tiempo total recorrido, tiempos de espera, el número vehículos o maximizar beneficios, servicio al cliente, entre otros.

El problema de ruteo de vehículos es un tema central en las redes de distribución con diversas aplicaciones en transporte, logística, distribución y telecomunicaciones. Pertenece a la clase NP-difícil pues el TSP pertenece a esta clase, y es un caso particular del VRP, cuando se tiene solo un vehículo disponible y no existen restricciones de capacidad ni distancia total recorrida (Lüer *et al.*, 2009).

Las variantes del VRP se clasifican en base a las características de los clientes, depósitos, vehículos y de las restricciones operativas. Algunas de estas se explican a continuación (Luna López, 2015; Recio Hernández, 2014):

- VRP múltiples depósitos (*Multiple Depot Vehicle Routing Problem*, MDVRP). Se cuenta con diversos depósitos desde los cuales se puede abastecer a los clientes y cada uno tiene asignado una flota de vehículos. A cada depósito se le asigna un número de clientes que son atendidos por los vehículos asignados al depósito, con el objetivo de reducir la distancia total recorrida o minimizar la flota de vehículos asignados al depósito.
- VRP capacitado (*Capacitated Vehicle Routing Problem*, CVRP). Cada vehículo cuenta con una capacidad y cada cliente tiene una demanda que debe ser cubierta por algún vehículo.
- VRP con ventanas de tiempo (*Vehicle Routing Problem with Time Windows*, VRPTW). Cada cliente debe ser visitado dentro de un cierto rango de tiempo predefinido, con el objetivo de minimizar la cantidad de vehículos utilizados, el tiempo total de viaje y el tiempo de espera necesario para abastecer a todos los clientes en sus respectivos horarios. Entre sus aplicaciones se encuentran la entrega de comida, recolección de residuos y problema de ruteo del transporte escolar.
- VRP con recogida y entrega (*Vehicle Routing Problem with Pick-up and Delivering*, VRPPD). Se incluye la recolección y entrega de mercancía en lugar de solo entrega. Se contempla además, la posibilidad de que los clientes puedan hacer devolución de la mercancía y por tal razón es necesario tomar en cuenta

que el vehículo tenga espacio suficiente para los productos a entregar o las devoluciones. El objetivo es encontrar rutas óptimas de visita a los lugares de entrega y recolección para una flota de vehículos.

- VRP con entregas divididas por diferentes vehículos (*Split Delivery Vehicle Routing Problem*, SDVRP). Los clientes pueden ser abastecidos por distintos vehículos, solamente si esto ayuda a reducir los costos totales de la ruta. El objetivo es el diseño de rutas tal que se minimice la distancia total recorrida.
- VRP estocástico (*Stochastic Vehicle Routing Problem*, SVRP). En esta variante se considera que algunos valores como el número de clientes, sus demandas, tiempo de servicio o tiempo de viaje tienen una probabilidad de presencia o de ausencia, son al azar o aleatorios. Debido a esto, no es posible requerir que todas las restricciones se satisfagan para todas las realizaciones de las variables aleatorias.
- VRP periódico (*Periodic Vehicle Routing Problem*, PVRP). A diferencia del VRP clásico donde la planeación que se realiza es de un día, en el PVRP se generaliza extendiendo el período de planeación a m días. Para esta variante se considera que un cliente debe ser visitado mínimo una vez dentro de un período de tiempo definido, los vehículos tienen capacidad de carga, se tiene que satisfacer la demanda de cada cliente y el vehículo debe regresar al depósito dentro del período de tiempo ya definido. El objetivo es minimizar el costo total de todas las rutas. Además, cada cliente tiene una demanda diaria conocida que debe ser satisfecha en sólo una visita y con un sólo vehículo.

Algunas otras variantes son el VRP con restricciones de distancia (*Distance Vehicle Routing Problem*, DVRP), VRP con restricciones de capacidad y distancia (*Distance and Capacitated Vehicle Routing Problem*, DCVRP), VRP con múltiples ventanas de tiempo (*Vehicle Routing Problem with Multiple Time Windows*, VRPMTW), VRP con ventanas de tiempo duras (*Vehicle Routing Problem with Time Deadlines*, VRPTD), VRP con ventanas de tiempo flexibles (*Vehicle Routing Problem with Soft Time Windows*, VRPSTW), VRP con

retornos (*Vehicle Routing Problem with Backhauls*, VRPB), VRP con entregas divididas y ventanas de tiempo (*Split Delivery Vehicle Routing Problem with Time Windows*, SDVRP), VRP con flota heterogénea (*Vehicle Routing Problem Heterogeneous Fleet*, VRPHF), VRP con recogida y entrega con ventanas de tiempo (*Vehicle Routing Problem with Pick-up and Delivering and Time Windows*, VRPPDTW), VRP con tamaño de flota (*Vehicle Routing Problem with Fleet Size*, FSVRP), VRP con múltiples capacidades (*Multi Capacity Vehicle Routing Problem*, MCVRP), entre otros.

En un marco de enrutamiento, puede no ser viable visitar a cada cliente por separado debido a limitaciones de recursos o problemas de eficiencia. En tales casos, utilizando la noción de cobertura; es decir, satisfacer la demanda de múltiples clientes visitando una ubicación de un solo cliente, puede ser ventajoso.

2.3 EL PROBLEMA DE COBERTURA Y SUS VARIANTES

La noción de cobertura se relaciona con muchas aplicaciones prácticas en escenarios reales en los que se cuenta con una cantidad de recursos limitada, de manera que no es posible satisfacer la demanda total de todos los clientes o no resulta viable visitar a cada cliente individualmente. Sin embargo, la cobertura se basa en la capacidad de los clientes para acercarse a las instalaciones o la capacidad de las instalaciones para cubrir a los clientes que se encuentran dentro de un área, radio o tiempo de servicio limitado (Ibarra-Rojas *et al.*, 2018). Bajo esta situación, problemas como la determinación de la cantidad y la ubicación de las escuelas públicas, estaciones de policía, bibliotecas, hospitales, edificios públicos, oficinas de correos, parques, jarrones militares, instalaciones de radar, sucursales, centros comerciales e instalaciones de eliminación de residuos se pueden formular como problemas de cobertura (Francis *et al.*, 1992), cuyo objetivo es maximizar la cantidad de demanda cubierta teniendo en cuenta estas limitaciones. A continuación, se presentan algunos

problemas relacionados y sus variantes.

Con respecto a la historia, Hakimi (1965) introduce por primera vez la noción de cobertura en los modelos de ubicación de instalaciones. El objetivo era determinar el mínimo número de policías necesarios para cubrir n nodos en una red de autopistas. Este ha sido uno de los problemas más populares debido a su aplicación en la vida real, especialmente en instalaciones de servicio y emergencia.

El problema de máxima cobertura (*Maximal Covering Problem*, MCP) fue introducido por Church y Velle (1974) en el cual se tiene un conjunto de clientes e instalaciones donde cada instalación puede satisfacer la demanda de algunos clientes que se encuentran dentro de una distancia predeterminada. El objetivo es maximizar el número de clientes cubiertos utilizando solamente algunas instalaciones. La variante de este problema en el que cada instalación cuenta con una capacidad limitada lo estudiaron en 1991 Pirkul y Schilling.

El problema del vendedor con cobertura (*Covering Salesman Problem*, CSP) es una extensión del problema del agente viajero (TSP), en el que se tiene un conjunto de clientes y cada cliente puede cubrir un subconjunto de otros clientes que se encuentran dentro de su radio de cobertura. El objetivo es construir una ruta de costo mínimo sobre un subconjunto de clientes, de manera que cada cliente no visitado debe estar dentro de un radio de cobertura de al menos un cliente que ha sido visitado por la ruta (Current y Schilling, 1989).

En el 2012, Golden *et al.* proponen una versión generalizada del CSP cuyo objetivo es encontrar una ruta de costo mínimo sobre un subconjunto de vértices de tal manera que cada vértice deber ser cubierto o visitado al menos k veces. Una de sus aplicaciones está en los equipos de atención de salud en zonas rurales donde los equipos solo pueden atender un número limitado de personas y por ello algunos puntos deben ser visitados más de una vez.

Posteriormente Gendreau *et al.* (1997) propusieron el problema del recorrido con cobertura (*Covering Tour Problem*, CTP) que es una generalización del vende-

dor con cobertura. En este problema se tienen tres tipos de vértices: obligatorios, opcionales e inalcanzables (es decir, están dentro de una distancia específica de un vértice visitado). El objetivo es construir una ruta de costo mínimo que debe visitar cada vértice obligatorio y un subconjunto de vértices opcionales para cubrir los vértices inalcanzables.

En el 2017 Flores-Garza *et al.* presentan el problema del tour con cobertura acumulativa con varios vehículos como una variante del problema del tour con cobertura con múltiples vehículos (*Multi-vehicle Covering Tour Problem, m-CTP*), el cual es aplicado en el contexto de la logística humanitaria y cuyo objetivo es determinar un conjunto de rutas para minimizar la suma de los tiempos de llegada (latencia) a las localidades visitadas. Una variante generalizada del *m-CTP* lo presentan Pham *et al.* (2017) y lo denominan como *Multi-vehicle Multi-Covering Tour Problem (mm-CTP)*.

Por otra parte, Current y Schilling (1994) introducen dos problemas bi-objetivo conocidos como problema del recorrido medio (*Median Tour Problem, MTP*) y el problema del recorrido de máxima cobertura (*Maximal Covering Tour Problem, MCTP*). En ambos problemas el recorrido debe visitar solo i de los n nodos en la red. Además, ambos tienen como primer objetivo minimizar de la duración total del recorrido. El segundo objetivo es maximizar el acceso de las rutas a los nodos que no fueron visitados. Este objetivo en el MTP, se realiza minimizando la distancia total recorrida desde los nodos no incluidos directamente en la ruta hasta la parada más cercana a la ruta, mientras que en el MCTP el objetivo se logra maximizando la demanda total dentro de una distancia máxima de viaje preespecificada desde una parada de la ruta. Estos dos problemas tienen aplicación en la prestación de asistencia sanitaria rural, el diseño de redes informáticas distribuidas y el diseño sistemas que prestan servicios móviles.

Naji-Azimi y Salari (2014) introducen el problema denominado *Time Constrained Maximal Covering Salesman Problem (TCMCSP)*, que es la generalización

del CSP y el OP. En este problema se tiene un depósito central, un conjunto de clientes e instalaciones. Cada instalación puede cubrir un subconjunto de clientes que se encuentran dentro de una distancia de cobertura predeterminada. El problema consiste en determinar cuáles son las instalaciones a visitar y cuál será el orden de las visitas que debe realizar el vehículo, de tal forma que la duración total de la ruta no exceda un tiempo límite y se maximice el número total de clientes cubiertos con las instalaciones visitadas. Entre las aplicaciones que mencionan se encuentra el diseño de redes de telecomunicaciones móviles, prestación de servicios de salud en zonas rurales, situaciones de desastre causadas por eventos naturales o causados por el hombre o durante el proceso de postulación de un político. Posteriormente, Ozbaygin *et al.* (2016) estudian este mismo problema con demandas ponderadas y cobertura parcial, para el cual proponen dos formulaciones en las que utilizan desigualdades válidas, diseñan y prueban enfoques de ramificación y corte para resolverlo.

2.4 PROBLEMA DE RUTEO CON MÁXIMA COBERTURA Y TIEMPO LÍMITE (TCMCRP)

En sus orígenes, el TCMCRP fue estudiado por primera vez en el 2019 por Amiri y Salari, como la generalización del TCMCSP, en el que se consideran varios vehículos en lugar de uno solo, debido a que en la práctica se suele contar con varios vehículos. Propusieron un modelo de programación lineal entera mixta y tres algoritmos heurísticos, búsqueda local iterada (*Iterated Local Search*, ILS), búsqueda tabú (*Tabu Search*, TS) y búsqueda de vecindad variable (*Variable Neighborhood Search*, VNS), para resolver el problema.

Un año más tarde, Sinnl (2019) presenta en su reporte técnico *Mixed-Integer Programming Approaches for the Time Constrained Maximal Covering Routing Problem* dos algoritmos de solución exacta, el primero es basado en un modelo compacto denominado C_+ , mientras que el otro modelo E_{++} contiene un número de restric-

ciones exponencial, además de utilizar técnicas de preprocesamiento y desigualdades validas para ambos modelos.

Como se ha mostrado, existen diversos trabajos que comparten similitudes con el problema en estudio, los cuales se resumen en la Tabla 2.1.

Autores	Número de rutas	Capacidad	Visitas obligatorias	Tiempo límite	Método de solución
Chao <i>et al.</i> (1996)	Múltiples	No	No	Si	LS
Ha <i>et al.</i> (2013)	Múltiples	No	No	Si	Branch and bound y ELS
Schittekat <i>et al.</i> (2013)	Múltiples	Si	No	No	GRASP
Naji-Azimi y Salari (2014)	Una	Si	No	No	LS
Shaelaie <i>et al.</i> (2014)	Una	No	No	No	MA y VNS
Kammoun <i>et al.</i> (2015)	Múltiples	Si	No	Si	VNS
Ozbaygin <i>et al.</i> (2016)	Una	Si	No	Si	Branch and cut
Amiri y Salari (2019)	Múltiples	No	No	Si	Branch and bound, ILS, TS y VNS
Simnl (2019)	Múltiples	No	No	Si	Branch and cut

Tabla 2.1: Estado del arte relacionado al TCMCRP

CAPÍTULO 3

DESCRIPCIÓN Y MODELACIÓN DEL PROBLEMA

En este capítulo se describe formalmente el problema bajo estudio y se presenta la modelación matemática del mismo.

El problema de estudio combina dos importantes problemas de optimización combinatoria: el problema del vendedor con cobertura (CSP) y el problema del vendedor selectivo múltiple (TOP). En ambos problemas se considera un conjunto de clientes que son visitados a lo más una vez. La diferencia entre ambos radica en el número de agentes o vehículos y en el criterio de desempeño que se considera en cada caso. En el problema del vendedor con cobertura se busca la construcción de una ruta de costo mínimo sobre un subconjunto de n clientes determinados, de manera que cada cliente no visitado se encuentre dentro de un radio de cobertura de al menos uno de los clientes que estén en la ruta. Por otra parte, en el problema del agente viajero selectivo múltiple se tiene un conjunto de agentes o vehículos y un conjunto de clientes, cada uno de ellos asociado a un puntaje que es recolectado únicamente si es visitado por uno de los vehículos. El objetivo del TOP es maximizar la puntuación colectada por todas las rutas, donde cada ruta está sujeta a una restricción de tiempo.

Como ya se mencionó anteriormente, existen diversas formulaciones reportadas

en la literatura para el problema de ruteo con máxima cobertura y tiempo límite. Sin embargo, el modelo que se tomará como referencia es el propuesto por Amiri y Salari (2019).

3.1 FORMULACIÓN MATEMÁTICA

A continuación se presenta un modelo de programación lineal entera mixta, que aborda el problema de estudio. Este modelo fue propuesto por Amiri y Salari (2019).

Primero, en la Sección 3.1.1 se introduce la notación matemática general que será utilizada a lo largo de la investigación, seguido por la formulación explicada a detalle en la Sección 3.1.2.

3.1.1 DESCRIPCIÓN FORMAL DEL PROBLEMA

Formalmente, el problema de estudio se define como sigue. Sea $G = (V, A)$ un grafo dirigido, donde $V = \{0\} \cup T \cup W$ es el conjunto de nodos, $A = \{(i, j) | i, j \in T \cup \{0\}\}$ es el conjunto de arcos y $\{0\}$ representa el depósito central.

3.1.2 MODELO MATEMÁTICO

Conjuntos

T : Conjunto de instalaciones.

W : Conjunto de clientes.

P : Conjunto de vehículos homogéneos.

Parámetros

L_k : Tiempo máximo de duración de la ruta $k \in P$

$D = [d_{ij}]$: Matriz de cobertura, donde $d_{ij} = 1$ si el cliente i se encuentra dentro de un radio de cobertura preespecificado de la instalación j , $d_{ij} = 0$ en otro caso.

t_{ij} : Tiempo de traslado de la instalación i a la instalación j .

Variables de decisión

$$x_{ijk} = \begin{cases} 1 & \text{si se visita la instalación } j \text{ después de la } i \\ & \text{por el vehículo } k \\ 0 & \text{en otro caso;} \end{cases} \quad \forall i, j \in \{0\} \cup T, k \in P$$

$$z_{ijk} = \begin{cases} 1 & \text{si el cliente } i \text{ es asignado a la instalación } j \\ & \text{cuando es visitada por el vehículo } k \\ 0 & \text{en otro caso;} \end{cases} \quad \forall i \in W, j \in T, k \in P$$

$$y_{jk} = \begin{cases} 1 & \text{si la instalación } j \text{ es visitada por el vehículo } k \\ 0 & \text{en otro caso;} \end{cases} \quad \forall j \in T, k \in P$$

f_{ijk} : Tiempo total acumulado desde el depósito hasta la instalación $j \in T \cup \{0\}$ cuando se visita la instalación j después de la i por el vehículo k .

Función objetivo

$$\text{Max} \sum_{i \in W} \sum_{j \in T} \sum_{k \in P} d_{ij} z_{ijk} \quad (3.1)$$

sujeto a:

$$\sum_{i \in T \cup \{0\}} \sum_{j \in T \cup \{0\}} t_{ij} x_{ijk} \leq L_k \quad \forall k \in P \quad (3.2)$$

$$\sum_{i \in T \cup \{0\}} x_{ijk} = \sum_{i \in T \cup \{0\}} x_{jik} = y_{jk} \quad \forall j \in T, \forall k \in P \quad (3.3)$$

$$\sum_{j \in T \cup \{0\}} x_{j0k} = 1 \quad \forall k \in P \quad (3.4)$$

$$\sum_{j \in T \cup \{0\}} x_{0jk} = 1 \quad \forall k \in P \quad (3.5)$$

$$z_{ijk} \leq y_{jk} \quad \forall i \in W, \forall j \in T, \forall k \in P \quad (3.6)$$

$$\sum_{k \in P} \sum_{j \in T} z_{ijk} \leq 1 \quad \forall i \in W \quad (3.7)$$

$$\sum_{k \in P} y_{jk} \leq 1 \quad \forall j \in T \quad (3.8)$$

$$\sum_{j \in (T \setminus \{i\}) \cup \{0\}} f_{ijk} - \sum_{j \in (T \setminus \{i\}) \cup \{0\}} f_{jik} = \sum_{j \in T \cup \{0\}} t_{ij} x_{ijk} \quad \forall i \in T, \forall k \in P \quad (3.9)$$

$$f_{0ik} = t_{0i} x_{0ik} \quad \forall i \in T, \forall k \in P \quad (3.10)$$

$$f_{i0k} \leq L_k x_{i0k} \quad \forall i \in T, \forall k \in P \quad (3.11)$$

$$f_{ijk} \leq (L_k - t_{j0}) x_{ijk} \quad \forall (i, j) \in A, j \neq 0, \forall k \in P \quad (3.12)$$

$$f_{ijk} \geq (t_{0i} + t_{ij}) x_{ijk} \quad \forall (i, j) \in A, j \neq 0, \forall k \in P \quad (3.13)$$

$$x_{ijk} + x_{jik} \leq 1 \quad \forall i, j \in T, \forall k \in P \quad (3.14)$$

$$x_{ijk} \leq y_{jk} \quad \forall i, j \in T, \forall k \in P \quad (3.15)$$

$$f_{ijk} \geq 0 \quad \forall (i, j) \in A, \forall k \in P \quad (3.16)$$

$$x_{ijk} \in \{0, 1\} \quad \forall i, j \in T \cup \{0\}, \forall k \in P \quad (3.17)$$

$$y_{jk} \in \{0, 1\} \quad \forall j \in T, \forall k \in P \quad (3.18)$$

$$z_{ijk} \in \{0, 1\} \quad \forall i \in W, j \in T, \forall k \in P \quad (3.19)$$

El objetivo (3.1) maximiza el número total de clientes cubiertos. Las restricciones (3.2) establecen que el visitar la instalación j después de la i no debe exceder del tiempo máximo de duración de la ruta $k \in P$. Las restricciones (3.3) mantienen la conectividad en cada ruta. El conjunto de restricciones (3.4) y (3.5) aseguran que todas las rutas salgan y regresen al depósito central. Las restricciones (3.6) aseguran que el cliente i solo puede ser asignado a la instalación j si es visitada por el vehículo k , es decir, si $y_{jk} = 1$. Por otra parte, en las restricciones (3.7), cada cliente es asignado a lo más a una instalación que es visitada por un vehículo. Las restricciones (3.8) indican que cada instalación puede ser visitada a lo más por un vehículo. El

conjunto de restricciones (3.9) - (3.13) prohíben los subtours. Las restricciones (3.9) indican la conservación de flujo. Las restricciones (3.10) - (3.15) son desigualdades válidas. Las restricciones (3.10) indican que el tiempo total acumulado desde el depósito a cualquier instalación $i \in T$, es igual al tiempo de traslado del depósito a i . Las restricciones (3.11) representan que el tiempo acumulado desde el depósito central a la instalación $i \in T$ cuando es utilizado el vehículo $k \in P$ no puede exceder de L_k . El conjunto de restricciones (3.12) establecen un límite superior para f_{ijk} que asegura que la última instalación en la ruta k pueda regresar al depósito, mientras que las restricciones (3.13) establecen un límite inferior para f_{ijk} que muestra que el tiempo total acumulado hasta j debe ser mayor o igual a la suma del tiempo desde el depósito hasta i y el tiempo entre i y j . Las restricciones (3.14) no permiten subtours entre dos instalaciones. Las restricciones (3.15) aseguran que se debe visitar la instalación j si la instalación j es visitada por el vehículo k . Finalmente, las restricciones (3.16) - (3.19) especifican el tipo de variables de decisión.

CAPÍTULO 4

METODOLOGÍA DE SOLUCIÓN

En este trabajo se propone una metaheurística tipo ALNS como método de solución al problema de estudio. Debido a la naturaleza del problema que hace que sea NP-difícil, las aplicaciones en situaciones de emergencia y que el tiempo computacional necesario para encontrar soluciones óptimas de este problema puede llegar a ser muy grande para instancias de más de 100 nodos, es importante el uso de métodos basados en metaheurísticas que generen soluciones de buena calidad en un tiempo razonable.

La búsqueda adaptativa de grandes vecindarios ó ALNS por sus siglas en inglés (Adaptive Large Neighborhood Search) es una metaheurística que está compuesta por una etapa de construcción de una solución inicial seguida de un postprocesamiento de la misma, haciendo uso de múltiples operadores de destrucción y reparación. El proceso continúa iterativamente hasta satisfacer un criterio de paro y se reporta la mejor solución encontrada. En este capítulo, se describe el funcionamiento y los componentes básicos del ALNS; del mismo modo, se explica la manera en que este algoritmo fue adaptado para el problema de estudio.

4.1 ADAPTIVE LARGE NEIGHBORHOOD SEARCH

El ALNS es una metaheurística propuesta por Ropke y Pisinger (2006) que permite el uso de múltiples operadores de destrucción y reparación, los cuales son seleccionados de manera aleatoria, siguiendo una distribución de probabilidad que se ajusta automáticamente conforme al desempeño del algoritmo, esto es, si con el uso de algún operador de destrucción y reparación se han obtenido buenas soluciones, se les asigna un peso o puntuación de manera que, la probabilidad de ser elegidos nuevamente sea alta.

Cabe mencionar que los operadores de destrucción, como su nombre lo indica se encargan de destruir una solución inicial de diferentes maneras, mientras que los operadores de construcción son procedimientos capaces de construir una solución completa dada una solución parcial. La cantidad de operadores de destrucción y reparación es indiferente, dado que se puede tener la misma cantidad de cada uno o diferente. Además, los operadores pueden tener la opción de compatibilidad entre ellos, es decir, cualquier operador de destrucción puede ser compatible con cualquier operador de reparación ó en su defecto cada uno de los operadores de destrucción es compatible únicamente con algunos de los operadores de reparación.

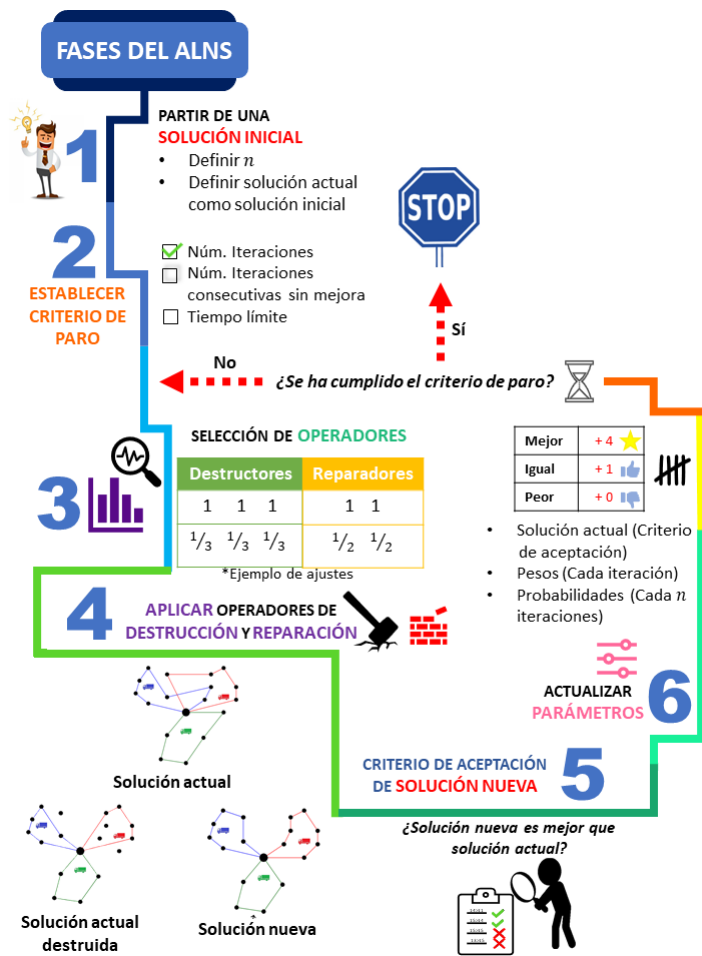


Figura 4.1: Descripción general del ALNS.

La Figura 4.1 describe gráficamente la estructura del ALNS propuesto en esta tesis. Los pasos 3-6 representan el ciclo que ocurre en cada iteración del algoritmo.

- Paso 1. Se comienza con una solución inicial que puede provenir de alguna heurística constructiva o metaheurística. Posteriormente, la solución inicial pasa a ser solución actual. Además, se define n como el número de iteraciones para la actualización de las probabilidades de elección de los operadores.
- Paso 2. Se establece el criterio de paro para el ALNS, este puede ser un número de iteraciones, tiempo límite o un número de iteraciones consecutivas sin mejora.

- Paso 3. A los operadores se les asignan pesos iniciales de manera uniforme y se realiza el ajuste de las probabilidades que tendrán con base a estos pesos. Sea $\Omega^- = \{a_i | i = 1, \dots, k\}$ el conjunto de k operadores de destrucción y $\Omega^+ = \{r_i | i = 1, \dots, l\}$ el conjunto de l operadores de reparación. Los pesos iniciales se denotan por $w(r_i)$ y $w(a_i)$, donde $r_i \in \Omega^+$ y $a_i \in \Omega^-$, de modo que las probabilidades de seleccionar los operadores son $p(r_i) = \frac{w(r_i)}{\sum_{j=1}^l w(r_j)}$ y $p(a_i) = \frac{w(a_i)}{\sum_{j=1}^k w(a_j)}$, respectivamente.
- Paso 4. Una vez que se hace la selección de operadores en base a sus probabilidades, se procede a aplicarlos para destruir la solución actual, generar una solución parcial y repararla, dando origen a una solución nueva. Cabe destacar que, las probabilidades de selección de cada uno de los operadores de destrucción y reparación dependen del desempeño obtenido en las iteraciones previas del ALNS.
- Paso 5. Solución nueva es evaluada bajo un criterio de aceptación donde se analiza la pregunta *¿Solución nueva es mejor que solución actual?*, es decir, si el valor de la función objetivo de solución nueva es mayor que el valor de solución actual.
- Paso 6. Se realiza la actualización de los siguientes parámetros:

 1. La actualización de la solución actual depende del criterio de aceptación del paso 5, es decir, únicamente se realiza la actualización si el valor de la función objetivo de la solución nueva es mayor que el valor de la función objetivo de la solución actual, en este caso, solución nueva pasa a ser solución actual. De lo contrario, solución actual se mantiene.
 2. Los pesos de los operadores se actualizan en cada iteración y aumentan dependiendo de la calidad de la solución nueva, misma que se genera al aplicar un operador de destrucción y reparación en cada iteración del ALNS. Un ejemplo es aumentar cuatro unidades si el valor de la función objetivo de la solución nueva supera el valor de la solución actual, aumen-

tar una unidad si hay un empate entre estos dos valores y como penalidad aumentar cero unidades si el valor de la función objetivo de solución nueva es menor que el valor de solución actual. Esto tiene como finalidad ir sumándole más peso a los operadores que van teniendo un buen desempeño en cada iteración y de esta manera, aumentar la probabilidad de ser elegidos.

3. La actualización de las probabilidades se hace cada n iteraciones para dar oportunidad de que se vayan acumulando los pesos de los operadores.

El ALNS continúa volviendo al paso 3 hasta que se cumpla el criterio de paro establecido.

4.1.1 SOLUCIÓN INICIAL

En el primer paso del ALNS se obtiene una solución inicial del problema haciendo uso de una heurística constructiva, es decir, se obtiene la solución de forma gradual incorporando en cada iteración el elemento con mejor contribución local a la calidad de la misma. Dicha contribución se mide mediante una función de evaluación.

La construcción de la solución inicial comienza de la siguiente manera: para cada ruta $k \in P$, se elige qué instalación $j \in T \setminus \{0\}$, que aún no ha sido incluida en alguna ruta, será visitada con base en la función de evaluación (4.1).

$$f(j) = |C_j| \tag{4.1}$$

Donde C_j es el conjunto de clientes que son cubiertos por la instalación j (cobertura actualizada), descartando los clientes que ya han sido cubiertos por alguna otra instalación que haya sido incluida anteriormente en alguna ruta. La instalación elegida j^* es aquella que cumple con la expresión (4.2), y es insertada en la ruta actual como última instalación visitada. Si ocurre un empate en la función de evaluación, se elige la instalación más cercana a la última en la ruta.

$$j^* = \operatorname{argmax}_{j \in T \setminus \{0\}} \{f(j)\} \quad (4.2)$$

El proceso anterior de inserción de instalaciones se realiza de manera iterativa hasta que ya no haya instalaciones por visitar o no sea posible insertar nuevas instalaciones dado que se excedería el tiempo límite establecido L_k .

El Algoritmo 1 presenta a detalle el proceso de construcción de la solución, a este algoritmo lo llamaremos Constructivo.

Algoritmo 1 Constructivo**Entrada:** Instancia**Salida:** Solución factible X

-
- 1: $X \leftarrow \emptyset$
 - 2: $t_{ij} \leftarrow$ Tiempo de traslado de la instalación i a la instalación j
 - 3: Sea Ta_k el tiempo acumulado utilizando el vehículo k
 - 4: $Ta_k \leftarrow 0$
 - 5: Sea N el conjunto de instalaciones no visitadas
 - 6: Sea i_0 el depósito inicial
 - 7: Sea i_f el depósito final
 - 8: **Para** cada vehículo k **hacer**
 - 9: **Mientras** $Ta_k < L_k$ y $|N| \neq 0$ **hacer**
 - 10: Vecindario $\leftarrow N$
 - 11: Buscar \leftarrow True
 - 12: **Mientras** Buscar y $|\text{Vecindario}| \neq 0$ **hacer**
 - 13: Elegir j^* tal que $j^* = \text{argmax}_{j \in \text{Vecindario}} f(j)$
 - 14: Si hay empate de j^* , tomar la de menor distancia a la última instalación visitada
 - 15: **Si** $t_{i_0 j^*} + t_{j^* i_f} \leq L_k$ **entonces**
 - 16: $X \leftarrow X \cup \{j^*\}$
 - 17: $Ta_k \leftarrow Ta_k + t_{i_0 j^*}$
 - 18: $N \leftarrow N \setminus \{j^*\}$
 - 19: $i_0 \leftarrow j^*$
 - 20: Actualizar $f(j)$ eliminando los clientes que ya han sido cubiertos por la instalación j^*
 - 21: Buscar \leftarrow False
 - 22: **Si no**
 - 23: Vecindario $\leftarrow \text{Vecindario} \setminus \{j^*\}$
 - 24: **Fin Si**
 - 25: **Fin Mientras**
 - 26: **Fin Mientras**
 - 27: **Fin Para**
-

En la Figura 4.2 se ilustra el proceso iterativo que sigue el algoritmo constructivo para cada uno de los vehículos. De manera específica, se tienen dos vehículos para los cuales se ingresan las instalaciones con mayor cobertura, por ejemplo, en la Figura 4.2 (a) se tiene que $f(1) = |C_1| = 5$ y $f(3) = |C_3| = 5$. Ambas cubren el mismo número de clientes, por lo que, en la Figura 4.2 (b) al haber un empate, se elige la de menor distancia, es decir, la instalación 1. Posteriormente, en la Figura 4.2 (c) se actualizan los radios de cobertura de las instalaciones visitadas en cada iteración y se muestran en color verde los clientes que ya han sido cubiertos. De esta manera, se tiene que $f(2) = |C_2| = 2$, $f(3) = |C_3| = 5$, $f(4) = |C_4| = 4$ y $f(5) = |C_5| = 2$, por lo que la siguiente instalación con mayor cobertura a verificar es la instalación 3. En la Figura 4.2 (d) se repite el proceso anterior mientras no se exceda el tiempo límite de la ruta y se realiza el mismo procedimiento de construcción para el segundo vehículo.

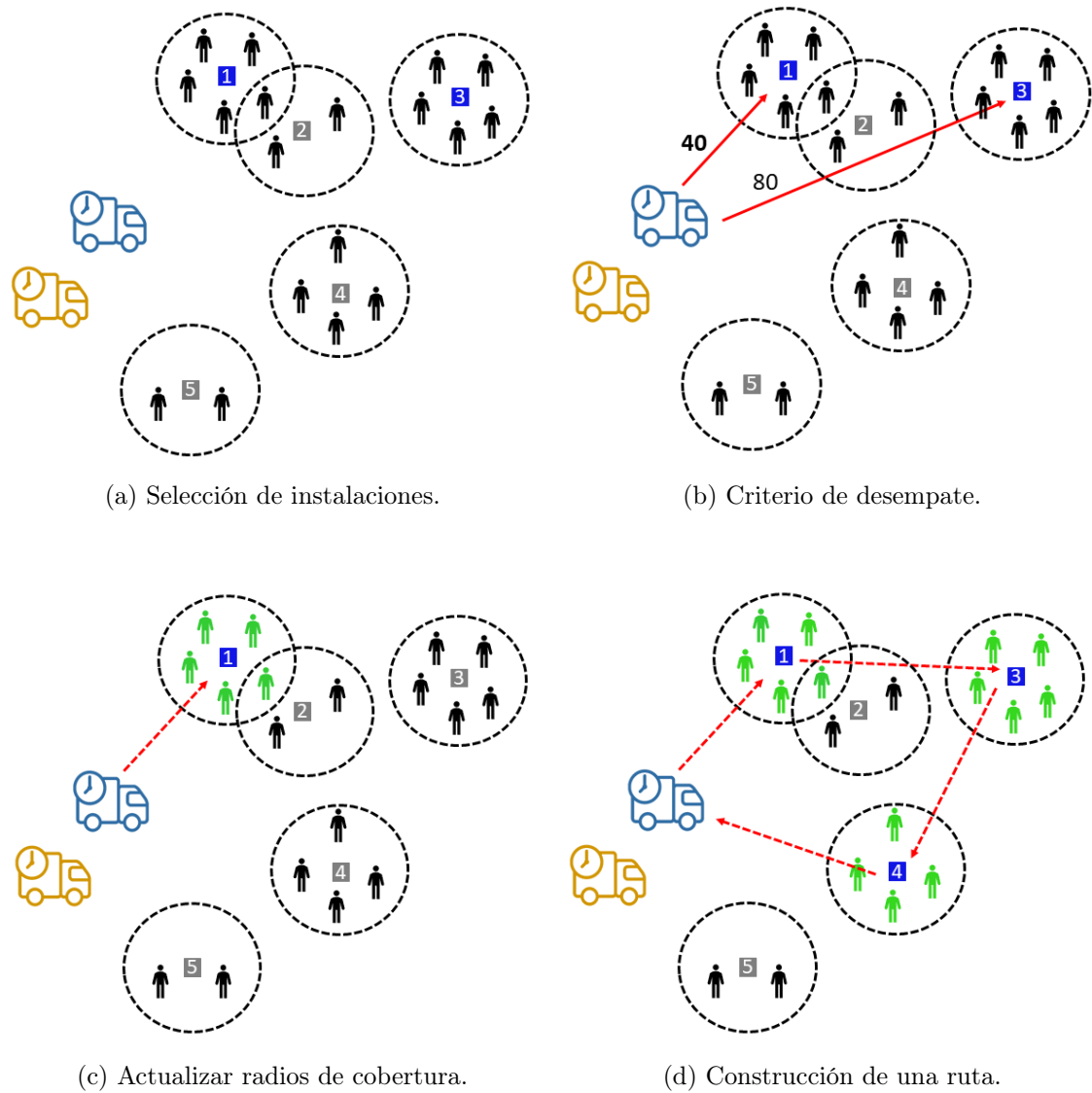


Figura 4.2: Proceso del algoritmo constructivo.

4.1.2 OPERADORES DE DESTRUCCIÓN

Antes de comenzar a explicar en que consiste cada uno de los operadores, se hace énfasis en algunas de las notaciones que se emplean en las métricas utilizadas, tomando como ejemplo la solución representada en la Figura 4.3.

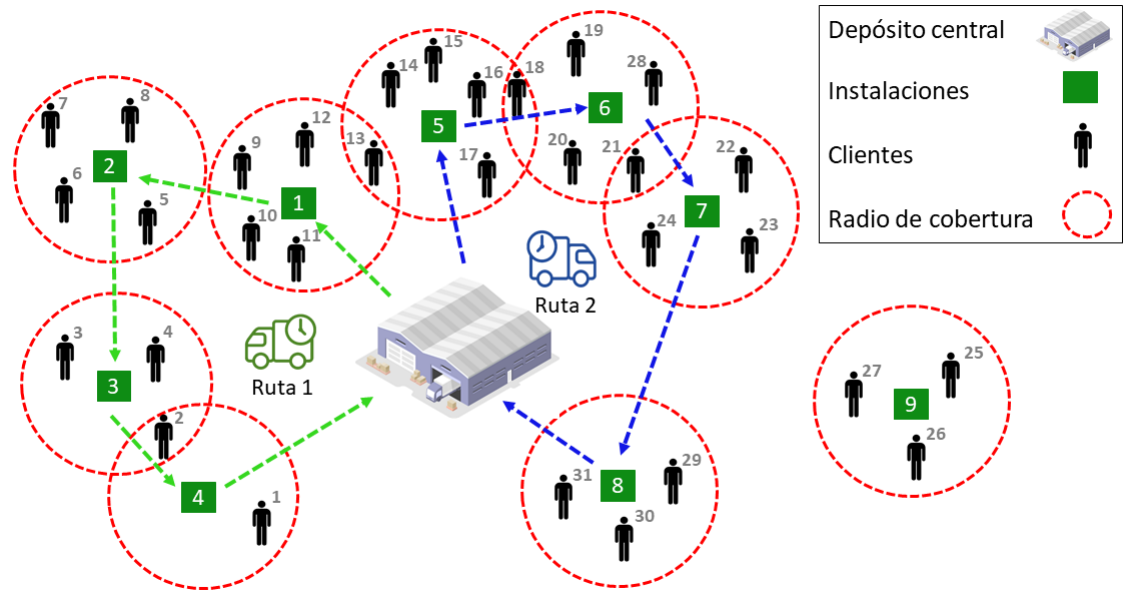


Figura 4.3: Ejemplo ilustrativo con 9 instalaciones, 31 clientes y 2 vehículos.

- C_{o_j} : Conjunto de clientes que son cubiertos por la instalación j cuando aún no se han insertado instalaciones a las rutas (cobertura original). Por ejemplo, en la Figura 4.3 se tiene que $C_{o_1} = \{9, 10, 11, 12, 13\}$.
- C_j : Como se explica en la Sección 4.1.1, representa la cobertura actualizada de la instalación j al momento de ser insertada a la ruta, de modo que cuando aún no se insertan instalaciones a las rutas, este conjunto es igual al conjunto C_{o_j} . De esta forma se tiene que $C_5 = \{14, 15, 16, 17, 18\}$, donde el cliente 13 a pesar de ser cubierto también por la instalación 5, este ya ha sido cubierto por la instalación 1 que fue visitada por el vehículo 1, de tal forma que $|C_5| = 5$.
- C_{u_j} : Conjunto de clientes que representan la cobertura única de la instalación j , es decir, el conjunto de clientes que cubre la instalación j menos la intersección con cada uno de los conjuntos de clientes que cubren el resto de las instalaciones visitadas (considerando coberturas originales). Dicho de otra manera, es el conjunto de clientes cubierto únicamente por la instalación j y que ninguna otra instalación cubre. De la Figura 4.3, se tiene que los clientes 18 y 21 que cubre la instalación 6, son cubiertos también por la instalación

5 y 7, respectivamente. De esta manera se tiene que la cobertura única de la instalación 6 esta dada por $Cu_6 = \{19, 20, 28\}$.

Para la solución del problema en estudio se propusieron un total de 11 operadores de destrucción, los cuales fueron diseñados tomando en cuenta características esenciales, como la cobertura y los tiempos de traslado de visitar alguna instalación. En general, tres de los operadores implementados son aleatorios, de manera que solo se enfocan en eliminar al azar instalaciones visitadas, mientras que los restantes eliminan las instalaciones visitadas siguiendo una métrica establecida.

1. Eliminar aleatoriamente instalaciones. Se realiza una destrucción moderada seleccionando al azar el 25 % y 50 % de las instalaciones visitadas y una destrucción masiva seleccionando al azar el 75 % de las instalaciones con la finalidad de volver a reconstruir la mayor parte de la solución con una heurística diferente. Esta destrucción masiva podrá ayudar a aquellas instancias para las cuales no se alcanzaron buenos resultados con la heurística constructiva.
2. Eliminar instalaciones con menor cobertura por tiempo. Este destructor asigna un coeficiente a cada instalación visitada j , dado por $\alpha_j = \frac{|C_j|}{t_{ij} + t_{jk}}$, donde i y k son instalaciones que fueron visitadas antes y después de j , respectivamente. Se elimina el 50 % de las instalaciones con menor cociente.
3. Eliminar instalaciones con menor cobertura. Con la finalidad de descartar aquellas instalaciones que cubren menos clientes a comparación del resto, se elimina el 50 % de las instalaciones visitadas con menor cobertura actualizada.
4. Eliminar instalaciones con menor cobertura única. Dado que más de una instalación puede visitar a un mismo cliente, se define $\beta_j = \frac{|Cu_j|}{|Co_j|}$ para cada instalación visitada j , como el porcentaje de clientes únicos. Se elimina el 50 % de las instalaciones visitadas con menor cociente.
5. Eliminar la instalación con mayor tiempo de visita. Para cada ruta, se determina cuál es la instalación j que tiene mayor tiempo de visita, es decir, el tiempo

que toma llegar de la instalación anterior a j . En la Figura 4.4 se muestra el proceso que sigue este operador en una ruta.

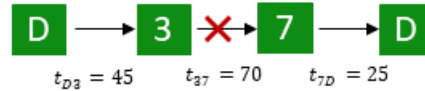


Figura 4.4: Ejemplo de ruta con los respectivos tiempos de visita.

6. Eliminar las instalaciones con menor cobertura actualizada. Se calcula el cociente de cada una de las instalaciones visitadas, dado por $\delta_j = \frac{|C_j|}{|C_{o_j}|}$. Se elimina el 50 % de las instalaciones visitadas con menor cociente.
7. Eliminar instalaciones . Por ruta, se eliminan las instalaciones cuya cobertura actualizada se encuentre dentro del intervalo $[|C_{min}|, |C_{min}| + 0.4(|C_{max}| - |C_{min}|)]$.
8. Eliminar las k instalaciones con peor cobertura. Para cada ruta, se eliminan las dos instalaciones con menor cobertura actualizada $|C_j|$.
9. Eliminar la localidad con mayor cobertura. Para cada ruta, con la finalidad de hacer una pequeña perturbación, se elimina la instalación j que tiene mayor cobertura actualizada $|C_j|$.

4.1.3 OPERADORES DE REPARACIÓN

Para la implementación del ALNS se utilizaron 5 operadores de reparación. Las inserciones se realizan recorriendo secuencialmente las rutas para los primeros dos operadores, mientras que para los restantes se elige qué instalación, en que ruta y posición debe realizarse la inserción. Solo se consideran inserciones factibles. Los operadores de reparación toman como punto de partida la solución obtenida por algún operador de destrucción.

1. Inserción de la instalación con menor incremento en tiempo por ruta. Se recorren las instalaciones que aún no han sido visitadas y se verifica si es posible insertar al menos una en alguna ruta, de este modo se elige qué instalación insertar y en qué posición, considerando que dicha inserción tenga el menor incremento en tiempo entre todas aquellas posibles inserciones. El incremento esta dado por $\Delta_{k,i,j} = t_{ik} + t_{kj} - t_{ij}$, donde k es la instalación a insertar entre el arco (i, j) .
2. Inserción de la instalación con mayor incremento en cobertura por ruta. Para cada una de las instalaciones que aún no han sido visitadas por alguna ruta, se verifica si es posible insertarlas en alguna posición de la misma, considerando que dicha inserción sea la que maximice el valor de la función objetivo de la solución y si existe un empate entre inserciones, se elige la primera.
3. Mejor inserción con menor incremento en tiempo. Para las instalaciones que aún no han sido visitadas, se verifica qué instalación insertar en qué ruta y posición, de tal manera que dicha inserción tenga el menor incremento en tiempo entre todas aquellas posibles inserciones.
4. Mejor inserción con mayor incremento en cobertura por tiempo. Para las instalaciones que aún no han sido visitadas, se verifica qué instalación insertar en qué ruta y posición, de tal manera que dicha inserción tenga el mayor incremento en cobertura por unidad de tiempo, dado por $\theta_{k,i,j} = \frac{|C_k|}{t_{ik} + t_{kj} - t_{ij}}$, entre todas aquellas posibles inserciones.
5. Inserción de la instalación con mayor relación de cobertura. Para cada instalación no visitada, se calcula un coeficiente dado por la suma ponderada $\tau_j = 0.3 (|I_j|) + 0.7 (|Cu_j|)$. Cabe recordar que, de manera similar como se explica en la Sección 4.1.2, Cu_j es el conjunto de clientes que representan la cobertura única de la instalación j , es decir, el conjunto de clientes que cubre la instalación j menos la intersección con cada uno de los conjuntos de clientes que cubren el resto de las instalaciones que no han sido visitadas (considerando cobertu-

ras originales) y de esta manera se tiene la relación $|I_j| = |Co_j| - |Cu_j|$. Se identifican aquellas instalaciones tal que su coeficiente esté dentro del intervalo $[\tau_{max} - 0.4(\tau_{max} - \tau_{min}), \tau_{max}]$ y se recorren aleatoriamente dichas instalaciones verificando en que ruta y posición realizar la inserción.

4.1.4 EVALUACIÓN DE LAS SOLUCIONES

Dentro de la descripción general del ALNS mostrada en la Figura 4.1, el paso 7 consiste en actualizar los pesos de los operadores en base a la calidad de las soluciones. La Tabla 4.1 muestra los pesos asignados a los operadores en base a su desempeño, asignando una puntuación o peso de cuatro unidades si la solución nueva es mejor que la solución actual, es decir, que el valor de la función objetivo de solución nueva es estrictamente mayor al valor de la función objetivo de solución actual. Para los casos en que el valor obtenido por la solución nueva sea igual al valor de la solución actual, solo se les incrementa una unidad a los pesos de los operadores utilizados, mientras que si el valor de la función objetivo de solución nueva es menor que el de solución actual, a los pesos de los operadores que hayan sido utilizados no se les suma unidades.

Tabla 4.1: Pesos asignados a los operadores

Solución	Peso
Mejor	+4
Alcance (igual)	+1
Peor	+0

CAPÍTULO 5

EXPERIMENTACIÓN Y RESULTADOS

En este capítulo se presenta el análisis de los resultados de la experimentación computacional realizada para la validación del modelo y la evaluación del desempeño del método de solución propuesto, ya que debido a la naturaleza heurística del mismo, no se tiene la garantía de obtener soluciones óptimas para el problema de estudio.

Por otra parte, cuando se implementa un algoritmo heurístico, es de gran importancia la elección de los parámetros que se utilizarán, ya que todo procedimiento metaheurístico es sensible a los valores de sus parámetros y de ello depende su desempeño. En particular, en un ALNS se debe elegir el período de actualización de la probabilidad de elección para cada operador de destrucción y reparación y el criterio de paro, que puede ser por tiempo de ejecución, número de iteraciones, número de iteraciones consecutivas sin mejora, entre otros.

Para hacer la elección de estos parámetros, se realizó experimentación computacional previa sobre un conjunto de instancias, así como pruebas de hipótesis que den soporte estadístico a dicha elección. Dichas pruebas, que fueron ejecutadas en Python 3.7, también se presentan en este capítulo. Además, se incluye el análisis realizado al evaluar el desempeño de los operadores de destrucción y reparación propuestos, así como el análisis de los resultados obtenidos utilizando solución inicial para CPLEX, tomando como punto de partida las soluciones obtenidas por el ALNS.

Todos los resultados presentados en este capítulo se redondean a dos cifras significativas.

5.1 DESCRIPCIÓN DE LAS INSTANCIAS

Para la evaluación del algoritmo presentado se realizó experimentación sobre un conjunto de 270 instancias proporcionadas por Amiri y Salari (2019), sin embargo, se descartaron dos instancias repetidas, una perteneciente al conjunto de instancias pequeñas y la otra del conjunto de instancias medianas, quedando un total de 268 instancias. Las características de cada conjunto de instancias se especifica en la Tabla 5.1, donde el conjunto de nodos está compuesto por el depósito, instalaciones y clientes. Por ejemplo, para el conjunto de instancias de 52 nodos que tienen 15, 20 y 25 instalaciones con 36, 31 y 26 clientes, respectivamente, donde el número de clientes se calcula como $|W| = |V| - |T| - 1$, recordando que, V es el conjunto de nodos, T el conjunto de instalaciones y W el conjunto de clientes. De esta manera, para este conjunto se tienen 8 instancias que consideran 2 vehículos y 9 instancias para 3 y 4 vehículos, dando un total de 26 instancias con esta cantidad de nodos.

Tabla 5.1: Descripción de las características de los conjuntos de instancias de prueba.

Instancias	Nodos	Instalaciones	Vehículos			Total
			2	3	4	
Pequeñas	52	15, 20 y 25	8	9	9	80
	76	22, 30 y 37	9	9	9	
	100	29, 39 y 49	9	9	9	
Medianas	150	44, 59 y 74	9	9	9	53
	200	59, 79 y 99	9	9	8	
Grandes	318	95, 127 y 158	9	9	9	135
	417	125, 166 y 208	9	9	9	
	575	172, 229 y 287	9	9	9	
	657	197, 262 y 328	9	9	9	
	724	217, 289 y 361	9	9	9	

En las Figuras 5.1, 5.2 y 5.3 se visualiza la dispersión geográfica del depósito, instalaciones y clientes de una instancia con 100, 200 y 724 nodos, respectivamente.

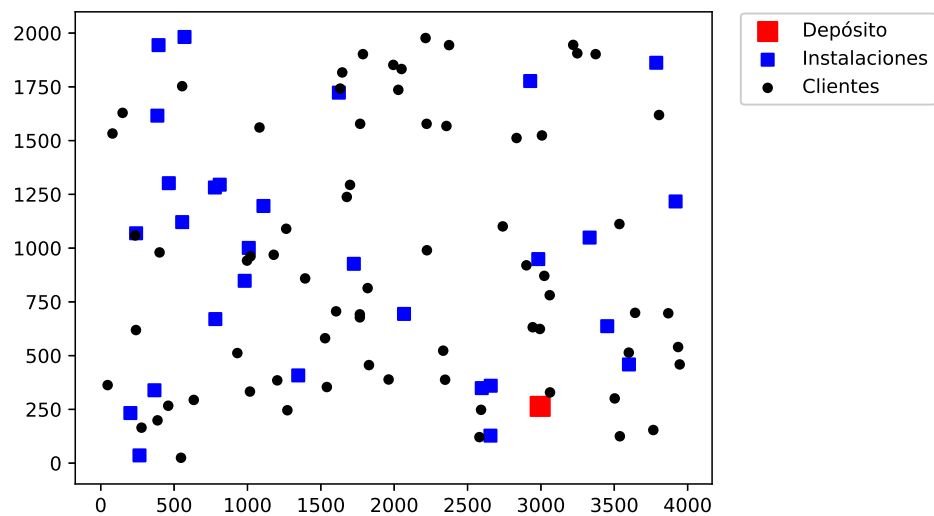


Figura 5.1: Ejemplo de topografía de una instancia con 29 instalaciones y 70 clientes.

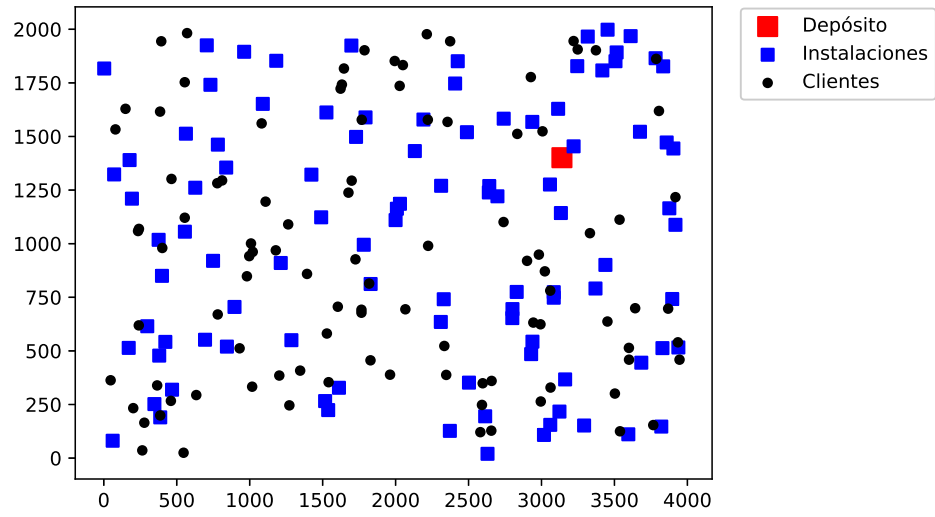


Figura 5.2: Ejemplo de topografía de una instancia con 99 instalaciones y 100 clientes.

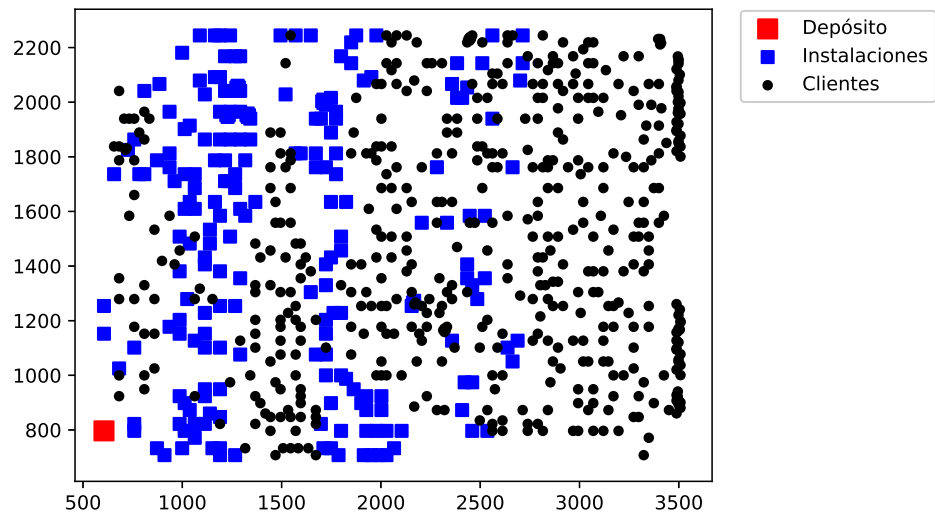


Figura 5.3: Ejemplo de topografía de una instancia con 217 instalaciones y 506 clientes.

5.2 ENTORNO EXPERIMENTAL

La validación del modelo se implementó en el lenguaje C++ y las instancias de prueba se resolvieron con el optimizador CPLEX 12.9. El algoritmo ALNS también se codificó en C++. La experimentación correspondiente se ejecutó en una Workstation HP Z620 con procesador Intel Xeon(R) CPU E5-2620 v2 a 2.10 GHz, con memoria RAM de 64 GB, bajo el sistema operativo Ubuntu 20.04 LTS.

5.3 PARÁMETROS UTILIZADOS EN EL ALNS

Como se mencionó anteriormente, para el uso de un esquema ALNS es necesario establecer un criterio de paro del algoritmo y el período de actualización de la probabilidad de elección para cada uno de los operadores de destrucción y reparación, por lo que se realiza experimentación para calibrar dichos parámetros.

Mediante experimentación preliminar en instancias con mayor número de instalaciones y clientes, se determinó un criterio de paro del algoritmo de 1000 iteraciones; sin embargo, para instancias de menor cardinalidad el número de iteraciones requerido por el método para converger a un óptimo local es menor. Por otro lado, también se determinó un período de 30 y 40 iteraciones para la actualización de las probabilidades de elección de los operadores en algunas instancias del conjunto de prueba. Observamos que la calidad de la solución parecía depender de esta variación en las instancias de prueba; no obstante, se tiene que tomar en cuenta que si se eligen períodos de pocas iteraciones no hay oportunidad de efectuar suficientes combinaciones de parejas destructor-reparador y a mayor número tanto de iteraciones para el criterio de paro como períodos de actualización, el tiempo incrementa. Por tal razón, para fijar de manera adecuada estos parámetros, se llevó a cabo experimentación computacional usando las 268 instancias de prueba.

Para la calibración de los parámetros criterio de paro y período de actualiza-

ción de las probabilidades, se realizaron tres diseños factoriales con dos factores para las instancias pequeñas, medianas y grandes, con el objetivo de analizar como la variabilidad de estos factores afecta la calidad de las soluciones y el tiempo computacional requerido por el ALNS. El primer factor es el criterio de paro, consideramos 500, 1000, 1500, 2000 y 2500 iteraciones; el segundo factor es el período utilizado para la actualización de las probabilidades, en el cual se consideraron periodos de 50 y 100 iteraciones. Para cada conjunto de instancias (pequeñas, medianas y grandes), con cada configuración, se realizaron diez réplicas y se calculó el número de veces que se alcanzó o superó el mejor valor de la función objetivo encontrado por CPLEX (ver Sección 5.4) para cada una de las instancias del conjunto, el tiempo de ejecución y el máximo $\text{Gap}_{A,C}$ (%) por réplica, donde el $\text{Gap}_{A,C}$ (%) se calcula de acuerdo a la ecuación (5.1). Debido a las inconsistencias que presentan algunas instancias y las tablas de resultados reportadas en el trabajo de Amiri y Salari (2019) se trabajó con los resultados obtenidos al validar el modelo con el optimizador CPLEX. Sin embargo, CPLEX no logra alcanzar una solución factible para varias instancias grandes, por lo que solo se han utilizado las instancias para las cuales CPLEX reporta una solución.

$$\text{Gap}_{A,C} (\%) = 100 \times \frac{Z_C - Z_A}{Z_C}, \quad (5.1)$$

Donde Z_C y Z_A representan el valor de la función objetivo obtenida por CPLEX y el ALNS, respectivamente.

En la Tabla 5.2 y Tabla 5.3 se muestra el número de iteraciones correspondiente a los distintos niveles del factor 1 y factor 2 utilizados en la experimentación, respectivamente.

Tabla 5.2: Niveles para el factor 1.

Niveles	Número de iteraciones para el criterio de paro
1	500
2	1000
3	1500
4	2000
5	2500

Tabla 5.3: Niveles para el factor 2.

Niveles	Período de actualizaciones
1	50 iteraciones
2	100 iteraciones

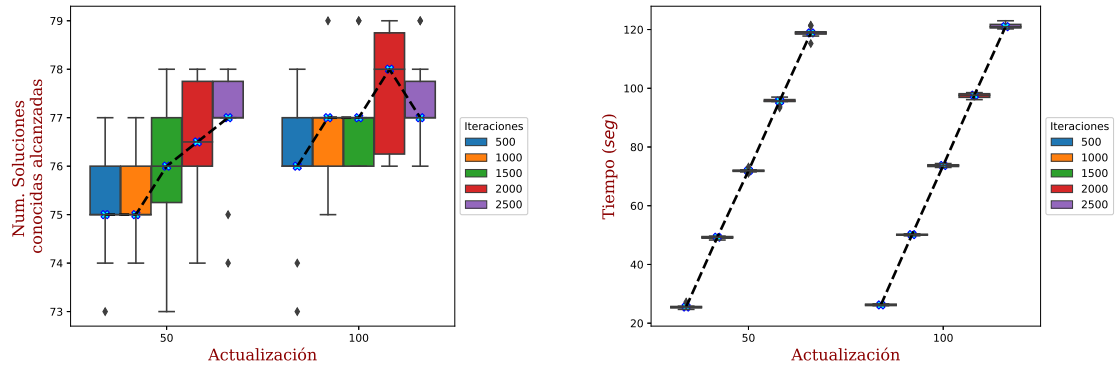
En las Figuras 5.4-5.6 se muestran los diagramas de caja y bigotes correspondientes a los diseños factoriales en interacción con el número de veces que el algoritmo ALNS alcanzó o superó la mejor solución, el tiempo de ejecución y el máximo $\text{Gap}_{A,C}$ (%), respectivamente. Para el número de veces que se alcanzó o superó la mejor solución utilizando períodos de actualización de 50 iteraciones en las instancias pequeñas y medianas se muestra un comportamiento creciente conforme aumenta el número de iteraciones para el criterio de paro, sin embargo, para los períodos de actualización de 100 iteraciones no sucede lo mismo. Por lo que no necesariamente para cada nivel en el período de actualización se tiene que, a mayor número de iteraciones en el criterio de paro, mejor desempeño del algoritmo.

Por otra parte, para cada nivel del factor 1 se observa un comportamiento creciente en el número de veces que se alcanza o supera la mejor solución variando los períodos de actualización en las instancias pequeñas, mientras que para las

instancias medianas en cada una de las iteraciones del criterio de paro no se tiene necesariamente que a mayor número de iteraciones en los períodos de actualización, mayor número de soluciones alcanzadas, esto se puede observar en la configuración con 1500 iteraciones para el criterio de paro con períodos de actualización de 50 y 100 iteraciones. Para las instancias grandes se alcanza o supera el mismo número de soluciones para las distintas configuraciones.

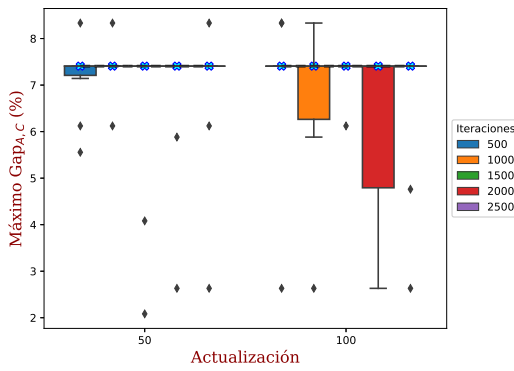
Aunado a este análisis, se tiene que para cada conjunto de instancias el máximo $\text{Gap}_{A,C}$ (%) es el mismo o presenta pequeñas variaciones en cada una de las posibles configuraciones.

Finalmente, el tiempo de ejecución para los tres conjuntos de instancias es creciente conforme aumenta el número de iteraciones en los períodos de actualización y criterio de paro.



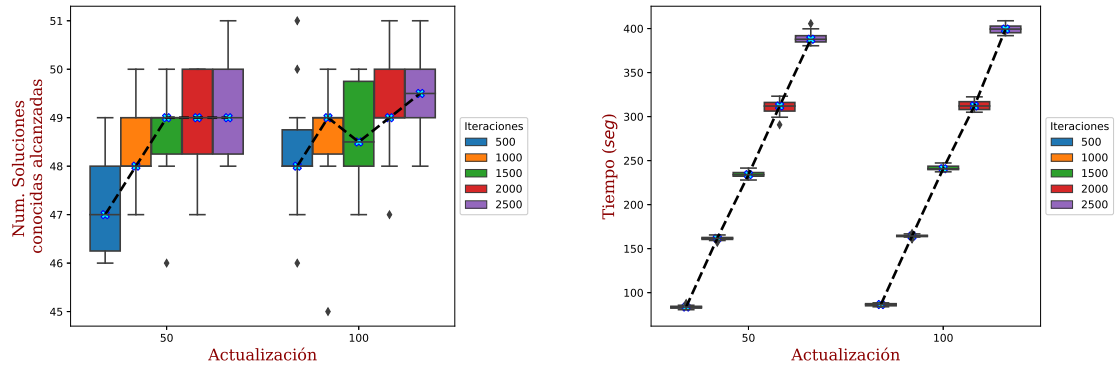
(a) Número de veces que se alcanzó o superó la mejor solución

(b) Tiempo de ejecución



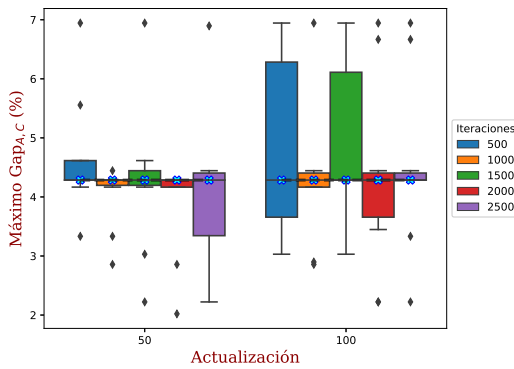
(c) Máximo Gap_{A,C} (%)

Figura 5.4: Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias pequeñas.



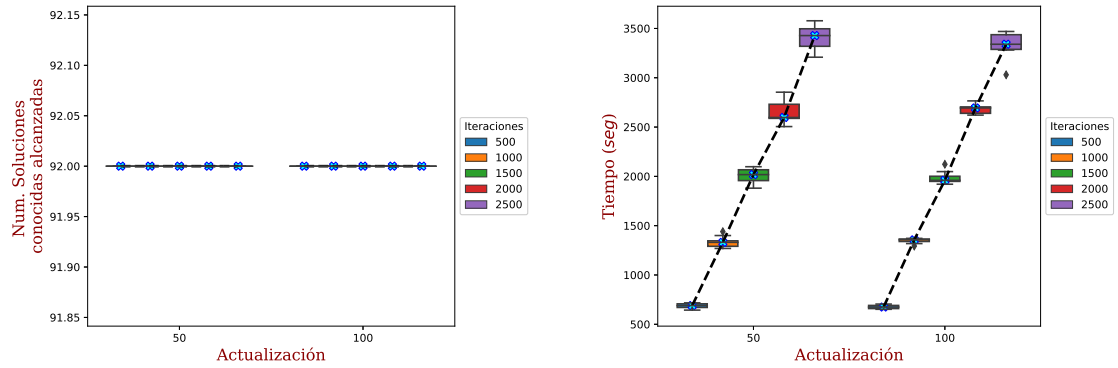
(a) Número de veces que se alcanzó o superó la mejor solución

(b) Tiempo de ejecución



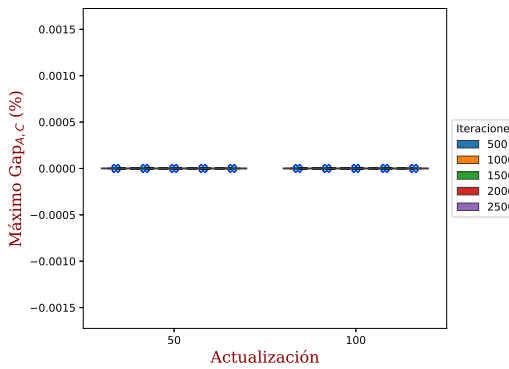
(c) Máximo Gap_{A,C} (%)

Figura 5.5: Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias medianas.



(a) Número de veces que se alcanzó o superó la mejor solución

(b) Tiempo de ejecución



(c) Máximo $\text{Gap}_{A,C}$ (%)

Figura 5.6: Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias grandes.

Dado que los valores obtenidos de las variables de respuesta resultantes de la experimentación, no cumplen con algunos de los supuestos de normalidad, homocedasticidad o independencia, se han utilizado las pruebas no paramétricas de Friedman y Kruskal Wallis (Jaykaran, 2010) para determinar si existe diferencia significativa en el efecto de los tratamientos sobre el número de veces que se alcanzó o superó la mejor solución, tiempo de ejecución y máximo $\text{Gap}_{A,C}$ (%), esto con un nivel de significancia de 0.05. Después, se procedió a realizar una prueba de Wilcoxon para determinar si existe diferencia significativa entre cada pareja de tratamientos,

de esta manera, se analizan cuáles medianas son iguales o diferentes y dependiendo del objetivo particular de cada una de las variables de respuesta y la relación entre ellas se selecciona la mejor configuración. Por ejemplo, para el número de veces que alcanzó o superó la mejor solución, se buscaría elegir aquella configuración que tuviera la mediana más alta, mientras que para el tiempo de ejecución y el máximo $\text{Gap}_{A,C}$ (%) la mediana más pequeña en dependencia del resultado de la prueba estadística y el análisis que se obtuvo para la primer variable de respuesta.

De este modo, para cada conjunto de instancias pequeñas, medianas o grandes se determinan las mejores configuraciones para el número de veces que se alcanzó o superó la mejor solución, tiempo de ejecución y máximo $\text{Gap}_{A,C}$ (%). Las Tablas 5.4-5.6 muestran las mejores configuraciones para las tres medidas de desempeño por conjunto de instancias.

Tabla 5.4: Mejores configuraciones obtenidas por pruebas no paramétricas para el conjunto de instancias pequeñas.

Medidas de desempeño			
Factores	Número de veces que se alcanzó o superó la mejor solución	Tiempo de ejecución	Máximo $\text{Gap}_{A,C}$ (%)
Iteración	2000	2000	2000
Actualización	100	100	100

Tabla 5.5: Mejores configuraciones obtenidas por pruebas no paramétricas para el conjunto de instancias medianas.

Medidas de desempeño			
Factores	Número de veces que se alcanzó o superó la mejor solución	Tiempo de ejecución	Máximo $\text{Gap}_{A,C}$ (%)
Iteración	2500	2000	2000
Actualización	100	100	100

Tabla 5.6: Mejores configuraciones obtenidas por pruebas no paramétricas para el conjunto de instancias grandes.

Medidas de desempeño			
Factores	Número de veces que se alcanzó o superó la mejor solución	Tiempo de ejecución	Máximo $\text{Gap}_{A,C}$ (%)
Iteración	2000	2000	2000
Actualización	100	100	100

De manera general, es de esperar que a medida que aumenta el número de nodos aumenta el tiempo de ejecución del algoritmo, por lo que los resultados de las pruebas estadísticas mostraron que hay diferencias significativas entre los efectos de las configuraciones para el criterio de paro y el período de actualización de probabilidades en el tiempo de ejecución del ALNS para los tres diferentes conjuntos. Sin embargo, la prueba estadística de Wilcoxon da como resultado que para el conjunto de instancias medianas la configuración de 2500 iteraciones para el criterio de paro con períodos de actualización de 50 y 100 iteraciones son estadísticamente iguales, mientras que en el conjunto de instancias grandes para cada nivel de iteración en el criterio de paro no se presentan diferencias significativas entre los niveles del período de actualización. Por otra parte, las distintas configuraciones no presentan diferencia significativa en el máximo $\text{Gap}_{A,C}$ (%) para las instancias pequeñas, medianas y grandes, mientras que para el alcance de la mejor solución existen diferencias significativas únicamente para las instancias pequeñas y medianas.

La Tabla 5.4 muestra las mejores configuraciones para las tres medidas de desempeño del conjunto de instancias pequeñas, resultando aquella con 2000 iteraciones ALNS y períodos de actualización de 100 iteraciones, en cada una de estas medidas.

En la Tabla 5.5 se muestra que la mejor configuración para el número de

veces que se alcanzó o superó la mejor solución es de 2500 iteraciones ALNS, no obstante, de acuerdo a la prueba de Wilcoxon no hay diferencias significativas entre esta configuración y la mejor configuración obtenida para el conjunto de instancias pequeñas. Además, si se realizan 2000 iteraciones ALNS se tiene una disminución de cerca de 90 segundos en el tiempo de ejecución. Por tal motivo, la mejor configuración para el conjunto de instancias medianas es 2000 iteraciones ALNS y con períodos de actualización de 100 iteraciones.

Finalmente, en la Tabla 5.6 se muestran las mejores configuraciones para las tres medidas de desempeño del conjunto de instancias grandes, cuyas elecciones se tomaron con base a las mejores configuraciones del conjunto de instancias pequeñas y medianas, esto para tener uniformidad y que estadísticamente no haya diferencias significativas en el efecto de los tratamientos sobre el número de veces que se alcanzó o superó la mejor solución y el máximo $\text{Gap}_{A,C}$ (%).

En conclusión, la mejor configuración para el algoritmo ALNS consta de un criterio de paro de 2000 iteraciones (factor 1), donde las probabilidades de elección de cada uno de los operadores (factor 2) son reevaluadas cada 100 iteraciones para los tres tipos de instancias.

5.4 ANÁLISIS DE LAS SOLUCIONES OBTENIDAS MEDIANTE CPLEX

Para llevar a cabo la validación del modelo, las instancias son resueltas mediante el optimizador CPLEX versión 12.8 usando un solo hilo. Son considerados dos criterios de paro para cada una de las instancias: el tiempo de cómputo (7200 s) o la calidad de la solución (Gap de 0.5 %).

Los resultados del modelo se presentan en la Tabla 5.7, donde la segunda, tercera y cuarta columna muestran el Gap (%) de optimalidad mínimo, promedio y

máximo, respectivamente. De la misma manera, se reportan los tiempos de ejecución (en segundos) mínimo, promedio y máximo, en la quinta, sexta y séptima columna, mientras que en la última columna se muestra el número de instancias que fueron resueltas a optimalidad dentro del tiempo límite para cada uno de los conjuntos de instancias. Cabe mencionar que para el conjunto de instancias grandes, solo se consideran en el análisis de los resultados (Gap (%) y tiempo de ejecución (en segundos)), aquellas instancias para las cuales CPLEX reportó solución dentro del tiempo límite dado.

Tabla 5.7: Resultados reportados por CPLEX para el modelo propuesto por Amiri y Salari (2019).

Instancias	Gap (%)			Tiempo (s)			Óptimos
	Min	Prom	Max	Min	Prom	Max	
Pequeñas	0.00	0.13	3.20	0.01	387.87	7199.71	77/80
Medianas	0.00	1.03	8.16	23.26	3216.25	7199.81	37/53
Grandes	0.00	0.11	6.40	69.63	2075.81	7199.58	89/135
Total							203/268

De la tabla anterior se puede notar que a medida que aumenta el número de nodos en la instancia, el tiempo de ejecución incrementa, esto puede verse reflejado en el conjunto de instancias pequeñas y medianas, donde el tiempo de ejecución promedio es de 387.87 segundos y 3216.25 segundos, respectivamente. Mientras que para las instancias grandes no sucede lo mismo, ya que el tiempo de ejecución promedio es de 2075.81 segundos.

Aunado a esto, los resultados en la calidad de las soluciones muestran que el desempeño del optimizador es bueno en las instancias pequeñas, ya que se encuentran los óptimos para casi todas las instancias de este conjunto, obteniendo un gap máximo de 3.20%. Dicho de otra manera, CPLEX encontró optimalidad para el 96.25% de las instancias pequeñas, mientras que un 69.81% para las instancias

medianas. En el caso de las instancias grandes, a pesar de que se tenga un gap promedio del 0.11 % y un gap máximo de 6.40 %, CPLEX encontró optimalidad para el 65.93 %, sin embargo, para el 31.85 % (43 instancias), este no fue capaz de encontrar una solución entera al problema dentro del tiempo límite y, en consecuencia, no es posible calcular un gap.

En total, el optimizador fue capaz de encontrar 203 óptimos para el modelo, esto es, aproximadamente un 76 % del total de instancias.

5.5 DESEMPEÑO DEL ALGORITMO DE SOLUCIÓN

El algoritmo propuesto se codificó en el lenguaje de programación C++. El desempeño del mismo es evaluado con el grupo de instancias descritas previamente, bajo el mismo entorno experimental descrito en la Sección 5.2.

En la Tabla 5.8 se reportan por separado, los tiempos totales de ejecución, en segundos, requeridos por CPLEX (incluyendo los tiempos de las instancias en las cuales este no encontró solución entera en el tiempo dado) y por el algoritmo ALNS para obtener las soluciones de cada uno de los conjuntos de instancias. De esta manera, se tiene un ahorro en tiempos de aproximadamente 99.65 %, 99.81 % y 98.80 % en las instancias pequeñas, medianas y grandes, respectivamente. Mientras que en total se tiene un ahorro del 99.09 %. Esto es de gran importancia debido a las diversas aplicaciones que puede tener este problema, donde los recursos, tales como el tiempo, son limitados.

Tabla 5.8: Tiempos de ejecución totales, en segundos, requeridos por el algoritmo ALNS y el optimizador CPLEX.

Instancias	CPLEX	ALNS
Pequeñas	31029.75	108.69
Medianas	170461.06	320.52
Grandes	498968.63	5968.34
Totales	700459.44	6397.55

El gap, en porcentaje, que existe entre la mejor solución reportada por CPLEX y la encontrada con el ALNS, es decir, $\text{Gap}_{A,C}$ (%), se mide de acuerdo a la ecuación (5.1).

De esta forma, dado que la función objetivo del modelo es maximizar, un gap negativo indica que el valor de la función objetivo obtenido por la metaheurística es más alto que el de la mejor solución encontrada por CPLEX en 7200 segundos, en tal caso la solución reportada por el optimizador no sería óptima, sino entera factible.

Los resultados obtenidos por el modelo TCMCRP se resumen en la Tabla 5.9. El $\text{Gap}_{A,C}$ (%) mínimo, promedio y máximo, se muestra en la segunda, tercera y cuarta columna, respectivamente. De la misma manera, se reportan los tiempos de ejecución (en segundos) mínimo, promedio y máximo, en la quinta, sexta y séptima columna, mientras que en las dos últimas columnas se muestran los óptimos obtenidos con el optimizador en cada uno de los conjuntos de instancias y los óptimos alcanzados con el ALNS en comparación con los de CPLEX, respectivamente.

Para un análisis más detallado se han incluido en el Apéndice A.1 los resultados de la experimentación realizada con el optimizador CPLEX y la metaheurística tipo ALNS, de cada una de las 268 instancias proporcionadas por Amiri y Salari (2019).

Tabla 5.9: Resultados del ALNS versus CPLEX para el modelo propuesto por Amiri y Salari (2019).

Instancias	Gap _{A,C} (%)			Tiempo (s)			Óptimos	
	Min	Prom	Max	Min	Prom	Max	CPLEX	ALNS/CPLEX
Pequeñas	0.00	0.30	7.41	0.60	1.36	3.19	77/80	73/77
Medianas	-7.14	-0.38	4.29	2.53	6.05	12.58	37/53	36/37
Grandes	-6.40	-0.11	0.00	4.64	44.21	162.90	89/135	89/89
Total								198/203

De la tabla anterior se puede notar que, en promedio, los resultados del algoritmo para las instancias pequeñas se muestran competitivos con los resultados obtenidos por CPLEX. Sin embargo, en los tiempos de ejecución de este conjunto se tiene un ahorro considerable, el ALNS utiliza aproximadamente 0.35% del tiempo total que el optimizador emplea para obtener las soluciones del conjunto de instancias pequeñas.

Por otra parte, el mejor desempeño de la metaheurística se alcanza en el conjunto de instancias grandes, para las cuales el algoritmo ALNS fue capaz de encontrar todas las soluciones óptimas conocidas e inclusive encontró solución para las 43 instancias en las que el optimizador no encontró ninguna solución entera al alcanzar el tiempo límite, además de mejorar una solución entera factible reportada por CPLEX en 6.40% y una mejora promedio del 0.11%. Algo semejante ocurre con el conjunto de instancias medianas, donde a pesar de que la metaheurística no encontró una de las soluciones óptimas, fue capaz de mejorar 10 soluciones enteras factibles encontradas por CPLEX, siendo la máxima mejora del 7.14% y una mejora promedio del 0.38%.

Ahora bien, cabe destacar que de un total de 203 óptimos reportados por CPLEX, el método propuesto logró alcanzar 198, es decir, aproximadamente 2% menos que el optimizador. Además, logró mejorar los valores de la función objetivo de 56 instancias, alcanzando un máximo de 7.14% de mejora. En los tiempos de

ejecución, en un total de 700459.44 segundos (8 días) CPLEX pudo encontrar 77 soluciones óptimas, 3 factibles y en 43 instancias no encontró una solución entera dentro del tiempo límite, mientras que el ALNS ofrece soluciones para todas las instancias en tan solo el 0.91 % (poco más de una hora y media) del tiempo que el optimizador emplea para resolver las 268 instancias.

La peor solución encontrada por el ALNS se aleja un 7.41 % de lo reportado por el optimizador y corresponde a una instancia del conjunto de instancias pequeñas.

Por último, en la Tabla 5.10 se muestra el porcentaje de mejora $Imp_{V,A}$ (%) mínimo, promedio y máximo entre las soluciones obtenidas por el ALNS y el algoritmo constructivo. Para ello, los resultados se obtuvieron de acuerdo a la fórmula (5.2).

$$Imp_{V,A} (\%) = 100 \times \frac{Z_V - Z_A}{Z_A}, \quad (5.2)$$

Donde Z_V representa el valor de la función objetivo obtenida por el constructivo.

De esta manera, una mejora negativa indica que el valor de la función objetivo obtenido por el ALNS es mayor que el valor obtenido por el constructivo.

En función de lo planteado, se observa que, para el conjunto de instancias grandes se reporta un $Imp_{V,A}$ (%) mínimo, promedio y máximo del 0 %, lo cual indica que el ALNS no mejoró las soluciones obtenidas por el constructivo, no obstante para el conjunto de instancias pequeñas y medianas se tiene una máxima mejora del 23.91 % y 31 %, respectivamente.

Tabla 5.10: Porcentajes de mejora de las soluciones obtenidas por ALNS con respecto al constructivo.

Instancias	Imp_{V,A} (%)		
	Min	Prom	Max
Pequeñas	-23.91	-3.41	0.00
Medianas	-31.00	-9.97	0.00
Grandes	0.00	0.00	0.00

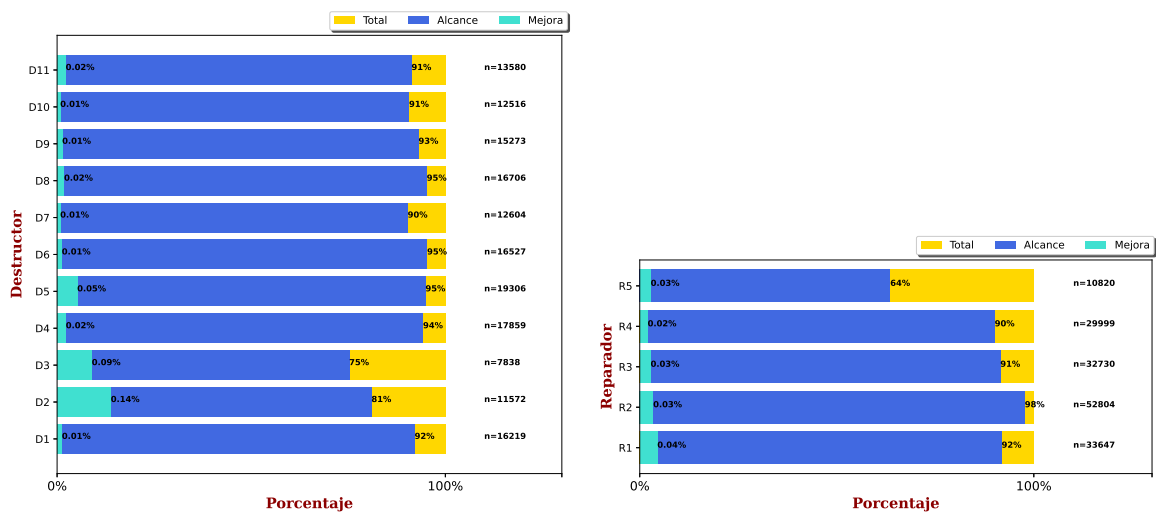
5.5.1 RENDIMIENTO DE LOS OPERADORES DE DESTRUCCIÓN Y REPARACIÓN

Para evaluar el rendimiento que tuvo cada uno de los operadores de destrucción y reparación implementados para el algoritmo ALNS, para cada uno de los conjuntos de instancias, se calculó el total de veces que fue utilizado cada operador y las veces que este alcanzó (igual) o mejoró (estrictamente mayor) el valor de la función objetivo de la solución actual. De esta manera, en las Figuras 5.7, 5.8 y 5.9 se presentan los porcentajes de mejora, alcance y uso total, en color amarillo, verde y azul, respectivamente. Además, el valor de n significa el total de veces que se usó dicho operador.

Evidentemente, los porcentajes de mejora reportados en los operadores de destrucción dependen del peso asignado a la calidad de la solución nueva, misma que se genera al aplicar un operador de destrucción y reparación en cada iteración del ALNS.

De la Figura 5.7 se resalta que los operadores que tuvieron mayor porcentaje de mejora en el conjunto de instancias pequeñas fueron el destructor 2, utilizado 11572 veces y el reparador 1 con 33647 veces, ambos con un porcentaje de mejora del 0.14 % y 0.04 % con un porcentaje de alcance del 81 % y 98 %, respectivamente. Así

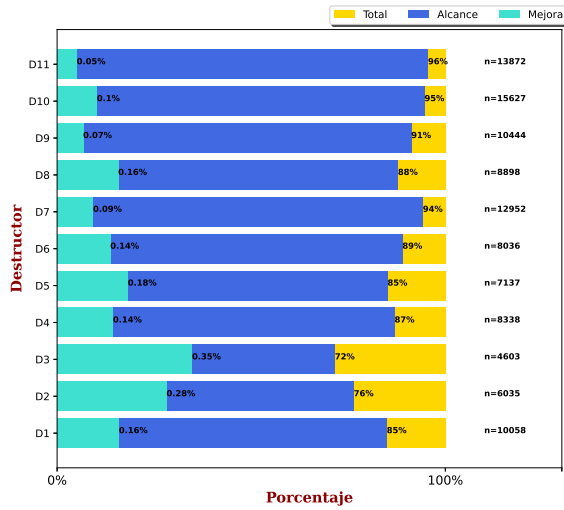
mismo, para las instancias medianas los operadores con mayor desempeño fueron el destructor 3 y reparador 2 con un porcentaje de mejora del 0.35 % y del 0.17 %, esto se puede observar en la Figura 5.8. Sin embargo, en la Figura 5.9 es evidente que para las instancias grandes no se reportaron mejoras de la solución actual, por ejemplo, de las 60314 veces que se utilizó el reparador 4, se tuvo un alcance del 99 %, esto quiere decir que, el 99 % de las veces, el valor objetivo de la solución resultante del ALNS fue igual al valor obtenido con la solución actual, que en este caso, dado que no hubo mejoras, corresponden a los valores obtenidos con el algoritmo constructivo, mientras que el 1 % restante, la solución tuvo un valor objetivo menor con respecto a la solución actual. De la misma manera, se realiza un análisis similar para el resto de los operadores.



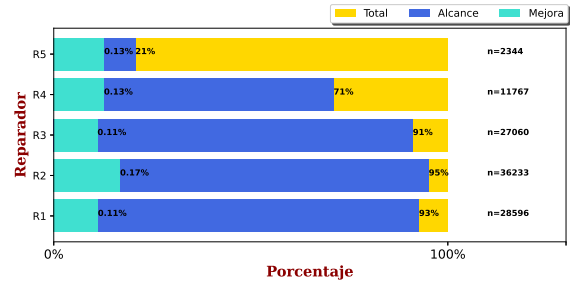
(a) Conjunto de destructores

(b) Conjunto de reparadores

Figura 5.7: Rendimiento individual de cada uno de los operadores en el conjunto de instancias pequeñas.

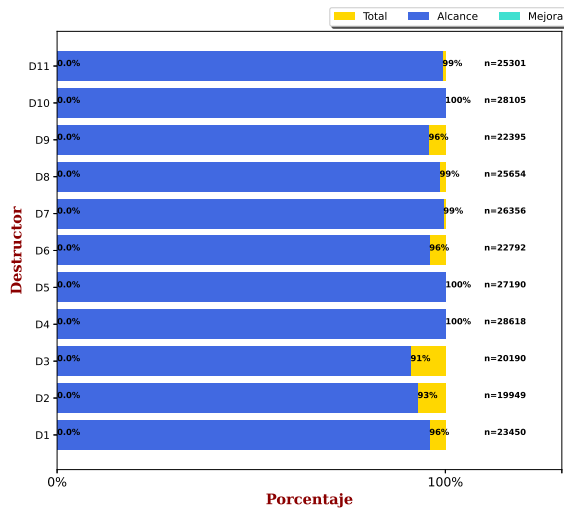


(a) Conjunto de destructores

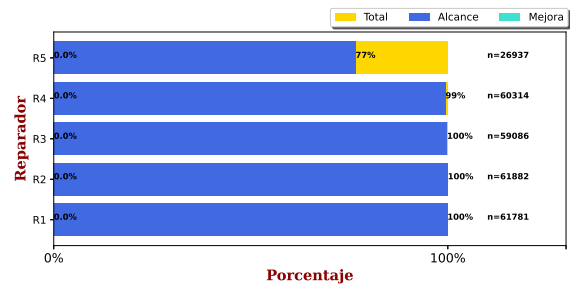


(b) Conjunto de reparadores

Figura 5.8: Rendimiento individual de cada uno de los operadores en el conjunto de instancias medianas.



(a) Conjunto de destructores



(b) Conjunto de reparadores

Figura 5.9: Rendimiento individual de cada uno de los operadores en el conjunto de instancias grandes.

Las Figuras 5.10, 5.11 y 5.12 muestran el rendimiento de las diferentes combinaciones de los operadores de destrucción y reparación. En el conjunto de instancias

pequeñas hay tres combinaciones de operadores que alcanzan el máximo porcentaje de mejora del 0.02 %, las cuales son el destructor 2 con reparador 1, destructor 3 con reparador 2 y destructor 2 con reparador 4, mientras que para el conjunto de instancias medianas el destructor 3 con el reparador 2 fue la mejor combinación con un porcentaje del 0.07 % de mejora. Para el conjunto de instancias grandes todas las combinaciones presentan una mejora de 0 %, por lo que no hay justificación para decir que una es mejor que otra. Sin embargo, hay combinaciones que tienen un porcentaje de alcance menor al 50 %, estas combinaciones son el destructor 3 con el reparador 5 y el destructor 2 con el reparador 5, estos con un porcentaje de alcance del 11 % y 35 %, respectivamente.

En general, el reparador 5 es el que obtiene el menor rendimiento junto con la combinación de destructor 3 con reparador 5.

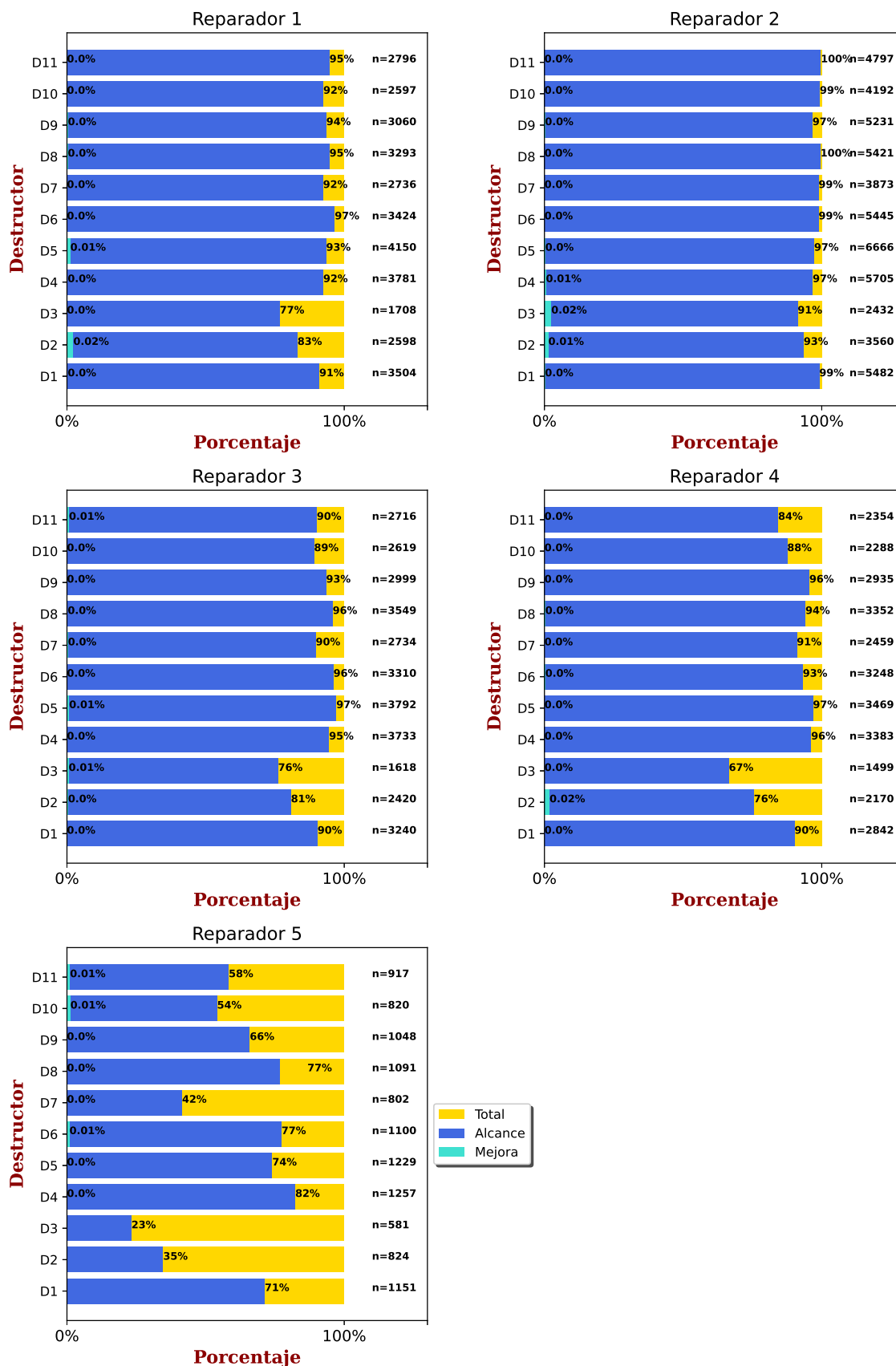


Figura 5.10: Rendimiento por parejas de cada uno de los operadores en el conjunto de instancias pequeñas.

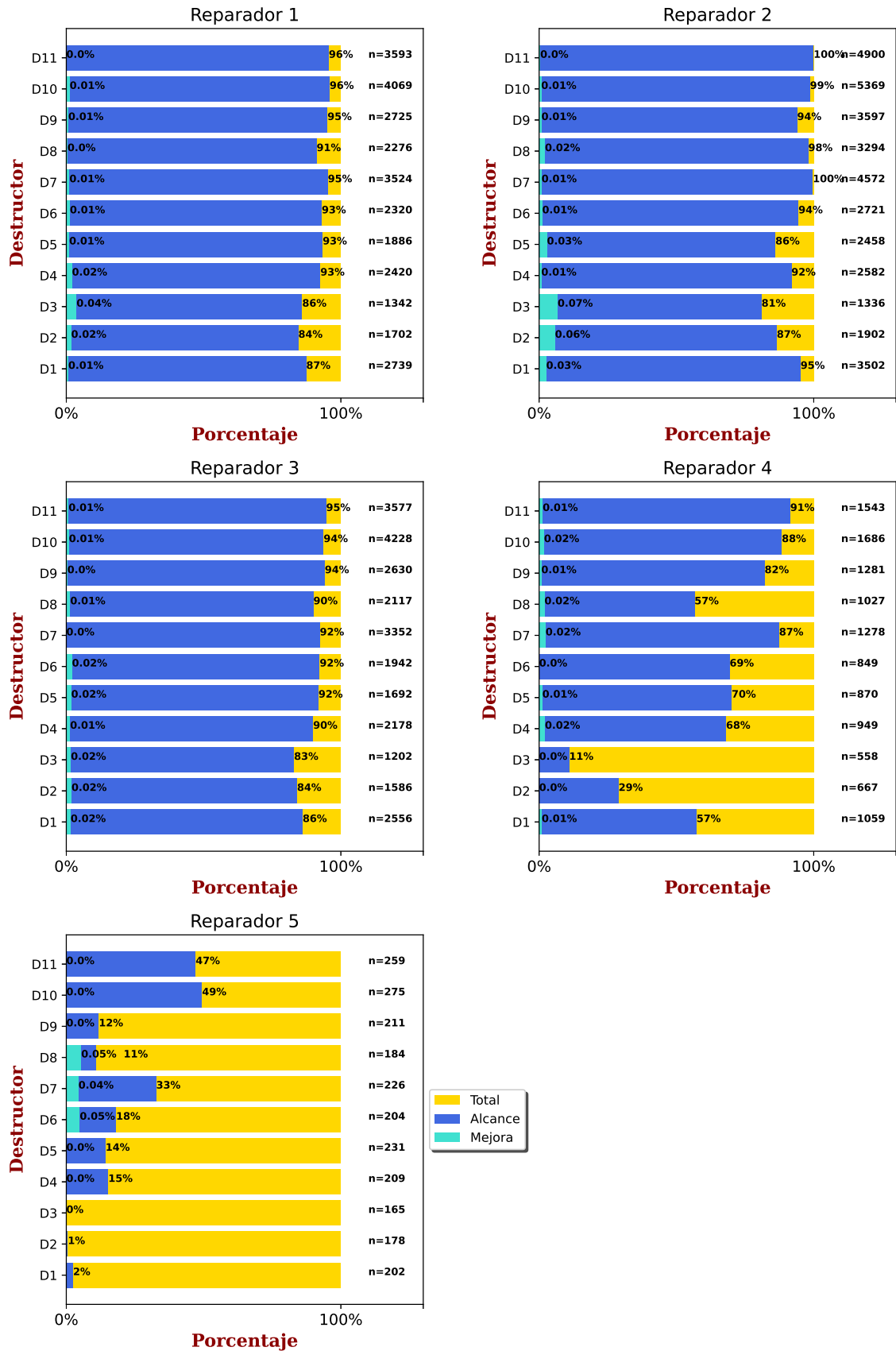


Figura 5.11: Rendimiento por parejas de cada uno de los operadores en el conjunto de instancias medianas.

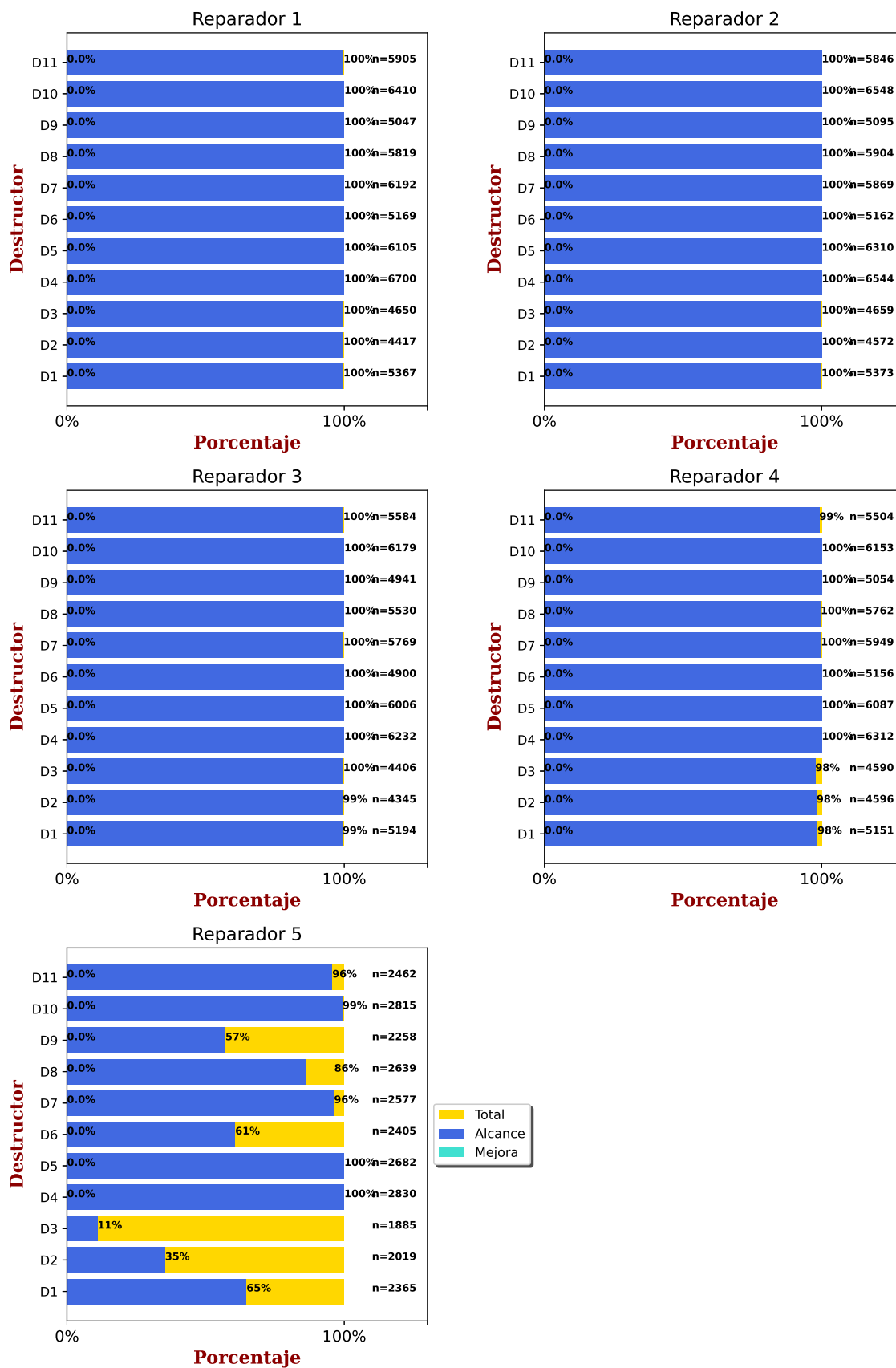


Figura 5.12: Rendimiento por parejas de cada uno de los operadores en el conjunto de instancias grandes.

5.6 ANÁLISIS DE LAS SOLUCIONES OBTENIDAS UTILIZANDO MIP START

Cuando se resuelve un problema de programación entera mixta (MIP), los inicios MIP (MIP Start, en inglés), también conocidos como inicios avanzados o inicios en caliente, permiten la opción de proporcionarle a CPLEX una solución inicial como ayuda para la optimización del MIP.

Se realizan pruebas computacionales haciendo uso de un inicio MIP, proporcionando a CPLEX como solución inicial, la solución obtenida por el algoritmo ALNS. Se consideran las mismas características descritas en la Sección 5.4, es decir, se utiliza CPLEX usando un solo hilo y se consideran los mismos dos criterios de paro para cada una de las instancias: el tiempo de cómputo (7200 s) y/o la calidad de la solución (Gap de 0.5%).

Cabe aclarar que para realizar esta experimentación, las soluciones iniciales dadas a CPLEX se obtuvieron ejecutando de nuevo el algoritmo ALNS con los parámetros ya establecidos, por lo que los resultados obtenidos por el ALNS en la Sección 5.5 no son los mismos.

Los resultados se presentan en la Tabla 5.11, donde la segunda, tercera y cuarta columna muestran el Gap (%) de optimalidad utilizando MIPStart mínimo, promedio y máximo, respectivamente. Análogamente, se reportan los tiempos de ejecución (en segundos) en la quinta, sexta y séptima columna, mientras que en las dos últimas columnas se muestran el número de instancias que fueron resueltas a optimalidad dentro del tiempo límite por CPLEX y CPLEX utilizando el parámetro MIPStart.

Tabla 5.11: Resultados de MIPStart versus CPLEX para el modelo propuesto por Amiri y Salari (2019).

Instancias	Gap (%)			Tiempo (s)			Óptimos	
	Min	Prom	Max	Min	Prom	Max	CPLEX	MIPStart
Pequeñas	0.00	0.06	2.38	0.02	380.95	7199.58	77/80	78/80
Medianas	0.00	0.45	5.89	0.38	1217.68	7199.65	37/53	45/53
Grandes	0.00	0.00	0.00	1.48	21.78	99.18	89/135	135/135
							Total	258/268

Los resultados en la calidad de las soluciones muestran que el desempeño MIPStart es bueno en las instancias grandes, ya que se encuentran todos los óptimos para las 135 instancias, en un tiempo promedio de ejecución de 21.78 segundos. Por otra parte, se encontró optimalidad para el 97.5% de las instancias pequeñas, lo que representa 1.25% más que las alcanzadas por CPLEX sin proporcionarle una solución inicial, con un máximo gap del 2.38% y un tiempo promedio de ejecución de 380.95 segundos. Finalmente, para el conjunto de instancias medianas se encontraron los óptimos para aproximadamente el 85% de las instancias, es decir, se alcanzó el 15% más que las alcanzadas sin MIPStart, obteniendo un máximo gap del 5.89% con un tiempo promedio de ejecución de 1217.68 segundos.

En total, haciendo uso de inicios MIP, el optimizador fue capaz de encontrar 258 óptimos usando el modelo, esto representa 96% del total de instancias y 10 soluciones enteras factibles, utilizando tan solo 14% (1 día) del tiempo que CPLEX por sí solo emplea para tratar de resolver las 268 instancias, de las cuales pudo encontrar 77 soluciones óptimas, 3 factibles y en 43 instancias no encontró una solución entera dentro del tiempo límite dado.

Por otra parte, cabe resaltar que de acuerdo a los resultados obtenidos por el ALNS en la Sección 5.5 se realiza nuevamente un análisis de estos resultados en comparación con lo que reporta CPLEX al usar el parámetro MIPStart. En la

Tabla 5.12 se muestra el $\text{Gap}_{A,M}$ (%) mínimo, promedio y máximo, en la segunda, tercera y cuarta columna, respectivamente, los cuales se obtuvieron de acuerdo con la siguiente ecuación (5.3).

$$\text{Gap}_{A,M} (\%) = 100 \times \frac{Z_A - Z_{MS}}{Z_{MS}}, \quad (5.3)$$

Donde Z_{MS} representa el valor de la función objetivo obtenida utilizando MIPStart. De esta manera, un gap negativo indica que el valor de la función objetivo obtenido por el ALNS es mayor (mejor) que el valor obtenido con MIPStart.

En las dos últimas columnas de la Tabla 5.12 se muestran los óptimos obtenidos con MIPStart y los alcanzados con el ALNS en comparación con los de MIPStart. Para las instancias pequeñas la metaheurística alcanzó 73 óptimos de los 78 reportados al utilizar MIPStart, 42 de los 45 óptimos reportados para las instancias medianas, mientras que para las instancias grandes el ALNS alcanzó todos los óptimos de los 135 reportados, dando un total de 250 óptimos que el ALNS fue capaz de alcanzar de los 258 reportados con MIPStart. En las Figuras 5.13-5.15 se muestran los valores de la función objetivo obtenidos por CPLEX, MIPStart y ALNS, para cada una de las instancias.

Tabla 5.12: Resultados del ALNS versus MIPStart para el modelo propuesto por Amiri y Salari (2019).

Instancias	Gap _{A,M} (%)			Óptimos	
	Min	Prom	Max	MIPStart	ALNS/MIPStart
Pequeñas	0.00	0.30	7.41	78/80	73/78
Medianas	0.00	0.16	4.29	45/53	42/45
Grandes	0.00	0.00	0.00	135/135	135/135
				Total	250/258

Por último, en la Tabla 5.13 se muestra el porcentaje de mejora $\text{Imp}_{M,C}$ (%)

mínimo, promedio y máximo de las soluciones obtenidas por CPLEX versus las soluciones obtenidas haciendo uso de MIPStart para CPLEX. Para ello, los resultados se obtuvieron con la ecuación (5.4).

$$Imp_{M,C} (\%) = 100 \times \frac{Z_C - Z_{MS}}{Z_C}, \quad (5.4)$$

De esta manera, un gap negativo indica que el valor de la función objetivo obtenido utilizando MIPStart es mayor (mejor) que el de la mejor solución encontrada por CPLEX.

Tabla 5.13: Porcentajes de mejora de las soluciones obtenidas por CPLEX con respecto al MIPStart.

Instancias	Imp_{M,C} (%)		
	Min	Prom	Max
Pequeñas	0.00	0.00	0.00
Medianas	-8.16	-0.54	2.02
Grandes	-6.40	-0.11	0.00

De acuerdo a la Tabla 5.13, se tiene que para el conjunto de instancias pequeñas se reporta una gap del 0% en los tres casos, lo cual indica que el MIPStart no mejoró las soluciones obtenidas por el ALNS, mientras que para el conjunto de instancias medianas y grandes se tiene una máxima mejora del -8.16% y -6.40%, respectivamente.

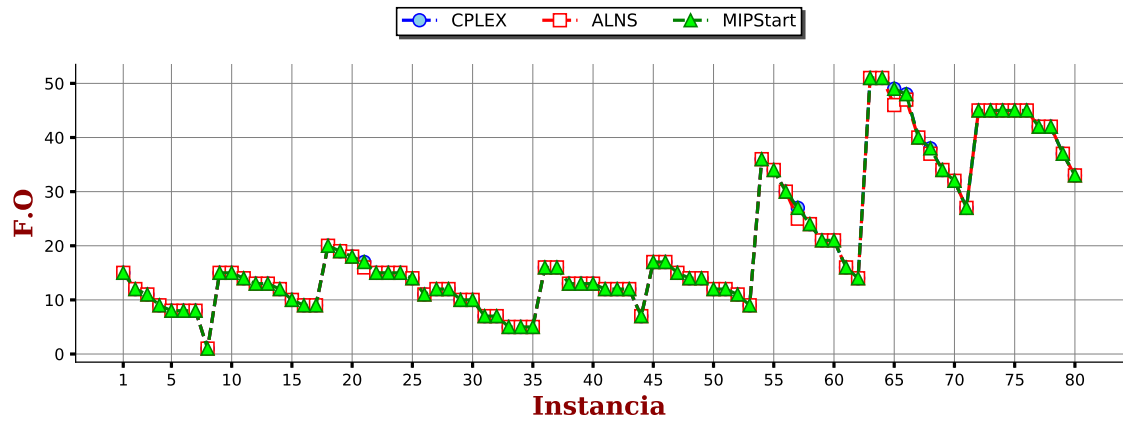


Figura 5.13: Valor de la función objetivo en instancias pequeñas.

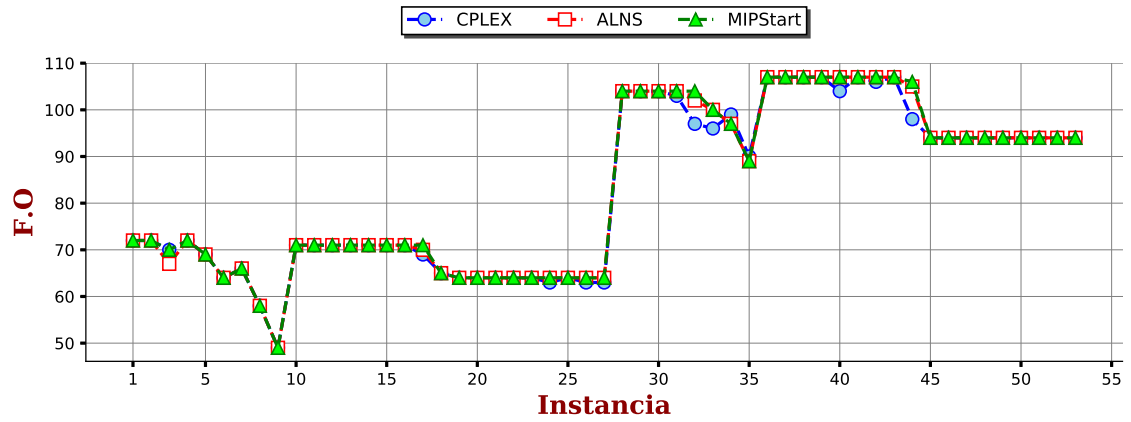


Figura 5.14: Valor de la función objetivo en instancias medianas.

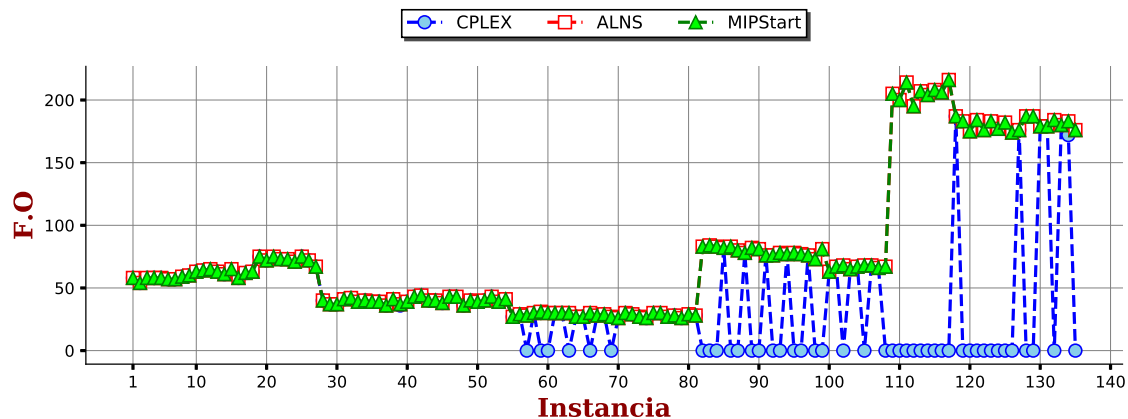


Figura 5.15: Valor de la función objetivo en instancias grandes.

En las Tablas A.1 a A.3 del Apéndice A.1 se exponen a detalle los resultados obtenidos para cada una de las instancias con CPLEX, ALNS y MIPStart.

5.7 COMPARACIÓN CON LOS ALGORITMOS HEURÍSTICOS PROPUESTOS EN LA LITERATURA

En esta sección se analizan algunos de los resultados expuestos por Amiri y Salari (2019), quienes propusieron tres algoritmos heurísticos, búsqueda local iterada (*Iterated Local Search*, ILS), búsqueda tabú (*Tabu Search*, TS) y búsqueda de vecindad variable (*Variable Neighborhood Search*, VNS), para resolver el TCMCRP.

Para llevar a cabo la comparación de los algoritmos heurísticos, se tomaron en cuenta únicamente las instancias grandes debido a que solo se cuenta con los resultados a detalle de este conjunto. Además, se descartaron para la comparación 29 instancias de las 135, ya que estas presentan inconsistencias en las tablas de resultados expuestas por Amiri y Salari (2019) con respecto a los archivos proporcionados de las mismas. Este tipo de inconsistencias se explican a detalle en el Apéndice A.2.

Además, dado que Amiri y Salari (2019) reportan en su artículo el valor de la

función objetivo promedio obtenida al haber ejecutado cinco veces cada algoritmo heurístico para el conjunto de instancias grandes, para el ALNS se realizó lo mismo, tomando en cuenta las primeras cinco réplicas de la experimentación realizada en la Sección 5.3 con la mejor configuración obtenida (criterio de paro de 2000 iteraciones con períodos de actualización de las probabilidades cada 100 iteraciones) y obteniendo el promedio de las mismas.

En la Tabla 5.14 se muestran los resultados del análisis de los algoritmos, donde en la segunda, tercera y cuarta columna se muestra para cada algoritmo el Gap promedio (%) mínimo, promedio y máximo, respectivamente. Dado que se conocen los óptimos para las 106 instancias con las que se realiza la comparación, el Gap promedio (%) de un algoritmo x se calcula con respecto al óptimo, de acuerdo a la ecuación (5.5).

$$Gap (\%) = 100 \times \frac{Z_{MS} - \bar{Z}_x}{Z_{MS}}, \quad (5.5)$$

Donde Z_{MS} representa el valor óptimo encontrado utilizando MIPStart, mientras que \bar{Z}_x es el promedio de las funciones objetivo obtenidas en las 5 ejecuciones del ALNS, ILS, VNS o TS, según sea el caso. Cabe recalcar que, los valores promedio encontrados con ILS, VNS y TS se tomaron de las tablas de resultados expuestas por Amiri y Salari (2019).

Finalmente, en la última columna se muestra la cantidad de óptimos alcanzados por cada metaheurística, de los 106 considerados para el análisis.

Tabla 5.14: Comparación del rendimiento del ALNS versus ILS, TS y VNS para el conjunto de instancias grandes.

Algoritmo	Gap (%)			Óptimos
	Min	Prom	Max	
ALNS	0.00	0.00	0.00	106/106
ILS	0.00	0.43	17.31	91/106
VNS	0.00	0.44	15.38	92/106
TS	0.00	0.00	0.00	106/106

Los resultados del algoritmo ALNS propuesto en este trabajo se muestran equitativos con los obtenidos con el TS, ambos alcanzando el óptimo para todas las instancias en las cinco réplicas. Por otra parte, el ILS alcanzó el 85.84% del total de óptimos con un gap promedio máximo del 17.31%, mientras que el VNS alcanzó el 86.79% con un gap promedio máximo del 15.38%. De esta manera, la metaheurística ALNS obtiene un mejor desempeño a comparación de los algoritmos ILS y VNS.

Para un análisis más detallado se han incluido en el Apéndice A.3 los resultados de la experimentación realizada y los obtenidos por Amiri y Salari (2019) para cada una de las 135 instancias.

CAPÍTULO 6

CONCLUSIONES Y TRABAJO FUTURO

En este Capítulo se presentan las conclusiones del trabajo y las posibles extensiones que podrían mejorar la calidad de los resultados obtenidos y así ampliar el alcance de la investigación.

6.1 CONCLUSIONES

En este trabajo se estudió el problema de ruteo con máxima cobertura y tiempo límite, también conocido en la literatura como *Time Constrained Maximal Covering Routing Problem* (TCMCRP), el cual es la combinación del problema del agente viajero con cobertura (CSP) y el problema del agente viajero selectivo múltiple (TOP). En este problema, se proporcionan un depósito central, un conjunto de vehículos y un conjunto de instalaciones, las cuales pueden cubrir a clientes que se encuentran dentro de un radio de cobertura dado. El problema consiste en determinar cuáles son las instalaciones a visitar y cuál será el orden de las visitas que deben realizar los vehículos disponibles, de tal forma que la duración total de cada ruta no exceda un tiempo límite y se cubra la mayor cantidad de clientes con las instalaciones visitadas.

El TCMCRP es un problema clasificado como NP-difícil, lo que implica que

sea relevante el desarrollo de algoritmos que permitan la resolución del problema de manera eficiente.

En la literatura existen diversas propuestas de resolución para el TCMCRP, tanto heurísticas como exactas; asimismo, es posible encontrar diferentes formulaciones y aplicaciones a problemáticas reales. No obstante, una metaheurística tipo ALNS como la propuesta en esta tesis, no ha sido previamente estudiada para este problema.

Se analizó el rendimiento del algoritmo ALNS sobre 268 instancias tomadas de la literatura con 52–724 nodos, divididas en conjuntos de tamaño pequeño, mediano y grande. Dicho algoritmo logró, obtener soluciones de buena calidad en tiempos de cómputo aceptables, resolviendo a optimalidad 198 de las 203 instancias reportadas haciendo uso del solver Cplex 12.9 y logrando mejorar los resultados en 56 instancias. Por otra parte, al hacer uso de inicios MIP, proporcionando a CPLEX como solución inicial la solución obtenida por el algoritmo ALNS, el optimizador fue capaz de alcanzar 258 óptimos y de esta manera pudimos validar que el ALNS propuesto resolvió a optimalidad 250 instancias de las 258.

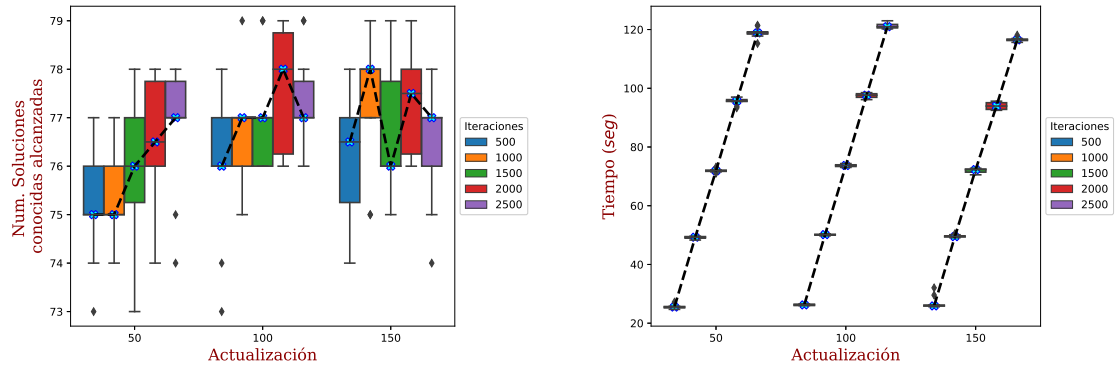
Por otra parte, para las 106 instancias grandes que son comparables con los resultados de la literatura, el ALNS muestra resultados competitivos con el TS (algoritmo que tuvo el mejor rendimiento de entre los propuestos por Amiri y Salari (2019)), ambos obteniendo un buen desempeño y alcanzando el óptimo en las cinco ejecuciones realizadas para cada instancia.

Además, de manera general con la experimentación realizada con CPLEX, utilizando MIPStart y la implementación del ALNS, se logró mejorar para tres instancias del problema, la mejor solución reportada hasta el momento en la literatura, alcanzando el óptimo para las mismas.

6.2 TRABAJO FUTURO

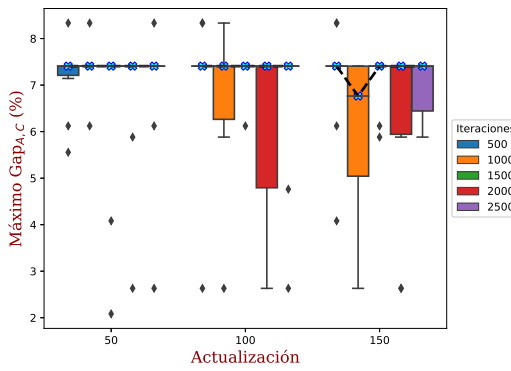
Como trabajo a futuro se propone evaluar el rendimiento del algoritmo propuesto en otro conjunto de instancias de la literatura o en casos reales. Además, se puede adaptar el algoritmo para resolver el problema del agente viajero con máxima cobertura y tiempo límite (TCMCSP) y el problema del agente viajero selectivo múltiple (TOP) para evaluar la efectividad del ALNS.

Asimismo, se pueden diseñar posibles mejoras al algoritmo propuesto, como cambiar la estructura de los operadores utilizados en este trabajo o implementar otros nuevos, así como reforzar la experimentación realizada para la elección de los parámetros del ALNS, esto debido a que se analizaron los resultados de agregar 150 iteraciones como nuevo nivel en el período de actualizaciones de las probabilidades, obteniendo que es posible encontrar una configuración que obtenga estadísticamente el mismo número de soluciones alcanzadas y además un ahorro en el tiempo de ejecución. En las Figuras 6.1-6.3 se muestran los diagramas de caja y bigotes correspondientes a los diseños factoriales en interacción con el número de veces que el algoritmo ALNS alcanzó o superó la mejor solución, el tiempo de ejecución y el máximo $\text{Gap}_{A,C}$ (%) para las instancias pequeñas, medianas y grandes.



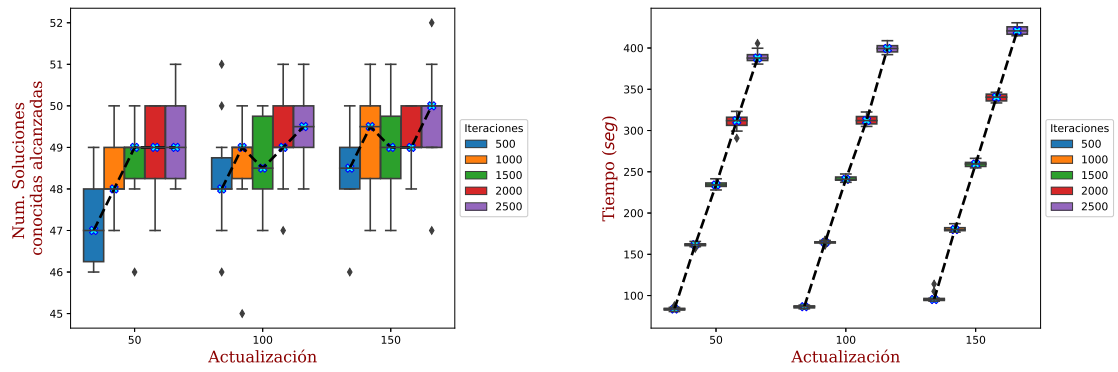
(a) Número de veces que se alcanzó o superó la mejor solución

(b) Tiempo de ejecución



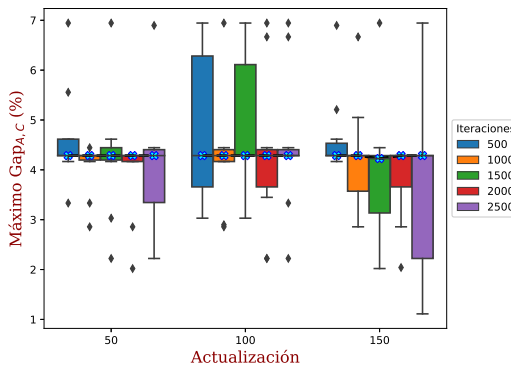
(c) Máximo Gap_{A,C} (%)

Figura 6.1: Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias pequeñas.



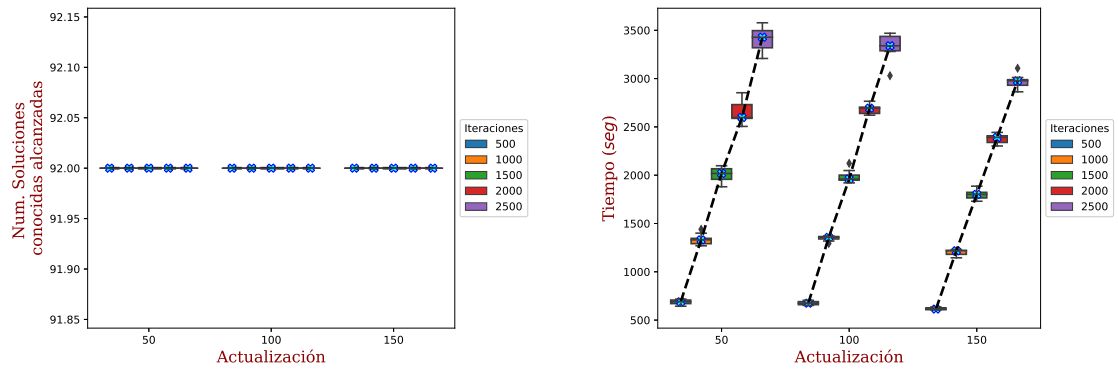
(a) Número de veces que se alcanzó o superó la mejor solución

(b) Tiempo de ejecución



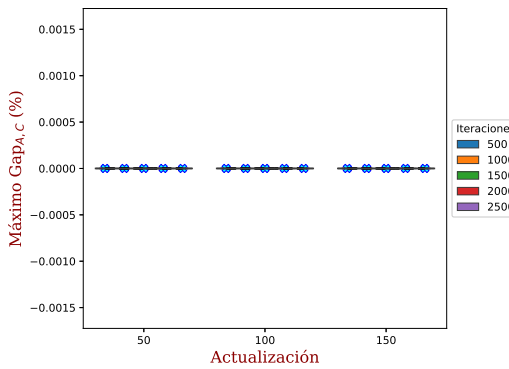
(c) Máximo Gap_{A,C} (%)

Figura 6.2: Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias medianas.



(a) Número de veces que se alcanzó o superó la mejor solución

(b) Tiempo de ejecución



(c) Máximo Gap_{A,C} (%)

Figura 6.3: Variables de respuesta por nivel de iteración y actualización para el conjunto de instancias grandes.

Por último, dada la importancia actual del medio ambiente, se pretende extender el problema de estudio diseñando un modelo matemático que logre incorporar aspectos ambientales. De esta manera, se pretende resolver el problema mediante la resolución secuencial de dos problemas de optimización, el primero de ellos teniendo como objetivo cubrir la máxima cantidad de clientes, mientras que el segundo intenta minimizar las emisiones de CO_2 .

APÉNDICE A

TABLAS DE RESULTADOS

En este apéndice se presentan las tablas que contienen los resultados detallados de los experimentos computacionales realizados en este trabajo.

A.1 RESULTADOS DEL MODELO TCMCRP

En las Tablas A.1-A.3 se presentan los resultados de las instancias pequeñas, medianas y grandes. En la primera columna se encuentra el nombre de la instancia, en la segunda, tercera y cuarta columna se muestra el valor de la función objetivo, Gap (%) y el tiempo de ejecución total en segundos reportado por el optimizador CPLEX, respectivamente. La quinta, sexta, séptima y octava columna muestran el valor de la función objetivo que se obtiene con el algoritmo ALNS, el $\text{Gap}_{A,C}$ (%) y $\text{Gap}_{A,M}$ (%), los cuales se calculan de acuerdo con (5.1) y (5.3), respectivamente y el tiempo de ejecución total del ALNS al resolver el modelo TCMCRP.

Las columnas restantes, contienen el valor de la función objetivo de la mejor solución encontrada, el Gap (%) y el tiempo computacional que reporta el optimizador CPLEX utilizando MIPStart, así como el mejor valor de la función objetivo obtenido de entre los tres métodos, CPLEX, ALNS y MIPStart representado por Z_B . Se resalta en **negritas** el mejor valor obtenido y si ocurre un empate se selecciona

aquel que haya tenido el menor tiempo de ejecución.

En los casos en los que no se muestra ningún valor es porque CPLEX no encontró ninguna solución factible al alcanzar el tiempo límite de 7200 segundos, y, en consecuencia, no es posible calcular el gap.

Cabe recordar que un gap negativo indica que el valor de la función objetivo obtenido por la metaheurística es más alto que el de la mejor solución encontrada por CPLEX en 7200 segundos, en tal caso la solución reportada por el optimizador no sería óptima, sino entera factible.

Tabla A.1: Resultados obtenidos por CPLEX, ALNS y MIPStart para el conjunto de instancias pequeñas.

Instancia	CPLEX			ALNS			MIPStart			Z_B	
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)		Tiempo (s)
Input2-0-0-1	15	0	0.30	15	0	0	0.63	15	0	0.18	15
Input2-0-0-2	12	0	0.06	12	0	0	0.60	12	0	0.04	12
Input2-0-0-4	11	0	0.07	11	0	0	0.79	11	0	0.04	11
Input2-0-0-5	9	0	0.02	9	0	0	0.63	9	0	0.03	9
Input2-0-0-6	8	0	0.02	8	0	0	0.62	8	0	0.03	8
Input2-0-0-7	8	0	0.02	8	0	0	0.67	8	0	0.03	8
Input2-0-0-8	8	0	0.02	8	0	0	0.62	8	0	0.03	8
Input2-0-0-9	1	0	0.01	1	0	0	0.60	1	0	0.02	1
Input2-1-0-1	15	0	5.01	15	0	0	0.73	15	0	18.32	15
Input2-1-0-2	15	0	0.80	15	0	0	0.71	15	0	0.95	15
Input2-1-0-3	14	0	0.79	14	0	0	0.78	14	0	5.20	14
Input2-1-0-4	13	0	0.10	13	0	0	0.82	13	0	0.07	13
Input2-1-0-5	13	0	0.05	13	0	0	0.78	13	0	0.06	13
Input2-1-0-6	12	0	0.32	12	0	0	0.85	12	0	0.39	12
Input2-1-0-7	10	0	0.05	10	0	0	0.79	10	0	0.07	10
Input2-1-0-8	9	0	0.06	9	0	0	0.83	9	0	0.06	9
Input2-1-0-9	9	0	0.05	9	0	0	0.84	9	0	0.05	9
Input2-2-0-1	20	0	204.64	20	0	0	1.03	20	0	4.13	20
Input2-2-0-2	19	0	18.84	19	0	0	0.97	19	0	3.01	19
Input2-2-0-3	18	0	25.10	18	0	0	0.95	18	0	2.81	18
Input2-2-0-4	17	0	35.71	16	5.88	5.88	1.00	17	0	58.14	17
Input2-2-0-5	15	0	0.24	15	0	0	0.97	15	0	0.13	15
Input2-2-0-6	15	0	0.10	15	0	0	0.96	15	0	0.11	15
Input2-2-0-7	15	0	0.39	15	0	0	1.12	15	0	0.43	15
Input2-2-0-8	14	0	1.78	14	0	0	1.10	14	0	4.30	14
Input2-2-0-9	11	0	0.11	11	0	0	1.06	11	0	0.08	11
Input5-0-0-1	12	0	0.05	12	0	0	0.83	12	0	0.04	12
Input5-0-0-2	12	0	0.05	12	0	0	0.98	12	0	0.04	12
Input5-0-0-3	10	0	0.02	10	0	0	0.84	10	0	0.03	10
Input5-0-0-4	10	0	0.02	10	0	0	0.92	10	0	0.04	10
Input5-0-0-5	7	0	0.02	7	0	0	0.92	7	0	0.04	7
Input5-0-0-6	7	0	0.02	7	0	0	1.07	7	0	0.03	7
Input5-0-0-7	5	0	0.02	5	0	0	1.31	5	0	0.04	5

Tabla A.1: Continuación.

Instancia	CPLEX			ALNS			MIPStart			Z_B	
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)		Tiempo (s)
Input5-0-0-8.	5	0	0.02	5	0	0	1.18	5	0	0.04	5
Input5-0-0-9.	5	0	0.02	5	0	0	1.25	5	0	0.04	5
Input5-1-0-1	16	0	0.15	16	0	0	1.12	16	0	0.17	16
Input5-1-0-2	16	0	0.48	16	0	0	1.06	16	0	0.13	16
Input5-1-0-3	13	0	20.25	13	0	0	1.19	13	0	26.07	13
Input5-1-0-4	13	0	0.11	13	0	0	1.31	13	0	0.09	13
Input5-1-0-5	13	0	0.09	13	0	0	1.18	13	0	0.08	13
Input5-1-0-6	12	0	0.06	12	0	0	1.20	12	0	0.06	12
Input5-1-0-7	12	0	0.05	12	0	0	1.21	12	0	0.09	12
Input5-1-0-8	12	0	0.04	12	0	0	1.29	12	0	0.08	12
Input5-1-0-9	7	0	0.03	7	0	0	1.38	7	0	0.07	7
Input5-2-0-1	17	0	1.71	17	0	0	1.69	17	0	0.26	17
Input5-2-0-2	17	0	19.76	17	0	0	1.50	17	0	0.25	17
Input5-2-0-3	15	0	0.74	15	0	0	1.70	15	0	0.19	15
Input5-2-0-4	14	0	0.21	14	0	0	1.60	14	0	0.21	14
Input5-2-0-5	14	0	0.23	14	0	0	1.25	14	0	0.15	14
Input5-2-0-6	12	0	0.09	12	0	0	1.41	12	0	0.11	12
Input5-2-0-7	12	0	0.12	12	0	0	1.53	12	0	0.14	12
Input5-2-0-8	11	0	0.09	11	0	0	1.42	11	0	0.14	11
Input5-2-0-9	9	0	0.06	9	0	0	2.32	9	0	0.10	9
Input10-0-0-1	36	0	455.81	36	0	0	1.17	36	0	542.77	36
Input10-0-0-2	34	0	4.53	34	0	0	1.20	34	0	3.41	34
Input10-0-0-3	30	0	3.28	30	0	0	1.33	30	0	1.97	30
Input10-0-0-4	27	0	1.46	25	7.41	7.41	1.17	27	0	0.12	27
Input10-0-0-5	24	0	0.23	24	0	0	1.09	24	0	0.10	24
Input10-0-0-6	21	0	0.15	21	0	0	1.30	21	0	0.09	21
Input10-0-0-7	21	0	0.16	21	0	0	1.16	21	0	0.12	21
Input10-0-0-8	16	0	0.11	16	0	0	1.18	16	0	0.11	16
Input10-0-0-9	14	0	0.09	14	0	0	1.13	14	0	0.10	14
Input10-1-0-1	51	0	231.38	51	0	0	2.01	51	0	0.24	51
Input10-1-0-2	51	0	182.43	51	0	0	1.77	51	0	158.79	51
Input10-1-0-3	49	0	5190.21	46	6.12	6.12	2.10	49	0	5977.05	49
Input10-1-0-4	48	2.08	7199.64	47	2.08	2.08	1.72	48	0	2776.43	48

Tabla A.1: Continuación.

Instancia	CPLEX			ALNS			MIPStart				
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)	Tiempo (s)	Z_B
Input10-1-0-5	40	0	1647.10	40	0	0	2.46	40	0	5620.75	40
Input10-1-0-6	38	0	286.67	37	2.63	2.63	2.09	38	0	863.60	38
Input10-1-0-7	34	0	0.99	34	0	0	1.96	34	0	0.30	34
Input10-1-0-8	32	0	0.58	32	0	0	2.01	32	0	0.26	32
Input10-1-0-9	27	0	0.37	27	0	0	1.85	27	0	0.22	27
Input10-2-0-1	45	0	22.54	45	0	0	2.37	45	0	0.36	45
Input10-2-0-2	45	0	49.56	45	0	0	2.63	45	0	0.38	45
Input10-2-0-3	45	0	130.91	45	0	0	2.56	45	0	0.37	45
Input10-2-0-4	45	0	284.82	45	0	0	2.36	45	0	0.50	45
Input10-2-0-5	45	0	566.96	45	0	0	2.66	45	0	0.46	45
Input10-2-0-6	42	2.38	7199.71	42	0	0	2.23	42	2.38	7199.58	42
Input10-2-0-7	42	2.38	7199.49	42	0	0	2.99	42	2.38	7199.54	42
Input10-2-0-8	37	0	20.17	37	0	0	3.19	37	0	0.36	37
Input10-2-0-9	33	0	11.23	33	0	0	2.87	33	0	0.46	33

Tabla A.2: Resultados obtenidos por CPLEX, ALNS y MIPStart para el conjunto de instancias medianas.

Instancia	CPLEX			ALNS				MIPStart			Z_B
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)	Tiempo (s)	
Input14-0-0-1	72	0	573.32	72	0	0	3.54	72	0	0.38	72
Input14-0-0-2	72	0	99.10	72	0	0	2.61	72	0	0.39	72
Input14-0-0-3	70	2.86	7199.81	67	4.29	4.29	3.11	70	2.86	7199.61	70
Input14-0-0-4	72	0	566.32	72	0	0	2.53	72	0	0.50	72
Input14-0-0-5	69	0	719.38	69	0	0	2.58	69	0	84.84	69
Input14-0-0-6	64	2.30	7199.78	64	0	0	2.54	64	3.13	7199.56	64
Input14-0-0-7	66	1.52	7199.32	66	0	0	3.02	66	1.52	7199.57	66
Input14-0-0-8	58	4.24	7199.60	58	0	0	3.22	58	1.72	7199.52	58
Input14-0-0-9	49	0	390.69	49	0	0	3.26	49	0	2750.61	49
Input14-1-0-1	71	0	23.26	71	0	0	4.25	71	0	0.55	71
Input14-1-0-2	71	0	54.70	71	0	0	5.16	71	0	0.54	71
Input14-1-0-3	71	0	728.98	71	0	0	4.54	71	0	0.53	71
Input14-1-0-4	71	0	147.62	71	0	0	4.99	71	0	0.79	71
Input14-1-0-5	71	0	771.17	71	0	0	3.62	71	0	0.80	71
Input14-1-0-6	71	0	2854.80	71	0	0	3.75	71	0	0.75	71
Input14-1-0-7	71	0	6246.03	71	0	0	3.93	71	0	0.97	71
Input14-1-0-8	69	2.90	7199.59	70	-1.45	1.41	4.49	71	0	131.02	71
Input14-1-0-9	65	4.62	7199.60	65	0	0	3.83	65	4.62	7199.65	65
Input14-2-0-1	64	0	55.94	64	0	0	4.61	64	0	0.81	64
Input14-2-0-2	64	0	939.50	64	0	0	4.97	64	0	0.86	64
Input14-2-0-3	64	0	70.58	64	0	0	5.34	64	0	0.81	64
Input14-2-0-4	64	0	115.15	64	0	0	4.38	64	0	1.28	64
Input14-2-0-5	64	0	879.71	64	0	0	4.66	64	0	1.30	64
Input14-2-0-6	63	1.59	7199.57	64	-1.59	0	4.67	64	0	1.31	64
Input14-2-0-7	64	0	1185.92	64	0	0	5.89	64	0	1.78	64
Input14-2-0-8	63	1.59	7199.57	64	-1.59	0	4.87	64	0	1.71	64
Input14-2-0-9	63	1.59	7199.57	64	-1.59	0	4.21	64	0	1.57	64
Input16-0-0-1	104	0	50.77	104	0	0	7.84	104	0	0.56	104
Input16-0-0-2	104	0	598.17	104	0	0	6.59	104	0	0.55	104
Input16-0-0-3	104	0	4244.15	104	0	0	4.38	104	0	0.55	104
Input16-0-0-4	103	0.97	7199.51	104	-0.97	0	4.78	104	0	0.83	104
Input16-0-0-5	97	7.22	7199.58	102	-5.15	1.92	4.39	104	0	3916.90	104
Input16-0-0-6	96	7.29	7199.61	100	-4.17	0	6.18	100	3.00	7199.55	100

Tabla A.2: Continuación.

Instancia	CPLEX			ALNS			MIPStart			Z_B	
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)		Tiempo (s)
Input16-0-0-7	99	3.79	7199.65	97	2.02	0	4.43	97	5.89	7199.60	99
Input16-0-0-8	90	0	3609.90	89	1.11	0	5.26	89	1.12	7199.53	90
Input16-1-0-1	107	0	42.74	107	0	0	9.12	107	0	1.00	107
Input16-1-0-2	107	0	1752.64	107	0	0	9.79	107	0	0.98	107
Input16-1-0-3	107	0	857.33	107	0	0	7.83	107	0	1.01	107
Input16-1-0-4	107	0	254.03	107	0	0	10.30	107	0	1.55	107
Input16-1-0-5	104	2.88	7199.48	107	-2.88	0	8.70	107	0	1.57	107
Input16-1-0-6	107	0	2814.26	107	0	0	7.16	107	0	1.55	107
Input16-1-0-7	106	0.94	7199.54	107	-0.94	0	10.86	107	0	2.16	107
Input16-1-0-8	107	0	6953.01	107	0	0	6.83	107	0	2.10	107
Input16-1-0-9	98	8.16	7199.35	105	-7.14	0.94	7.66	106	0	1.90	106
Input16-2-0-1	94	0	564.02	94	0	0	10.11	94	0	1.57	94
Input16-2-0-2	94	0	84.11	94	0	0	9.38	94	0	1.58	94
Input16-2-0-3	94	0	418.91	94	0	0	9.55	94	0	1.69	94
Input16-2-0-4	94	0	317.92	94	0	0	11.11	94	0	2.51	94
Input16-2-0-5	94	0	899.39	94	0	0	9.20	94	0	2.49	94
Input16-2-0-6	94	0	4661.54	94	0	0	8.61	94	0	2.50	94
Input16-2-0-7	94	0	620.09	94	0	0	12.58	94	0	3.54	94
Input16-2-0-8	94	0	6892.13	94	0	0	8.96	94	0	3.56	94
Input16-2-0-9	94	0	3210.68	94	0	0	10.33	94	0	3.56	94

Tabla A.3: Resultados obtenidos por CPLEX, ALNS y MIPStart para el conjunto de instancias grandes.

Instancia	CPLEX			ALNS				MIPStart			Z_B
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)	Tiempo (s)	
G-Input2-0-0-1	58	0	674.52	58	0	0	31.77	58	0	8.80	58
G-Input2-0-0-2	54	0	2643.31	54	0	0	26.98	54	0	14.37	54
G-Input2-0-0-3	58	0	6678.06	58	0	0	40.53	58	0	19.99	58
G-Input2-0-0-4	58	0	813.40	58	0	0	58.28	58	0	8.82	58
G-Input2-0-0-5	58	0	2989.99	58	0	0	32.56	58	0	14.44	58
G-Input2-0-0-6	57	0	6431.42	57	0	0	31.97	57	0	20.06	57
G-Input2-0-0-7	57	0	871.79	57	0	0	19.32	57	0	8.93	57
G-Input2-0-0-8	59	0	1801.71	59	0	0	39.12	59	0	14.26	59
G-Input2-0-0-9	60	0	4902.39	60	0	0	32.55	60	0	19.97	60
G-Input2-1-0-1	63	0	308.00	63	0	0	23.63	63	0	5.29	63
G-Input2-1-0-2	64	0	468.88	64	0	0	24.43	64	0	8.52	64
G-Input2-1-0-3	65	0	2989.48	65	0	0	26.55	65	0	11.94	65
G-Input2-1-0-4	63	0	325.02	63	0	0	16.39	63	0	5.25	63
G-Input2-1-0-5	61	0	873.65	61	0	0	28.10	61	0	8.53	61
G-Input2-1-0-6	65	0	2555.32	65	0	0	17.69	65	0	11.95	65
G-Input2-1-0-7	58	0	395.43	58	0	0	23.46	58	0	5.27	58
G-Input2-1-0-8	62	0	572.57	62	0	0	22.75	62	0	8.67	62
G-Input2-1-0-9	63	0	1632.23	63	0	0	21.64	63	0	11.94	63
G-Input2-2-0-1	75	0	265.44	75	0	0	16.41	75	0	2.83	75
G-Input2-2-0-2	72	0	336.81	72	0	0	13.04	72	0	4.52	72
G-Input2-2-0-3	74	1.35	7199.58	75	-1.35	0	15.49	75	0	6.30	75
G-Input2-2-0-4	73	0	2091.80	73	0	0	16.26	73	0	2.81	73
G-Input2-2-0-5	73	0	295.15	73	0	0	13.77	73	0	4.51	73
G-Input2-2-0-6	71	0	6491.04	71	0	0	17.78	71	0	6.32	71
G-Input2-2-0-7	75	0	104.68	75	0	0	14.12	75	0	2.80	75
G-Input2-2-0-8	72	0	250.52	72	0	0	20.58	72	0	4.51	72
G-Input2-2-0-9	67	0	3917.91	67	0	0	9.43	67	0	6.32	67
G-Input5-0-0-1	40	0	325.99	40	0	0	23.41	40	0	4.61	40
G-Input5-0-0-2	37	0	1322.31	37	0	0	23.74	37	0	7.50	37
G-Input5-0-0-3	37	0	2589.61	37	0	0	20.21	37	0	10.62	37
G-Input5-0-0-4	41	0	333.56	41	0	0	24.83	41	0	4.63	41
G-Input5-0-0-5	42	0	978.20	42	0	0	17.34	42	0	7.53	42
G-Input5-0-0-6	39	0	2134.10	39	0	0	15.80	39	0	10.63	39

Tabla A.3: Continuación.

Instancia	CPLEX			ALNS			MIPStart				
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)	Tiempo (s)	Z_B
G-Input5-0-0-7	40	0	415.31	40	0	0	23.52	40	0	4.62	40
G-Input5-0-0-8	39	0	1451.25	39	0	0	16.26	39	0	7.53	39
G-Input5-0-0-9	39	0	2592.45	39	0	0	17.18	39	0	10.62	39
G-Input5-1-0-1	36	0	199.80	36	0	0	9.79	36	0	2.78	36
G-Input5-1-0-2	41	0	527.34	41	0	0	11.53	41	0	4.56	41
G-Input5-1-0-3	36	2.78	7199.53	37	-2.78	0	10.09	37	0	6.34	37
G-Input5-1-0-4	39	0	207.43	39	0	0	16.57	39	0	2.80	39
G-Input5-1-0-5	43	0	441.49	43	0	0	17.14	43	0	4.58	43
G-Input5-1-0-6	44	0	881.52	44	0	0	10.07	44	0	6.46	44
G-Input5-1-0-7	40	0	239.08	40	0	0	23.82	40	0	2.79	40
G-Input5-1-0-8	40	0	392.79	40	0	0	14.81	40	0	4.57	40
G-Input5-1-0-9	38	0	1365.21	38	0	0	12.57	38	0	6.44	38
G-Input5-2-0-1	43	0	81.47	43	0	0	7.95	43	0	1.50	43
G-Input5-2-0-2	43	0	451.56	43	0	0	5.03	43	0	2.34	43
G-Input5-2-0-3	36	0	320.33	36	0	0	4.83	36	0	2.68	36
G-Input5-2-0-4	40	0	69.63	40	0	0	6.22	40	0	1.48	40
G-Input5-2-0-5	39	0	91.37	39	0	0	5.79	39	0	2.36	39
G-Input5-2-0-6	40	0	566.96	40	0	0	4.64	40	0	3.15	40
G-Input5-2-0-7	43	0	125.79	43	0	0	5.47	43	0	1.48	43
G-Input5-2-0-8	39	0	157.42	39	0	0	8.53	39	0	2.34	39
G-Input5-2-0-9	41	0	251.29	41	0	0	7.67	41	0	3.33	41
G-Input10-0-0-1	27	0	3978.11	27	0	0	93.45	27	0	19.96	27
G-Input10-0-0-2	29	0	4631.47	29	0	0	53.60	29	0	31.46	29
G-Input10-0-0-3	-	-	7200.80	28	-	0	73.67	28	0	43.32	28
G-Input10-0-0-4	30	0	1875.52	30	0	0	71.30	30	0	19.90	30
G-Input10-0-0-5	-	-	7199.74	31	-	0	51.50	31	0	31.82	31
G-Input10-0-0-6	-	-	7199.74	30	-	0	116.91	30	0	43.57	30
G-Input10-0-0-7	30	0	1725.88	30	0	0	54.72	30	0	19.96	30
G-Input10-0-0-8	30	0	6216.18	30	0	0	78.29	30	0	31.46	30
G-Input10-0-0-9	-	-	7200.09	30	-	0	42.42	30	0	44.41	30
G-Input10-1-0-1	27	0	1574.46	27	0	0	63.45	27	0	11.57	27
G-Input10-1-0-2	27	0	4573.95	27	0	0	55.08	27	0	18.55	27
G-Input10-1-0-3	-	-	7200.23	30	-	0	36.78	30	0	25.90	30

Tabla A.3: Continuación.

Instancia	CPLEX			ALNS			MIPStart			Z_B	
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap $_{A,C}$ (%)	Gap $_{A,M}$ (%)	Tiempo (s)	Z_{MS}	Gap (%)		Tiempo (s)
G-Input10-1-0-4	28	0	1171.57	28	0	0	35.48	28	0	11.57	28
G-Input10-1-0-5	29	0	1999.69	29	0	0	32.28	29	0	18.70	29
G-Input10-1-0-6	-	-	7200.31	27	-	0	44.49	27	0	25.37	27
G-Input10-1-0-7	26	0	949.38	26	0	0	43.33	26	0	11.62	26
G-Input10-1-0-8	30	0	3988.82	30	0	0	46.43	30	0	18.55	30
G-Input10-1-0-9	29	0	3595.22	29	0	0	36.92	29	0	25.57	29
G-Input10-2-0-1	27	0	597.64	27	0	0	32.09	27	0	6.05	27
G-Input10-2-0-2	26	0	1409.02	26	0	0	20.97	26	0	9.68	26
G-Input10-2-0-3	30	0	2054.47	30	0	0	21.82	30	0	13.48	30
G-Input10-2-0-4	30	0	353.85	30	0	0	16.57	30	0	6.08	30
G-Input10-2-0-5	27	0	563.23	27	0	0	24.94	27	0	9.68	27
G-Input10-2-0-6	28	0	1828.07	28	0	0	20.80	28	0	13.43	28
G-Input10-2-0-7	26	0	263.21	26	0	0	14.80	26	0	6.04	26
G-Input10-2-0-8	29	0	1012.62	29	0	0	29.83	29	0	9.70	29
G-Input10-2-0-9	28	0	1878.75	28	0	0	31.71	28	0	13.43	28
G-Input14-0-0-1	-	-	7199.76	83	-	0	66.75	83	0	28.76	83
G-Input14-0-0-2	-	-	7199.91	84	-	0	82.42	84	0	54.98	84
G-Input14-0-0-3	-	-	7199.97	83	-	0	80.86	83	0	76.26	83
G-Input14-0-0-4	82	0	4905.47	82	0	0	92.00	82	0	34.89	82
G-Input14-0-0-5	-	-	7199.78	83	-	0	120.17	83	0	55.22	83
G-Input14-0-0-6	-	-	7199.87	80	-	0	74.16	80	0	63.15	80
G-Input14-0-0-7	78	0	1780.50	78	0	0	65.53	78	0	34.79	78
G-Input14-0-0-8	-	-	7200.00	82	-	0	66.82	82	0	49.70	82
G-Input14-0-0-9	-	-	7200.10	81	-	0	105.55	81	0	63.33	81
G-Input14-1-0-1	76	0	3695.49	76	0	0	44.75	76	0	19.09	76
G-Input14-1-0-2	-	-	7199.64	76	-	0	32.05	76	0	26.19	76
G-Input14-1-0-3	-	-	7199.92	78	-	0	43.61	78	0	44.25	78
G-Input14-1-0-4	77	0	3679.26	77	0	0	45.10	77	0	20.02	77
G-Input14-1-0-5	-	-	7200.31	78	-	0	60.78	78	0	26.17	78
G-Input14-1-0-6	-	-	7199.94	77	-	0	46.00	77	0	43.88	77
G-Input14-1-0-7	76	0	1345.30	76	0	0	51.23	76	0	19.86	76
G-Input14-1-0-8	-	-	7199.77	73	-	0	45.33	73	0	31.78	73
G-Input14-1-0-9	-	-	7200.53	81	-	0	32.67	81	0	44.06	81

Tabla A.3: Continuación.

Instancia	CPLEX			ALNS			MIPStart				
	Z_C	Gap (%)	Tiempo (s)	Z_A	Gap _{A,C} (%)	Gap _{A,M} (%)	Tiempo (s)	Z_{MS}	Gap (%)	Tiempo (s)	Z_B
G-Input14-2-0-1	63	0	360.18	63	0	0	21.34	63	0	10.23	63
G-Input14-2-0-2	67	0	5616.98	67	0	0	19.69	67	0	16.45	67
G-Input14-2-0-3	-	-	7199.67	68	-	0	11.98	68	0	18.62	68
G-Input14-2-0-4	65	0	651.98	65	0	0	22.33	65	0	9.90	65
G-Input14-2-0-5	67	0	4017.77	67	0	0	16.33	67	0	16.41	67
G-Input14-2-0-6	-	-	7200.03	68	-	0	11.11	68	0	22.84	68
G-Input14-2-0-7	68	0	587.05	68	0	0	32.41	68	0	10.56	68
G-Input14-2-0-8	66	0	2464.21	66	0	0	46.91	66	0	16.47	66
G-Input14-2-0-9	-	-	7199.95	67	-	0	32.97	67	0	23.22	67
G-Input16-0-0-1	-	-	7198.92	205	-	0	82.76	205	0	44.71	205
G-Input16-0-0-2	-	-	7199.78	200	-	0	144.77	200	0	58.15	200
G-Input16-0-0-3	-	-	7199.88	214	-	0	162.90	214	0	80.08	214
G-Input16-0-0-4	-	-	7199.61	195	-	0	122.23	195	0	36.61	195
G-Input16-0-0-5	-	-	7200.07	207	-	0	108.74	207	0	70.83	207
G-Input16-0-0-6	-	-	7200.27	204	-	0	116.13	204	0	98.01	204
G-Input16-0-0-7	-	-	7199.71	208	-	0	108.55	208	0	41.54	208
G-Input16-0-0-8	-	-	7200.03	206	-	0	147.41	206	0	70.57	206
G-Input16-0-0-9	-	-	7200.07	216	-	0	154.08	216	0	99.18	216
G-Input16-1-0-1	187	0	6020.97	187	0	0	65.53	187	0	20.73	187
G-Input16-1-0-2	-	-	7200.06	183	-	0	63.70	183	0	40.16	183
G-Input16-1-0-3	-	-	7200.02	175	-	0	112.26	175	0	57.34	175
G-Input16-1-0-4	-	-	7199.85	184	-	0	83.04	184	0	25.09	184
G-Input16-1-0-5	-	-	7200.20	176	-	0	69.04	176	0	40.14	176
G-Input16-1-0-6	-	-	7200.01	183	-	0	86.47	183	0	55.78	183
G-Input16-1-0-7	-	-	7199.70	177	-	0	78.01	177	0	24.41	177
G-Input16-1-0-8	-	-	7199.85	182	-	0	108.31	182	0	40.18	182
G-Input16-1-0-9	-	-	7199.90	174	-	0	69.54	174	0	58.43	174
G-Input16-2-0-1	176	0	2361.58	176	0	0	42.34	176	0	13.18	176
G-Input16-2-0-2	-	-	7200.00	187	-	0	37.46	187	0	16.83	187
G-Input16-2-0-3	-	-	7199.70	187	-	0	49.42	187	0	28.49	187
G-Input16-2-0-4	179	0	4752.04	179	0	0	44.35	179	0	13.06	179
G-Input16-2-0-5	179	0	6680.60	179	0	0	45.41	179	0	16.83	179
G-Input16-2-0-6	-	-	7199.89	184	-	0	49.23	184	0	28.49	184
G-Input16-2-0-7	180	0	2418.36	180	0	0	71.17	180	0	12.94	180
G-Input16-2-0-8	172	6.40	7199.57	183	-6.40	0	54.19	183	0	20.45	183
G-Input16-2-0-9	-	-	7199.76	176	-	0	57.46	176	0	28.50	176

A.2 COMPARACIÓN DE RESULTADOS DEL MODELO TCMCRP

En las Tablas A.6-A.8 se muestran los resultados reportados en la literatura contra el mejor valor obtenido entre CPLEX, ALNS y MIPStart. En la primera columna se encuentra el nombre de la instancia, en la segunda se muestra Z_B , valor que se reportó en las Tablas A.1-A.3, en la tercer columna se muestra el valor objetivo obtenido por Amiri y Salari (2019), quienes introducen por primera vez el problema TCMCRP, y cuyos valores fueron obtenidos utilizando ILOG CPLEX 12.3 considerando como criterio de paro para cada una de las instancias cinco horas como tiempo de cómputo. En la cuarta y quinta columna se muestran los valores de la función objetivo obtenidos por Sinnl (2019), en el cual se desarrollaron dos algoritmos de solución exacta, el primero es basado en un modelo compacto denominado C_+ , mientras que el otro modelo E_{++} contiene un número de restricciones exponencial, además desarrollaron técnicas de preprocesamiento y desigualdades validas para ambos modelos. Se utilizó CPLEX 12.8 con un tiempo límite de cómputo de 600 segundos para cada instancia. Cabe mencionar, que los resultados de Sinnl son expuestos en un reporte técnico que está en proceso de evaluación.

Finalmente, en las columnas restantes se muestra el Gap (%) que existe entre Z_B y la solución reportada por Amiri y Salari (2019) y Sinnl (2019), el cual se calcula con las fórmulas (A.1), (A.2) y (A.3), donde un gap negativo indica que el mejor valor obtenido entre CPLEX, ALNS y MIPStart es más alto que soluciones encontradas hasta el momento en la literatura.

$$Gap (\%) = 100 \times \frac{Z_{AS} - Z_B}{Z_{AS}}, \quad (A.1)$$

$$Gap (\%) = 100 \times \frac{Z_{C_+} - Z_B}{Z_{C_+}}, \quad (A.2)$$

$$Gap (\%) = 100 \times \frac{Z_{E_{++}} - Z_B}{Z_{E_{++}}}, \tag{A.3}$$

Donde Z_B representa el mejor valor de la función objetivo obtenido entre CPLEX, ALNS y MIPStart, Z_{AS} el valor de la función objetivo encontrado por Amiri y Salari (2019), mientras que Z_{C_+} y $Z_{E_{++}}$ representan los valores objetivo obtenidos por Sinnl (2019).

En las tablas se resalta en **negritas** si el valor objetivo encontrado representa el valor objetivo óptimo para esa instancia, es decir, que se tenga un gap del 0%. Además, debido a que en las tablas de resultados de Amiri y Salari (2019) se presentan algunas inconsistencias referentes a las características de las instancias o a los valores objetivo reportados, se hizo un análisis de estas inconsistencias dividiéndolas en dos grupos, las cuales se explicarán a detalle a continuación.

1. Diferencia en los tiempos límite de los vehículos. En la Tabla A.4 se muestra un ejemplo de instancia con este tipo de inconsistencia, en la cual el tiempo límite dado en el archivo de la instancia no coincide con el expuesto en las tablas de resultados de Amiri y Salari (2019), sin embargo, el valor objetivo (en este caso óptimo) coincide con el encontrado en este trabajo por CPLEX ó MIPStart y con el obtenido por Sinnl (2019), pero no ocurre esto mismo para todas las instancias con este tipo de inconsistencia. Por esta razón, se le asigna una marca con un asterisco (*) aquellas instancias que presentan esta misma inconsistencia y no se calcula el gap correspondiente, debido a que no se tiene claro como realizar una comparación que sea válida y se desconoce la razón de estas diferencias.

Tabla A.4: Ejemplo de inconsistencia en el tiempo límite de los vehículos.

Instancia	Características				Tiempo límite			Valor objetivo		
	Nodos	Instalaciones	Clientes	Vehículos	Amiry y Salari (2019)	Sinnl (2019)	Archivo de la instancia	Amiry y Salari (2019)	Sinnl (2019)	Este trabajo
Input5-1-0-5*	76	30	45	3	10583.61	10538.61	10538.61	13	13	13

2. Diferencia de óptimos. En la Tabla A.5 se muestra un ejemplo de instancia en la que las características y el tiempo límite en el archivo de la instancia coincide exactamente con lo expuesto en las tablas de resultados de Amiri y Salari (2019), sin embargo, se reportan óptimos diferentes. Para estas instancias cuando Sinnl (2019) alcanza el óptimo, dicho valor coincide con el encontrado en este trabajo por CPLEX o MIPStart.

Tabla A.5: Ejemplo de inconsistencia en valor óptimo reportado.

Instancia	Características				Tiempo límite			Valor objetivo		
	Nodos	Instalaciones	Clientes	Vehículos	Amiri y Salari (2019)	Sinnl (2019)	Archivo de la instancia	Amiri y Salari (2019)	Sinnl (2019)	Este trabajo
Input10-0-0-3**	100	29	70	2	4955.06	4955.06	4955.06	36	30	30

Tabla A.6: Detalles de resultados obtenidos por Amiri y Salari (2019) y Sinnl (2019) para el conjunto de instancias pequeñas.

Instancia	Amiri y Salari (2019)		Sinnl (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
Input2-0-0-1	15	15	15	15	0	0	0
Input2-0-0-2	12	12	12	12	0	0	0
Input2-0-0-4	11	11	11	11	0	0	0
Input2-0-0-5	9	9	9	9	0	0	0
Input2-0-0-6	8	8	8	8	0	0	0
Input2-0-0-7	8	8	8	8	0	0	0
Input2-0-0-8	8	8	8	8	0	0	0
Input2-0-0-9	1	1	1	1	0	0	0
Input2-1-0-1	15	15	15	15	0	0	0
Input2-1-0-2	15	15	15	15	0	0	0
Input2-1-0-3	14	14	14	14	0	0	0
Input2-1-0-4	13	13	13	13	0	0	0
Input2-1-0-5	13	13	13	13	0	0	0
Input2-1-0-6	12	12	12	12	0	0	0
Input2-1-0-7	10	10	10	10	0	0	0
Input2-1-0-8	9	9	9	9	0	0	0
Input2-1-0-9	9	9	9	9	0	0	0
Input2-2-0-1	20	20	20	20	0	0	0
Input2-2-0-2	19	19	19	19	0	0	0
Input2-2-0-3	18	18	18	18	0	0	0
Input2-2-0-4	17	17	17	17	0	0	0
Input2-2-0-5	15	15	15	15	0	0	0
Input2-2-0-6	15	15	15	15	0	0	0
Input2-2-0-7**	15	14	15	15	-	0	0
Input2-2-0-8**	14	11	14	14	-	0	0
Input2-2-0-9**	11	12	11	11	-	0	0
Input5-0-0-1	12	12	12	12	0	0	0
Input5-0-0-2	12	12	12	12	0	0	0
Input5-0-0-3	10	10	10	10	0	0	0
Input5-0-0-4	10	10	10	10	0	0	0
Input5-0-0-5	7	7	7	7	0	0	0
Input5-0-0-6	7	7	7	7	0	0	0
Input5-0-0-7	5	5	5	5	0	0	0

Tabla A.6: Continuación.

Instancia	Amiri y Salari (2019)		Sinnl (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
Input5-0-0-8	5	5	5	5	0	0	0
Input5-0-0-9*	5	5	5	5	-	0	0
Input5-1-0-1	16	16	16	16	0	0	0
Input5-1-0-2	16	16	16	16	0	0	0
Input5-1-0-3	13	13	13	13	0	0	0
Input5-1-0-4	13	13	13	13	0	0	0
Input5-1-0-5*	13	13	13	13	-	0	0
Input5-1-0-6	12	12	12	12	0	0	0
Input5-1-0-7	12	12	12	12	0	0	0
Input5-1-0-8	12	12	12	12	0	0	0
Input5-1-0-9	7	7	7	7	0	0	0
Input5-2-0-1	17	17	17	17	0	0	0
Input5-2-0-2	17	17	17	17	0	0	0
Input5-2-0-3	15	15	15	15	0	0	0
Input5-2-0-4	14	14	14	14	0	0	0
Input5-2-0-5	14	14	14	14	0	0	0
Input5-2-0-6*	12	12	12	12	-	0	0
Input5-2-0-7	12	12	12	12	0	0	0
Input5-2-0-8	11	11	11	11	0	0	0
Input5-2-0-9	9	9	9	9	0	0	0
Input10-0-0-1	36	36	36	36	0	0	0
Input10-0-0-2**	34	36	34	34	-	0	0
Input10-0-0-3**	30	36	30	30	-	0	0
Input10-0-0-4**	27	36	27	27	-	0	0
Input10-0-0-5**	24	36	24	24	-	0	0
Input10-0-0-6	21	21	21	21	0	0	0
Input10-0-0-7	21	21	21	21	0	0	0
Input10-0-0-8	16	16	16	16	0	0	0
Input10-0-0-9	14	14	14	14	0	0	0
Input10-1-0-1	51	51	51	51	0	0	0
Input10-1-0-2	51	51	51	51	0	0	0
Input10-1-0-3	49	49	49	49	0	0	0
Input10-1-0-4	48	48	48	48	0	0	0

Tabla A.6: Continuación.

Instancia	Amiri y Salari (2019)		Sinnl (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
Input10-1-0-5	40	40	40	40	0	0	0
Input10-1-0-6	38	38	38	38	0	0	0
Input10-1-0-7	34	34	34	34	0	0	0
Input10-1-0-8	32	32	32	32	0	0	0
Input10-1-0-9	27	27	27	27	0	0	0
Input10-2-0-1	45	45	45	45	0	0	0
Input10-2-0-2	45	45	45	45	0	0	0
Input10-2-0-3	45	45	45	45	0	0	0
Input10-2-0-4	45	45	45	45	0	0	0
Input10-2-0-5	45	45	45	45	0	0	0
Input10-2-0-6	42	42	42	42	0	0	0
Input10-2-0-7	42	42	42	42	0	0	0
Input10-2-0-8	37	37	37	37	0	0	0
Input10-2-0-9	33	33	33	33	0	0	0

Tabla A.7: Detalles de resultados obtenidos por Amiri y Salari (2019) y Sinnl (2019) para el conjunto de instancias medianas.

Instancia	Amiri y Salari (2019) Sinnl (2019)				Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
Input14-0-0-1	72	72	72	72	0	0	0
Input14-0-0-2	72	72	72	72	0	0	0
Input14-0-0-3	70	70	70	70	0	0	0
Input14-0-0-4	72	69	72	72	-4.35	0	0
Input14-0-0-5	69	69	69	69	0	0	0
Input14-0-0-6	64	64	64	64	0	0	0
Input14-0-0-7	66	66	66	66	0	0	0
Input14-0-0-8	58	58	58	58	0	0	0
Input14-0-0-9	49	49	49	49	0	0	0
Input14-1-0-1	71	71	71	71	0	0	0
Input14-1-0-2	71	71	71	71	0	0	0
Input14-1-0-3	71	71	71	71	0	0	0
Input14-1-0-4	71	70	71	71	-1.43	0	0
Input14-1-0-5	71	71	71	71	0	0	0
Input14-1-0-6	71	69	71	71	-2.90	0	0
Input14-1-0-7	71	71	71	71	0	0	0
Input14-1-0-8	71	71	71	71	0	0	0
Input14-1-0-9	65	65	65	65	0	0	0
Input14-2-0-1	64	64	64	64	0	0	0
Input14-2-0-2	64	64	64	64	0	0	0
Input14-2-0-3	64	64	64	64	0	0	0
Input14-2-0-4	64	64	64	64	0	0	0
Input14-2-0-5	64	64	64	64	0	0	0
Input14-2-0-6*	64	63	64	64	-	0	0
Input14-2-0-7	64	64	64	64	0	0	0
Input14-2-0-8	64	64	64	64	0	0	0
Input14-2-0-9	64	64	64	64	0	0	0
Input16-0-0-1	104	104	104	104	0	0	0
Input16-0-0-2	104	104	104	104	0	0	0
Input16-0-0-3	104	102	102	104	-1.96	-1.96	0
Input16-0-0-4	104	92	98	104	-13.04	-6.12	0
Input16-0-0-5	104	104	102	104	0	-1.96	0
Input16-0-0-6	100	101	99	101	0.99	-1.01	0.99

Tabla A.7: Continuación.

Instancia	Amiri y Salari (2019)		Sinnl (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
Input16-0-0-7	99	100	95	89	1	-4.21	-11.24
Input16-0-0-8	90	88	89	89	-2.27	-1.12	-1.12
Input16-1-0-1	107	107	107	107	0	0	0
Input16-1-0-2	107	107	107	107	0	0	0
Input16-1-0-3	107	107	107	107	0	0	0
Input16-1-0-4	107	107	107	107	0	0	0
Input16-1-0-5	107	103	107	107	-3.88	0	0
Input16-1-0-6	107	100	107	107	-7	0	0
Input16-1-0-7	107	104	93	107	-2.88	-15.05	0
Input16-1-0-8	107	103	103	-	-3.88	-3.88	-
Input16-1-0-9	106	95	105	-	-11.58	-0.95	-
Input16-2-0-1	94	94	94	94	0	0	0
Input16-2-0-2	94	94	94	94	0	0	0
Input16-2-0-3	94	94	94	94	0	0	0
Input16-2-0-4	94	94	94	94	0	0	0
Input16-2-0-5	94	94	94	94	0	0	0
Input16-2-0-6	94	94	82	94	0	-14.63	0
Input16-2-0-7	94	94	94	94	0	0	0
Input16-2-0-8	94	94	94	94	0	0	0
Input16-2-0-9*	94	94	88	94	-	-6.82	0

Tabla A.8: Detalles de resultados obtenidos por Amiri y Salari (2019) y Sinml (2019) para el conjunto de instancias grandes.

Instancia	Amiri y Salari (2019)		Sinml (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
G-Input2-0-0-1	58	58	58	58	0	0	0
G-Input2-0-0-2	54	54	54	54	0	0	0
G-Input2-0-0-3	58	-	58	58	-	0	0
G-Input2-0-0-4	58	58	58	58	0	0	0
G-Input2-0-0-5	58	-	58	58	-	0	0
G-Input2-0-0-6	57	-	57	57	-	0	0
G-Input2-0-0-7	57	-	57	57	-	0	0
G-Input2-0-0-8	59	-	59	59	-	0	0
G-Input2-0-0-9	60	-	60	60	-	0	0
G-Input2-1-0-1	63	62	63	63	-1.61	0	0
G-Input2-1-0-2	64	64	64	64	0	0	0
G-Input2-1-0-3	65	-	65	65	-	0	0
G-Input2-1-0-4	63	63	63	63	0	0	0
G-Input2-1-0-5	61	-	61	61	-	0	0
G-Input2-1-0-6	65	-	65	65	-	0	0
G-Input2-1-0-7	58	-	58	58	-	0	0
G-Input2-1-0-8	62	-	62	62	-	0	0
G-Input2-1-0-9	63	-	63	63	-	0	0
G-Input2-2-0-1	75	75	75	75	0	0	0
G-Input2-2-0-2	72	72	72	72	0	0	0
G-Input2-2-0-3	75	58	75	75	-29.31	0	0
G-Input2-2-0-4	73	73	73	73	0	0	0
G-Input2-2-0-5	73	73	73	73	0	0	0
G-Input2-2-0-6	71	-	71	71	-	0	0
G-Input2-2-0-7	75	75	75	75	0	0	0
G-Input2-2-0-8	72	-	72	72	-	0	0
G-Input2-2-0-9*	67	-	67	67	-	0	0
G-Input5-0-0-1*	40	40	40	40	-	0	0
G-Input5-0-0-2*	37	37	37	37	-	0	0
G-Input5-0-0-3*	37	-	37	37	-	0	0
G-Input5-0-0-4*	41	41	41	41	-	0	0
G-Input5-0-0-5*	42	40	42	42	-	0	0
G-Input5-0-0-6*	39	-	39	39	-	0	0

Tabla A.8: Continuación.

Instancia	Amiri y Salari (2019)		Sinml (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
G-Input5-0-0-7*	40	-	40	40	-	0	0
G-Input5-0-0-8*	39	-	39	39	-	0	0
G-Input5-0-0-9*	39	-	39	39	-	0	0
G-Input5-1-0-1*	36	36	36	36	-	0	0
G-Input5-1-0-2*	41	41	41	41	-	0	0
G-Input5-1-0-3*	37	33	37	37	-	0	0
G-Input5-1-0-4*	39	39	39	39	-	0	0
G-Input5-1-0-5*	43	43	43	43	-	0	0
G-Input5-1-0-6*	44	-	44	44	-	0	0
G-Input5-1-0-7*	40	40	40	40	-	0	0
G-Input5-1-0-8*	40	-	40	40	-	0	0
G-Input5-1-0-9*	38	-	38	38	-	0	0
G-Input5-2-0-1*	43	43	43	43	-	0	0
G-Input5-2-0-2*	43	38	43	43	-	0	0
G-Input5-2-0-3*	36	43	36	36	-	0	0
G-Input5-2-0-4*	40	41	40	40	-	0	0
G-Input5-2-0-5*	39	37	39	39	-	0	0
G-Input5-2-0-6*	40	37	40	40	-	0	0
G-Input5-2-0-7*	43	41	43	43	-	0	0
G-Input5-2-0-8*	39	40	39	39	-	0	0
G-Input5-2-0-9*	41	-	41	41	-	0	0
G-Input10-0-0-1	27	-	27	27	-	0	0
G-Input10-0-0-2	29	-	29	29	-	0	0
G-Input10-0-0-3	28	-	28	28	-	0	0
G-Input10-0-0-4	30	-	30	30	-	0	0
G-Input10-0-0-5	31	-	31	31	-	0	0
G-Input10-0-0-6	30	-	30	30	-	0	0
G-Input10-0-0-7	30	-	30	30	-	0	0
G-Input10-0-0-8	30	-	24	30	-	-25	0
G-Input10-0-0-9	30	-	30	30	-	0	0
G-Input10-1-0-1	27	27	27	27	0	0	0
G-Input10-1-0-2	27	-	27	27	-	0	0
G-Input10-1-0-3	30	-	30	30	-	0	0

Tabla A.8: Continuación.

Instancia	Amiri y Salari (2019)		Sinml (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
G-Input10-1-0-4	28	-	28	28	-	0	0
G-Input10-1-0-5	29	-	29	29	-	0	0
G-Input10-1-0-6	27	-	27	27	-	0	0
G-Input10-1-0-7	26	-	26	26	-	0	0
G-Input10-1-0-8	30	-	30	30	-	0	0
G-Input10-1-0-9	29	-	29	29	-	0	0
G-Input10-2-0-1	27	27	27	27	0	0	0
G-Input10-2-0-2	26	26	26	26	0	0	0
G-Input10-2-0-3	30	-	30	30	-	0	0
G-Input10-2-0-4	30	30	30	30	0	0	0
G-Input10-2-0-5	27	-	27	27	-	0	0
G-Input10-2-0-6	28	-	28	28	-	0	0
G-Input10-2-0-7	26	19	26	26	-36.84	0	0
G-Input10-2-0-8	29	-	29	29	-	0	0
G-Input10-2-0-9	28	-	28	28	-	0	0
G-Input14-0-0-1	83	-	83	83	-	0	0
G-Input14-0-0-2	84	-	-	84	-	-	0
G-Input14-0-0-3	83	-	-	83	-	-	0
G-Input14-0-0-4	82	-	-	82	-	-	0
G-Input14-0-0-5	83	-	-	83	-	-	0
G-Input14-0-0-6	80	-	-	80	-	-	0
G-Input14-0-0-7	78	-	78	78	-	-	0
G-Input14-0-0-8	82	-	82	82	-	-	0
G-Input14-0-0-9	81	-	81	81	-	-	0
G-Input14-1-0-1	76	76	76	76	0	0	0
G-Input14-1-0-2	76	-	64	76	-	-	0
G-Input14-1-0-3	78	-	-	78	-	-	0
G-Input14-1-0-4	77	-	77	77	-	0	0
G-Input14-1-0-5	78	-	78	78	-	0	0
G-Input14-1-0-6	77	-	63	77	-	-	0
G-Input14-1-0-7	76	-	76	76	-	0	0
G-Input14-1-0-8	73	-	73	73	-	0	0
G-Input14-1-0-9	81	-	81	81	-	0	0

Tabla A.8: Continuación.

Instancia	Amiri y Salari (2019)		Sinnl (2019)		Gap (%)		
	Z_B	Z_{AS}	Z_{C+}	Z_{E++}	Z_B vs Z_{AS}	Z_B vs Z_{C+}	Z_B vs Z_{E++}
G-Input14-2-0-1	63	63	63	63	0	0	0
G-Input14-2-0-2	67	-	67	67	-	0	0
G-Input14-2-0-3	68	-	-	68	-	-	0
G-Input14-2-0-4	65	65	65	65	0	0	0
G-Input14-2-0-5	67	-	67	67	-	0	0
G-Input14-2-0-6	68	-	68	68	-	0	0
G-Input14-2-0-7	68	-	68	68	-	0	0
G-Input14-2-0-8	66	-	66	66	-	0	0
G-Input14-2-0-9	67	-	67	67	-	0	0
G-Input16-0-0-1	205	-	-	205	-	-	0
G-Input16-0-0-2	200	-	-	200	-	-	0
G-Input16-0-0-3	214	-	-	214	-	-	0
G-Input16-0-0-4	195	-	-	195	-	-	0
G-Input16-0-0-5	207	-	-	207	-	-	0
G-Input16-0-0-6	204	-	-	204	-	-	0
G-Input16-0-0-7*	208	-	-	208	-	-	0
G-Input16-0-0-8	206	-	-	206	-	-	0
G-Input16-0-0-9	216	-	-	216	-	-	0
G-Input16-1-0-1	187	-	-	187	-	-	0
G-Input16-1-0-2	183	-	-	183	-	-	0
G-Input16-1-0-3	175	-	-	175	-	-	0
G-Input16-1-0-4	184	-	145	184	-	-26.90	0
G-Input16-1-0-5	176	-	-	176	-	-	0
G-Input16-1-0-6	183	-	-	183	-	-	0
G-Input16-1-0-7	177	-	-	177	-	-	0
G-Input16-1-0-8	182	-	-	182	-	-	0
G-Input16-1-0-9	174	-	-	174	-	-	0
G-Input16-2-0-1	176	-	90	176	-	-95.56	0
G-Input16-2-0-2	187	-	-	187	-	-	0
G-Input16-2-0-3	187	-	-	187	-	-	0
G-Input16-2-0-4	179	-	154	179	-	-16.23	0
G-Input16-2-0-5	179	-	133	179	-	-34.59	0
G-Input16-2-0-6	184	-	122	184	-	-50.82	0
G-Input16-2-0-7	180	-	178	180	-	-1.12	0
G-Input16-2-0-8	183	-	165	183	-	-10.91	0
G-Input16-2-0-9	176	-	97	176	-	-81.44	0

A.3 COMPARACIÓN DEL ALNS CON LOS ALGORITMOS HEURÍSTICOS PROPUESTOS POR AMIRI Y SALARI (2019)

En la Tabla A.9 se muestra el promedio de la función objetivo para cada instancia, obtenido al ejecutar cada algoritmo cinco veces.

En la primer columna se encuentra el nombre de la instancia, en la segunda y tercera columna se reporta la función objetivo promedio y el Gap promedio (%), ambos correspondientes al algoritmo ALNS. Análogamente, en el resto de las columnas se encuentran los resultados obtenidos por Amiri y Salari (2019) de los algoritmos ILS, VNS y TS. El Gap promedio (%) de cada uno de los algoritmos se calcula con respecto al óptimo, de acuerdo a la ecuación (A.4)

$$Gap (\%) = 100 \times \frac{Z_{MS} - \bar{Z}_x}{Z_{MS}}, \quad (A.4)$$

Donde Z_{MS} representa el valor óptimo encontrado utilizando MIPStart, mientras que \bar{Z}_x es el promedio de las funciones objetivo obtenidas en las 5 ejecuciones del ALNS, ILS, VNS y TS, respectivamente. Cabe recalcar que, los valores promedio encontrados con ILS, VNS y TS se tomaron de las tablas de resultados expuestas por Amiri y Salari (2019).

Para cada instancia se resalta en **negritas** el valor de la función objetivo promedio de los algoritmos que alcanzan el óptimo en las cinco ejecuciones y se identifican las instancias que presentan inconsistencias, tal y como se explicó en el Apéndice A.2.

Tabla A.9: Comparación de los algoritmos heurísticos propuestos para el TCMCRP en el conjunto de instancias grandes.

Instancia	ALNS		ILS		VNS		TS	
	\bar{Z}_{ALNS}	Gap (%)	\bar{Z}_{ILS}	Gap (%)	\bar{Z}_{VNS}	Gap (%)	\bar{Z}_{TS}	Gap (%)
G-Input2-0-0-1	58	0	58	0	58	0	58	0
G-Input2-0-0-2	54	0	54	0	54	0	54	0
G-Input2-0-0-3	58	0	58	0	58	0	58	0
G-Input2-0-0-4	58	0	58	0	57	1.72	58	0
G-Input2-0-0-5	58	0	58	0	58	0	58	0
G-Input2-0-0-6	57	0	56.8	0.35	56.6	0.70	57	0
G-Input2-0-0-7	57	0	57	0	57	0	57	0
G-Input2-0-0-8	59	0	58	1.69	58	1.69	59	0
G-Input2-0-0-9	60	0	58.6	2.33	58.8	2	60	0
G-Input2-1-0-1	63	0	63	0	63	0	63	0
G-Input2-1-0-2	64	0	64	0	64	0	64	0
G-Input2-1-0-3	65	0	65	0	65	0	65	0
G-Input2-1-0-4	63	0	63	0	63	0	63	0
G-Input2-1-0-5	61	0	61	0	61	0	61	0
G-Input2-1-0-6	65	0	65	0	65	0	65	0
G-Input2-1-0-7	58	0	58	0	58	0	58	0
G-Input2-1-0-8	62	0	61.5	0.81	62	0	62	0
G-Input2-1-0-9	63	0	63	0	63	0	63	0
G-Input2-2-0-1	75	0	75	0	75	0	75	0
G-Input2-2-0-2	72	0	72	0	72	0	72	0
G-Input2-2-0-3	75	0	71	5.33	71	5.33	75	0
G-Input2-2-0-4	73	0	73	0	73	0	73	0
G-Input2-2-0-5	73	0	73	0	73	0	73	0
G-Input2-2-0-6	71	0	71	0	71	0	71	0
G-Input2-2-0-7	75	0	75	0	75	0	75	0
G-Input2-2-0-8	72	0	72	0	72	0	72	0
G-Input2-2-0-9*	67	0	67	0	67	0	67	0
G-Input5-0-0-1*	40	0	40	0	40	0	40	0
G-Input5-0-0-2*	37	0	37	0	37	0	37	0
G-Input5-0-0-3	37	0	37	0	37	0	37	0
G-Input5-0-0-4*	41	0	41	0	41	0	41	0
G-Input5-0-0-5*	42	0	40	4.76	40.6	3.33	42	0
G-Input5-0-0-6*	39	0	39	0	39	0	39	0

Tabla A.9: Continuación.

Instancia	ALNS		ILS		VNS		TS	
	\bar{Z}_{ALNS}	Gap (%)	\bar{Z}_{ILS}	Gap (%)	\bar{Z}_{VNS}	Gap (%)	\bar{Z}_{TS}	Gap (%)
G-Input5-0-0-7*	40	0	40	0	40	0	40	0
G-Input5-0-0-8*	39	0	39	0	39	0	39	0
G-Input5-0-0-9*	39	0	39	0	39	0	39	0
G-Input5-1-0-1*	36	0	36	0	36	0	36	0
G-Input5-1-0-2*	41	0	41	0	41	0	41	0
G-Input5-1-0-3*	37	0	36	2.70	37	0	37	0
G-Input5-1-0-4*	39	0	39	0	39	0	39	0
G-Input5-1-0-5*	43	0	43	0	43	0	43	0
G-Input5-1-0-6*	44	0	44	0	44	0	44	0
G-Input5-1-0-7*	40	0	40	0	40	0	40	0
G-Input5-1-0-8*	40	0	40	0	40	0	40	0
G-Input5-1-0-9*	38	0	38	0	38	0	38	0
G-Input5-2-0-1*	43	0	43	0	43	0	43	0
G-Input5-2-0-2*	43	0	43	0	43	0	43	0
G-Input5-2-0-3*	36	0	36	0	36	0	36	0
G-Input5-2-0-4*	40	0	40	0	39	2.5	41	-2.5
G-Input5-2-0-5*	39	0	39	0	39	0	37.8	3.08
G-Input5-2-0-6*	40	0	40	0	37	7.5	40	0
G-Input5-2-0-7*	43	0	43	0	43	0	41.8	2.79
G-Input5-2-0-8*	39	0	39	0	39	0	40	-2.56
G-Input5-2-0-9*	41	0	40	2.44	39	4.88	41	0
G-Input10-0-0-1	27	0	27	0	27	0	27	0
G-Input10-0-0-2	29	0	29	0	29	0	29	0
G-Input10-0-0-3	28	0	28	0	28	0	28	0
G-Input10-0-0-4	30	0	30	0	30	0	30	0
G-Input10-0-0-5	31	0	31	0	31	0	31	0
G-Input10-0-0-6	30	0	28.5	5	30	0	30	0
G-Input10-0-0-7	30	0	30	0	30	0	30	0
G-Input10-0-0-8	30	0	30	0	30	0	30	0
G-Input10-0-0-9	30	0	30	0	30	0	30	0
G-Input10-1-0-1	27	0	27	0	27	0	27	0
G-Input10-1-0-2	27	0	27	0	27	0	27	0
G-Input10-1-0-3	30	0	30	0	30	0	30	0

Tabla A.9: Continuación.

Instancia	ALNS		ILS		VNS		TS	
	\bar{Z}_{ALNS}	Gap (%)	\bar{Z}_{ILS}	Gap (%)	\bar{Z}_{VNS}	Gap (%)	\bar{Z}_{TS}	Gap (%)
G-Input10-1-0-4	28	0	28	0	28	0	28	0
G-Input10-1-0-5	29	0	29	0	29	0	29	0
G-Input10-1-0-6	27	0	27	0	25.5	5.56	27	0
G-Input10-1-0-7	26	0	26	0	26	0	26	0
G-Input10-1-0-8	30	0	30	0	30	0	30	0
G-Input10-1-0-9	29	0	29	0	29	0	29	0
G-Input10-2-0-1	27	0	27	0	27	0	27	0
G-Input10-2-0-2	26	0	26	0	26	0	26	0
G-Input10-2-0-3	30	0	30	0	30	0	30	0
G-Input10-2-0-4	30	0	30	0	30	0	30	0
G-Input10-2-0-5	27	0	27	0	27	0	27	0
G-Input10-2-0-6	28	0	28	0	28	0	28	0
G-Input10-2-0-7	26	0	21.5	17.31	22	15.38	26	0
G-Input10-2-0-8	29	0	29	0	29	0	29	0
G-Input10-2-0-9	28	0	28	0	28	0	28	0
G-Input14-0-0-1	83	0	82	1.20	82	1.20	83	0
G-Input14-0-0-2	84	0	84	0	84	0	84	0
G-Input14-0-0-3	83	0	83	0	83	0	83	0
G-Input14-0-0-4	82	0	82	0	82	0	82	0
G-Input14-0-0-5	83	0	83	0	81	2.41	83	0
G-Input14-0-0-6	80	0	80	0	80	0	80	0
G-Input14-0-0-7	78	0	78	0	78	0	78	0
G-Input14-0-0-8	82	0	82	0	82	0	82	0
G-Input14-0-0-9	81	0	81	0	81	0	81	0
G-Input14-1-0-1	76	0	76	0	76	0	76	0
G-Input14-1-0-2	76	0	76	0	76	0	76	0
G-Input14-1-0-3	78	0	78	0	78	0	78	0
G-Input14-1-0-4	77	0	77	0	77	0	77	0
G-Input14-1-0-5	78	0	78	0	77	1.28	78	0
G-Input14-1-0-6	77	0	77	0	77	0	77	0
G-Input14-1-0-7	76	0	76	0	76	0	76	0
G-Input14-1-0-8	73	0	73	0	73	0	73	0
G-Input14-1-0-9	81	0	81	0	81	0	81	0

Tabla A.9: Continuación.

Instancia	ALNS		ILS		VNS		TS	
	\bar{Z}_{ALNS}	Gap (%)	\bar{Z}_{ILS}	Gap (%)	\bar{Z}_{VNS}	Gap (%)	\bar{Z}_{TS}	Gap (%)
G-Input14-2-0-1	63	0	63	0	63	0	63	0
G-Input14-2-0-2	67	0	67	0	67	0	67	0
G-Input14-2-0-3	68	0	68	0	68	0	68	0
G-Input14-2-0-4	65	0	65	0	65	0	65	0
G-Input14-2-0-5	67	0	67	0	67	0	67	0
G-Input14-2-0-6	68	0	68	0	68	0	68	0
G-Input14-2-0-7	68	0	68	0	68	0	68	0
G-Input14-2-0-8	66	0	66	0	66	0	66	0
G-Input14-2-0-9	67	0	67	0	67	0	67	0
G-Input16-0-0-1	205	0	204	0.49	205	0	205	0
G-Input16-0-0-2	200	0	200	0	200	0	200	0
G-Input16-0-0-3	214	0	212.5	0.70	211	1.40	214	0
G-Input16-0-0-4	195	0	195	0	195	0	195	0
G-Input16-0-0-5	207	0	207	0	207	0	207	0
G-Input16-0-0-6	204	0	204	0	204	0	204	0
G-Input16-0-0-7*	208	0	206	0.96	206	0.96	208	0
G-Input16-0-0-8	206	0	206	0	206	0	206	0
G-Input16-0-0-9	216	0	212.5	1.62	212.5	1.62	216	0
G-Input16-1-0-1	187	0	187	0	187	0	187	0
G-Input16-1-0-2	183	0	181.4	0.87	181.5	0.82	183	0
G-Input16-1-0-3	175	0	175	0	175	0	175	0
G-Input16-1-0-4	184	0	182.6	0.76	183.4	0.33	184	0
G-Input16-1-0-5	176	0	176	0	176	0	176	0
G-Input16-1-0-6	183	0	183	0	181.5	0.82	183	0
G-Input16-1-0-7	177	0	177	0	176.6	0.23	177	0
G-Input16-1-0-8	182	0	182	0	182	0	182	0
G-Input16-1-0-9	174	0	174	0	174	0	174	0
G-Input16-2-0-1	176	0	176	0	175	0.57	176	0
G-Input16-2-0-2	187	0	187	0	187	0	187	0
G-Input16-2-0-3	187	0	183	2.14	187	0	187	0
G-Input16-2-0-4	179	0	179	0	179	0	179	0
G-Input16-2-0-5	179	0	179	0	179	0	179	0
G-Input16-2-0-6	184	0	180	2.17	181	1.63	184	0
G-Input16-2-0-7	180	0	180	0	180	0	180	0
G-Input16-2-0-8	183	0	183	0	183	0	183	0
G-Input16-2-0-9	176	0	171.5	2.56	172	2.27	176	0

BIBLIOGRAFÍA

- AMIRI, A. y M. SALARI (2019), «Time-constrained maximal covering routing problem», *OR Spectrum*, **41**(2), págs. 415–468, <https://doi.org/10.1007/s00291-018-0541-3>.
- ANGEL, R. D., W. L. CAUDLE, R. NOONAN y A. WHINSTON (1972), «Computer-assisted school bus scheduling», *Management Science*, **18**(6), págs. B-279–B-288, <https://doi.org/10.1287/mnsc.18.6.B279>.
- APPLEGATE, D. L., R. E. BIXBY, V. CHVÁTAL y W. J. COOK (2006), *The Traveling Salesman Problem: A Computational Study*, Princeton Series in Applied Mathematics, Princeton University Press, ISBN: 9780691129938, DOI: 10.2307/j.ctt7s8xg, <http://www.jstor.org/stable/j.ctt7s8xg>.
- ARCHETTI, C., D. FEILLET, A. HERTZ y M. G. SPERANZA (2009), «The capacitated team orienteering and profitable tour problems», *Journal of the Operational Research Society*, **60**(6), págs. 831–842, <https://doi.org/10.1057/palgrave.jors.2602603>.
- ARKIN, E. M., J. S. MITCHELL y G. NARASIMHAN (1998), «Resource-constrained geometric network optimization», en *Proceedings of the Fourteenth Annual Symposium on Computational Geometry*, SCG '98, Citeseer, Association for Computing Machinery, New York, NY, USA, págs. 307–316, ISBN: 0897919734, <https://doi.org/10.1145/276884.276919>.
- BEKTAS, T. (2006), «The multiple traveling salesman problem: an overview of

- formulations and solution procedures», *Omega*, **34**(3), págs. 209–219, <https://doi.org/10.1016/j.omega.2004.10.004>.
- BOCK, A. y L. SANITÀ (2015), «The capacitated orienteering problem», *Discrete Applied Mathematics*, **195**, págs. 31–42, 12th Cologne-Twente Workshop on Graphs and Combinatorial Optimization (CTW 2013), <https://doi.org/10.1016/j.dam.2014.10.001>.
- BUTT, S. E. y T. M. CAVALIER (1994), «A heuristic for the multiple tour maximum collection problem», *Computers & Operations Research*, **21**(1), págs. 101–111, [https://doi.org/10.1016/0305-0548\(94\)90065-5](https://doi.org/10.1016/0305-0548(94)90065-5).
- CARTER, A. E. y C. T. RAGSDALE (2002), «Scheduling pre-printed newspaper advertising inserts using genetic algorithms», *Omega*, **30**(6), págs. 415–421, [https://doi.org/10.1016/S0305-0483\(02\)00059-2](https://doi.org/10.1016/S0305-0483(02)00059-2).
- CHAO, I.-M., B. L. GOLDEN y E. A. WASIL (1996), «The team orienteering problem», *European Journal of Operational Research*, **88**(3), págs. 464–474, [https://doi.org/10.1016/0377-2217\(94\)00289-4](https://doi.org/10.1016/0377-2217(94)00289-4).
- CHURCH, R. y C. R. VELLE (1974), «The maximal covering location problem», *Papers in Regional Science*, **32**(1), págs. 101–118, springer-Verlag, http://www.geog.ucsb.edu/~forest/G294download/MAX_COVER_RLC_CSR.pdf.
- CLARKE, G. y J. W. WRIGHT (1964), «Scheduling of vehicles from a central depot to a number of delivery points», *Operations Research*, **12**(4), págs. 568–581, <https://doi.org/10.1287/opre.12.4.568>.
- CURRENT, J. R. y D. A. SCHILLING (1989), «The covering salesman problem», *Transportation science*, **23**(3), págs. 208–213, <https://doi.org/10.1287/trsc.23.3.208>.
- CURRENT, J. R. y D. A. SCHILLING (1994), «The median tour and maximal covering tour problems: Formulations and heuristics», *European Journal of Operatio-*

- nal Research*, **73**(1), págs. 114–126, [https://doi.org/10.1016/0377-2217\(94\)90149-X](https://doi.org/10.1016/0377-2217(94)90149-X).
- DANTZIG, G. B. y J. H. RAMSER (1959), «The truck dispatching problem», *Management Science*, **6**(1), págs. 80–91, <https://doi.org/10.1287/mnsc.6.1.80>.
- DAVENDRA, D. (2010), *Traveling Salesman Problem: Theory and Applications*, ISBN: 9789533074269, <https://doi.org/10.5772/547>.
- FLOOD, M. M. (1956), «The traveling-salesman problem», *Operations Research*, **4**(1), págs. 61–75, <https://doi.org/10.1287/opre.4.1.61>.
- FLORES-GARZA, D. A., M. A. SALAZAR-AGUILAR, S. U. NGUEVEU y G. LAPORTE (2017), «The multi-vehicle cumulative covering tour problem», *Annals of Operations Research*, **258**(2), págs. 761–780, <https://doi.org/10.1007/s10479-015-2062-7>.
- FRANCIS, R. L., J. A. WHITE y L. F. MCGINNIS (1992), *Facility Layout and Location: An Analytical Approach*, *Prentice-Hall international series in industrial and systems engineering*, tomo 31, segunda edición, Prentice Hall, ISBN: 9780132992312, <https://lib.ugent.be/catalog/rug01:001942828>.
- GARFINKEL, R. S. (1977), «Minimizing wallpaper waste, part 1: a class of traveling salesman problems», *Operations Research*, **25**(5), págs. 741–751, <https://doi.org/10.1287/opre.25.5.741>.
- GENDREAU, M., G. LAPORTE y F. SEMET (1997), «The covering tour problem», *Operations Research*, **45**(4), págs. 568–576, <https://doi.org/10.1287/opre.45.4.568>.
- GILMORE, P. C. y R. E. GOMORY (1964), «Sequencing a one state-variable machine: A solvable case of the traveling salesman problem», *Operations Research*, **12**(5), págs. 655–679, <https://doi.org/10.1287/opre.12.5.655>.

- GOLDEN, B., Z. NAJI-AZIMI, S. RAGHAVAN, M. SALARI y P. TOTH (2012), «The generalized covering salesman problem», *INFORMS Journal on Computing*, **24**(4), págs. 534–553, <https://doi.org/10.1287/ijoc.1110.0480>.
- GOLDEN, B. L., L. LEVY y R. VOHRA (1987), «The orienteering problem», *Naval Research Logistics (NRL)*, **34**(3), págs. 307–318, DOI: 10.1002/1520-6750(198706)34:3<307::AID-NAV3220340302>3.0.CO;2-D, <https://onlinelibrary.wiley.com/doi/abs/10.1002/1520-6750%28198706%2934%3A3%3C307%3A%3AAID-NAV3220340302%3E3.0.CO%3B2-D>.
- GONZÁLEZ VELARDE, J. L. y R. Z. RÍOS MERCADO (1999), «Investigación de operaciones en acción: Aplicación del TSP en problemas de manufactura y logística», *Ingenierías*, **2**(4), págs. 18–23, http://eprints.uanl.mx/9986/1/4_Jose_L_Gonzalez_investigacion_de_operaciones.pdf.
- GORENSTEIN, S. (1970), «Printing press scheduling for multi-edition periodicals», *Management Science*, **16**(6), págs. B-373–B-383, <https://doi.org/10.1287/mnsc.16.6.B373>.
- GRÖTSCHER, M. y O. HOLLAND (1991), «Solution of large-scale symmetric travelling salesman problems», *Mathematical Programming*, **51**(1-3), págs. 141–202, <https://doi.org/10.1007/BF01586932>.
- GUNAWAN, A., H. C. LAU y P. VANSTEENWEGEN (2016), «Orienteering problem: A survey of recent variants, solution approaches and applications», *European Journal of Operational Research*, **255**(2), págs. 315–332, <https://doi.org/10.1016/j.ejor.2016.04.059>.
- GUTIN, G. y A. P. PUNNEN (2007), *The traveling salesman problem and its variations, Combinatorial Optimization*, tomo 12, Springer, Boston, MA, ISBN: 978-0-306-48213-7, <https://doi.org/10.1007/b101971>.
- HA, M. H., N. BOSTEL, A. LANGEVIN y L.-M. ROUSSEAU (2013), «An exact algorithm and a metaheuristic for the multi-vehicle covering tour problem with a

- constraint on the number of vertices», *European Journal of Operational Research*, **226**(2), págs. 211–220, doi="https://doi.org/10.1016/j.ejor.2012.11.012.
- HAKIMI, S. L. (1965), «Optimum distribution of switching centers in a communication network and some related graph theoretic problems», *Operations Research*, **13**(3), págs. 462–475, <https://doi.org/10.1287/opre.13.3.462>.
- IBARRA-ROJAS, O., L. HERNANDEZ LANDA y L. OZUNA (2018), «The accessibility vehicle routing problem», *Journal of Cleaner Production*, **172**, págs. 1514–1528, <https://doi.org/10.1016/j.jclepro.2017.10.249>.
- JAIN, A. K. y S. MAHESWARI (2012), «Survey of recent clustering techniques in data mining», *Int J Comput Sci Manag Res*, **3**(2), págs. 68–75.
- JAYKARAN, C. (2010), «How to select appropriate statistical test?», *Journal of Pharmaceutical Negative Results*, **1**(2), pág. 61, <https://doi.org/10.4103/0976-9234.75708>.
- KAMMOUN, M., H. DERBEL, M. RATLI y B. JARBOUI (2015), «A variable neighborhood search for solving the multi-vehicle covering tour problem», *Electronic Notes in Discrete Mathematics*, **47**, págs. 285–292, The 3rd International Conference on Variable Neighborhood Search (VNS'14), <https://doi.org/10.1016/j.endm.2014.11.037>.
- KANTOR, M. G. y M. B. ROSENWEIN (1992), «The orienteering problem with time windows», *Journal of the Operational Research Society*, **43**(6), págs. 629–635, <https://doi.org/10.1057/jors.1992.88>.
- KATAOKA, S. y S. MORITO (1988), «An algorithm for single constraint maximum collection problem», *Journal of the Operations Research Society of Japan*, **31**(4), págs. 515–531, <https://doi.org/10.15807/jorsj.31.515>.
- LAPORTE, G. y S. MARTELLO (1990), «The selective travelling salesman problem», *Discrete Applied Mathematics*, **26**(2-3), págs. 193–207, [https://doi.org/10.1016/0166-218X\(90\)90100-Q](https://doi.org/10.1016/0166-218X(90)90100-Q).

- LENSTRA, J. K. (1974), «Clustering a data array and the traveling-salesman problem», *Operations Research*, **22**(2), págs. 413–414, <https://doi.org/10.1287/opre.22.2.413>.
- LENSTRA, J. K. y A. R. KAN (1975), «Some simple applications of the travelling salesman problem», *Journal of the Operational Research Society*, **26**(4), págs. 717–733, <https://doi.org/10.1057/jors.1975.151>.
- LÜER, A., M. BENAVENTE, J. BUSTOS y B. VENEGAS (2009), «El Problema de Rutas de Vehículos: Extensiones y Métodos de Resolución, estado del Arte.», en *EIG*, <https://pdfs.semanticscholar.org/e47e/71d4f7470d69dd987190286cfd48eeabae61.pdf>.
- LUNA LÓPEZ, L. C. (2015), *Localización de paradas y diseño óptimo de rutas para transporte de personal*, Tesis de Maestría, Universidad Autónoma de Nuevo León, <http://eprints.uanl.mx/9541/1/1080214944.pdf>.
- MADSEN, O. B. (1988), «An application of travelling-salesman routines to solve pattern-allocation problems in the glass industry», *Journal of the Operational Research Society*, **39**(3), págs. 249–256, <https://doi.org/10.1057/jors.1988.42>.
- MCCORMICK JR, W. T., P. J. SCHWEITZER y T. W. WHITE (1972), «Problem decomposition and data reorganization by a clustering technique», *Operations Research*, **20**(5), págs. 993–1009, <https://doi.org/10.1287/opre.20.5.993>.
- MONTEMANNI, R. y L. M. GAMBARDELLA (2009), «An ant colony system for team orienteering problems with time windows», *Foundation Of Computing And Decision Sciences*, **34**(4), págs. 287–306.
- NAJI-AZIMI, Z. y M. SALARI (2014), «The time constrained maximal covering salesman problem», *Applied Mathematical Modelling*, **38**(15-16), págs. 3945–3957, <https://doi.org/10.1016/j.apm.2014.01.001>.

- OZBAYGIN, G., H. YAMAN y O. E. KARASAN (2016), «Time constrained maximal covering salesman problem with weighted demands and partial coverage», *Computers & Operations Research*, **76**, págs. 226–237, <https://doi.org/10.1016/j.cor.2016.06.019>.
- PHAM, T. A., M. H. HÀ y X. H. NGUYEN (2017), «Solving the multi-vehicle multi-covering tour problem», *Computers & Operations Research*, **88**, págs. 258–278, <https://doi.org/10.1016/j.cor.2017.07.009>.
- PIRKUL, H. y D. A. SCHILLING (1991), «The maximal covering location problem with capacities on total workload», *Management Science*, **37**(2), págs. 233–248, <https://doi.org/10.1287/mnsc.37.2.233>.
- POHLE, T., P. KNEES, M. SCHEDL, E. PAMPALK y G. WIDMER (2007), «Reinventing the wheel: a novel approach to music player interfaces», *IEEE Transactions on Multimedia*, **9**(3), págs. 567–575, <https://doi.org/10.1109/TMM.2006.887991>.
- RATLIFF, H. D. y A. S. ROSENTHAL (1983), «Order-picking in a rectangular warehouse: a solvable case of the traveling salesman problem», *Operations Research*, **31**(3), págs. 507–521, <https://doi.org/10.1287/opre.31.3.507>.
- RECIO HERNÁNDEZ, A. (2014), *Optimización de un algoritmo de colonia de hormigas mediante CUDA para resolver problemas de ruteo de vehículos*, Tesis de Maestría, Universidad Autónoma de Aguascalientes. Centro de Ciencias Básicas. Departamento de Ciencias de la Computación, <http://hdl.handle.net/11317/401>.
- REINELT, G. (1994), *The traveling salesman: computational solutions for TSP applications*, *Lecture Notes in Computer Science*, tomo 840, Springer-Verlag Berlin Heidelberg, ISBN: 978-3-540-48661-9, <https://doi.org/10.1007/3-540-48661-5>.
- RIGHINI, G. y M. SALANI (2009), «Decremental state space relaxation strategies and initialization heuristics for solving the orienteering problem with time windows with dynamic programming», *Computers & Operations Research*, **36**(4), págs. 1191–1203, <https://doi.org/10.1016/j.cor.2008.01.003>.

- ROPKE, S. y D. PISINGER (2006), «An adaptive large neighborhood search heuristic for the pickup and delivery problem with time windows», *Transportation Science*, **40**(4), págs. 455–472, <https://doi.org/10.1287/trsc.1050.0135>.
- SCHITTEKAT, P., J. KINABLE, K. SÖRENSEN, M. SEVAUX, F. SPIEKSMÁ y J. SPRINGAEL (2013), «A metaheuristic for the school bus routing problem with bus stop selection», *European Journal of Operational Research*, **229**(2), págs. 518–528, <https://doi.org/10.1016/j.ejor.2013.02.025>.
- SHAELAIE, M. H., M. SALARI y Z. NAJI-AZIMI (2014), «The generalized covering traveling salesman problem», *Applied Soft Computing*, **24**, págs. 867–878, <https://doi.org/10.1016/j.asoc.2014.08.057>.
- SINNL, M. (2019), «Mixed-Integer Programming Approaches for the Time-Constrained Maximal Covering Routing Problem», *Informe técnico*, Institute of Production and Logistics Management, Johannes Kepler University Linz.
- TSILIGIRIDES, T. (1984), «Heuristic methods applied to orienteering», *Journal of the Operational Research Society*, **35**(9), págs. 797–809, <https://doi.org/10.1057/jors.1984.162>.
- VANSTEENWEGEN, P., W. SOUFFRIAU, G. V. BERGHE y D. VAN OUDHEUSDEN (2009), «Iterated local search for the team orienteering problem with time windows», *Computers & Operations Research*, **36**(12), págs. 3281–3290, <https://doi.org/10.1016/j.cor.2009.03.008>.
- VANSTEENWEGEN, P., W. SOUFFRIAU y D. VAN OUDHEUSDEN (2011), «The orienteering problem: A survey», *European Journal of Operational Research*, **209**(1), págs. 1–10, <https://doi.org/10.1016/j.ejor.2010.03.045>.
- VEGA OCAÑA, O. (2008), *Asignación de rutas de viajantes de comercio*, Tesis Doctoral, UPC, Escola Tècnica Superior d'Enginyeria Industrial de Barcelona, Departament d'Organització d'Empreses, <http://hdl.handle.net/2099.1/5719>.

WOEGINGER, G. J. (2003), «Exact algorithms for NP-hard problems: A survey», en *Combinatorial optimization—eureka, you shrink!. Lecture Notes in Computer Science*, tomo 2570, Springer, Berlin, Heidelberg, págs. 185–207, ISBN: 978-3-540-36478-8, https://doi.org/10.1007/3-540-36478-1_17.

RESUMEN AUTOBIOGRÁFICO

Brenda Yaneth Sotelo Benítez

Candidato para obtener el grado de
Maestría en Ciencias de la Ingeniería
con Orientación en Sistemas

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica

Tesis:

PROBLEMA DE RUTEO CON MÁXIMA COBERTURA Y TIEMPO LÍMITE

Nací el 17 de febrero de 1995 en la ciudad de Monterrey, Nuevo León; mis padres son Norma Elia Benítez Dominguez e Isaias Sotelo Abundiz.

Cursé mis estudios de educación básica en el período 2001-2007 en la escuela primaria “Francisco Javier Mina”, los estudios de educación media en la Escuela Secundaria No. 41 “Jose T. Puente” durante los años 2007-2010 y en el período 2010-2012 realicé los estudios de educación media superior en la Preparatoria No. 9 de la Universidad Autónoma de Nuevo León.

En el año 2012 ingresé a la Facultad de Ciencias Físico Matemáticas de la Universidad Autónoma de Nuevo León y egresé en el 2017 obteniendo el título de Licenciada en Matemáticas.

Posteriormente, ingresé como estudiante de maestría al Programa de Posgrado en Ingeniería de Sistemas de la Facultad de Ingeniería Mecánica y Eléctrica de la Universidad Autónoma de Nuevo León en el año 2018.

Durante mis estudios de posgrado he tenido la oportunidad de participar en congresos nacionales, tales como el XXVIII Escuela Nacional de Optimización y Análisis Numérico (ENOAN 2019) en el Consejo Zacatecano de Ciencia, Tecnología e Innovación en la ciudad de Zacatecas, del 26 al 30 de agosto de 2019 y el 52 Congreso Nacional de la Sociedad Matemática Mexicana (SMM) que se llevó a cabo del 21 al 25 de octubre de 2019, en la Facultad de Ciencias Físico Matemáticas de la Universidad Autónoma de Nuevo León.