Icono

Descripción generada automáticamente

**UNIVERSIDAD ANDRÉS BELLO**

**FACULTAD DE INGENIERÍA**

**INGENIERIA CIVIL INFORMÁTICA**

**ANALISIS, OPTIMIZACIÓN Y DOCUMENTACIÓN DE VRP**

**Tesis de pregrado para optar al título de Ingeniero Civil Informático**

Bastian Ignacio Moya Carrasco

Felipe Ismael Moya Carrasco

Profesor tutor:

Dr. Gustavo Gatica

Santiago, Chile

2024

# Resumen

Este trabajo se centra en replicar y evaluar el modelo presentado en el paper *"The finished vehicle routing problem with a heterogeneous transport fleet"* (HVRP–FVL), que aborda una variante del problema de ruteo de vehículos. Este modelo considera una flota heterogénea en logística de vehículos terminados e incorpora diversos costos de transporte, como peajes, costos laborales, consumo energético, penalizaciones por tiempo y emisiones de carbono. El objetivo del HVRP–FVL es minimizar el costo total de la flota mediante una metaheurística basada en algoritmos genéticos (GA) con codificación dual, cruzamiento adaptativo, mutación y operadores de escalada.

En el desarrollo de este trabajo, se revisaron tres versiones de este estudio, junto con fuentes adicionales, para esclarecer variables no declaradas o mal dimensionadas. A pesar de las inconsistencias encontradas en las fórmulas y valores reportados, se trabajó en replicar mecánicamente los casos de estudio provistos en el paper. Esto permitió evaluar la funcionalidad del modelo a nivel práctico, aunque no fue posible obtener los valores de fitness esperados ni validar completamente el modelo matemático.

El objetivo principal fue diseñar y replicar, mediante el uso de Python, la funcionalidad propuesta, con la que se buscó rutas óptimas que minimizaran el uso de camiones, garantizaran la puntualidad dentro de las ventanas de tiempo y maximizasen la capacidad de carga de cada vehículo. Si bien no se logró corroborar la veracidad total del modelo teórico, se obtuvieron rutas optimizadas que cumplen con los criterios logísticos establecidos, demostrando la aplicabilidad de este enfoque en escenarios reales de transporte.

# Índice

[Resumen 1](#_Toc183413459)

[Índice 2](#_Toc183413460)

[Tabla de contenidos 3](#_Toc183413461)

[Palabras Clave 4](#_Toc183413462)

[Introducción 5](#_Toc183413463)

[Marco teórico 6](#_Toc183413464)

[Evolución Histórica de la formulación de VRP 6](#_Toc183413465)

[Metodología 7](#_Toc183413466)

[Funcionalidades del Software 9](#_Toc183413467)

[Variables especiales: 9](#_Toc183413468)

[Proceso de asignación: 9](#_Toc183413469)

[Salida de datos: 9](#_Toc183413470)

[Implementación Técnica 9](#_Toc183413471)

[Modelo Matemático 10](#_Toc183413472)

[Descripción general: 10](#_Toc183413473)

[Conjuntos y parámetros: 10](#_Toc183413474)

[Variable de decisión: 10](#_Toc183413475)

[Función Objetivo: 10](#_Toc183413476)

[Restricciones: 10](#_Toc183413477)

[Código en Python 12](#_Toc183413478)

[Código uso mutuo 12](#_Toc183413479)

[Código uso general 15](#_Toc183413480)

[Código comprobación casos de estudio 16](#_Toc183413481)

[Resultados 17](#_Toc183413482)

[Conclusiones 19](#_Toc183413483)

[Referencias 20](#_Toc183413484)

# Tabla de contenidos

[Ilustración 1 Modelos originarios del problema VRP 6](#_Toc183413485)

[Ilustración 2 Tiempos y Direcciones 12](#_Toc183413486)

[Ilustración 3 Función fitness 1 13](#_Toc183413487)

[Ilustración 4 Función fitness 2 13](#_Toc183413488)

[Ilustración 5 Función fitness 3 14](#_Toc183413489)

[Ilustración 6 Fitness total y probabilidades de cambios 14](#_Toc183413490)

[Ilustración 7 Solución optimizada de acuerdo con el paper 17](#_Toc183413491)

[Ilustración 8 Solución optimizada del código creado 17](#_Toc183413492)

[Ilustración 9 Solución optimizada de 100 ordenes 18](#_Toc183413493)

[Ilustración 10 Solución de 100 órdenes del código creado 18](#_Toc183413494)

# Palabras Clave

\*VRP: Vehicle routing problem (Problema de Ruteo de Vehículos).

\*Heurísticas: Atajos que otorgan decisiones más rápidas que métodos tradicionales de computación.

\*Pseudocódigo: Descripción de un algoritmo o programa escrito en lenguaje natural estructurado.

\*Fitness: Características que miden la calidad del código.

# Introducción

Este informe presenta el análisis, optimización y documentación del código implementado para resolver el **Vehicle Routing Problem (VRP)**, tomando como base el artículo [*"The Finished Vehicle Routing Problem with a Heterogeneous Transport Fleet"*](https://www.researchsquare.com/article/rs-2942819/v1). Este artículo sirvió como referencia principal para la realización de este trabajo y la presente tesis.

El artículo original aborda una variante del VRP enfocada en la planificación de rutas óptimas para la distribución de productos mediante una flota heterogénea de vehículos. Cada vehículo parte desde un único punto, o centro de distribución, hacia múltiples clientes, considerando diversas restricciones relacionadas con los costos, las características de los vehículos a llevar, los propios camiones y las ventanas de tiempo. Aunque el artículo proporciona un esquema en pseudocódigo, este no detalla completamente la implementación de ciertos aspectos del modelo ni especifica cómo se valida el fitness esperado.

Sabiendo eso, se realizó un análisis exhaustivo del artículo y se diseñó un código funcional desde cero, incorporando mejoras y ajustes necesarios para adaptar el modelo a un escenario práctico. Durante este proceso, se logró replicar las rutas optimizadas siguiendo el esquema presentado en el artículo, aunque con algunas diferencias en los valores de Fitness obtenidos.

El objetivo principal es desarrollar un algoritmo para el VRP que optimice las rutas de una flota heterogénea de vehículos, considerando diversas restricciones operativas y logísticas:

* **Capacidad máxima de carga**: Cada vehículo tiene un límite de carga, tanto en términos de peso como de dimensiones, que no debe ser excedido.
* **Ventanas de tiempo**: Cada cliente debe ser atendido dentro de un intervalo de tiempo específico.
* **Cobertura de clientes**: Cada cliente debe ser visitado exactamente una vez con su entrega completa, y cada vehículo debe regresar al centro de distribución después de completar sus entregas.

A través de este trabajo, se busca comprobar la eficacia del modelo propuesto en el artículo y su aplicabilidad en contextos logísticos reales.

# Marco teórico

## Evolución Histórica de la formulación de VRP

Los problemas tipo VRP ha recibido varias variaciones desde la primera vez que fue planteado en 1956, pero siempre se sigue con la base de salir de un punto hacia los distintos destinos, lo cuales solo se visita una vez, para luego volver al inicio.

Desde el primer VRP planteado por Flood nacieron variaciones, como se muestra en la siguiente figura:

Diagrama

Descripción generada automáticamente

Ilustración 1 Modelos originarios del problema VRP

Cada una de las variaciones tenía distintos cambios, algunos viendo un cierto número de rutas, revisando el mínimo costo requerido u otros ligeros cambios entre ellos (Rocha Medina, L. B., González La Rota, E. C., & Orjuela Castro, J. A. (2011)).

[1] Rocha Medina, L. B., González La Rota, E. C., & Orjuela Castro, J. A. (2011). Una revisión al estado del arte del problema de ruteo de vehículos: Evolución histórica y métodos de solución. Ingeniería, 16(2), 35-55.

# Metodología

El desarrollo del presente trabajo se realizó utilizando el lenguaje de programación **Python** en el entorno de desarrollo **Visual Studio Code**.

**Herramientas y entorno de desarrollo**

* **Lenguaje de programación**: Python fue elegido por su flexibilidad y su amplia disponibilidad de bibliotecas para procesamiento de datos, optimización y simulación.
* **Entorno de desarrollo**: Visual Studio Code, una herramienta versátil y ampliamente utilizada en el desarrollo de proyectos de software.
* **Gestión de datos**: Los datos necesarios para el problema provistos en el paper, como las características de los vehículos, los pedidos y las distancias entre nodos, se almacenaron en archivos de texto, permitiendo una fácil modificación y análisis.

**Proceso de desarrollo**

1. **Análisis preliminar**:
   * Se revisaron las inconsistencias y elementos faltantes en el artículo original. Este análisis reveló vacíos conceptuales y valores sin declarar, lo que requirió realizar múltiples ajustes y reelaboraciones durante el desarrollo del código.
2. **Diseño del modelo y lectura de datos**:
   * Se implementó un sistema para la lectura de datos desde archivos de texto, lo que permitió estructurar y validar la información básica del problema (capacidad de los vehículos, restricciones de carga, ventanas de tiempo, entre otros).
   * Se adaptaron las características del modelo al contexto del caso de estudio.
3. **Implementación de las funciones de fitness**:
   * Las funciones de fitness fueron diseñadas siguiendo los criterios planteados en el artículo de referencia. Estas funciones permiten evaluar la calidad de las soluciones generadas, considerando factores como:
     + Minimización del número de camiones.
     + Cumplimiento de ventanas de tiempo.
     + Maximización de la carga transportada por cada vehículo.
   * Las funciones fueron validadas a través de pruebas con datos reales.
4. **Optimización mediante algoritmos genéticos**:
   * El algoritmo genético implementado incluyó las siguientes etapas:
     + **Cálculo del fitness inicial**: Se evaluó la población inicial generada.
     + **Selección**: Se seleccionaron las soluciones con mayor probabilidad de mejorar el resultado.
     + **Cruzamiento y mutación**: Se aplicaron operadores adaptativos para explorar nuevas configuraciones de rutas.
     + **Actualización de la población**: Se integraron las nuevas soluciones generadas, repitiendo el ciclo hasta converger en una solución óptima.
   * Las probabilidades de cruzamiento y mutación se calcularon dinámicamente en cada iteración para garantizar un equilibrio entre exploración y explotación.
5. **Validación y ajuste del modelo**:
   * Se evaluaron las soluciones obtenidas con base en los resultados esperados y se ajustaron los parámetros del modelo según las observaciones.

**Resultados esperados**

El proceso descrito permitió comprobar 2 cosas:

* El funcionamiento del código en su función principal (cálculo de fitness, generación de poblaciones, etc.)
* El funcionamiento del código en los casos de estudios provistos, los cuales sirvieron como base para probar el mismo.

# Funcionalidades del Software

## Variables especiales:

* N1
* N2

## Proceso de asignación:

* Input de data: El software recibe la data a utilizar en los algoritmos desde varios archivos de texto (.txt).
* Evaluación y análisis: La evaluación de cada variable de vehículo o cliente se evaluará mediante un algoritmo desarrollado en Python para determinar la manera óptima de viaje y entrega de vehículos.
* Criterios de asignación: Los criterios de entrega serán dependientes de la carga máxima de los tipos de vehículos, considerando también las dimensiones de estos.

## Salida de datos:

* El programa retorna el trayecto optimo y junto con el valor de Fitness de cada parte ()

# Implementación Técnica

El software se desarrolló usando Python, junto con librerías de ampliación de funciones matemáticas, tratamiento de texto y copia de datos. Estas serían math, re y copy respectivamente, además de otras para funciones específicas.

* Entrada de datos: Los datos de los clientes, los camiones, autos y parámetros utilizados se extrajeron del paper y fuentes afines, y se introducen en el programa a través de archivos txt estructurados
* Creación variables N1 y N2: Utilizando los datos extraídos, el programa sigue las instrucciones del paper, creando N1 y N2 los cuáles serán las bases para el GA
* Generación rutas optimas: Habiendo hecho las n° iteraciones el programa le entrega al usuario la solución encontrada, el cual consiste en la ruta y el fitness.

# Modelo Matemático

Se utiliza el modelo matemático provisto en el archivo, pero con algunos cambios y aclaraciones:

## Descripción general:

El problema considera un conjunto de nodos que deben ser visitados por una flota heterogénea de vehículos con distintas dimensiones y cantidad . El objetivo es minimizar el costo total de transporte

## Conjuntos y parámetros:

* : Conjunto de vehículos.
* : Conjunto de nodos (clientes), donde es el depósito.
* ​: Capacidad máxima del vehículo.
* : Demanda del cliente (en toneladas).

* ​: Ventana de tiempo para iniciar el servicio en el cliente (inicio ​, fin ​).
* : Tiempo de viaje entre el nodo y el nodo .
* ​: Costo de viaje entre el nodo y el nodo .
* ​: Distancia entre el nodo y el nodo .
* ​: Demanda del cliente de vehículos tipo .
* , , ,: Limitaciones de carga, largo, ancho y alto del vehículo de tipo .

## Variable de decisión:

* : Variable binaria que toma valor 1 si el vehículo del tipo se encuentra en el arco , y 0 en caso contrario
* : Variable binaria que toma valor 1 si el vehículo del tipo visitara al cliente , 0 en caso contrario

## Función Objetivo:

La función objetivo es minimizar el coste de toda la operación

Donde son los costos de transporte, las penalizaciones de tiempo y los costes por emisiones de carbono.

## Restricciones:

* Asignación única: Los clientes deben ser visitados por un único vehículo, sin separar los envíos
* Conservación del movimiento: para cada cliente , el número de vehículos que entran debe ser igual al salir
* Restricción de capacidad: El peso total al unir clientes a una ruta no debe superar el peso permitido por el vehículo
* Restricción de espacio: El largo total al unir clientes a una ruta no debe superar el peso permitido por el vehículo

# Código en Python

## Código uso mutuo

A continuación, se presentarán las secciones más importantes de código usadas en ambos casos (general y comprobación de caso)

Texto

Descripción generada automáticamente

Ilustración 2 Tiempos y Direcciones

Texto

Descripción generada automáticamente con confianza baja

Ilustración 3 Función fitness 1

Texto

Descripción generada automáticamente

Ilustración 4 Función fitness 2

Texto

Descripción generada automáticamente con confianza media

Ilustración 5 Función fitness 3

Texto

Descripción generada automáticamente

Ilustración 6 Fitness total y probabilidades de cambios

De las figuras anteriores se puede observar el proceso para obtener cada función de Fitness, junto con los procesos para calcular la probabilidad de cruzamiento y de mutación de los nodos.

## Código uso general

Se presenta a continuación como está estructurado el código para el uso general

Texto

Descripción generada automáticamente

Ilustración 7 Creación N1 y N2, y tratamiento de datos

Texto

Descripción generada automáticamente

Ilustración 8 Estructuración GA

## Código comprobación casos de estudio

Por último, se muestra cómo se estructuro el código para comprobar los casos de estudio. En este caso solo se comprobó la lógica del código al recibir la misma entrada de N1 y N2, y se comprobó el funcionamiento lógico del mismo. En vez del código expresado en la ilustración 8, solo se comprobó directamente con las funciones decodificar\_pedidos y codificado real, de las cuales la primera introduce los 0 dentro de N1 y la segunda usando el N1 modificado, guarda todos los datos a usar en las funciones siguientes.

Texto

Descripción generada automáticamente

Ilustración 9 Caso de Estudio 21 clientes

Texto

Descripción generada automáticamente

Ilustración 10 Caso de Estudio 100 clientes

# Resultados

En el paper en el que se basa este trabajo se entregaron 2 casos de prueba con sus resultados. Por la particularidad de este trabajo, se tomó en cuenta la salida primaria, la cual es la ruta que tomaría cada camión, representado con los valores/ID´s de cada deposito, los cuales estarían entre Ceros, los cuales representan el depósito:

Tabla

Descripción generada automáticamente

Ilustración 7 Solución optimizada de acuerdo con el paper



Ilustración 8 Solución optimizada del código creado

Tabla

Descripción generada automáticamente

Ilustración 9 Solución optimizada de 100 ordenes

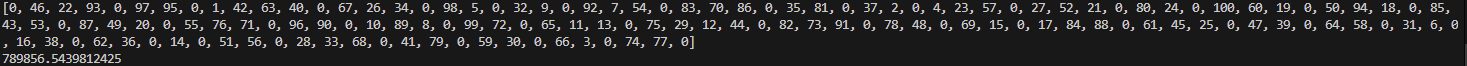


Ilustración 10 Solución de 100 órdenes del código creado

# Conclusiones

El análisis y desarrollo realizado en este trabajo permitieron implementar y optimizar un algoritmo basado en heurísticas para resolver el **Problema de Rutas de Vehículos (VRP)** con una flota heterogénea. Comparando los resultados obtenidos con los del artículo base, se evidencia que el algoritmo propuesto es capaz de replicar e incluso mejorar ciertas soluciones dadas por el paper original, respetando las restricciones establecidas, como la capacidad de los vehículos y las ventanas de tiempo.

1. **Resultados alcanzados**:
   * Se logró implementar un algoritmo genético desde cero, respetando las funciones de fitness descritas en el paper base. Esto permitió optimizar las rutas considerando múltiples restricciones, como la capacidad de los vehículos y las ventanas de tiempo de los clientes.
   * El algoritmo entregó soluciones comparables al artículo base en términos de las rutas a usar por cada camión, los que nos muestra que se logró el resultado esperado.
2. **Limitaciones**:
   * Aunque el algoritmo genético ofreció buenos resultados, el tiempo de ejecución podría ser mejorado mediante la implementación de optimizaciones adicionales, ya que los datos no clarificados por parte de la fuente pueden cambiar drásticamente el resultado de los ejemplos.
   * No se exploraron otras metaheurísticas, como el recocido simulado o algoritmos basados en colonias de hormigas, que podrían ofrecer perspectivas diferentes al problema.
3. **Propuestas futuras**:
   * Integrar herramientas como Gurobi para resolver el problema de manera exacta y comparar los resultados con los obtenidos mediante heurísticas.
   * Explorar métodos híbridos que combinen metaheurísticas y enfoques exactos para balancear mejor la calidad de las soluciones y el tiempo de ejecución.

En resumen, este trabajo demostró la viabilidad de aplicar algoritmos genéticos para resolver problemas complejos como el VRP con flota heterogénea, y sienta las bases para futuras investigaciones en optimización logística utilizando métodos computacionales avanzados.

# Referencias

* Rocha Medina, L. B., González La Rota, E. C., & Orjuela Castro, J. A. (2011). Una revisión al estado del arte del problema de ruteo de vehículos: Evolución histórica y métodos de solución. Ingeniería, 16(2), 35-55.
* Liu, Jiansheng Liu and Yuan, Bin and Hu, Yingcong and Smith, Alice E., The Finished Vehicle Routing Problem with a Heterogeneous Transport Fleet. Available at SSRN: <https://ssrn.com/abstract=4239703> or [http://dx.doi.org/10.2139/ssrn.4239703](https://dx.doi.org/10.2139/ssrn.4239703)
* Tolga Bektaş, Gilbert Laporte, The Pollution-Routing Problem, Transportation Research Part B: Methodological, Volume 45, Issue 8, 2011, Pages 1232-1250, ISSN 0191-2615, <https://doi.org/10.1016/j.trb.2011.02.004>. (<https://www.sciencedirect.com/science/article/pii/S019126151100018X>)
* Emrah Demir, Tolga Bektaş, Gilbert Laporte, A comparative analysis of several vehicle emission models for road freight transportation, Transportation Research Part D: Transport and Environment, Volume 16, Issue 5, 2011, Pages 347-357, ISSN 1361-9209, <https://doi.org/10.1016/j.trd.2011.01.011>. (https://www.sciencedirect.com/science/article/pii/S136192091100023X)