1. Preliminares

En este capitulo se presentan los elementos sobre los cuales se construye la presente investigación. En la sección 1.1 se presenta el Problema de Enrutamiento de Vehículos (VRP por sus siglas en ingles) asi como sus algunas de sus variantes. En la sección 1.2 y 1.3 se presentan conceptos relacionados con los criterios de vecindad y región de una vecindad, esenciales para la exploración de nuevas soluciones. En la sección 1.4 se introduce una metaheurística de búsqueda local, elemento de gran relevancia para el presente trabajo. Por último las secciones 1.5 y 1.6 se abordan los elementos de la biblioteca de software GLPK y el lenguaje de programación Lisp, ambas relevantes para el desarrollo de este trabajo.

1.1. Problema de Enrutamiento de Vehículos (VRP)

El Problema de Enrutamiento de Vehículos (VRP) es uno de los problemas de optimización combinatoria, más estudiado por su importancia tanto práctica como teórica. Formulado por Dantzig y Ramser en 1959 [7], el VRP consiste en diseñar rutas óptimas para una flota de vehículos que deben satisfacer la demanda de un conjunto de clientes dispersos geográficamente [19]. Estos vehículos parten y regresan a un depósito, con el objetivo de minimizar un criterio de costo, como la distancia total recorrida [3]. El VRP tiene aplicaciones prácticas en áreas como la logística, el transporte urbano y la distribución de bienes en grandes cadenas de suministro [9, 20, 21].

Una característica esencial del VRP es su clasificación como problema NP-duro, lo que significa que encontrar una solución óptima es computacionalmente intratable para instancias grandes. Por ello, en la práctica, se recurre a heurísticas y metaheurísticas para obtener soluciones aproximadas con alta calidad en tiempos razonables [12].

Dado que las aplicaciones del VRP en el mundo real presentan restricciones y características específicas, han surgido múltiples variantes del problema. Entre las más estudiadas se encuentran: el Problema de Enrutamiento de Vehículos Capacitado (CVRP), el Problema de Enrutamiento de Vehículos con Ventanas de Tiempo (VRPTW), Problema de Enrutamiento de Vehículos con Múltiples Depósitos (MDVRP). A continuación se detallan los mencionados anteriormente.

1. **Problema de Enrutamiento de Vehículos Capacitado (CVRP):** En esta variante, cada vehículo tiene una capacidad máxima que no puede ser excedida por la suma de las demandas de los clientes asignados a su ruta. El CVRP es relevante en contextos logísticos donde el volumen o peso de los productos transportados juega un rol crucial, como en la distribución de alimentos o materiales de construcción. El principal desafío en el CVRP es balancear el uso eficiente de los vehículos mientras se minimizan los costos totales, respetando las restricciones de capacidad [5].

2. **Problema de Enrutamiento de Vehículos con Ventanas de Tiempo (VRPTW):**
   Aquí, además de optimizar las rutas, se debe cumplir un intervalo de tiempo específico durante el cual un cliente debe ser visitado para realizar la entrega. Esta variante es común en servicios de entrega, como el correo o las empresas de paquetería, donde los clientes especifican franjas horarias para recibir sus pedidos. El VRPTW introduce complejidad adicional al problema, ya que las soluciones deben sincronizar las rutas y los tiempos de servicio, minimizando penalizaciones por retrasos o tiempos de espera [16].

3. **Problema de Enrutamiento de Vehículos con Múltiples Depósitos (MDVRP):**
   En el MDVRP, los vehículos pueden partir de múltiples depósitos en lugar de un único punto central. Esto refleja escenarios reales donde empresas con centros de distribución regionales buscan optimizar sus operaciones de manera integrada. La asignación de clientes a depósitos y la planificación de rutas eficaces son desafíos clave en esta variante [6].

Debido a su clasificación como un problema NP-Duro, se han desarrollado múltiples enfoques para resolverlos. Los métodos exactos, como la programación lineal entera y la ramificación y acotación, son efectivos para instancias pequeñas, pero su escalabilidad es limitada [2]. En instancias más grandes, se emplean heurísticas y metaheurísticas como la búsqueda de vecindad variable (VNS), la búsqueda en vecindades amplias (LNS) [17], los algoritmos genéticos y la colonia de hormigas [12]. Estas técnicas buscan un balance entre la exploración del espacio de soluciones y la explotación de las áreas más prometedoras.

La creación de conjuntos de datos de prueba más complejos y diversos, como los benchmarks de [20], han permitido evaluar y comparar las soluciones de diferentes enfoques de manera más rigurosa.

En la siguiente sección se presenta la definición de solución del VRP y las operaciones que se pueden hacer sobre las mismas.

1.2. Criterios de vecindad

Para poder definir que es un criterio de vecindad, necesitamos definir primero que es una solución del VRP. De manera general una solución consiste en un conjunto de rutas r a las cuales, en la mayoría de variantes, pertenecen un grupo de clientes c como se muestra en la figura 1. Según las características del problema la estructura de la solución puede variar.

$$S = (r_1 : (c_1, c_2, c_3), r_2 : (c_4, c_5), r_2 : (c_6, c_7, c_8))$$

Figura 1: Ejemplo de solución de un VRP

Luego de haber definido que es una solucion del VRP, se puede definir que un criterio de vecindad es un conjunto de operaciones que modifican una solucion, permitiendo crear nuevas. Formalmente se puede definir en el espacio de soluciones X como una aplicacion $\mu : X \to 2^X$, tal que a cada solucion $x \in X$ le asigna un conjunto de soluciones $\mu(x) \subset X$, las cuales se llaman soluciones vecinas de x o vecindad de x

Cuando se usan algoritmos de búsqueda local para resolver el VRP, los criterios de vecindad juegan un papel fundamental. Estos criterios permiten mejorar la solución gradualmente o escapar de óptimos locales. Por lo tanto la elección de criterios adecuados es crucial para la eficacia de los métodos de optimización.

A continuación se presentan algunos ejemplos de Criterios de Vecindad:

- **Reubicación de un cliente:** Este criterio consiste en mover un cliente de una ruta a otra o a una posición diferente dentro de la misma ruta. Por ejemplo, un cliente puede ser reubicado para equilibrar las cargas de los vehículos o para reducir el costo de la ruta.
- **Intercambio de clientes:** En este caso, se intercambian dos clientes entre diferentes rutas o dentro de la misma ruta. Esto permite ajustar las rutas para optimizar la distancia total recorrida o cumplir restricciones específicas, como las de capacidad o tiempo.
- **Intercambio de subrutas:** Estos criterios implican modificar subrutas intercambiando dos o tres aristas, respectivamente. Por ejemplo, en un movimiento 2-opt, se eliminan dos aristas de la solución actual y se reconectan los nodos para formar una ruta diferente. Este tipo de criterio es común en problemas como el TSP y se adapta bien al VRP para minimizar la distancia total.
- **Inserción de clientes:** Este criterio selecciona un cliente que no está incluido en la solución actual o que ha sido eliminado temporalmente, y lo inserta en una posición que minimice el costo de la ruta. Este enfoque es particularmente útil en heurísticas constructivas y métodos como la Búsqueda en Vecindades Amplias (LNS).
- **Intercambio múltiple (Generalized Swap):** Extiende el concepto de intercambio a grupos de clientes. Por ejemplo, se pueden intercambiar subrutas completas de clientes entre rutas para explorar vecindarios más amplios.

Los criterios de vecindad mantienen el equilibrio entre la intensificación y la diversificación en la búsqueda. La intensificación se enfoca en explorar soluciones cercanas para mejorar la calidad local, mientras que la diversificación permite moverse a regiones distantes del espacio de soluciones para evitar quedarse atrapado en óptimos locales. Sin embargo en una vecindad existen regiones que poseen un mayor numero de soluciones que mejoran a la actual, por lo tanto, explorar dichas regiones puede conducirnos a la solución de manera mas rápida. En la siguiente sección se define que es una región.

1.3. Región de una vecindad

En el contexto de la exploración de vecindades, una región se define como un subconjunto del espacio de soluciones vecinas, agrupadas en función de características comunes determinadas por el criterio de vecindad aplicado. Estas regiones surgen como resultado de particionar el espacio de soluciones, permitiendo realizar búsquedas focalizadas e intensivas en áreas específicas que se consideran prometedoras en términos de encontrar soluciones óptimas o cercanas al óptimo.

La partición en regiones facilita la exploración eficiente al reducir la complejidad asociada con el análisis exhaustivo de vecindades grandes. Este enfoque divide el espacio de búsqueda en unidades manejables, lo que permite identificar y priorizar regiones con alto potencial, evitando la dispersión de los recursos computacionales en zonas menos relevantes.

El concepto de región no solo organiza la exploración del espacio de soluciones, sino que también se alinea con estrategias de optimización híbridas que separan fases de exploración y explotación. Durante la exploración, las regiones se analizan superficialmente para clasificar su calidad. Posteriormente, en la fase de explotación, se intensifica la búsqueda dentro de las regiones clasificadas como más prometedoras, maximizando la eficiencia del proceso de optimización.

Las regiones pueden ser combinadas con la metaheurística de Búsqueda en Vecindad Variable, permitiendo guiar la búsqueda a áreas especificas de las vecindades generadas. En la siguiente sección se profundiza en el trabajo de Búsqueda de Vecindad Variable.

1.4. Búsqueda en Vecindades Variables (VNS)

La Búsqueda en Vecindades Variables (VNS) es una metaheurística de optimización introducida por Mladenović y Hansen en 1997 [13], diseñada para resolver problemas de optimización combinatoria y global mediante la exploración sistemática de múltiples vecindades en el espacio de soluciones. A diferencia de otros métodos de búsqueda local, que suelen trabajar con una única vecindad, VNS se basa en la idea de que cambiar dinámicamente entre diferentes estructuras de vecindad.

Esta metaheurística se basa en dos principios fundamentales:

1. Un mínimo local para una estructura de vecindad no lo es necesariamente para otra.

2. Un mínimo se considera global si es un mínimo local para cualquiera sea el criterio de vecindad.

Estos principios indican el uso de varios criterios de vecindad para resolver el problema, debido a que si se encuentra un optimo local en uno de ellos, sea posible cambiar de criterio y continuar la exploración, en cambio si hay un optimo local para todos los criterios entonces se puede afirmar que es el optimo global del problema. Esta es la idea de la VNS.

El algoritmo parte de una solución inicial y utiliza un conjunto finito de criterios de vecindad predefinidos. Durante cada iteración, se explora el entorno de la solución actual con una estructura específica; si se encuentra una mejora, la solución se actualiza y el proceso reinicia con la primera estructura, pero si no hay mejoras, se cambia a la siguiente estructura hasta cumplir con las condiciones de parada, es decir hasta que se utilicen todos los criterios.

En el contexto del VRP y sus variantes, VNS ha sido utilizado para encontrar soluciones cercanas al óptimo [4, 18]. Por ejemplo, en el Capacitated Vehicle Routing Problem (CVRP), las modificaciones suelen implicar el intercambio de clientes entre rutas o la reasignación de clientes dentro de una misma ruta [1]. En el VRP con ventanas de tiempo (VRPTW), VNS se adapta para cumplir con restricciones temporales, incorporando operadores que ajusten las secuencias de visitas según las ventanas de tiempo permitidas [4]. Adicionalmente, en variantes como el *Multi-Depot VRP* (MDVRP), los vecindarios incluyen movimientos que redistribuyen clientes entre depósitos, optimizando las rutas de manera global [18].

Una de las principales ventajas de VNS es su simplicidad y flexibilidad, ya que puede integrarse fácilmente con otros enfoques de optimización [13]. Sin embargo, su efectividad depende de un diseño adecuado de los vecindarios y de los operadores utilizados, ya que estos determinan la calidad de las soluciones exploradas y la capacidad del algoritmo para escapar de óptimos locales [14]. Se han introducido variantes avanzadas a la metaheurística como la Búsqueda de Vecindad Infinitamente Variable (IVNS) [14], que permite explorar vecindarios dinámicamente generados mediante un lenguaje formal, ampliando aún más el alcance de esta técnica.

VNS sigue siendo una herramienta en la optimización combinatoria, particularmente en problemas donde la calidad de la solución es un aspecto crítico debido a su capacidad para encontrar un optimo global respecto a los criterios de los cuales se disponga. Su enfoque sistemático para explorar el espacio de soluciones lo convierte en un método robusto y eficiente, adecuado para una amplia gama de aplicaciones prácticas e industriales [13, 8, 4]. Otra herramienta para la solución de problemas de optimización es una biblioteca de *software* de la cual se trata en la sección siguiente.

1.5. Búsqueda en Vecindades Grandes (LNS)

La Búsqueda en Vecindarios Amplios (Large Neighborhood Search, LNS) es una metaheurística diseñada para resolver problemas de optimización combinatoria, como el Problema de Enrutamiento de Vehículos (VRP), permitiendo la exploración de grandes porciones del espacio de soluciones. Introducida por Shaw en 1998, LNS se basa en un enfoque innovador que combina la destrucción y reconstrucción de soluciones para escapar de óptimos locales, un desafío recurrente en problemas complejos de optimización.

El principio central de LNS es dividir el proceso de búsqueda en dos etapas complementarias. En la primera etapa, denominada destrucción, se elimina una parte sustancial de la solución actual, generando una solución parcial que abre

la posibilidad de explorar nuevas configuraciones. Posteriormente, en la etapa de reconstrucción, la solución incompleta se completa nuevamente mediante el uso de heurísticas específicas o técnicas optimizadoras que intentan mejorar el resultado inicial. Este ciclo de destrucción y reconstrucción es clave para explorar un vecindario amplio del espacio de búsqueda, lo que diferencia a LNS de métodos tradicionales que solo realizan cambios pequeños y locales.

En el contexto del VRP, LNS ha sido ampliamente aplicado debido a su capacidad para manejar instancias grandes y problemas con múltiples restricciones. Por ejemplo, en el CVRP, el método se utiliza para ajustar las rutas de los vehículos mientras se respeta la capacidad máxima de carga. De manera similar, para variantes como el VRP con ventanas de tiempo (VRPTW), LNS reconfigura los itinerarios para garantizar que las entregas cumplan con los intervalos de tiempo especificados por los clientes. Además, en situaciones dinámicas donde los pedidos pueden surgir en tiempo real, LNS permite reorganizar las rutas adaptándose rápidamente a los cambios.

Entre las principales ventajas de LNS destaca su flexibilidad, ya que se puede personalizar fácilmente para abordar variantes específicas del problema, adaptando los mecanismos de destrucción y reconstrucción según las características del escenario. Además, su capacidad para diversificar el espacio de búsqueda lo hace especialmente útil para evitar el estancamiento en óptimos locales, un problema común en métodos de búsqueda más restrictivos. Sin embargo, también presenta limitaciones. La exploración de grandes vecindarios puede ser computacionalmente intensiva, especialmente en problemas de gran escala, lo que puede requerir una cantidad significativa de recursos para alcanzar soluciones de calidad. Asimismo, la eficacia de LNS depende en gran medida de las heurísticas utilizadas durante la etapa de reconstrucción, ya que estas determinan la calidad de las nuevas soluciones generadas.[17]

La implementación de LNS sigue un flujo bien definido. El proceso inicia con la creación de una solución inicial factible que sirve como punto de partida. A continuación, se alterna entre la eliminación de una parte de la solución y la reconstrucción de los elementos eliminados. En cada iteración, se analiza si la nueva solución debe sustituir a la actual, basándose en criterios como el costo total de las rutas o la viabilidad del plan. Este ciclo continúa hasta que se cumple un límite previamente establecido, ya sea en tiempo o en número de iteraciones. Este enfoque equilibra la exploración de nuevas áreas del espacio de soluciones con la explotación de las soluciones prometedoras ya descubiertas.

En términos de impacto, LNS ha demostrado ser una de las herramientas más eficaces para abordar problemas complejos de ruteo, especialmente aquellos donde las restricciones operativas y las necesidades de optimización dificultan la aplicación de métodos exactos. Su capacidad para combinar flexibilidad, escalabilidad y adaptabilidad lo ha convertido en una técnica preferida tanto en la investigación académica como en aplicaciones prácticas, como la logística, la distribución y la gestión del transporte.

1.6. GLPK

El GNU Linear Programming Kit (GLPK) es una biblioteca de *software* diseñada para resolver problemas de programación lineal (PL) y programación lineal entera mixta (MIP)[11]. GLPK es parte del proyecto GNU y se distribuye bajo la Licencia Pública General de GNU [10], lo que lo convierte en una opción de código abierto ampliamente adoptada tanto en entornos académicos como industriales. Ofrece herramientas robustas y flexibles para modelar y resolver problemas de optimización combinatoria, siendo particularmente útil en aplicaciones como logística, transporte y gestión de recursos.

GLPK se centra en la resolución de problemas formulados en términos de modelos matemáticos, donde se maximiza o minimiza una función objetivo bajo un conjunto de restricciones lineales. Estos modelos se especifican típicamente utilizando el lenguaje de modelado GNU Math Programming Language (GMPL) [15]. GMPL permite describir problemas de manera declarativa y concisa, separando la definición del modelo de los datos específicos, lo que facilita la reutilización y la adaptación del modelo a diferentes escenarios.

Una de las características destacadas de GLPK es su implementación del método simplex revisado y del método primal-dual para resolver problemas de programación lineal. Para problemas más complejos, como los de programación lineal entera mixta, GLPK utiliza un enfoque de ramificación y acotación (branch-and-cut). Este enfoque combina técnicas de ramificación para dividir el problema en subproblemas más pequeños y métodos de corte para eliminar regiones inviables del espacio de búsqueda, mejorando la eficiencia del proceso.

La siguiente sección esta dedicada al lenguaje de programación LISP y el porqué es eficiente para la solución del VRP.

1.7. Lisp

LISP (LISt Processing) es un lenguaje de programación utilizado en el ámbito de la optimización combinatoria, especialmente en la resolución de problemas como el Vehicle Routing Problem (VRP) y sus variantes debido a su eficiencia en el manejo de listas [6, 19, 3, 17, 14]. Una de las aplicaciones de LISP en este contexto es su uso para implementar estrategias de exploración en vecindades, crucial en métodos como la Búsqueda de Vecindad Variable (VNS) y sus extensiones.

La implementación en LISP permite realizar operaciones de manera eficiente gracias a su sistema de listas, macros y funciones genéricas. Debido a esto, LISP se utiliza para estructurar y realizar operaciones complejas dentro de los vecindarios de soluciones del VRP. Esto incluye la selección y manipulación de rutas y subrutas, el cálculo iterativo de costos para evaluar cambios en la solución, y la identificación de regiones óptimas en base a criterios definidos.

LISP ha sido utilizado para generar automáticamente criterios de vecindad mediante lenguajes formales [14]. Estos lenguajes, definidos con gramáticas libres del contexto, permiten crear cadenas que representan infinitos criterios de vecindad adaptables a las características específicas de cada variante del VRP.

Esto no solo agiliza la exploración de vecindades, sino que también garantiza la consistencia en las operaciones realizadas.

Debido a su capacidad para crear criterios de vecindad y manejar macros, lo convierten en un lenguaje capaz de buscar óptimos de manera exhaustiva rápida y eficiente.

Referencias

[1] Marwa Amous. A variable neighborhood search algorithm for the capacitated vehicle routing problem. 2017.

[2] Roberto baldacci. Recent exact algorithms for solving the vehicle routing problem under capacity and time window constraints. 2012.

[3] Daniela González Beltrán. Generación automática de gramáticas para la obtención de infinitos criterios de vecindad en el problema de enrutamiento de vehículos. mayo 2019.

[4] Olli Bräysy. A reactive variable neighborhood search for the vehicle-routing problem with time windows. 2003.

[5] Juan Pablo Orrego Cardozo. Solución al problema de ruteo de vehículos con capacidad limitada (cvrp) usando una técnica metaheurística. 2016.

[6] Gabriela Mijenes Carrera. Análisis de vecindades en problemas de enrutamiento de vehículos con múltiples depósitos y flotas heterogéneas. noviembre 2021.

[7] G. B. Dantzig and J. H. Ramser. The truck dispatching problem. 1959.

[8] João Paulo Queiroz dos Santos. Reactive search strategies using reinforcement learning, local search algorithms and variable neighborhood search. 2014.

[9] Burak Eksioglu. The vehicle routing problem: A taxonomic review. 2009.

[10] Free Software Foundation. Gnu general public license, version 3, 2007. Last accessed: 2024-12-14.

[11] Andrew Makhorin. *GLPK (GNU Linear Programming Kit) Reference Manual*. Free Software Foundation, 2001. Last accessed: 2024-12-14.

[12] Abdul Kadar Muhammad Masum. Solving the vehicle routing problem using genetic algorithm. 2011.

[13] Nenad Mladenovic and Pierre Hansen. Variable neighborhood search. 1997.

[14] Camila Pérez Mosquera. Primeras aproximaciones a la búsqueda de vecindad infinitamente variable. 2017.

[15] GNU Project. Glpk modeling language (gmpl), 2024. Last accessed: 2024-12-14.

[16] Yutao Qi. A decomposition based memetic algorithm for multi-objective vehicle routing problem with time windows. 2015.

[17] Héctor Felipe Massón Rosquete. Exploración de vecindades grandes en el problema de enrutamiento de vehículos usando técnicas estadísticas. mayo 2020.

[18] Mir Ehsan Hesam Sadati. A hybrid variable neighborhood search approach for the multi-depot green vehicle routing problem. 2021.

[19] José Jorge Rodríguez Salgado. Una propuesta para la evaluación automática de soluciones vecinas en un problema de enrutamiento de vehículos a partir del grafo de evaluación de una solución. junio 2020.

[20] Eduardo Uchoa. New benchmark instances for the capacitated vehicle routing problem. 2016.

[21] Vidal. Hybrid genetic search for the cvrp: Open-source implementation and swap* neighborhood. 2021.