ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ciencias Naturales y Matemáticas

El problema de enrutamiento vehicular en institutos de educación: modelización y algoritmo de solución

PROYECTO INTEGRADOR

Previo a la obtención del Título de:

Matemático

Presentado por:

Carlos Andrés Ordóñez Palacios

Patrick Jhosue Arévalo Guzmán

GUAYAQUIL - ECUADOR

Año: 2023

DEDICATORIA

Este trabajo esta dedicado a mis padres, que me han brindado su apoyo y cariño en cada momento.

Carlos Ordóñez P.

DEDICATORIA

Dedico esta tesis a mi amiga Camila, mi fenocopia. Eres un recordatorio constante de la belleza y el potencial que existe en las personas que a menudo son mal comprendidas por el mundo.

Este trabajo también está dedicado a todas las personas que, como Camila y yo, enfrentan desafíos únicos en la vida. A aquellos cuyos pensamientos y perspectivas son diferentes, pero que tienen un potencial innegable para contribuir de manera significativa al mundo que los rodea. Que esta tesis sea un tributo a su resiliencia, creatividad y valentía al navegar en un mundo que no siempre reconoce su brillantez.

Patrick Arévalo G.

AGRADECIMIENTOS

Quiero expresar mi gratitud por las personas que me han apoyado durante el periodo como estudiante, tanto a los profesores y compañeros que me han acompañado durante este recorrido. Al tutor Ph.D. Xavier Cabezas por su guia y apoyo a lo largo del proyecto, del mismo modo que la coordinadora de la carrera Ph.D. Elimar Marchan.

Mi más profundo agradecimiento para mi familia: mis padres, hermanos y mi querida mascota. Han sido mi fuente de inspiración y apoyo incondicional. A mi pareja, gracias por estar a mi lado

Carlos Ordóñez P.

AGRADECIMIENTOS

Agradezco al PhD. Xavier Cabezas, mi tutor, por su orientación invaluable. También a la PhD. Elimar Marchan, coordinadora de carrera, por su colaboración. Mi gratitud a Mira Murati, Nanawo Akari, Neuro-Sama, ZUN por su respaldo emocional. A Physics Landscape, a Mates Ecuadorff y a las personas con las que tuve valiosas discusiones académicas. A mi familia y a quienes me ayudaron con trámites. A todas las personas que me brindaron oportunidades en diversas esferas luego de conocerme.

Patrick Arévalo G.

DECLARACIÓN EXPRESA

"Los derechos de titularidad y explotación, nos corresponde conforme al reglamento de propiedad intelectual de la institución; *Carlos Andrés Ordóñez Palacios, Patrick Jhosue Arévalo Guzmán,*y damos nuestro consentimiento para que la ESPOL realice la comunicación pública de la obra por cualquier medio con el fin de promover la consulta, difusión y uso público de la producción intelectual"

Carlos Ordóñez P.

Patrick Arévalo G.

EVALUADORES

Firmado electrónicamente por:
LUZ ELIMAR MARCHAN MENDOZA

Ph.D. Luz Elimar Marchan Mendoza

PROFESOR DE LA MATERIA

Firmado electrónicamente por:
JOSE XAVIER CABEZAS GARCIA

Ph.D. José Xavier Cabezas Garcia

TUTOR DEL PROYECTO

RESUMEN

Para las instituciones educativas la movilización de los estudiantes es de los aspectos con gran relevancia, así como la educación. Pues en este se involucra tanto la seguridad como la puntualidad de sus estudiantes. Asi, surge la necesidad de brindar un buen servicio basado en los recursos que posee la institución. En este trabajo se plantea una mejora en la forma de escoger las rutas por medio de un modelo de optimización matemática que resulte en la reducción del tiempo en los recorridos usados por la flota de buses escolar. Para trabajar el problema se ejecutó un modelo de solución exacta que permitió obtener las respectivas rutas que cada bus debería recorrer, considerando la ubicación de los estudiantes y distintas estaciones de bus. Sin embargo, si bien es cierto que un método resulta ser computacionalmente costoso, dependiendo de la instancia del problema puede brindar resultados en un tiempo razonable con el beneficio de saber que es la mejor respuesta posible para el problema.

Palabras Clave: Optimización de rutas, Transporte Escolar, Algoritmos exactos, Problema de asignación.

ABSTRACT

For an educational institution the way theirs students are being transported is as important as the education. The reason for that is it involves the security and punctuality aspects. From this it comes a necessity of having a good service according to the institution resources. For this it was proposed an upgrade on how to decide a route by a mathematical optimization model that further will reduce the travel time for the scholar bus. To develop the problem an exact algorithm solution was the one that give the routes that each bus will have to travel, considering where the students and the bus stops are located. However, this method results to be computationally expensive, so by depending on the instance it could give the results on a reasonable time with the benefit of knowing that is an optimal solution.

Keywords: *Routes optimization, school transportation, exact algorithms, Assignation problem.*

ÍNDICE GENERAL

RESUMEN I

ABSTRACT II

ABREVIATURAS V

SIMBOLOGÍA VI

ÍNDICE DE FIGURAS VII

ÍNDICE DE TABLAS VIII

CAPÍTULO 1 1

1. INTRODUCCIÓN 1

1.1 Descripción del problema 2

1.2 Justificación del problema 3

1.3 Objetivos 3

1.3.1 Objetivo General 3

1.3.2 Objetivos Específicos 4

1.4 Marco teórico 4

1.4.1 Antecedentes 4

1.4.2 Teoría de la optimización en el transporte 5

1.4.3 VRP en el transporte escolar 7

1.4.4 Definición de términos básicos 11

CAPÍTULO 2 13

2. METODOLOGÍA 13
2.0.1 Algoritmos de solución 13
2.0.2 Lazy constrains 14
2.0.3 Implementación del SBRP 14
2.0.4 Breaking subtours constrains 15
CAPÍTULO 3 20
3. RESULTADOS Y ANÁLISIS 20
3.1 Pruebas para instancias pequeñas 21
3.2 Pruebas instancias medianas 26
CAPÍTULO 4 29
4. CONCLUSIONES Y RECOMENDACIONES 29
4.1 Conclusiones 29
4.2 Recomendaciones 30
BIBLIOGRAFÍA
APÉNDICES

ABREVIATURAS

BB	Branch and Bound
	Ramificación y acotación
BC	Branch and Cut
	Ramificar y cortar
DFJ	Danzig–Fulkerson–Johnson
ESPOL	Escuela Superior Politécnica del Litoral
GG	Gavish–Graves
MILP	Mixed-Integer Linear Programming
	Programación Lineal Mixta con Variables Enteras
SBRP	School Bus Routing Problem
	Problema de enrutamiento vehicular para buses escolares
TMZ	Miller–Tucker–Zemlin
TSP	Travelling Salesman Problem
	Problema del agente viajero
VRP	Vehicle Routing Problem
	Problema de enrutamiento vehicular

SIMBOLOGÍA

$\sum$	sumatoria
$\mathcal{P}(A)$	Conjunto potencia de un conjunto A
x_{ijk}	Variable binaria que indica si un vehículo k recorre el arco desde i hasta j
y_{ik}	Variable binaria que indica si el vehículo k visita la parada i
z_{ilk}	Variable binaria que indica si el estudiante l es recogido por vehículo
u_{ik}	Variable entera que indica el orden de visita por estación i del vehiculo k
c_{ij}	Costo desde la parada i hasta la parada j
K	Total de vehículos disponibles
C	Capacidad de un vehículo
V	Conjunto de paradas de bus
S	Conjunto de estudiantes
s_{li}	Variable binaria que indica si el estudiante l puede caminar hasta la parada i

ÍNDICE DE FIGURAS

Figura 3.1 Solución del modelo sin incluir restricciones para romper subrutinas, Fuente: creación propia 21

Figura 3.2 Solución óptima para instancia de 12 nodos, Fuente: creación propia 23

Figura 3.3 Soluciones para la instancia de 20 nodos, Fuente: creación propia 26

Figura 3.4 Soluciones para las instancias de 25 y 30 estaciones de bus, Fuente: creación propia 28

ÍNDICE DE TABLAS

Tabla 3.1 Nodos activos por autobus, , Fuente: creación propia 22

Tabla 3.2 Detalles de las soluciones para el DFJ y respuesta con subtours, Fuente: creación propia 23

Tabla 3.3 Comparativa entre los distintos algoritmos de solución, Fuente: creación propia . 24

Tabla 3.4 Comparativa entre los distintos algoritmos de solución, Fuente: creación propia . 25

Tabla 3.5 Pruebas de formulación TMZ para instancias de 25, 30 y 40 estaciones de bus, Fuente: creación propia 28

CAPÍTULO 1

1. INTRODUCCIÓN

El enrutamiento vehicular para buses escolares, también conocido como SBRP por sus siglas en inglés (School Bus Routing Problem), puede considerarse como un caso particular del problema clásico de enrutamiento vehicular (VRP por sus siglas en inglés). Este problema fue inicialmente descrito en 1959 por Dantzing y Ramser bajo el nombre de ”Truck Dispatching Problem”.

El VRP pertenece a una clase de problemas de optimización que involucran la planificación de rutas eficientes para un conjunto de vehículos que deben de dirigirse a ubicaciones específicas, cumpliendo con determinadas restricciones. El objetivo principal del VRP es encontrar la asignación óptima de rutas y secuencias de visitas que minimice medidas de costo como: la distancia total recorrida, el tiempo total de viaje o el número de vehículos utilizados;mientras se cumplen todas las restricciones requeridas, como: la capacidad de carga de los vehículos y las ventanas de tiempo de los clientes.

Con el transcurso del tiempo, se han desarrollado diferentes métodos para abordar este problema, tanto mediante el uso de técnicas exactas, como de las metaheurísticas. Entre los métodos exactos tendremos: el branch and bound, branch and cut, programación dinámica y programación lineal entera. Sin embargo, dado que el problema es de naturaleza NP, la eficiencia computacional de las soluciones exactas se ve comprometida, especialmente en

instancias grandes Corona León (2005)

Dada esta complejidad, implementar soluciones para el VRP representa un desafío, el cual se intensifica al considerar características más específicas. Como el caso del SBRP que requiere además, tener en cuenta las particularidades propias de cada institución escolar al diseñar los algoritmos, con el objetivo de simular la realidad en la medida de lo posible.

1.1 Descripción del problema

El VRP es un problema que se desarrolló en un inicio con la intención de mejorar una situación de la industria. Sin embargo, con el paso del tiempo se ha encontrado la forma de adaptarlo a diferentes situaciones de la vida. Uno de los entornos en el que se ha podido adaptar el VRP es en las instituciones educativas, específicamente en la sección de buses escolares.

Si una institución educativa presenta dificultades de gestión de su transporte escolar, se tiene como raíz del problema: el hecho de no aplicar algoritmos de optimización. Una práctica usual para el proceso de la obtención de rutas para buses escolares es por medio de la experiencia de los conductores, lo que no garantiza precisión ni los mejores resultados; como consecuencia de esto se tienen dificultades en la puntualidad o la falta de coordinación en la recogida y entrega del alumnado. Estas situaciones terminan afectandp de manera directa a los colegios y sus integrantes: estudiantes, padres y también a los mismos conductores.

El problema por resolver, se centra en la necesidad de diseñar e implementar rutas óptimas y eficientes, considerando las restricciones de tiempo y recursos disponibles que vienen determinados por cada institución educativa. La optimización se realiza mediante algoritmos especializados, bajo los supuestos de un modelo matemático, soluciones exactas o aproximadas de las rutas de buses. Además, la modelización y el enfoque de solución debe considerar:

preferencias de los padres, direcciones de los hogares, puntos de partida y obstáculos en las vías.

1.2 Justificación del problema

La resolución eficiente del problema del transporte escolar en una institución educativa establece fundamentos y metodologías que pueden aplicarse en otros escenarios logísticos similares. Estos incluyen: la distribución de mercancías, el transporte público, los servicios de entrega, entre otros. La optimización de rutas, la asignación eficiente de recursos y la consideración de preferencias y restricciones son elementos comunes en los problemas de transporte.

Por lo tanto, al abordar el problema del transporte escolar en las escuelas y colegios, no solo se benefician directamente los estudiantes, los padres, los conductores y el personal administrativo; sino que también se establece una base sólida para enfrentar otros desafíos logísticos en diferentes industrias y sectores que requieren una gestión eficiente de la flota de vehículos.

1.3 Objetivos

1.3.1 Objetivo General

Mejorar las rutas de los buses escolares mediante un modelo de ruteo de vehículos para la reducción en el tiempo de los recorridos.

1.3.2 Objetivos Específicos

- Realizar un diagnóstico de las condiciones actuales en las que se manejan las rutas de los vehículos escolares en la institución
- Seleccionar un modelo de enrutamiento vehicular adecuado a las condiciones de la institución para obtener una solución óptima al problema.
- Adaptar el modelo de enrutamiento vehicular a la institución para mejorar las rutas de transporte.

1.4 Marco teórico

1.4.1 Antecedentes

El VRP ha sido ampliamente estudiado en diferentes ocasiones para resolver situaciones de la vida real. En 1959, Dantzing y Ramser propusieron en su trabajo ”Trucking dispatching problem” una definición formal inicial del VRP como una generalización del problema del agente viajero (TSP), con el objetivo de resolver un caso real que involucraba una flota de vehículos utilizados para el transporte de gasolina Dantzig and Ramser (1959).

Con el paso del tiempo, el VRP se diversificó y surgieron diferentes variantes que involucraban distintos algoritmos de solución y casos específicos. Uno de los ámbitos clásicos de aplicación del VRP es en las instituciones educativas, donde se enfocan los esfuerzos en la optimización de las rutas para los buses escolares. Esto garantiza que los estudiantes sean recogidos y dejados de manera eficiente y segura, minimizando así los tiempos de viaje y optimizando la utilización de los vehículos.

Además, el VRP ha dado lugar al estudio del SBRP, que se enfoca específicamente en la optimización de las rutas de los buses escolares. Desde su primera publicación en 1969 por Newton y Thomas, el SBRP ha sido objeto de estudio. Sin embargo, debido a la aparición de casos de la vida real con características y restricciones específicas, se ha reconocido la necesidad de desarrollar enfoques diferentes para el SBRP en diversos estudios Park and Kim (2010).

En la actualidad, el SBVRP continúa siendo objeto de investigación con el objetivo de obtener mejores resultados mediante la implementación de diversas técnicas de solución, ya sean exactas o heurísticas. La optimización en el transporte, particularmente en el contexto del VRP y el SBRP, es de suma importancia, ya que permite reducir costos operativos, mejorar la eficiencia de las operaciones y minimizar los tiempos de entrega, lo cual tiene un impacto significativo en diversos sectores.

1.4.2 Teoría de la optimización en el transporte

Definición de VRP (Vehicle Routing Problem)

El VRP es un problema complejo de optimización que busca diseñar rutas eficientes para una flota de vehículos que deben realizar entregas o recolecciones a distintos clientes o ubicaciones. El objetivo es minimizar el costo total de transporte, que puede estar relacionado con: la distancia total recorrida, el tiempo de transporte, la cantidad de vehículos utilizados, entre otros.

Tipos de VRP

Existen algunas variantes de VRP que se utilizan para representar diversas situaciones del mundo real Parra and Chavez (1998):

- **CVRP (Capacitated Vehicle Routing Problem):** En esta variante, se tiene una flota fija de vehículos de capacidad uniforme que deben satisfacer las entregas a una serie de clientes.
- **MDVRP (Multiple Depot VRP):** Esta variante considera múltiples centros de abastecimiento.
- **VRPHEs (VRP with heterogeneous Fleet):** En este caso, la flota de vehículos consta de vehículos de diversas capacidades.
- **VRPTW (Vehicle Routing Problem with Time Windows):** Esta variante agrega restricciones de tiempo, donde las entregas deben realizarse dentro de ciertos intervalos de tiempo definidos.
- **VRPPD (VRP with Pick-Up and Delivering):** Aquí, se debe considerar la posibilidad de que los clientes devuelvan algún producto.
- **SDVRP (Split Delivery VRP):** Esta variante permite que un cliente sea visitado por varios vehículos si eso reduce los costos totales.

Algoritmos de optimización utilizados en VRP

Existen varios enfoques para resolver problemas de VRP, que se pueden categorizar en métodos exactos, heurísticas y metaheurísticas:

- **Métodos exactos:** Incluyen algoritmos de ramificación y acotamiento (branch and bound), ramificación y corte (branch and cut), programación dinámica y programación lineal entera. Estos métodos garantizan encontrar la solución óptima, pero pueden ser computacionalmente complejos para problemas de gran escala.

- **Heurísticas:** Son métodos que buscan soluciones de alta calidad de manera eficiente, sin garantizar la optimalidad. Ejemplos de heurísticas para VRP incluyen: el método de construcción, el algoritmo de dos fases y el de mejora iterativa, es decir usar como entrada una solución que haya sido plantada por una heurística anterior.Hou et al. (2022).

- **Metaheurísticas:** Son técnicas de optimización de alto nivel que guían la búsqueda de soluciones en el espacio de soluciones para explorar regiones de alta calidad. Ejemplos de metaheurísticas incluyen la colonia de hormigas, la programación por restricciones, los algoritmos genéticos, el recocido simulado, la búsqueda tabú y las redes neuronales.Arias-Rojas et al. (2012)

1.4.3 VRP en el transporte escolar

Estudios previos sobre la aplicación de VRP en el transporte escolar

Existen varios trabajos que han aplicado el VRP al problema del transporte escolar, también conocido como School Bus Routing Problem (SBRP). Por ejemplo, Schittekat et al. (2006) adaptaron el VRP a un caso en Bélgica, donde los autobuses escolares deben recoger a los estudiantes en varias paradas y llevarlos a la escuela. Utilizaron un método de solución exacta basado en programación entera para resolver el problema.

Otro trabajo relevante es el de Escobar Morales et al. (2018), quienes propusieron un método de tres fases para el SBRP. En la primera fase, se clasifican las ubicaciones de los estudiantes y las distancias entre ellas. En la segunda fase, se asignan los estudiantes a rutas utilizando una heurística de barrido. En la tercera fase, se asignan las rutas a los autobuses, considerando cada ruta como un problema de agente viajero (TSP). Para resolver el TSP,

utilizaron el software Concorde debido a la complejidad computacional del problema.

Ventajas y beneficios de la aplicación de VRP en el transporte escolar

La aplicación de VRP en el transporte escolar puede traer varios beneficios. Primero, puede reducir los costos de transporte al minimizar la distancia total recorrida por los autobuses. Segundo, puede mejorar la eficiencia del servicio al minimizar el tiempo de viaje de los estudiantes y equilibrar la carga de trabajo entre los autobuses. Tercero, puede mejorar la seguridad de los estudiantes al diseñar rutas que eviten áreas peligrosas y al asegurar que los autobuses no estén sobrecargados.

Planteamiento del SBRP

Se considera un conjunto de paradas potenciales donde los autobuses pueden recoger y dejar estudiantes. Cada parada tiene asociada una capacidad máxima de estudiantes que puede atender. El objetivo es determinar las rutas óptimas para los autobuses escolares de manera que se minimice el coste total de viaje que realizan todos los vehículos. Para ello tenemos que considerar las siguientes variables de decisión y datos del problema:

Variables de Decisión

- x_{ijk}: Variable binaria que es igual a 1 si el vehículo k recorre el arco desde la parada i hasta la parada j, y 0 en caso contrario.
- y_{ik}: Variable binaria que es igual a 1 si el vehículo k visita la parada i, y 0 en caso contrario.
- z_{ilk}: Variable binaria que es igual a 1 si el estudiante l es recogido por el vehículo k en la parada i, y 0 en caso contrario.

Datos:

- c_{ij}: Distancia desde la parada i hasta la parada j.
- K: Número total de vehículos disponibles.
- C: Capacidad de un vehículo.
- V: Conjunto de todas las paradas.
- S: Conjunto de todos los estudiantes.
- s_{li}: Variable binaria que es igual a 1 si el estudiante l puede caminar hasta la parada i, y 0 en caso contrario.

Función objetivo: Con el costo c_{ij} representando la distancia desde la parada i hasta la parada j, la función objetivo representa los costos de recorrer los arcos (i, j) multiplicado por el número de veces que el vehículo k recorre ese arco i.e. el costo total por realizar todos los recorridos.

$$\sum_{i \in V} \sum_{j \in V} c_{ij} \sum_{k=1}^{K} x_{ijk} \tag{1.1}$$

La formulación matemática del problema se puede expresar de la siguiente manera:

$$\min \sum_{i \in V} \sum_{j \in V} c_{ij} \sum_{k=1}^{K} x_{ijk} \tag{1.2}$$

sujeto a las siguientes restricciones:

1. **Restricción de inicio de ruta**: Cada vehículo k debe comenzar su ruta desde la parada inicial.

$$\sum_{k=1}^{K} y_{0k} \leqslant K, \quad k = 1, \ldots, K \tag{1.3}$$

2. **Restricción de flujo**: Si un vehículo visita una parada, entonces debe haber un arco que entre y salga de esa parada.

$$\sum_{j \in V} x_{ijk} = \sum_{j \in V} x_{jik} = y_{ik}, \quad \forall i \in V, k = 1, \ldots, K \tag{1.4}$$

3. **Restricción de subrutinas**: Previene la formación de subrutinas entre un subconjunto de paradas.

$$\sum_{i \in S} \sum_{j \notin S} x_{ijk} \geqslant y_{hk}, \quad \forall S \subseteq V \backslash \{0\}, h \in S, k = 1, \ldots, K \tag{1.5}$$

4. **Restricción de visita única**: Cada parada, excepto la parada inicial, debe ser visitada por un solo vehículo.

$$\sum_{k=1}^{K} y_{ik} \leqslant 1, \quad \forall i \in V \backslash \{0\} \tag{1.6}$$

5. **Restricción de caminar a la parada**: Si un estudiante puede caminar hasta la parada i, entonces puede ser recogido por algún vehículo en esa parada.

$$\sum_{k=1}^{K} z_{ilk} \leqslant s_{li}, \quad \forall l \in S, \forall i \in V \tag{1.7}$$

6. **Restricción de capacidad del vehículo**: La capacidad de un vehículo no debe ser excedida por el número de estudiantes que recoge.

$$\sum_{i \in V} \sum_{l \in S} z_{ilk} \leqslant C, \quad \forall k = 1, \ldots, K \tag{1.8}$$

7. **Restricción de asignación estudiante-parada**: Si un estudiante es recogido por un vehículo en una parada, entonces el vehículo debe visitar esa parada.

$$z_{ilk} \leqslant y_{ik}, \quad \forall i, l, k \tag{1.9}$$

8. **Restricción de recogida única**: Cada estudiante debe ser recogido exactamente una vez.

$$\sum_{i \in V} \sum_{k=1}^{K} z_{ilk} = 1, \quad \forall l \in S \tag{1.10}$$

Además, todas las variables de decisión son binarias, es decir,

$$y_{ik}, x_{ijk}, z_{ilk} \in \{0, 1\}, \quad \forall i, j \in V \quad con \quad i \neq j \tag{1.11}$$

1.4.4 Definición de términos básicos

- **Problema de optimización:** Es un tipo de problema en el que se busca encontrar la mejor solución posible dentro de un conjunto de soluciones factibles. Este proceso implica minimizar o maximizar una función objetivo sujeta a un conjunto de restricciones.
- **Función objetivo:** Es una función matemática que describe el objetivo de un problema de optimización.
- **Restricciones:** Son las condiciones que deben cumplir las soluciones factibles de un problema de optimización. Las restricciones definen el espacio factible de soluciones del problema. En el SBRP, las restricciones están relacionadas con la capacidad de los vehículos, el número de vehículos disponibles, entre otros.
- **Problema NP-completo:** Es una clase de problemas de decisión en la teoría de la computación. Un problema se considera NP-completo si pertenece a la clase NP (problemas cuyas soluciones se pueden verificar en tiempo polinómico), y cualquier problema en la clase NP se puede reducir a él en tiempo polinómico. Los problemas NP-completos no tienen algoritmos de tiempo polinómico conocidos para resolverlos, tal que enbcontrar una solución óptima puede requerir tiempo exponencial.Bazrafshan et al. (2021).

- **Instancia del problema VRP:** Una instancia del problema es un conjunto específico de datos que se le proporciona a cada parámetro del problema. Esto incluye el número de paradas, el número de estudiantes, la capacidad de los vehículos, la distancia entre cada parada, entre otros datos.

- **Variables de decisión:** Son las incógnitas que se deben determinar en un problema de optimización. Representan las decisiones que el tomador de decisiones puede controlar. En el SBRP, las variables de decisión son el número de veces que un vehículo recorre un arco específico, si un vehículo visita una parada específica y si un estudiante es recogido por un vehículo en una parada específica.

- **Problema de Programación Lineal Entera Mixta (MILP):** Son problemas de optimización en los que la función objetivo es lineal, tambien se tendra que una parte o la totalidad de las variables del modelo son enteras y pueden haber tanto restricciones de igualdad como de desigualdad. El SBRP se puede formular como un MILP.

CAPÍTULO 2

2. METODOLOGÍA

Considerando resultados de trabajos realizados previamente por otros investigadores se decidió implementar un método de solución exacta para resolver el SBRP.

2.0.1 Algoritmos de solución

Algoritmo Branch and bound

El algoritmo de Branch and bound (BB por sus siglas en inglés) es un método de solución exacta para problemas de optimización el cual esta constituido con dos procesos fundamentales.

- **Branch** o **Ramificación**: Lo que hace es dividir el problema en distintos subproblemas que sean mas sencillos de manejar.
- **Bound** o **Acotar**: Al momento que uno de los subproblemas que se encuentra analizando en el algoritmo como una de las soluciones se vuelve muy complicado de resolver, entonces, una cota definida previamente nos permite eliminar esa ramificación en específico, de modo que no se la sigue analizando.

Para esto, se puede tomar el valor de la cota de un programa entero usando su relajación lineal. De tal modo que un subproplema o ramificación se considera sencillo si la relajación lineal tiene una solución entera. Sino, se toma la decisión de cortar la ramificación. Achterberg et al. (2005)

Algoritmo Branch and cut

El algoritmo de Branch and cut (BC por sus siglas en inglés) es un método de solución exacta para resolver problemas de optimización en programación lineal entera mixta, por lo que se puede acoplar al SBRP.

El BC corresponde a la unión de dos algoritmos de solución que son conocidos como Branch and Bound y Cutting Plane. Padberg and Rinaldi (1991)

2.0.2 Lazy constrains

Uno de los enfoques usados para resolver algunos problemas de optimización donde se tiene un costo computacional alto es utilizar el concepto de las “lazy constrains”. Esto implica añadir únicamente las restricciones que ayudan a que el modelo no proporcione una respuesta errónea.

Un beneficio de realizar este método es que resulta computacionalmente más rápido proporcionar únicamente al modelo con las restricciones que generen alguna contradicción para la solución. Además, esto puede beneficiar aun más para modelos que cuentan con un gran número de restricciones Pferschy and Staněk (2017).

2.0.3 Implementación del SBRP

Para el trabajo se hizo uso de un código en Python, además de una librería de optimización conocida como docplex. De esta libreria se hizo uso de docplex.mp, la cual permite escribir un modelo matemático con sus respectivas variables de decisión, función objetivo y restricciones. En la librería docplex se hizo uso de CPLEX en su version académica como solver, el cual es un solver de IBM y permite desarrollar y resolver modelos de optimización con programación matemática,

correspondiente a las funciones de la libreria docplex.mp, o programación de restricciones para las funciones de docplex.cp.

Se debe mencionar que la librería se tiene implementado el algoritmo del BC, para esto se plantea tanto las variables de decisión, función objetivo y las restricciones. Sin embargo, se debe mencionar que al momento de correr el modelo de enrutamiento previamente planteado no se considera la restricción 1.5 al conseguir una solución por primera vez.

Con una primera solución exacta del problema del SBRP. Sin embargo, dentro de la solución se generaron recorridos aislados del recorrido principal de cada vehículo, esto se conoce como los subtours o subrutina.

2.0.4 Breaking subtours constrains

Para el problema del TSP espera como solución un circuito cerrado o Hamiltoniano. En el caso del VRP se espera una cantidad de circuitos cerrados dependiendo de la cantidad de vehículos que se tienen disponibles. Sin embargo, al momento de optimizar los costos de los recorridos se suele presentar subrutas que generan más circuitos de los necesarios. Esto hace que la solución a pesar de ser la óptima, no sea una solución viable, por lo que se necesitan nuevas restricciones para evitar tener esas soluciones. Para esto se puede tomar entre las restricciones del tipo DFJ o TMZ. Bazrafshan et al. (2021)

Formulación de Danzig–Fulkerson–Johnson (DFJ)

En el año 1954 los autores Danzig–Fulkerson–Johnson en el artículo Solution of a Large-Scale Traveling-Salesman Problem proponen la primera formulación de un problema de

programación lineal entera. Schittekat et al. (2006)

$$\sum_{i \in Q} \sum_{j \in Q} y_{ij} \leqslant |Q| - 1, \quad \forall Q \subseteq \{1, \ldots, n\} \wedge 2 < |Q| \geqslant n - 1 \tag{2.1}$$

Al agregar estas restricciones, se evita tener subrutina para cada subconjunto Q, los cuales son subconjuntos de vertices que tienen cardinalidad entre 3 y un $n - 1$. Esto debido a que la cantidad mínima de nodos que permiten generar un tour es de 3. La variable y_{ij} es una variable binaria que se usa para saber si los nodos i, j son visitados.

Un detalle sobre este método para eliminar los subrutina es el hecho que para realizar la restricción se necesita usar todos los subconjuntos de un conjunto de nodos que en este caso serian las paradas de bus. Siendo que se tiene al conjunto de paradas de autobus como V, entonces el conjunto que se busca es $\mathcal{P}(V)$. Por lo tanto, el problema combinatorio aumenta el número de restricciones de forma exponencial. Esto debido a que la cardinalidad de $\mathcal{P}(V)$ es de 2^n. Lo que ocasiona que la metodología de la formulación DFJ no sea la ideal en caso de tener instancias de gran escala.

Formulación de Miller–Tucker–Zemlin (TMZ)

Una de las formulaciones para el problema del TSP fue dada por los autores Miller, Tucker y Zemlin en el año 1960 en el artículo Integer programming formulations and travelling salesman problems. En este a su vez plantean las restricciones para eliminar subrutina de la siguiente manera. Bektaş and Elmastaş (2007)

$$u_i - u_j + n y_{ij} \leqslant n - 1 \quad \forall j \neq 1 \wedge j \neq i \tag{2.2}$$

En donde u_i y u_j corresponden a variables enteras que definen el orden con el que los nodos son visitados en el recorrido.

La variable y_{ij} hace referencia a una variable binaria la cual es 1 en caso de que los nodos i, j sean visitados, 0 sino.

Esto indica que las restricciones TMZ generan etiquetas para los distintos nodos se modo que estos seguir una secuencia entre ellos, eliminando asi las subrutinas que se generaron.

En el caso del SBRP las restricciones TMZ quedan de la siguiente manera: Las variables u_i representan la ”posición” (orden de visita) de la parada i en la ruta del vehículo k. La restricción MTZ fuerza a que si el vehículo k va desde la parada i a la parada j (i.e., $x_{ijk} = 1$), entonces la posición de la parada j debe ser al menos una unidad mayor que la posición de la parada i.

Entonces, esta restricción debe ser aplicada para todos los pares de paradas $i, j \in V$ con $i \neq j$ y para todos los vehículos k, con la excepción de la parada inicial (generalmente asumida como la parada 0) ya que esta parada no tiene una posición previa. Además, n es el número total de paradas. Por lo tanto, sus restricciones se verían así:

$$u_{ik} - u_{jk} + nx_{ijk} \leqslant n - 1, \quad \forall i, j \in V, i \neq j, i \neq 0, j \neq 0, k = 1, \ldots, K \tag{2.3}$$

Las variables de posición (orden) u_{ik} también deben ser limitadas para asegurar que tomen valores significativos. Así, se añadieron las siguientes restricciones :

$$1 \leqslant u_{ik} \leqslant n, \quad \forall i \in V \backslash 0, \quad \forall k \in K : y_{ik} = 1 \tag{2.4}$$

$$u_{ik} = 0, \quad y_{ik} = 0 \quad i \in V, k \in K \tag{2.5}$$

Por último, se añadió una restricción que fuerza a la parada inicial a tener la posición (orden) 1 para cada vehículo:

$$u_{0k} = 1, \quad \forall k = 1, \ldots, K \tag{2.6}$$

Formulación DFJ modificada

Las restricciones para romper las subrutinas de DFJ añaden una gran complejidad computacional al problema del VRP, siendo esta la parte combinatoria del mismo. Con eso en mente una forma de poder aplicar las restricciones, pero con una menor carga computacional sería usar utilizar el concepto de lazy contrains, tal que se puedan agregar únicamente las restricciones de los subconjuntos necesarios para romper la subrutina que se generan en la solución.

Para esto, se necesita una solución inicial la cual tendrá subrutinas, luego por vehículo se buscan las subrutinas y en específico los subconjuntos que representan para $V\backslash 0$ tal que se escriben únicamente las restricciones correspondientes a ese subconjunto, de modo que al finalizar de añadir las restricciones se vuelve a resolver el problema. Para esto se utiliza la misma versión de la ecuación 1.5, haciendo el respectivo cambio con el conjunto S que representa a los subconjuntos de V. Aun así, al realizarlo de esta manera no brindara una solución de manera inmediata al igual que DFJ o TMZ, en este caso se tendría que iterar el proceso hasta obtener una solución que vaya acorde al problema.

Formulación Gavish–Graves (GG)

Para el caso de la formulación del TSP asimetrico se tiene la formulación conocida como comodidad de flujo. De estas se tienen algunas variaciones como la de comodidad única, comodida doble y comodidad multiple.

La formulación GG para un problema de una comodidad que incluye las restricciones de

subrutina son:

$$\sum_{j=1} z_{ij} - \sum_{j\neq 1} z_{ij} = 1 \quad i = 2, \ldots, n \tag{2.7}$$

$$z_{ij} \leqslant (n-1)\, y_{ij}, \quad i = 2, \ldots, n \wedge j = 1, \ldots, n \tag{2.8}$$

$$z_{ij} \geqslant 0 \quad \forall i, j \tag{2.9}$$

Para esto se tiene que z es una variable positiva y yij es una variable binaria tal que es igual a 1 cuando los nodos i, j son visitados y 0 sino. Bazrafshan et al. (2021)

Comparativa de las restricciones DFJ, TMZ y GG

Para realizar el trabajo se consideró las caracterísitcas de cada una de las restricciones. Por ejemplo, las restricciones del tipo TMZ y GG representan una menor cantidad de restricciones lo que implica que trabajan mejor con una instancia de mayor escala. Sin embargo, las restricciones del tipo DFJ brinda soluciones bastante cercanos a lo más óptimo.

Para el caso de las restricciones TMZ generaron un resultado cercano al óptimo. Sin embargo, estas generar muchas variables enteras, lo que aumenta la complejidad del problema.

Por último, en el caso de la formulación GG no brinda una solución tan cercana a lo óptimo, pero esta solución se obtiene con un menor número de restricciones. Además, de no generar variables enteras.

En el caso de la formulación DFJ modificada se debe mencionar que no necesariamente se genera un número fijo de restricciones para cada instancia. Con eso en mente, para el desarrollo del proyecto se hace uso de tres de estos tipos de restricciones, tanto las DFJ, TMZ y DFJ modificadas con la intención de poder compararlas entre ellas y analizar cuales podrian brindar una solución eficaz en mejor tiempo para un problema como el de las instituciones eduactivas.

CAPÍTULO 3

3. RESULTADOS Y ANÁLISIS

Un propósito del trabajo es poder resolver el problema del SBRP con un algoritmo de solución exacta para una instancia que se pueda considerar pequeña. En este caso las pruebas se realizaron utilizando Python con la librería de Docplex que tiene a CPLEX funcionando como solver en un computador de características: Intel(R) Core(TM) i7-4510U CPU @ 2.00GHz 8.00 GB RAM, cuyo sistema operativo es Windows de 64 bits.

Al querer conocer cómo se comportaría el modelo para este tipo de problemas, se generaron datos usando lo siguiente. Primero, se distribuyeron los puntos que servirán como estación de bus uniformemente en el espacio definido y luego para cada una de estas estaciones de bus, conociendo que cada estudiante tiene una distancia máxima en metros que este puede caminar, se define un radio con esa distancia. Tal que, la ubicación de la estudiante seria asignada de forma aleatoria dentro de ese radio máximo. Esto nos permitió realizar distintas pruebas para el problema y verificar que los modelos de optimización sean viables como método de solución para las instancias que una institución educativa pueda necesitar.

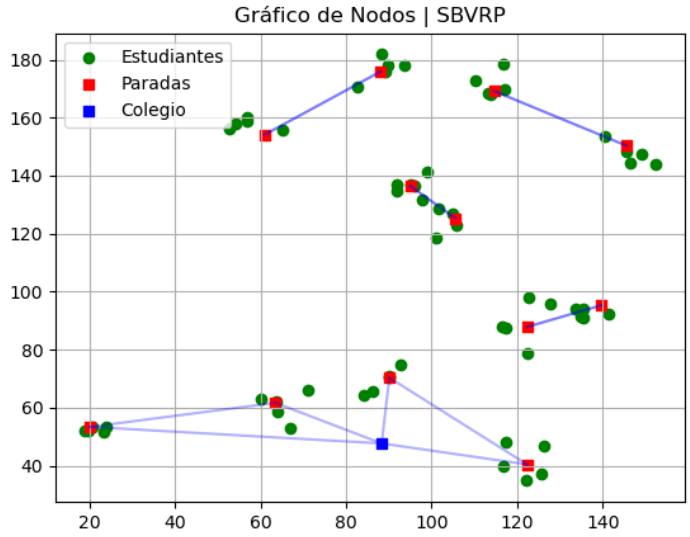
3.1 Pruebas para instancias pequeñas

Al tener un planteamiento del modelo para el SBRP ya definido previamente en el capítulo 1, se procede a escribirlo en un código de Python. Para esto se agrega tanto la función objetivo 1.2 como las distintas restricciones para el tipo de problema incluyendo desde la ecuación 1.3 hasta la ecuación 1.11. Sin embargo, en esta primera solución no se considera la ecuación 1.5 perteneciente a la restricción para romper a las subrutinas.

Como primera instancia a resolver en forma de prueba para el modelo, se tiene un total de 12 estaciones de buses. Con un total de 60 estudiantes y 2 buses con una capacidad máxima de 30 estudiantes por cada bus.

Una vez realizada esa primera prueba se tiene una primera respuesta que se considera óptima para el problema. Sin embargo, no es el tipo de respuesta que se esperaría como una solución. Debido a que no es algo que se pueda cumplir de forma realista.

Figura 3.1.
Solución del modelo sin incluir restricciones para romper subrutinas, Fuente: creación propia



Lo mostrado en la figura 3.1 es una solución óptima para el problema, pero no es enteramente lo que se espera como la solución. Pues implicaría que un mismo bus escolar tendría que hacer un recorrido que incluya salir de la escuela por un estudiante A, volver a la escuela y luego volver a salir por otro estudiante B.

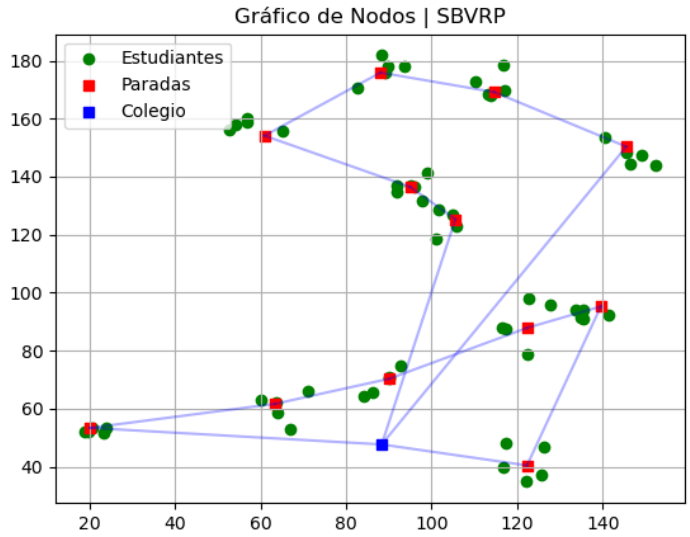
Tabla 3.1.
Nodos activos por autobus, , Fuente: creación propia

Bus escolar	Nodos Activos
1	(8, 2), (7, 4), (0, 8), (4, 7), (9, 6), (2, 0), (6, 9)
2	(5, 0), (1, 5), (12, 10), (10, 12), (3, 11), (0, 1), (11, 3)

Por ello, se utilizan las distintas formulaciones para poder romper estas subrutinas presentes en la solución.

De esto se implementan dentro del algoritmo de solución unas restricciones del tipo DFJ con las que al observar la solución graficada se puede notar que precisamente, esta no presenta subrutinas en ninguno de los recorridos.

Figura 3.2.
Solución óptima para instancia de 12 nodos, Fuente: creación propia



Sin embargo, hubo aspectos que mostraron cambios con respecto a la primera solución. Entre estos por ejemplo uno de los más notorios fue el tiempo de solución.

Tabla 3.2.
Detalles de las soluciones para el DFJ y respuesta con subtours, Fuente: creación propia

Formulación	Tiempo (sec)	N. variables	N. restricciones	N. variables binarias	Costo
DFJ	37.937	1778	51439	1778	637.157
Sin formulación	0.200	1778	2287	1778	452.459

Luego, se realizo una comparación de esta instancia particular con las otras formulaciones para romper las subrutinas de DFJ modificada y del tipo TMZ. Esto con la intención de conocer si la solución obtenida varia en alguno de ellas y además de conocer cual nos permite resolver la instancia en el menor tiempo posible, una vez se finaliza de correr los distintos procesos se puede

obtener el siguiente resumen para los tres métodos

Tabla 3.3.
Comparativa entre los distintos algoritmos de solución, Fuente: creación propia

Formulación	Tiempo (sec)	N. variables	N. restricciones	N. var. enteras	N. var binarias	Costo
Sin formulación	0.200	1778	2287	0	1778	452.459
DFJ	37.937	1778	51439	0	1778	637.155
TMZ	10.391	1804	2603	26	1778	637.155
DFJ modificada	12.356	1778	2353	0	1778	637.155

De esto obtenemos que para la instancia especifica, con la formulación de TMZ se pudo obtener una reducción del 63.14% en el tiempo de solución con respecto a la formulación DFJ y en el caso de la formulación DFJ modificada se obtuvo un 51.41% de mejora con respecto a la DFJ completa.

Tambien se intentó probar resolver el problema para una instancia un poco más grande como en el caso de 20 nodos, 80 estudiantes y 3 buses escolares con una capacidad de 30 estudiantes por cada bus.

Al utilizar los tres algoritmos se llega a una solución óptima con dos de estos. Los detalles se muestran en la tabla 3.4:

Tabla 3.4.
Comparativa entre los distintos algoritmos de solución, Fuente: creación propia

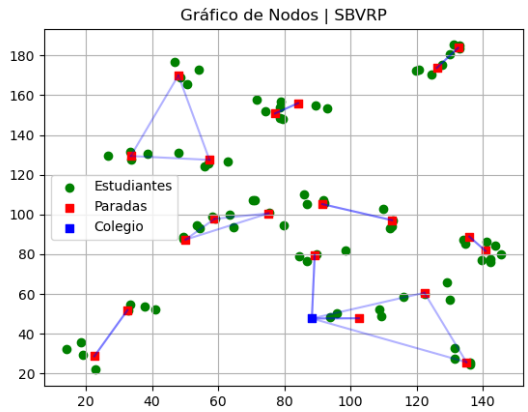
Formulación	Tiempo (sec)	N. variables	N. restricciones	N. var. enteras	N. var binarias	Costo
Sin formulación	0.410	6123	6630	0	6123	540.862
DFJ	-	-	Out of memory	-	-	-
TMZ	1167.955	6186	7896	63	6123	839.739
DFJ modificada	1255.155	6123	6963	0	6123	839.739

Como se puede ver, en este caso uno de los detalles más evidente es con respecto a la formulación DFJ, en esta ocurrió un error conocido como out of memory al momento generar los subconjuntos con los cuales luego trabajaran las restricciones. Por lo tanto, con este método no se pudo comenzar a llegar a resolver el problema, ni llegar a una solución posible no óptima.

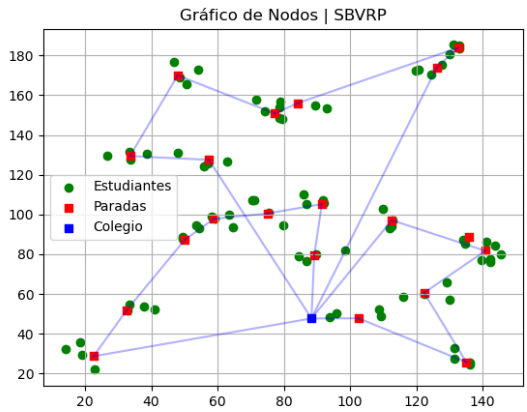
Por otra parte, en el caso de las restricciones del tipo TMZ y las DFJ modificadas se consiguieron soluciones óptimas para la instancia dada, llegando los dos métodos a la misma solución sin subrutinas. De esto se puede ver como de hecho nuevamente al utilizar las restricciones TMZ para el problema se tiene un mejor rendimiento centrandonos directamente en el tiempo que le toma en llegar a una solución para el problema. Puesto que en este caso el TMZ logra tener una ventaja de 87.2 segundos.

Figura 3.3.
Soluciones para la instancia de 20 nodos, Fuente: creación propia

(a)
Solución con subrutinas



(b)
Solución utilizando TMZ y DFJ modificado



Sin embargo, el tiempo que se presenta solo considera el proceso de solución del modelo con CPLEX. Al usar la formulación DFJ modificada, se debe considerar que este es un proceso iterativo que dentro de cada repetición resuelve el modelo con las restricciones, ademas de otros procesos. Entre ellos uno de los principales para el funcionamiento es la busqueda de los subtours. Para esto se hace uso de la libreria Networkx de Python. En esta se usa una función que permite conocer los ciclos cerrados que se forman en un grafo, en este caso es la ruta de cada bus escolar. Una vez identificados, se los debe de agregar como restricciones del modelo de optimización con el fin de eliminar esa subrutina.

3.2 Pruebas instancias medianas

Realizando test para el modelo utilizando las restricciones del tipo TMZ, debido a que fue la que obtuvo mejor desempeño en las pruebas realizadas previamente. Se pudo tener algunos

resultados para distintas instancias. Donde se mantenía con tiempo constantes para problemas con un tamaño de 15 estaciones de bus.

Sin embargo, se intento probar con distintas instancias generadas con las siguientes caracteristicas:

- 25 estaciones de bus, 125 estudiantes y 5 buses escolares con capacidad para 30 estudiantes cada uno
- 30 estaciones de bus, 120 estudiantes y 6 buses escolares con capacidad para 25 estudiantes cada uno
- 40 estaciones de bus, 200 estudiantes y 10 buses escolares con capacidad para 25 estudiantes cada uno

Sin embargo, al intentar obtener una solución sin un límite de tiempo, el modelo generaba muchas restricciones por lo que mostraba un error de Out of memory

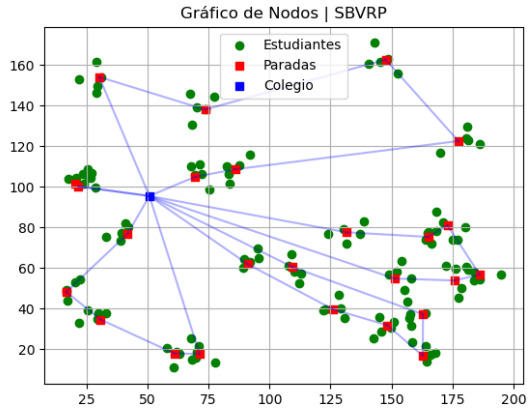
Para poder obtener respuestas de esta instancia se establece un tiempo limite de 43200 segundos. De modo que si el modelo de optimización no ha encontrado la solución óptima para el problema, entonces pasara a mostrar la mejor respuesta entre las respuestas que son factibles. pero sin la certeza de que sea la solución óptima del problema.

Tabla 3.5.
Pruebas de formulación TMZ para instancias de 25, 30 y 40 estaciones de bus, Fuente: creación propia

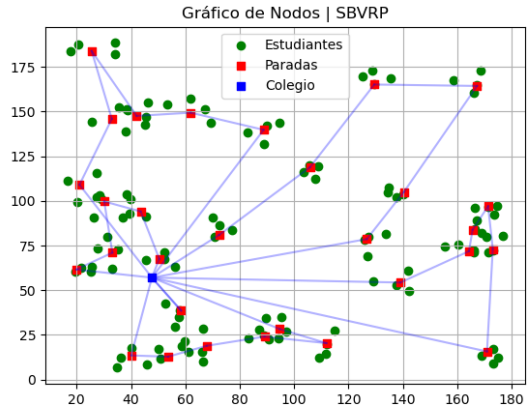
Instancia	Tiempo (sec)	N. variables	N. restricciones	N. var. enteras	N. var binarias	Costo
25 estaciones	43228.700	19135	22426	130	19005	1211.971
30 estaciones	43240.400	27552	31321	186	27366	1389.611
40 estaciones	14654.160	-	Out of memory	-	-	-

Figura 3.4.
Soluciones para las instancias de 25 y 30 estaciones de bus, Fuente: creación propia

(a)
25 estaciones de bus, 125 estudiantes



(b)
30 estaciones de bus, 120 estudiantes



CAPÍTULO 4

4. CONCLUSIONES Y RECOMENDACIONES

4.1 Conclusiones

La mejora de las rutas escolares mediante la implementación de soluciones exactas es una posibilidad real y efectiva. A lo largo de este análisis, se han identificado varios aspectos clave que respaldan esta afirmación y se ha evaluado si se cumplieron los objetivos establecidos.

- Adaptación al Contexto Educativo: El primer objetivo era evaluar si las soluciones de enrutamiento vehicular pueden adaptarse de manera efectiva a las condiciones específicas de una institución educativa. En este sentido se ha mostrado que es esencial tener en cuenta las restricciones y recursos únicos de cada escuela al seleccionar un modelo de solución. Se logró personalizar el enfoque para que sean adaptables a los desafíos particulares de cada entidad, lo que respalda la efectividad de las iniciativas de mejora de rutas escolares.

- Eficacia del Algoritmo ”Branch and Cut”: El segundo objetivo era proponer y evaluar el algoritmo ”Branch and Cut” como un método para alcanzar una solución exacta para optimizar las rutas de autobuses escolares. En este aspecto, se ha observado que este algoritmo es capaz de optimizar el enrutamiento vehicular con gran precisión. Se ha demostrado su aplicabilidad efectiva en la minimización de costos y tiempos de viaje, lo

que beneficia a estudiantes, padres y personal educativo. Sin embargo, es importante señalar que su eficacia está limitada por la capacidad computacional disponible y la complejidad en la formulación del problema seleccionada.

- Desafíos en la Escalabilidad: Finalmente, se ha evaluado si las soluciones son escalables para resolver problemas de mayor tamaño. Aunque los algoritmos exactos funcionan de manera óptima en instancias pequeñas, enfrentan desafíos significativos en términos de escalabilidad. El aumento en la complejidad y el tamaño del problema exige una cantidad significativa de recursos computacionales, lo que puede resultar en tiempos de ejecución prolongados o incluso la infeasibilidad/no factibilidad de resolver instancias moderadamente grandes. Este desafío debe ser abordado en futuras investigaciones.

4.2 Recomendaciones

- Se recomienda el uso de la formulación TMZ para la resolución del SVRP, especialmente en instancias de tamaño mediano o grande. Esta formulación ha demostrado ser más eficiente en términos de tiempo de ejecución y capacidad para obtener soluciones óptimas en comparación con las formulaciones DFJ. Además, su enfoque basado en orden permite adaptar el problema si se consideran zonas de tiempo, lo cuál lo hace adecuado para problemas con restricciones temporales.

- La mejora de las rutas escolares puede ser un proceso incremental. En lugar de tratar de resolver todo el problema de una sola vez, se puede comenzar por optimizar una parte del problema, como la asignación de estudiantes a paradas, y luego abordar gradualmente las rutas de los buses. Se puede utilizar un método exacto para resolver la asignación, mientras

que en la segunda parte, se pueden aplicar algoritmos de TSP aproximados o heurísticas eficientes para optimizar la ruta de cada vehículo. Este sería un enfoque con varias fases.

- Dado que el SBRP es un problema complejo y en constante evolución, se sugiere que la investigación futura se enfoque en el desarrollo de algoritmos híbridos que combinen enfoques exactos con heurísticas eficientes. También se pueden explorar técnicas de aprendizaje automático para la toma de decisiones en tiempo real y la adaptación dinámica de las rutas escolares en función de las condiciones cambiantes.

BIBLIOGRAFÍA

Achterberg, T., Koch, T., and Martin, A. (2005). Branching rules revisited. *Operations Research Letters*, 33(1):42–54.

Arias-Rojas, J. S., Jimenez, J.-F., and Montoya-Torres, J. R. (2012). Solving of school bus routing problem by ant colony optimization.

Bazrafshan, R., Zolfani, S., and Mirzapour Al-e hashem, S. (2021). Comparison of the sub-tour elimination methods for the asymmetric traveling salesman problem applying the seca method. *Axioms*, 10:19.

Bektaş, T. and Elmastaş, S. (2007). Solving school bus routing problems through integer programming. *Journal of The Operational Research Society - J OPER RES SOC*, 58:1599–1604.

Corona León, J. A. (2005). Hiperheurísticas a través de programación genética para la resolución de problemas de ruteo de vehículos.

Dantzig, G. B. and Ramser, J. H. (1959). The truck dispatching problem. *Management Science*, 6:80–91.

Escobar Morales, D. F., Gaviria Cano, J. E., and Orejuela Cabrera, J. P. (2018). Método de tres fases para la solución del ruteo de buses escolares. *Revista Espacios*, 39(50):6.

Hou, Y.-e., Gu, W., Wang, C., Yang, K., and Wang, Y. (2022). A selection hyper-heuristic based on q-learning for school bus routing problem. *IAENG International Journal of Applied Mathematics*, 52(4):817–825.

Padberg, M. and Rinaldi, G. (1991). A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Review*, 33(1):60–100.

Park, J. and Kim, B.-I. (2010). The school bus routing problem: A review. *European Journal of Operational Research*, 202(2):311–319.

Parra, O. D. and Chavez, M. C. (1998). El problema del transporte. *Centro de Investigación en Ingeniería y Ciencias Aplicadas, Cuernavaca. Morelos./Papadimitriou, CH, Steiglitz, K. Combinatorial*.

Pferschy, U. and Staněk, R. (2017). Generating subtour elimination constraints for the tsp from pure integer solutions. *Central European journal of operations research*, 25:231–260.

Schittekat, P., Sevaux, M., and Sörensen, K. (2006). A mathematical formulation for a school bus routing problem. volume 2, pages 1552 – 1557.

APÉNDICES

APÉNDICE A

A.1 Implementación de las Condiciones TMZ para SBRP

En esta sección, se proporciona una descripción detallada de las condiciones TMZ adaptadas para abordar el Problema de Enrutamiento de Vehículos Escolares (SBRP) que fue tratado en este documento.

A.1.1 Nuevas variables introducidas

u_ik: Variable de decisión que representa la posición de un vehículo escolar k en relación con la parada i. Esta variable es un número entero que indica el orden en que el vehículo k visita la parada i.

A.1.2 Restricciones TMZ

Las restricciones TMZ se implementan como sigue:

3. Breaking subtour constraints (TMZ)

$$u_{ik} - u_{jk} + Mx_{ijk} \leqslant M - 1, \quad \forall i, j \in V, i \neq j, j \neq 0, k = 1, \ldots, K$$

Donde: u_{ik} y u_{jk} son las variables u que representan la posición de los vehículos k en las paradas i y j, respectivamente. x_{ijk} es la variable binaria que indica si el vehículo k viaja directamente desde la parada i a la parada j. M representa el número de paradas. Estas restricciones aseguran que no se formen subtours, es decir, rutas que no visiten todas las

paradas requeridas. Cada vez que un vehículo pasa de una parada i a una parada j (donde i y j son diferentes y diferentes de 0), la variable $u$correspondiente se actualiza para garantizar que el vehículo continúe su ruta de manera adecuada.

A.1.3 Regularización de Variables u

Además de las restricciones TMZ, es importante regularizar las variables u para garantizar que si un vehículo k no pasa por una parada i, la variable u_{ik} se establezca en cero. Esto se logra con la siguiente restricción:

$$u_{ik} \leqslant M \cdot y_{ik}, \quad \forall i \in V, \forall k \in K$$

A.1.4 Inicio de Rutas desde la Escuela

Para asegurarse de que todos los vehículos comiencen su ruta en la parada cero (escuela), se establece la siguiente restricción:

$$u_{0k} = 0, \quad \forall k \in K$$

Esta restricción garantiza que los vehículos escolares empiecen y terminen su ruta en la escuela, cumpliendo con los requisitos del SBRP.

```
M=len(V)
# Variables relevantes
u = {(i, k): mdl.integer_var(lb=0, ub=M-1, name='u_{0}_{1}'.format(i, k))
    for i in V for k in K}

# Restricción de regularización para u
for i in V:
    for k in K:
        mdl.add_constraint(u[i, k] <= M * y[i, k])

# Restricciones TMZ
for i in V:
    for j in V:
        if i != j and j != 0:
            for k in K:
                mdl.add_constraint(u[(i, k)] - u[(j, k)] + M * x[(i, j, k)]
                    <= M - 1)

# Inicio de rutas desde la escuela
for k in K:
    mdl.add_constraint(u[(0, k)] == 0)
```

Código A.1. Implementación con docplex de las restricciones TMZ para SBRP

A.2 Implementación de las Condiciones DFJ

```
from itertools import combinations
subconjuntos = []
for i in range(len(V[1:]) + 1):
    subconjuntos.extend(list(combinations(V[1:], i)))
for s in subconjuntos[1:]:
    for h in s:
        for d in K:
            mdl.add_constraint(mdl.sum(x[i,j,d] for i in s for j in V if j
                not in s) >= y[s[0],d] )
```

Código A.2. Implementación con docplex de las restricciones DFJ para SBRP

A.3 Implementación de las Condiciones DFJ modificadas (lazy constrains)

```
bus_arcos={}
for d in K:
    arcos = [(i, j) for (i, j, k) in x if x[(i, j, k)].solution_value > 0.9
        and k == d]
    bus_arcos[d] = arcos

stud_stop={}
for j in V[1:]:
    estu = [l for (i,l,k) in z if z[(i,l,k)].solution_value>0.9 and i==j]
    stud_stop[j] = estu

#se genera el primer subconjuntos necesario
import networkx as nx
restri = {}
for bus in bus_arcos:
    G=nx.DiGraph()
    G.add_edges_from(bus_arcos[bus])
    ciclos = list(nx.simple_cycles(G))
    for ciclo in ciclos:
        if 0 in ciclo:
            ciclos.remove(ciclo)
    if len(ciclos)>=1:
        restri[bus] = ciclos
numrestri = 0
```

```
for res in restri:
    numrestri+= len(restri[res])
itera=1
while restri != {}:

    #agregamos las restricciones al modelo
    for bus in restri:
        for subconj in restri[bus]:
            for d in K:
                mdl.add_constraint(mdl.sum(x[i,j,d] for i in subconj for j
                    in V if j not in subconj) >= y[subconj[0],d] )

    #solucionamos el problema
    solucion=mdl.solve(log_output=True)

    #sumamos el tiempo
    tiempo += mdl.get_solve_details().time

    #diccionarios de solucion
    bus_arcos={}
    for d in K:
        arcos = [(i, j) for (i, j, k) in x if x[(i, j, k)].solution_value >
            0.9 and k == d]
        bus_arcos[d] = arcos

```

```
47      #volvemos a ver cuantos subtours hay para volver a repetir
48      restri = {}
49      for bus in bus_arcos:
50          G = nx.DiGraph()
51          G.add_edges_from(bus_arcos[bus])
52          ciclos = list(nx.simple_cycles(G))
53          for ciclo in ciclos:
54              if 0 in ciclo:
55                  ciclos.remove(ciclo)
56          if len(ciclos)>=1:
57              restri[bus] = ciclos
58      #volvemos a contar mas subconjuntos a las restricciones
59      for res in restri:
60          numrestri+= len(restri[res])
61
62      #condicion para romper el while
63      if restri == {}:
64          break
65      itera+=1
```

Código A.3. Proceso iterativo modificado para las condiciones DFJ