UNIVERSIDAD DE CONCEPCIÓN

FACULTAD DE INGENIERÍA

DEPARTAMENTO DE INGENIERÍA INDUSTRIAL

RUTEO DE VEHÍCULOS PARA LA RECOLECCIÓN DE RESIDUOS ELECTRÓNICOS. UN CASO DE ESTUDIO UTILIZANDO DATOS GEOGRÁFICOS

Nicolás Eduardo Netz Carrasco

Memoria de Título presentada a la Facultad de Ingeniería de la Universidad de Concepción para optar al título profesional de Ingeniera Civil Industrial

Profesora Guía

Lorena Pradenas Rojas

Profesora Comisión

Rosa Medina Durán

Diciembre 2023

Concepción, Chile

Dedicatoria

A mi madre y mi padre, que siempre me han apoyado.

Agradecimientos

A mis padres, por su amor, apoyo y sacrificios.

A todos mis profesores, por sus enseñanzas y dedicación.

A mis seres queridos y amigos cercanos, por acompañarme en este camino de aprendizaje y formación, y por su constante apoyo que ha sido fundamental en mi vida personal.

A Andrew Hurst, Supervising Senior Environmental Scientist del Covered Electronic Waste Recycling Program del "Department of Resources Recycling and Recovery (CalRecycle)" del estado de California, Estados Unidos, por facilitarme el acceso a los datos geográficos de sus bases de datos, que permitieron desarrollar este trabajo.

Especialmente quiero agradecer a la profesora Lorena, cuyos conocimientos, orientación y apoyo han sido de gran importancia durante todo el proceso de investigación y redacción de esta memoria de título. Su guía ha sido fundamental para alcanzar los objetivos propuestos.

A todos ellos, mi más profundo agradecimiento por su contribución y apoyo en este importante logro académico.

Resumen

Este estudio aborda la creciente problemática de los residuos eléctricos y electrónicos (RAEE o *e-waste*, en inglés), derivada de la expansión del uso de dispositivos tecnológicos a nivel global. En respuesta a los desafíos ambientales y de salud pública asociados, se propone un enfoque basado en herramientas de Ingeniería. Se usa un modelo de programación matemática, específicamente el *Capacitated Vehicle Routing Problem with Time Windows* (*CVRPTW*), para la recolección eficiente de desechos. Dada la complejidad computacional, se desarrolla una metaheurística basada en *Iterated Local Search.* Se identifican y caracterizan los elementos relevantes del sistema logístico, destacando la cadena de suministro inversa. Los experimentos computacionales respaldan la efectividad de la metaheurística propuesta, aunque la complejidad computacional del modelo de programación matemática restringe su aplicabilidad en ciertos casos. La aplicación práctica con datos geográficos reales presentados en este estudio, muestra resultados exitosos con la metaheurística usada, señalando la necesidad de considerar múltiples depósitos en futuras investigaciones.

Palabras clave: Residuos electrónicos, Programación Matemática, *Capacitated Vehicle Routing Problem with Time Windows*, *Iterated Local Search*.

Abstract

This study addresses the growing issue of electronic and electrical waste (e-waste), stemming from the global proliferation of technological devices. In response to associated environmental and public health challenges, an engineering-based approach is proposed. A mathematical programming model, specifically the Capacitated Vehicle Routing Problem with Time Windows (CVRPTW), is employed for the efficient collection of waste. Given the computational complexity, a metaheuristic based on Iterated Local Search is developed. Key elements of the logistics system are identified and characterized, with a focus on the reverse supply chain. On one hand, computational experiments validate the effectiveness of the proposed metaheuristic, while the computational complexity of the mathematical model limits its applicability in some cases. Practical application with real geographical data demonstrates successful outcomes with the metaheuristic, emphasizing the need to consider multiple depots in future research.

Keywords: Electronic waste, Mathematical Programming, Metaheuristics, Capacitated Vehicle Routing Problem with Time Windows, Iterated Local Search

Índice de contenido

Índice de tablas

Índice de figuras

Índice de Algoritmos

1. Introducción

1.1. Antecedentes generales

En las últimas décadas, y desde la revolución industrial, el estilo de vida de las personas ha cambiado significativamente, usando una gran cantidad de objetos tecnológicos que facilitan las tareas del *día a día*. Por otro lado, gracias a los avances en los sistemas de producción, los aparatos tecnológicos son cada vez más baratos, siendo estos accedidos por una mayor fracción de la población mundial. Esto tiene implicancias positivas, permiten realizar tareas que antes eran imposibles (Raggio, 2021), pero al mismo tiempo implica que la cantidad de residuos producidos por año sea cada día mayor (Baldé, 2017).

El problema no es solo que existan más residuos, sino que los residuos generados por aparatos eléctricos y electrónicos descartados por sus dueños, contienen químicos tóxicos que provienen de los metales que lo componen, generando un problema mayor de contaminación en los vertederos y otros lugares no deseados. Algunos estudios, como el presentado por Forti et al. (2020) indica que cada año se generan 7.3 kg por habitante por año en 2019, y que aumente a 9 kg por habitante por año, lo que es una cantidad significativa, y un mal manejo de estos, tiene efectos negativos en: el suelo, el océano, el turismo y la salud pública (Kaza, 2018).

El manejo de residuos es un problema de logística ampliamente estudiado, se han aplicado diversos métodos de optimización para gestionar el uso. Diversos autores también han propuesto métodos de solución al ruteo de vehículos en la recolección de residuos y minimizando distintas funciones objetivos entre otros, se pueden mencionar: minimizar el costo total de transporte (Wan, et al., 2023), minimizar el costo total en emisiones e impacto social (Mohammadi, et al., 2023), minimización de costos totales de transporte y costos de contaminación atmosférica (Rahmanifar, et al., 2023), maximizar la probabilidad de utilidades con el desecho recolectado (Hashemi-Amiri, et al., 2023), minimizar las emisiones generadas (Zhang, 2020), minimizar el riesgo de exposición a desechos peligrosos (Yu, 2020), minimizar el retraso en la recolección (Tirkolaee, 2019) y maximizar el desecho total recolectado en un horizonte de tiempo (Expósito-Márquez, 2019).

Algunos autores han detallado que el proceso de recolección de este tipo de desecho puede representar hasta el 70% del presupuesto necesario para el proceso completo (Kaza, 2018), superando otras acciones operativas como almacenamiento o tratamiento final.

1.2. Objetivos

1.2.1. Objetivo general

Estudiar el problema logístico de recolección de desecho electrónico y proponer un sistema que utilice herramientas de optimización y computación apropiado para recolectar residuos electrónicos de manera eficiente en una ciudad, minimizando los costos de recolección en puntos limpios.

1.2.2. Objetivos específicos

- (OE1) Representar e implementar computacionalmente una solución al problema a través de un modelo de programación matemática.
- (OE2) Plantear e implementar computacionalmente una solución metaheurística al problema.
- (OE3) Aplicar la programación matemática y la metaheurística a instancias de prueba, para validar los resultados
- (OE4) Identificar y caracterizar el sistema logístico que es requerido para implementar la solución.
- (OE5) Analizar resultados económicos de una implementación bajo supuestos en una ciudad.

1.3. Justificación

Dado que la problemática de manejo de residuos, es relevante para un futuro sostenible, la recolección, recuperación y manejo de éste tipo de desecho debe ser economicamente atractiva. Por lo tanto, la formulación de un sistema logístico eficiente que utilice herramientas de optimización y computación para gestionar los recursos es atractiva de ser estudiada y eventualmente implementada, con el objetivo de generar un impacto positivo en el medio ambiente desde la utilización de herramientas de la ingeniería.

1.4. Estructura de la memoria

El presente documento considera la siguiente estructura: En la sección 1. se introduce el tema y los objetivos. En la sección 2 se describe el problema y la situación de los residuos electrónicos a nivel nacional y mundial. En la sección 3, se presenta la revisión bibliográfica

atingente al tema, los métodos de optimización y antecedentes de sistemas de logística propuestos en la literatura para la recolección de residuos. En la sección 4, se describe la metodología aplicada para resolver la problemática. En la sección 5, se analizan los resultados de los experimentos computacionales, el caso de estudio y el sistema logístico. Luego, en el capítulo 6, se presentan las conclusiones y finalmente el glosario, las referencias bibliográficas y el anexo.

2. Descripción del problema

2.1. Aparatos eléctricos y electrónicos

En distintas partes del mundo se tienen diferentes definiciones para enmarcar, lo que es un aparato eléctrico y electrónico. Por ejemplo, en la Unión Europea se definen, bajo la Directiva 2012/19/EU en más de una categoría como: "aparatos que para funcionar requieren de corriente eléctrica o campos electromagnéticos y equipamiento, para la generación, transmisión y medición de tales corrientes y campos y diseñados para ser utilizados con un voltaje menor a 1000 volts, para corriente alterna y 1500 volts para corriente continua" (European Union, 2012).

Por otro lado, organizaciones como la *StEP Initiative* (*Solution to the Electronic Waste Problem Initiative*) han acuñado sus propias definiciones, independientes de cualquier legislación sobre desechos electrónicos o responsabilidad del productor. Así, *StEP* define los aparatos eléctricos y electrónicos como "Cualquier artículo doméstico o comercial con circuitos o componentes eléctricos con suministro de energía o batería." (Iniciativa StEP, 2014)

Figura 2.1- Una calculadora, ejemplo de aparato electrónico

Fuente: Citizen Systems (2023). Business Line. Calculadoras de Sobremesa. https://www.citizen-systems.com/es/products/calculator/desktop

Entonces, según lo identificado tanto por la Unión Europea y la iniciativa *StEP*, se identifican las abreviaciones AEE para los aparatos eléctricos y electrónicos en español y EEE en inglés (*Electric and Electronic Equipment*).

2.2.Residuos de aparatos eléctricos y electrónicos

La definición de cuando un AEE pasa a ser residuo es inconsistente, ya que depende de cuando el dueño del aparato descarta su funcionamiento (Iniciativa StEP, 2014). Este instante es conocido como el fin de la vida o *End of Life* (Oguchi, 2011). Así, la Iniciativa *StEP* define los RAEE como:

> *"RAEE (o E-Waste) es un término utilizado para cubrir todo tipo de equipos eléctricos y electrónicos (AEE) y sus piezas que han sido desechados por el propietario como residuos sin intención de reutilizarlos. (Iniciativa StEP, 2014)*

Por otro lado, una definición más reciente propuesta por M. Wagner, et al. (2022) describe los "Residuos eléctricos y electrónicos o RAEE" de la siguiente manera:

> *"Son equipos eléctricos o electrónicos, incluidos todos los componentes, subconjuntos y artículos consumibles que forman parte del equipo y que han sido desechados por el propietario como residuos sin intención de reutilización."*

Figura 2.2 - Centro de acopio de e-waste

Fuente: Adaptado de Homepage [Fotografía], (CCL North Ltd., 2023), https://www.cclnorth.com

Estos aparatos contienen diferentes compuestos. Algunos son valorizables y en algunos casos también, pueden ser peligrosos, por lo que su gestión adecuada no solo permite reducir el impacto ambiental de los mismos, sino que también permite el aprovechamiento de los recursos materiales. (Amphos 21 Consulting Chile Ltda, 2015) .

Oguchi, et al. (2011), determina que los residuos de aparatos eléctricos y electrónicos contienen metales comunes, metales preciosos y metales poco comunes e identifica aquellos que son de interés según 21 tipos de AEE, incluyendo Aluminio, Cobre, Fierro, Plomo, Estaño, Zinc, Oro, Plata, Paladio, Bario, Bismuto, Cobalto, Galio, Estroncio y Tántalo.

La Directiva RAEE de la Unión Europea y las "Directrices sobre estadísticas de residuos electrónicos" (European Union, 2012) utilizan, una categorización orientada al tratamiento, con seis categorías principales:

- Aparatos de intercambio de temperatura
- Pantallas y monitores
- Lámparas
- Grandes aparatos
- Pequeños aparatos
- Pequeños aparatos informáticos y de telecomunicaciones

2.3.Situación mundial

El *e-waste* es una de las corrientes de residuo de más rápido crecimiento en el mundo. Se estima que 53.6Mt de RAEE fueron generados a nivel mundial en el 2019, de los cuales sólo un 17% se recoge y se recicla oficialmente. (M. Wagner, 2022).

La cantidad de *e-waste* en el mundo ha crecido velozmente. Como se observa en la Figura 2.3, los países de la OECD mantienen elevados valores de *e-waste* generado a nivel municipal, se destaca la posición de Estados Unidos, en comparación del resto de los países.

El crecimiento del *e-waste* varía, si el país es desarrollado o si está en vías de desarrollo (Baldé et. Al 2017) y eso también se puede observar en la Figura 2.3. Países en vías del desarrollo mantienen crecimientos de 10% al 25% en peso, mientras que países desarrollados crecen a un 1% a 5%.

El consumo global de nuevos aparatos eléctricos y electrónicos fue alrededor de 60Mt en el año 2016.

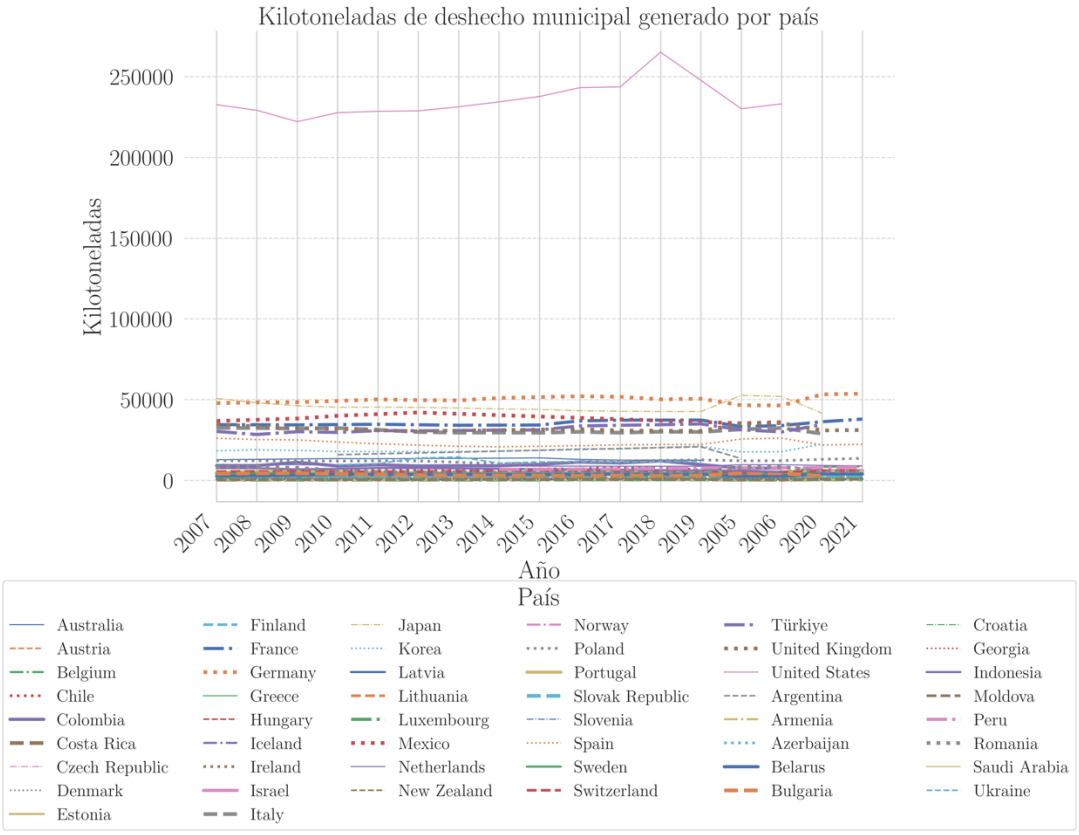

Figura 2.3 – Desecho municipal generado por país por año (miles de toneladas)

Fuente: Adaptado de "Waste from electrical and electronic equipment (WEEE – e-waste)" (OECD, 2023) https://stats.oecd.org/Index.aspx?DataSetCode=EWASTE

Algunas veces el *e-waste* es simplemente registrado como desecho de metales entonces, los valores pueden estar subestimados. (Iniciativa StEP, 2014)

Otra problemática a nivel mundial es sobre la exportación e importación de este tipo de desechos. Los desechos electrónicos cruzan fronteras, en general hacia países subdesarrollados. El *E-Waste Monitor* cuantificó que cerca del 10% del total del *e-waste* global cruzó fronteras en el año 2019. Las implicancias de este movimiento se relacionan a la contaminación y la pérdida de éstos, en una posible economía circular.

Diversas legislaciones se han impulsado para que los productores e importadores de AEE se encarguen de los desechos (RAEE). Un ejemplo serían las leyes de Responsabilidad Extendida del Productor.

2.4. Situación regional

En Latinoamérica existen escasos datos que permitan seguir el estado de este tipo de desecho, además las legislaciones son relativamente nuevas comparadas con Europa. Sólo algunos países, que pertenecen a la OECD, tienen en proceso leyes que impulsarían un cambio hacia una economía circular más fuerte y con procedimientos para minimizar la generación de este tipo de residuo (M. Wagner, 2022).

En la Figura 2.4, se observa que los países con datos de toneladas de *e-waste* generados por año, son, Argentina, seguido por Colombia, luego Perú y finalmente, Chile.

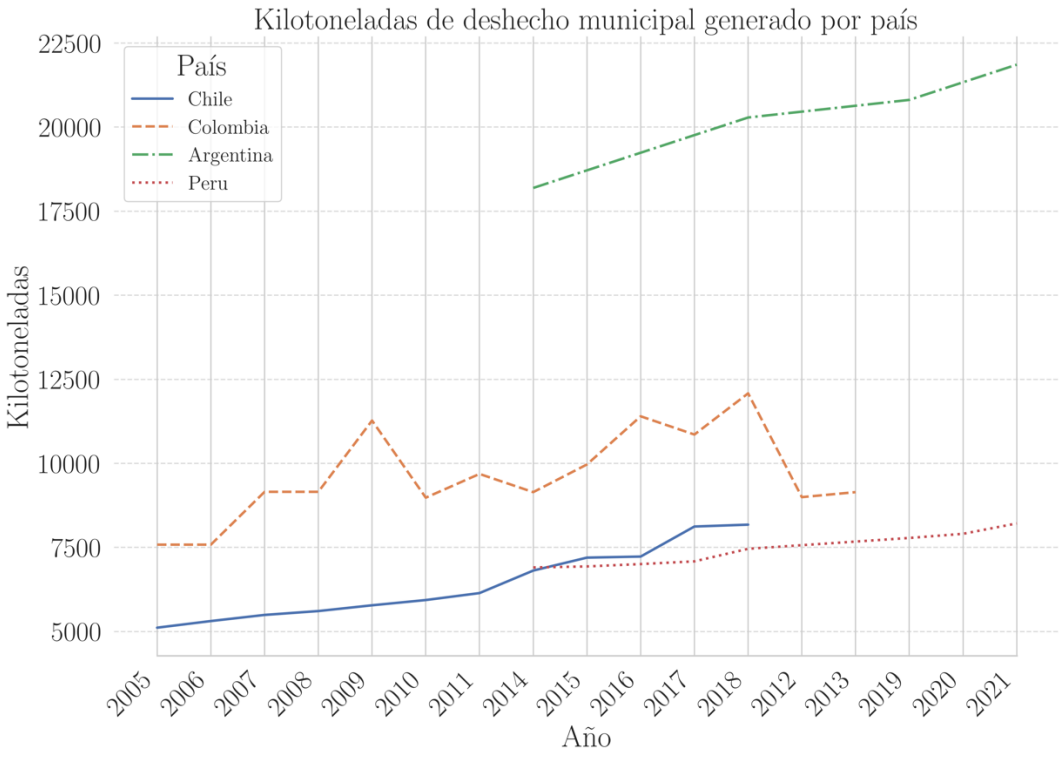

Figura 2.4 - *E-waste* generado a nivel municipal en Latinoamérica

Fuente: Adaptado de "Waste from electrical and electronic equipment (WEEE – e-waste)" (OECD, 2023) https://stats.oecd.org/Index.aspx?DataSetCode=EWASTE

2.5. Situación nacional

En Chile, las tendencias no son diferente de la mundial, ni menos de la regional. Chile es uno de los países con más rápido crecimiento de toneladas de *e-waste* en Latinoamérica, y relacionado, principalmente con su PIB per cápita, la apertura de sus mercados y los cortos ciclos de vida de los productos.

En la Figura 2.5, se observa el crecimiento *e-waste* per cápita en Chile, en Kg, en los últimos años. La tendencia muestra, que se ha pasado de 7kg/per cápita en el 2010, a cerca de 10kg/per cápita en el 2021.

Las legislaciones que han impulsado en los últimos años, como la Ley REP, buscan:

> *"Obligar a fabricantes e importadores de seis productos prioritarios a recuperar un porcentaje de ellos una vez que terminan su vida útil"*

De esta manera, las políticas estatales proyectan para el 2050, que la economía nacional sea circular.

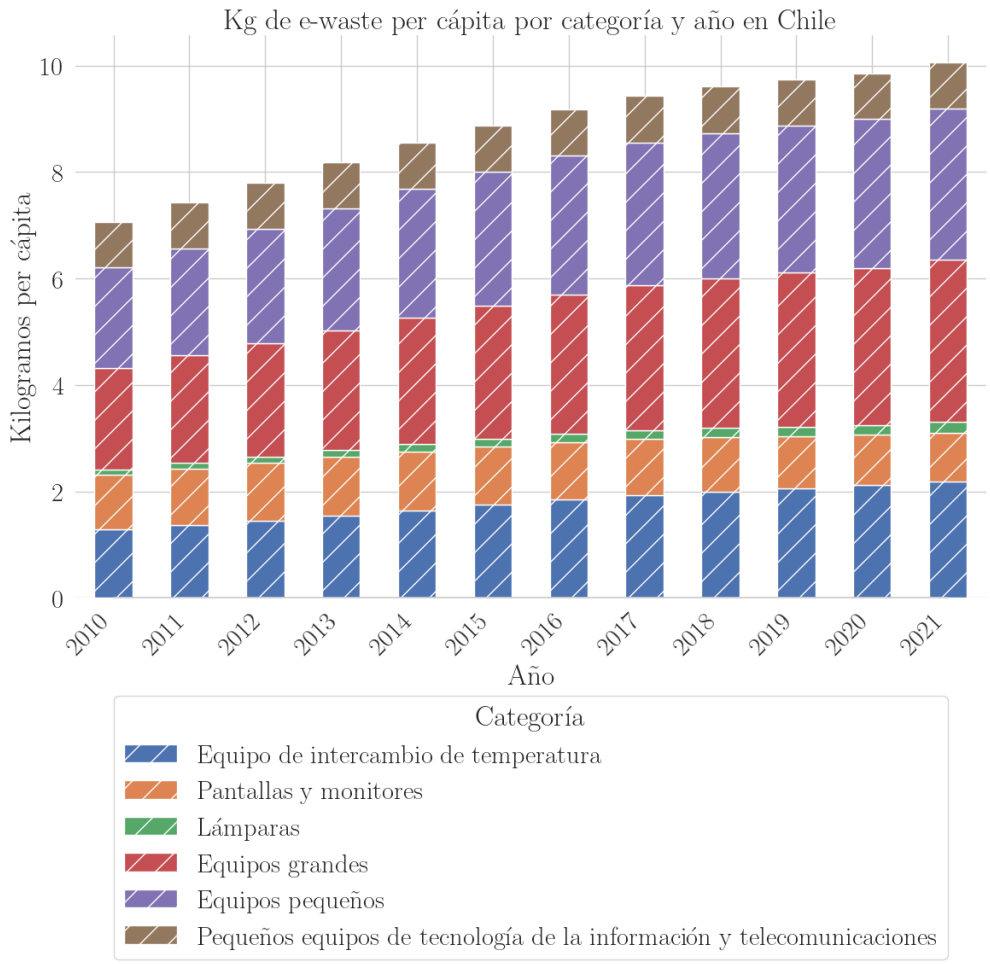

Figura 2.5 - *E-waste* por grupos generado en Chile por años

Fuente: Adaptado de "Waste from electrical and electronic equipment (WEEE – e-waste)" (OECD, 2023) https://stats.oecd.org/Index.aspx?DataSetCode=EWASTE

En Chile, no existe un medio u organismo gubernamental que recolecte o procese este tipo de desecho. Principalmente, hay localizaciones (privadas, municipales y de organizaciones no gubernamentales) donde reciben este tipo de desecho. Estos lugares son conocidos como "Puntos Limpios".

Ahora que rige la ley REP, han surgido empresas que compiten ofreciendo servicios de recolección o procesamiento de *e-waste*. Dentro de ellas se identifica: Recycle, Kyklos, Electrónica Circular, Texnico, Evernex.

Se espera, que en el futuro aún más empresas y servicios, se ofrezcan en el mercado, pues productores e importadores deberán ofrecer servicios que cumplan la nueva normativa.

3. Revisión bibliográfica

En la presente sección se revisan documentos relevantes para el estudio, con diferentes orígenes y objetivos.

3.1.Ruteo de vehículos

La creciente velocidad en el consumen bienes ha llevado a un aumento rápido de residuos en todo el mundo, representando una amenaza ambiental, pero estos residuos pueden tener un valor económico. Por lo tanto, los gobiernos impulsan medidas para recopilarlos e incluirlos en procesos de reciclaje. Estas medidas son generalmente sistemas de ruteo de vehículos con operaciones de cadena de suministro inversa (Sar & Ghadimi, 2023).

El problema de ruteo de vehículos aparece por primera vez en 1959, propuesto por Dantzig & Ramser, (1959), con el propósito de distribuir productos a clientes y crear rutas óptimas. (Han & Ponce-Cueto, 2015). Este modelo se ha adaptado a diversas aplicaciones reales, como la recolección de residuos, la industria alimentaria, y otros. (Golden, Assad, & Wasil, 2014). Existen en la literatura distintos modelos aplicados a la recolección de residuos, con distintas funciones objetivo.

Algunas funciones objetivo consideradas en la literatura incluyen: minimizar el costo total de transporte (Wan, et al., 2023), minimizar el costo total en emisiones e impacto social (Mohammadi, et al., 2023), minimización de costos totales de transporte y costos de contaminación atmosférica (Rahmanifar, et al., 2023), maximizar la probabilidad de utilidades con el desecho recolectado (Hashemi-Amiri, et al., 2023), minimizar el costo total de recolección (Buhrkal, 2012), minimizar las emisiones generadas (Zhang, 2020), minimizar el riesgo de exposición a desechos peligrosos (Yu, 2020), minimizar el retraso en la recolección (Tirkolaee, 2019) y maximizar el desecho total recolectado en un horizonte de tiempo (Expósito-Márquez, 2019).

Diversos modelos presentados en la literatura incluyen restricciones de capacidad (Sar & Ghadimi, 2023), ventanas de tiempo (Kassem & Chen, 2013), multiples depósitos y multiples viajes de cada vehículo (Kim, Kim, & Sahoo, 2006) entre otros.

Nuevos paradigmas de tecnología también son considerados, como el Internet Of Things. Kapadia & Metha (2023) usan contenedores de desecho y vehículos inteligentes, que usan sensores para enviar información en tiempo real usando internet.

Las metaheurísticas que han sido estudiadas en el contexto de la recolección de residuos incluyen heurísticas y meta-heurísticas como *iterated local search*, *tabu search*, *genetic algorithms, variable neighbourhood search, large neighbourhood search*, *simulated annealing* y *ant colony optimization*. (Han & Ponce-Cueto, 2015).

La meta-heurística de Iterated Local Search consiste en construir iterativamente una secuencia de soluciones generadas por una heurística, que lleva a mejores soluciones que si se usaran ensayos aleatorios de la heurística. (Lourenço, Martin, & Stützle, 2010)

En la literatura del problema de ruteo de vehículos se usan ampliamente las instancias de Solomon (1987) para realizar *benchmarks* de las metaheurísticas. Otras instancias de *benchmark* presentes en la literatura son las Gehring & Homberger (1999), que se diferencian de las de Solomon en que tienen un mayor tamaño. (Sar & Ghadimi, 2023)

3.2.Sistema logístico

Un sistema logístico de una cadena de suministro es un conjunto de actividades funcionales que se repiten muchas veces, que convierten la materia prima en productos terminados y añade valor para el consumidor (Ballou, 2004). Las materias primas, fábricas, almacenes y puntos de venta no están ubicados en los mismos lugares. Luego, las actividades de logística y transporte permiten el funcionamiento de estos sistemas (Bowersox, Closs, & Cooper, 2007).

Las cadenas de suministro inversas permiten manejar el *e-waste* y en las últimas dos décadas han sido muy relevantes para gobiernos, expertos e investigadores. Los estudios relacionados a la cadena de suministro inversa para el *e-waste* se dividen en 4 grandes grupos: Factores de implementación; Evaluación de rendimiento; Mejorar el rendimiento de la recolección y diseño de redes (Linh, Amer, Lee, Phuc, & Dat, 2019).

En la Figura 3.1, Figura 3.2, Figura 3.3 se presentan distintos sistemas de logística para cadenas de suministro de *e-waste* de la literatura. Si bien, todos identifican los elementos

importantes de la cadena de suministro, no existe, en la literatura un sistema que destaque el proceso de recolección.

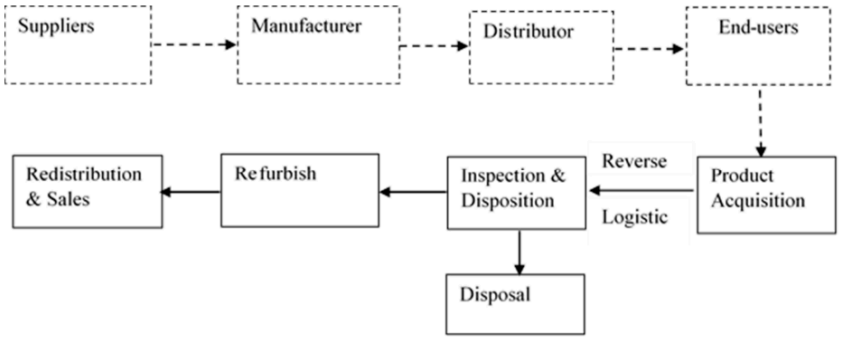

Figura 3.1 - Una cadena de suministro de ciclo cerrado para *e-waste*

Fuente: Tomado de Linh, Amer, Lee, Phuc, & Dat (2019), E-Waste Reverse Supply Chain: A Review and Future Perspectives https://doi.org/10.3390/app9235195

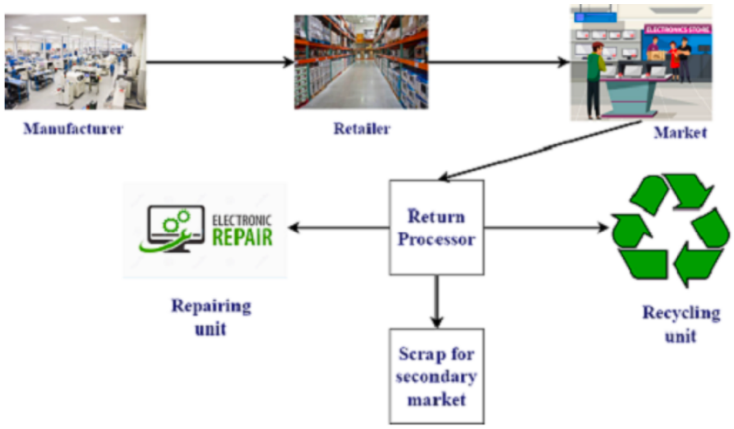

Figura 3.2 - Framework propuesto para la cadena de suministro forward e inversa de productos electrónicos

Fuente: Tomado de Kumar et al., 2019, Supply Chain Management of E-Waste for End-of-Life Electronic Products with Reverse Logistics

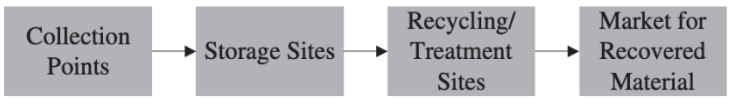

Figura 3.3 - Flujo de materiales de punto a punto en el sector formal de desecho de *e-waste*

Fuente: Tomado de Li et al., 2021, A Reverse Logistics Model For Recovery Options Of Ewaste Considering the Integration of the Formal and Informal Waste Sectors

4. Metodología

En la presente sección, se describe la metodología utilizada para resolver el problema de ruteo de vehículos para la recolección de residuos electrónicos y la aplicación al caso de estudio. Se presenta formalmente el problema, el modelo de optimización y la metaheurística propuesta.

4.1. Abstracción del problema

Con el objetivo de brindar una solución a la recolección de los residuos electrónicos en grandes ciudades, se identifican ciertas características del problema, que permiten abstraerse y utilizar**,** modelos conocidos para obtener una solución.

Considerando que el proceso de recolección de desechos puede representar hasta el 70% del presupuesto requerido para recolectar los desechos electrónicos (Kazajo et al., 2018), incluso superando otras actividades como almacenamiento y tratamiento final, se busca minimizar los costos asociados a la recolección de estos.

Por otro lado, existe tendencia que las ciudades dispongan de lugares para depositar los residuos electrónicos, comúnmente denominados puntos de recolección o puntos limpios. Estos puntos de recolección disponen de horario de apertura y de cierre, además, de cantidad incierta de desechos electrónicos en ellas.

Así, para la recolección y traslado a un punto de la red, se necesita una flota de vehículos, para desplazarse entre los puntos de recolección, en horarios permitidos y continúen al siguiente punto de recolección mientras tengan capacidad en el vehículo.

Se identifican las siguientes características del problema:

1. Un único depósito central de acopio.
2. Se dispone de un conjunto de puntos de recolección.
3. Cada punto de recolección tiene una cantidad de desecho electrónico (en kg) y una ventana de tiempo para ser visitado.
4. Los vehículos de recolección comienzan y terminan su ruta en el depósito central.
5. Un vehículo puede realizar como máximo una ruta, por lo tanto, se pueden dejar vehículos sin usar.

6. Cada punto de recolección puede ser visitado por sólo un vehículo, que se lleva todo el deshecho en esa única visita.
7. No se puede dividir en distintos vehículos la carga de recolección de un solo punto.
8. Cada vehículo tiene capacidad máxima de desecho electrónico (en kg) a trasladar.
9. No hay penalización por nodos que no se alcanzan a visitar.

4.2.Sistema logístico

El diseño del sistema logístico, se llevó a cabo mediante un enfoque de "*Mind Mapping*", conocido por ser un método efectivo para generar ideas por asociación (University of Adelaide, 2023). Por lo general, se empieza con la idea central, a partir de la cual se construye un diagrama, que incluye palabras clave, frases, conceptos y figuras. Este enfoque proporciona una estructura sistemática para el diseño eficiente de sistemas logísticos, abarcando con éxito la identificación de requisitos.

4.3.Problema de ruteo de vehículos capacitado con ventanas de tiempo

A partir de lo anteriormente expuesto, el problema tratado en el presente estudio, es problema de ruteo de vehículos capacitado con ventanas de tiempo (*Capacitated Vehicle Routing Problem with Time Windows: CVRPTW*).

El *CVRPTW* es una de las muchas variantes del *Vehicle Routing Problem* (VRP), y aparece en numerosas áreas de la industria y está definido como, la respuesta a la pregunta "¿Cuál es el mejor conjunto de rutas para que un conjunto de vehículos entregue paquetes a un conjunto de clientes?". Los principales objetivos del problema, según Montoya T, et al. (2015) consisten en: minimizar el costo de la ruta, la distancia recorrida, el número de vehículos necesarios para cubrir a todos los clientes, entre otras. La solución es un conjunto de rutas para cada vehículo, que asegure la entrega y óptimo para la función objetivo seleccionada.

En este caso los clientes disponen de ventanas de tiempo en las cuales se debe completar las entregas ("*Time Windows*") y se restringe la capacidad (G. Gutiérrez-Jarpa, 2010). En el presente estudio, el problema no es la entrega de paquetes sino, la recolección de éstos. Las restricciones usadas en la literatura se relacionan con: la capacidad de los

vehículos, ventanas de tiempo, vehículos heterogéneos, múltiples almacenes, múltiples viajes de vehículos, etc.

El problema tratado, se plantea como:

> *"Dado un conjunto de vehículos y un conjunto de solicitudes de transporte en los nodos de la red, se debe encontrar un conjunto de rutas que realicen algunas o todas las rutas de transporte"*

El objetivo es minimizar la distancia total de la ruta de recolección.

En la Figura 4.1 se presenta una representación gráfica del problema. Se indican: los Puntos Limpios, el depósito y las variables que indican, la información de cada Punto Limpio. En la Figura 4.2 se presenta una representación posible de solución. Se ven las rutas de los vehículos (*k1* y *k2)*, que cumplen las restricciones y minimizan la función objetivo.

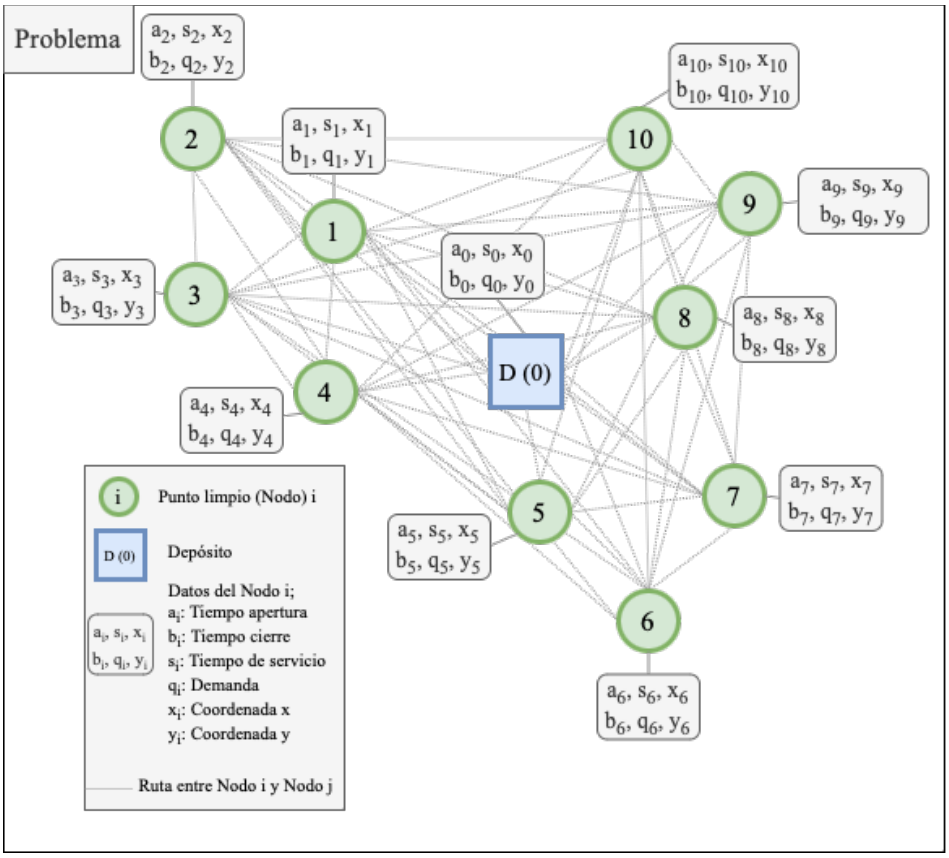

Figura 4.1 - Abstracción del problema

Fuente: Elaboración propia

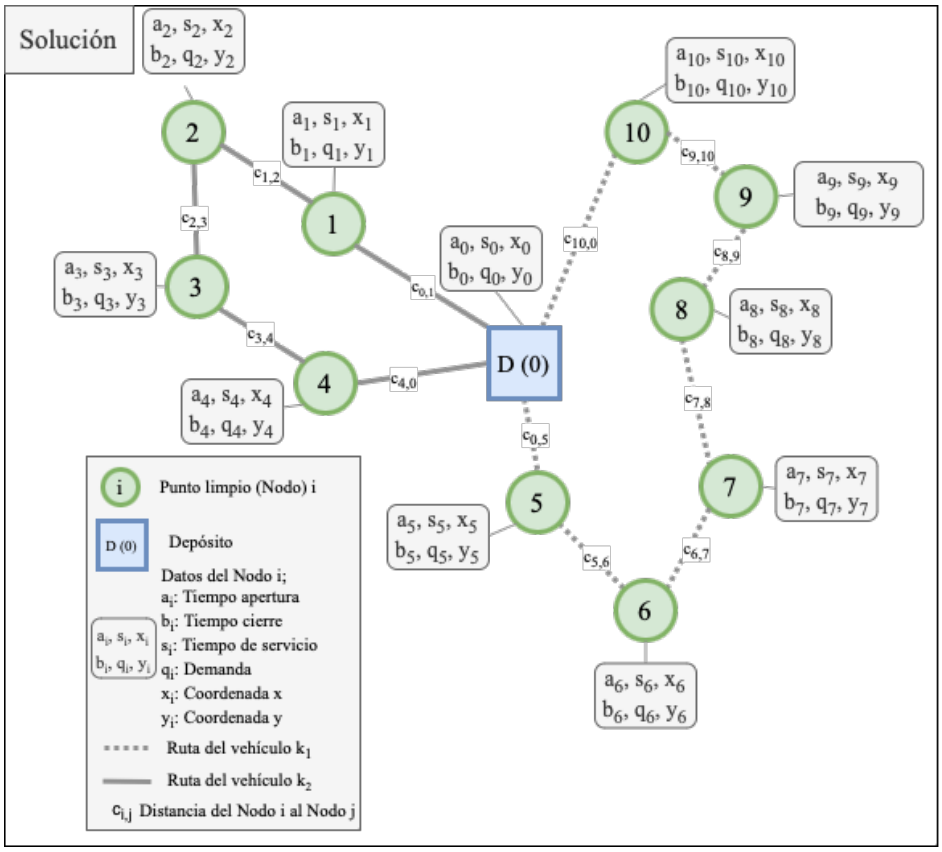

Figura 4.2 - Representación de la solución del CVRPTW

Fuente: Elaboración propia

Para el CVRPTW, los algoritmos híbridos han probado ser los más efectivos en términos de calidad de resultados y velocidad. Los algoritmos metaheurísticos son la técnica de solución preferida por muchos investigadores, debido a su velocidad y capacidad de manejar instancias grandes (Sar & Ghadimi, 2023). Estos permiten obtener rápidamente soluciones factibles de buena calidad y su desempeño se juzga empíricamente (Desaulniers, Madsen, & Ropke, 2014).

4.4.Modelo de programación matemática

El problema tratado en este estudio corresponde, al *Capacitated Vehicle Routing Problem with Time Windows* (*CVRPTW*) y en la presente sección se describen: los índices, parámetros, conjuntos, variables de decisión y restricciones utilizadas. El objetivo es minimizar la distancia total de la ruta de recolección.

El problema se puede describir mediante dos conjuntos. El primero, de puntos de recolección y depósito, cuyas características incluyen los tiempos de apertura, cierre, servicio, coordenadas en el plano y demanda. El segundo, corresponde a los vehículos y sus capacidades, esto es, cuantos kilogramos de desecho pueden llevar.

El modelo mostrado a continuación, se basa en VRPTW1 de Toth & Vigo (2014) para el Ruteo de Vehículos con Ventanas de Tiempo. Los parámetros involucrados son presentados en la Tabla 4.1.

Los índices que se presentan, *i,j,k*, permiten indexar las variables de decisión. Y los parámetros del problema representan características de las instancias, tales como: los puntos de recolección, los vehículos disponibles, los tiempos de inicio, fin y servicio, la demanda de los puntos de recolección y la capacidad de los vehículos. Los conjuntos definidos corresponden a colecciones de puntos de recolección, depósito y vehículos. Los conjuntos son utilizados en las ecuaciones (2) a (13), con el objetivo de generar grupos de ecuaciones, utilizando la notación de los índices sobre ellos.

Índices		
i	Nodo i que representa un punto de recolección o depósito	
j	Nodo j que representa un punto de recolección o depósito	
k	Vehículo k	
Parámetros		
n	Número de puntos de recolección	
v	Número de vehículos disponibles	
x_i	Coordenada X del nodo i	
y_i	Coordenada Y del nodo j	
c_{ij}	Distancia euclidiana entre nodo i y nodo j [km]	
a_i	Tiempo inicial para atender al nodo i [min]	
b_i	Tiempo final para atender el nodo i [min]	
s_i	Tiempo de servicio en el nodo i [min]	
Q_k	Capacidad del vehículo k [kg]	
q_i	Demanda del nodo i [kg]	
Conjuntos		
V	Conjunto de todos los puntos de recolección y depósito	$V = \{0,1,2,\dots,n,n+1\}$
N	Conjunto de todos los puntos de recolección	$N = \{1,2,3,\dots,n\}$
K	Conjunto de todos los vehículos	$K = \{0,1,2,\dots v\}$
A	Arcos del problema	$A = \{(i,j)\ \forall i \in V\ \forall j \in V\ i \neq j\}$
Variables de decisión		
x_{ijk}	Si el vehículo k viaja del nodo i al nodo j	$k \in K$ $i,j \in N$
T_{ik}	Tiempo que dedica el vehículo k al punto i	$k \in K$ $i \in C$

Tabla 4.1 - Parámetros del modelo de programación matemática

Fuente: Elaboración propia

La distancia entre los nodos *i* y *j*, c_{ij}, se calcula utilizando la parte entera de la distancia euclidiana definida en la ecuación (1).

$$c_{ij} = \lfloor \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2} \rfloor \quad \textbf{(1)}$$

El modelo usado, se presenta en las ecuaciones: (2) a (12) y mostradas a continuación.

Función Objetivo:

$$\min \sum_{i \in N} \sum_{j \in N} \sum_{k \in K} c_{ij} x_{ijk} \tag{2}$$

Sujeto a:

$$\sum_{\substack{j \in N \cup \{0\} \\ i \neq j}} \sum_{k \in K} x_{ijk} = 1 \qquad \forall i \in N \tag{3}$$

$$\sum_{\substack{j \in V \\ j \neq 0}} x_{0jk} = 1 \qquad \forall k \in K \tag{4}$$

$$\sum_{\substack{i \in N \cup \{0\} \\ i \neq j}} x_{ijk} = \sum_{\substack{i \in N \cup \{n+1\} \\ i \neq j}} x_{jik} \qquad \forall j \in N, \forall k \in K \tag{5}$$

$$\sum_{\substack{i \in V \\ i \neq n+1}} x_{i(n+1)k} = 1 \qquad \forall k \in K \tag{6}$$

$$T_{ik} + c_{ij} + s_i - M_{ij}\left(1 - x_{ijk}\right) \leq T_{jk} \qquad \forall i \in N, \forall j \in V, \forall k \in K \tag{7}$$

$$T_{ik} \geq a_i \qquad \forall i \in V, \forall k \in K \tag{8}$$

$$T_{ik} \leq b_i \qquad \forall i \in V, \forall k \in K \tag{9}$$

$$\sum_{\substack{j \in N \cup \{n+1\} \\ j \neq i}} \sum_{i \in N} q_i x_{ijk} \leq Q_k \qquad \forall k \in K \tag{10}$$

$$x_{ijk} \in \{0,1\} \qquad \forall i, j \in V, \forall k \in K \tag{11}$$

$$T_{ik} \geq 0 \qquad \forall i \in V, \forall k \in V \tag{12}$$

La ecuación (2) corresponde a la función objetivo, que minimiza la distancia total de las rutas de recolección. Las restricciones del tipo (3) aseguran, que cada punto de recolección sea visitado exactamente una vez y por un solo vehículo. Las restricciones del conjunto (4) aseguran, que cada vehículo tenga una dirección de destino, que puede ser un punto de recolección o el depósito. Las restricciones descritas en (5) aseguran que, si un vehículo llega a un punto de recolección, este vehículo debe continuar en alguna dirección, ya sea, otro punto de recolección o el depósito. Las restricciones del conjunto (6) aseguran, que todos los vehículos deben regresar al depósito. Las restricciones de los conjuntos (7), (8) y (9) aseguran, que el vehículo cumpla con visitar los puntos de recolección dentro, de las ventanas de tiempo y cumpliendo, los tiempos de servicio. Además, se considera que las restricciones del conjunto (7) restringen, las sub-rutas donde, M debe ser lo suficientemente grande, y se define en (13). Las restricciones del conjunto (10) restringen, la capacidad de los vehículos limitando, la demanda a cubrir en cada ruta, según las demandas individuales de cada punto de recolección. Finalmente, las restricciones de tipo (11) y (12) establecen los rangos de las variables de decisión. Notar que las variables de decisión x_{ijk} son binarias y que T_{ik} puede tomar un valor real mayor que 0.

$$M_{ij} = b_i - a_j + s_i + t_{ij} \quad \textbf{(13)}$$

Los problemas: CVRP y VRPTW se clasifican como NP-Hard (Toth & Vigo, 2014). Esto se debe a que existe una reducción del problema del CVRP a otro perteneciente a la clase NP-Hard (Garey & Johnson, 1979). Incluso encontrar una solución factible al VRPTW es un problema NP-Completo (Savelsbergh, 1985). La complejidad computacional es al menos tan compleja como tiempo polinomial no determinista, por lo que si bien, resolver el problema mediante la programación matemática asegura la optimalidad de la solución, esto no es práctico para problemas de tamaño real, pues tomaría mucho tiempo resolverlos.

4.5. Metaheurísticas

Con el objetivo de obtener soluciones de óptimos locales al problema con bajos tiempos computacionales, se utiliza una metaheurística. La metaheurística usada para la resolución del problema corresponde a *Iterated Local Search* (Lourenço, Martin, & Stützle, 2010).

Iterated Local Search es una metaheurística que ofrece una estructura simple y permite la búsqueda de varios vecindarios mediante búsquedas locales y perturbaciones. Ha sido ampliamente utilizada para la resolución de variantes del VRP. Öztaş & Tuş (2022) proponen una metaheurística híbrida basada en ILS para la resolución del VRPSPD. Máximo, Cordeau, & Nascimento (2022) proponen un Adaptative Iterated Local Search para la resolución del HVRP. Raggio (2020) resuelve el CCCVRPTWSD con ILS y Simulated Annealing. Munari (2016) resuelve el VRPTWMD utilizando ILS. Hashimoto, Yagiura, & Ibaraki (2008) proponen un ILS para el *time-dependent* VRPTW.

El *Iterated Local Search* propuesto, está basado en lo presentado por Raggio (2020), Munari (2016), Cote (2012), Lourenço, Martin, & Stützle (2010). Se presentan dos búsquedas locales y dos perturbaciones. El algoritmo es implementado en *Python3*.

4.5.1. Representación de la solución

La solución es representada utilizando 4 arreglos.

- *next:* Un arreglo con los ID que indican el siguiente nodo desde el nodo en la posición *i*.
- *prev:* Un arreglo con los ID que indican el nodo anterior desde el nodo en la posición *i*.
- *assigned_to:* Un arreglo con los ID que indican a que vehículo está asignado el nodo en la posición *i*.
- *unassigneds:* Un arreglo con los ID que indican qué nodos no están asignados en la solución.

En el Anexo, Figura B.1 se presenta una solución y los arreglos correspondientes a ella.

El código implementado en *Python3* permite modificar mediante el uso de objetos, métodos y funciones estos arreglos. El código se basa en la estructura que implementó Cote (2012), en C++ para el problema del VRP.

Los principales objetos implementados son: *Problem*, *Solution*, *Node*, *Driver*, *CostFunction*, *Route, Move, Opt2, SequentialInsertion, Shift1k, RandomRemove, RandomSwap* y *Ils*.

4.5.2. Pseudo código ILS

En este segmento, se describe la estructura general del *ILS*. El algoritmo incluye distintas funciones que serán explicadas más adelante en su propia sección.

A partir de un problema *p*, primero se ejecuta un algoritmo de inserción secuencial, que genera una solución factible inicial. Luego, se aplican los algoritmos de búsqueda local 1 y búsqueda local 2, con el fin de mejorar esta solución. En la línea 5, se inicia un ciclo limitado por la duración permitida máxima, el cual efectúa los algoritmos de perturbación 1 y perturbación 2, con el objetivo de diversificar el espacio de solución. Una vez que el espacio de solución se diversificó, se efectúan búsquedas locales, 1 y 2, con el objetivo de intensificar la búsqueda. En las líneas 10 a 12, se evalúa la solución encontrada, donde, si esta tiene un costo menor, entonces se guarda como la mejor encontrada (s*). Finalmente, en las líneas 13 a 15, está el criterio de aceptación, donde se acepta la nueva solución, si esta representa una mejora de al menos un 5% sobre la anterior. El procedimiento de las líneas 6 a 15 se repite hasta cumplir un tiempo máximo de ejecución.

```
    Algoritmo 1: Pseudocódigo ILS
    Input: p, γ
1   s = InserciónSecuencial(p);
2   s = BusquedaLocal1(s);
3   s = BusquedaLocal2(s);
4   s*=s;
5   while TiempoDeEjecucion do:
6       | s = Perturbacion1(s);
7       | s = Perturbacion2(s)
8       | s = BusquedaLocal1(s);
9       | s = BusquedaLocal2(s);
10      | if Costo(s) < Costo(s*) then:
11      |  | s*=s;
12      | end
13      | if Costo(s*) − Costo(s)/ Costo(s*) > 0.05 then:
14      |  | s = s*;
15      | end
16  end
    Output: s
```

Fuente: Basado en el trabajo de Lourenço, Martin, & Stützle (2010).

4.5.3. Factibilidad de la solución metaheurística

La función del Algoritmo 2 verifica la factibilidad de una solución. El algoritmo recibe una solución y retorna un valor booleano de verdadero si es factible o falso en caso contrario.

En la línea 1 se inicializa la lista *factible* que almacenará valores booleanos. En las líneas 2 a 12 se lleva a cabo un bucle que revisa la ruta de cada vehículo, revisando tres condiciones:

1. *ruta_estructura*: evalúa que cada ruta comience y termine en el depósito.
2. *ruta_tiempo*: evalúa que cada vehículo visite los puntos de recolección dentro de las ventanas de tiempo y cumpla los tiempos de servicio.
3. *ruta_capacidad*: evalúa que cada vehículo cumpla la demanda de la ruta.

Luego, en las líneas 15 a 17, la función *solución_estructura* evalúa que cada punto de recolección sea visitado a lo más una vez y por un solo vehículo. En las líneas 19 a 22, se revisa el resultado del algoritmo. Si en la lista factible hay un valor falso, entonces la solución no es factible y el algoritmo retorna el valor falso. Si no hay ningún valor falso en la lista *factible*, entonces la solución es factible y el algoritmo retorna el valor verdadero.

Notar que este algoritmo permite no cumplir con la restricción de visitar a todos los nodos del problema. Desaulniers, Madsen, & Ropke (2014) destacan la importancia de permitir soluciones infactibles en la búsqueda, pues permite explorar mejor el espacio de búsqueda y es una herramienta importante, para encontrar soluciones de alta calidad.

Algoritmo 2: EsFactible

Input: s

```
1   factible = []
2   for each vehiculo en vehículos do:
3   |  ruta = ruta del vehículo en s
4   |  if ruta_estructura(ruta) = Falso then:
5   |  |  factible = factible + Falso
6   |  End
7   |  if ruta_tiempo(ruta) = Falso then:
8   |  |  factible = factible + Falso
9   |  End
10  |  if ruta_capacidad(ruta) = Falso then:
11  |  |  factible = factible + Falso
12  |  End
13  End
15  if solucion_estructura(s) = Falso then:
16  |  factible = factible + Falso
17  end
19  if Falso in factible then:
20  |  return Falso
21  else:
22  |  return Verdadero
```

Output: Verdadero o Falso

Fuente: Elaboración propia

4.5.4. Heurística inicial: Inserción secuencial

El algoritmo de inserción secuencial es una heurística de construcción que permite generar una solución factible inicial, donde se busca una solución de manera secuencial insertando nodos en rutas existentes de manera iterativa.

Este algoritmo toma como entrada el problema sin inicializar o una solución cualquiera y retorna otra solución donde se han insertado nodos que no están en ninguna ruta, siempre y cuando sea factible. Si no logra insertar ningún nodo de manera factible, retorna la misma solución que recibió.

En las líneas 1 y 2 se inicializan las listas *nodos* y *rehusados*. En nodos se guardan aquellos que no son parte de ninguna ruta, y rehusados, se inicializa vacía, pues contendrá los nodos que no se pueden insertar en ninguna ruta. En las líneas 3 a 16 se inicia un bucle que busca el mejor cambio para insertar el nodo en alguna ruta. Para cada vehículo disponible, se calcula el costo incremental de incluir el nodo en la ruta actual. Se selecciona el cambio que minimiza dicho costo. Si este cambio es factible y su costo es menor que el mejor cambio previamente encontrado, entonces se actualiza *mejor_cambio*. Luego, se verifica la factibilidad de la solución que considera el *cambio*, si es factible, entonces se aplica a la solución. Si no es factible, el nodo se añade a *rehusados* y se deja fuera de la solución. Finalmente, se retorna la solución final.

```
    Algoritmo 3: Inserción secuencial
    Input: s
1   nodos = Lista de nodos que no son parte de una ruta;
2   rehusados = []
3   for each nodo en nodos do:
4   |   mejor_cambio = ∅ ;
5   |   for each vehículo en vehículos do:
6   |   |   cambio = Ruta para vehículo que incluye a nodo con el menor
    |   |  costo incremental;
7   |   |   If  EsFactible(cambio)  and  Costo(cambio)<
    |   |  Costo(mejor_cambio) then:
8   |   |   |  mejor_cambio = cambio;
9   |   |   end
10  |   |   if EsFactible(mejor_cambio) then:
11  |   |   |  s = Aplicar mejor_cambio a s
12  |   |   else
13  |   |   |  s = Añadir nodo a reusados
14  |   |   end
15  |   end
16  end
    Output: s
```

Fuente: Basado en el trabajo de Jean-Francois Cote (2012)

4.5.5. Búsqueda Local 1

El algoritmo *shift(1,0)* es una estrategia de búsqueda local que aplica cambios de nodos entre rutas. Brevemente, el algoritmo mueve un nodo desde una ruta hacia otra.

El algoritmo comienza con la solución actual, y crea dos listas, N y R, que representan los nodos y las rutas del problema. El proceso iterativo se lleva a cabo entre las líneas 4 y 13, donde cada nodo es evaluado para su posible inserción en cada ruta existente. Para cada nodo existente y cada ruta, se evalúa la posibilidad de quitar cada nodo de la ruta e insertar el nodo existente en la posición de este otro. El cambio se evalúa en la línea 8, donde en caso de ser factible y tener un menor costo, se actualiza el mejor cambio encontrado. Finalmente, en la línea 14, si el mejor cambio es factible, entonces se aplica el cambio a la solución.

En la Figura 4.3, se presenta la aplicación del algoritmo de forma gráfica. En este caso, el algoritmo elige el mejor cambio de quitar el *Nodo 1* de la ruta k_1 e insertarlo en la ruta k_2 entre el *Depósito 0* y el *Nodo 2*.

Algoritmo 4: Búsqueda Local 1: Shift(1,0)

Input: s

```
1   N = Lista con nodos en s;
2   R = Lista con rutas en s;
3   mejor_cambio = ∅;
4   for nodo_i en N do:
5       for ruta en R do:
6           for j entre 0 y len(ruta) − 1 do:
7               cambio = Quitar nodo_i de ruta e insertar nodo_i en la
                posición j si y solo sí ruta[j]≠ i;
8               if EsFactible(cambio) y Costo(cambio) < Costo(mejor_cambio)
                then:
9                   mejor_cambio = cambio;
10              End
11          end
12      end
13  end
14  if EsFactible(mejor_cambio) then:
15      s = Aplicar mejor_cambio a s
16  end
```

Output: s

Fuente: Basado en el trabajo de Jean-Francois Cote (2012)

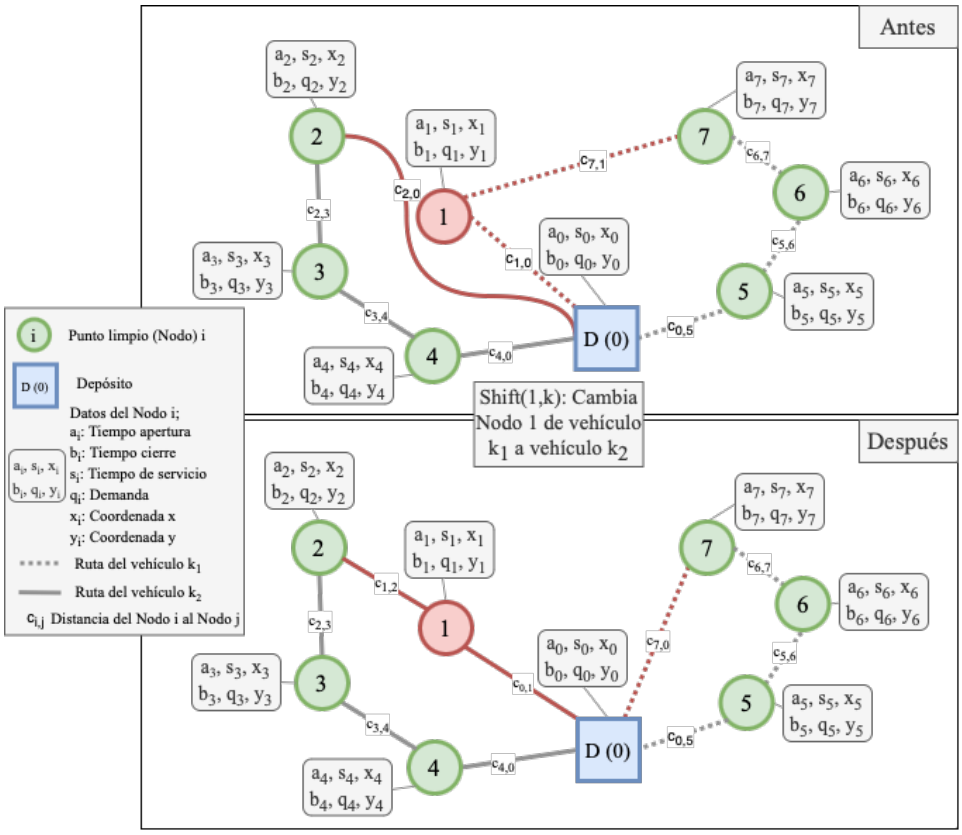

Figura 4.3 - Representación gráfica de Algoritmo Shift(1,0)

Fuente: Elaboración propia

4.5.6. Búsqueda Local 2

El algoritmo de búsqueda local 2, corresponde a un *2-opt* (Bräysy & Gendreau, 2005), que busca mejorar una ruta dada mediante intercambios de arcos no adyacentes, es decir, que no están uno tras el otro.

En resumen, el cambio que realiza es que dos arcos no adyacentes (i,i+) y (j,j+) se quitan y otros dos arcos, (i,j) y (i+,j+) son ingresados, de tal manera que las rutas se ven alteradas.

En cada iteración, el algoritmo evalúa todas las combinaciones posibles de arcos no adyacentes en la ruta y realiza un intercambio si resulta en una nueva ruta factible y de menor costo. El proceso se repite hasta que no se pueden encontrar más mejoras en la ruta actual. Si se encuentra una mejora, se actualiza la mejor ruta y se repite el proceso.

Este tipo de búsqueda local entre rutas permite favorecer la intensificación de la solución encontrada, acercándola a un mínimo local.

La Figura 4.4 muestra el resultado del algoritmo *2-opt* en la ruta k_2, donde intercambia los arcos no adyacentes *(1,3)* y *(2,4)* por *(1,2)* y *(3,4)*.

Algoritmo 5: Búsqueda Local 2: 2-opt

```
    Input: s, ruta
1   mejor_ruta = ruta
2   n = Largo ruta
3   mejora = True
4   while mejora do:
5   |  mejora = False
6   |  for i <- 1 hasta n do:
7   |  |  for j<- i+1 hasta n do:
8   |  |  |  nueva_ruta = Desconecta las aristas (i,i+1) y (j,j+1) de la
    |  |  |  ruta, luego las reemplaza por (i,j) y (i+1,j+1)
9   |  |  |  if  EsFactible(nueva_ruta)  y  Costo(nueva_ruta)  <
    |  |  |  Costo(mejor_ruta) then:
10  |  |  |  |  mejor_ruta = nueva_ruta;
11  |  |  |  |  mejora = True;
12  |  |  |  |  break
13  |  |  |  end
14  |  |  end
15  |  |  if mejora = True then:
16  |  |  |  break
17  |  |  end
18  |  end
19  end
20  if mejora = True then:
21  |  s = Aplicar mejor_ruta a s
22  end
    Output: s
```

Fuente: Basado en el trabajo de Bräysy & Gendreau, (2005)

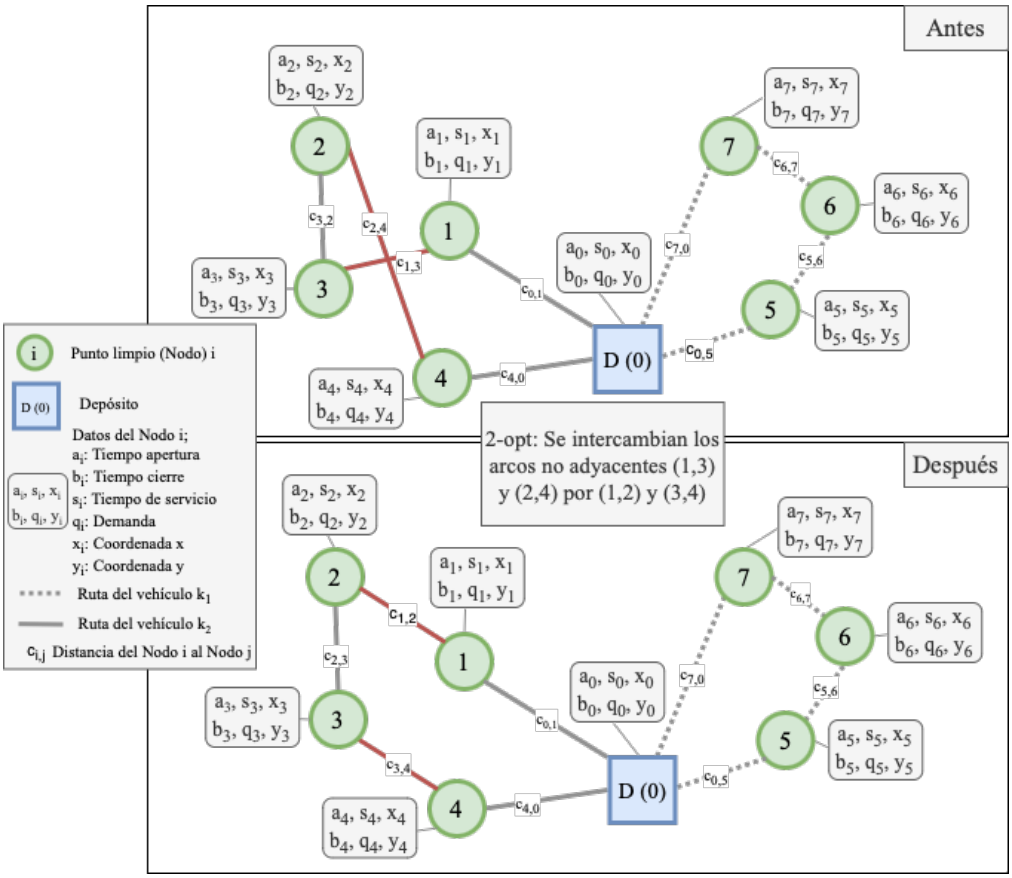

Figura 4.4 - Representación gráfica del Algoritmo 2-opt

Fuente: Elaboración Propia

4.5.7. Perturbación 1

El algoritmo de perturbación 1, llamado Intercambio Aleatorio toma como entrada una solución y un parámetro v, que indica el número de perturbaciones que se realizarán.

En la línea 2 se inicializa una lista que guarda los intercambios probados en tuplas (ej. (i,j) representa que se cambió el Nodo i por el Nodo j). Entre la línea 4 y la 14 se hace un bucle. En cada iteración del bucle, se explora una perturbación de intercambio aleatorio, donde se eligen aleatoriamente dos nodos, i y j, que cumplan con las condiciones de la línea 5. Los nodos deben pertenecer a la solución, no pueden ser el mismo y no deben pertenecer a la lista de intercambios. En la línea 6 se actualiza la lista con los intercambios. En la línea 7 se realiza el intercambio y en la línea 8 se evalúa su factibilidad. En caso de ser factible, se aplica el cambio. En caso contrario, se deshace el cambio y se continúa buscando

aleatoriamente un intercambio factible, hasta que se cumpla la cantidad de intercambios v. Finalmente, una vez realizados los intercambios, el algoritmo retorna una solución factible.

La perturbación presentada tiene por objetivo introducir variaciones en la solución, que permiten explorar configuraciones y posibles soluciones mejores en el espacio de búsqueda mediante la diversificación.

La Figura 4.5 muestra el resultado la perturbación 1, donde se eligen aleatoriamente los Nodos 5 y 6 en la ruta k_1 y se intercambian de posición.

```
    Algoritmo 6: Perturbación 1: Intercambio Aleatorio
    Input: s, v
1   s'= s;
2   intercambios = [];
4   while v > 0 do:
5       i,j = Elegir dos nodos aleatorios i y j de s' tal que i≠j y (i,j)
        ó (j,i) ∉ intercambios
6       intercambios = añadir (i,j) y (j,i) a intercambios
7       s = Intercambiar nodos i y j en s;
8       if EsFactible(s) then:
9           s' = s;
10          v = v − 1;
11      else
12          s = s';
13      end
14  end
    Output: s'
```

Fuente: Elaboración propia

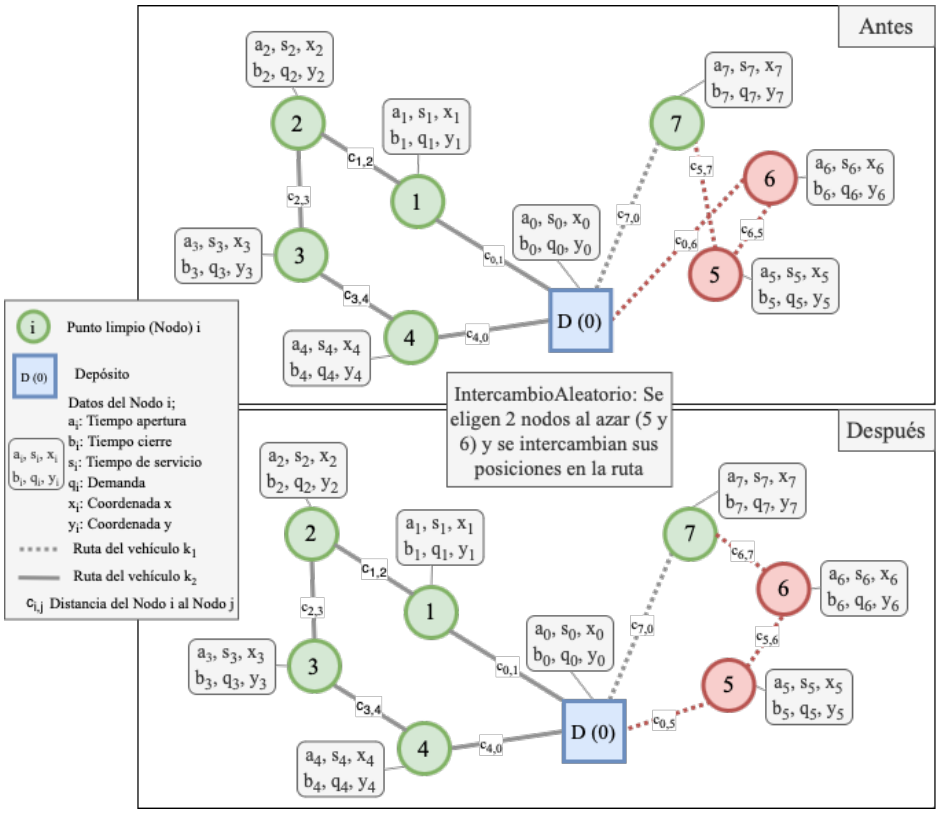

Figura 4.5 - Representación gráfica del Algoritmo Intercambio Aleatorio

Fuente: Elaboración propia

4.5.8. Perturbación 2

El Algoritmo 6 describe otra técnica de perturbación llamada "Eliminación Aleatoria e Inserción". El algoritmo toma como entrada una solución inicial s y un parámetro n, que indica el número de nodos que se eliminarán de la solución. Seguido del proceso de eliminación, el algoritmo utiliza la heurística de Inserción Secuencial para re-construir la solución.

Finalmente, el algoritmo entrega una solución perturbada, que mantiene la factibilidad.

La Figura 4.6 muestra el resultado la perturbación 2, donde se elige aleatoriamente el *Nodo 1* en la ruta k_1 y se inserta en la ruta k_2 usando el algoritmo de Inserción Secuencial.

```
    Algoritmo 7: Perturbación 2: Eliminación Aleatoria E Inserción
    Input: s, n
1   s'= s;
2   while n > 0 do:
3      i = Elegir un nodo aleatorio de s;
4      s' = Eliminar nodo i de s;
5      n = n – 1;
6   end
7   s' = InserciónSecuencial(s')
    Output: s'
```

Fuente: Elaboración propia

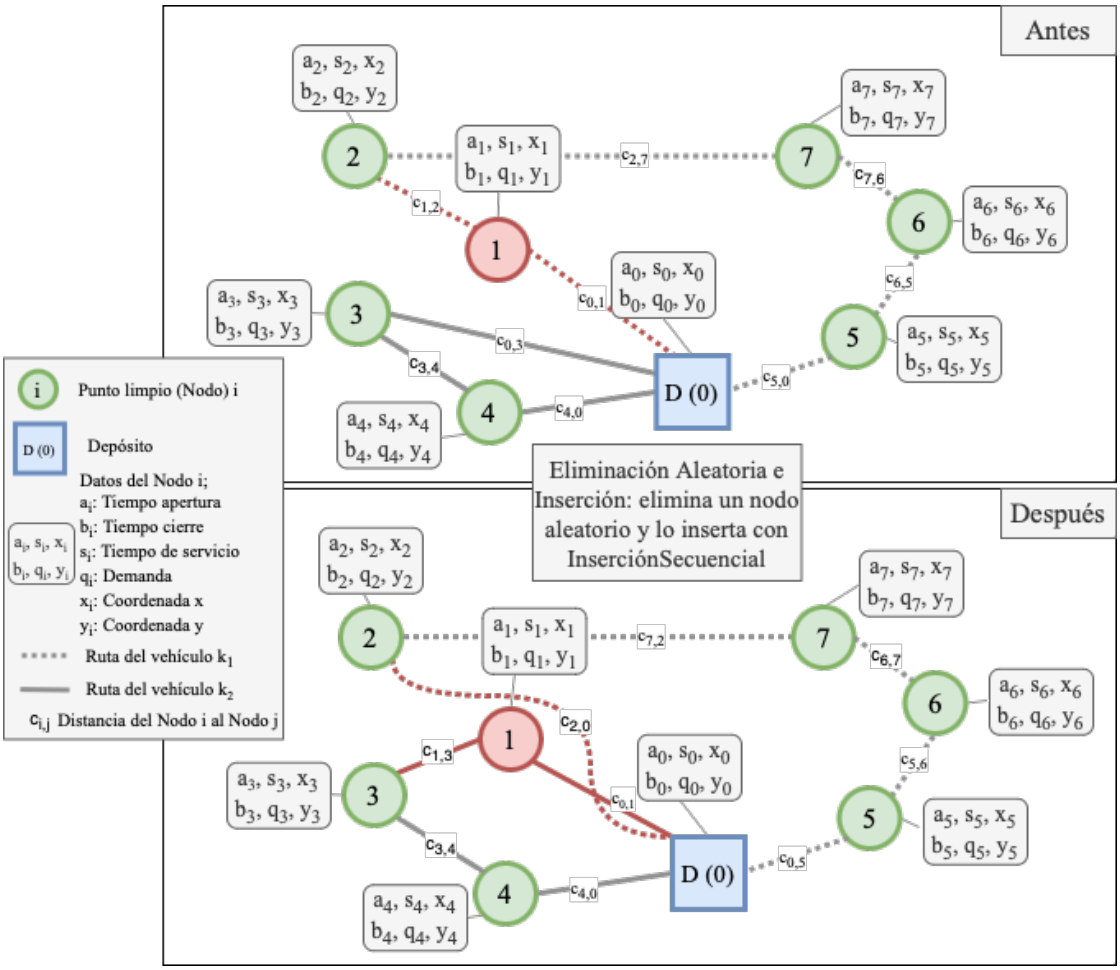

Figura 4.6 - Representación gráfica de Algoritmo Eliminación Aleatoria e Inserción

Fuente: Elaboración propia

4.6.Caso de estudio con datos geográficos

En esta sección, se presenta el caso de estudio en el cual se aplicarán los algoritmos expuestos en la sección anterior y utilizando datos geográficos actualizados de la ciudad de Los Ángeles (Hurst & CalRecycle, 2023). Una tabla con parte de la base de datos se presenta en el Anexo, Tabla A.1.

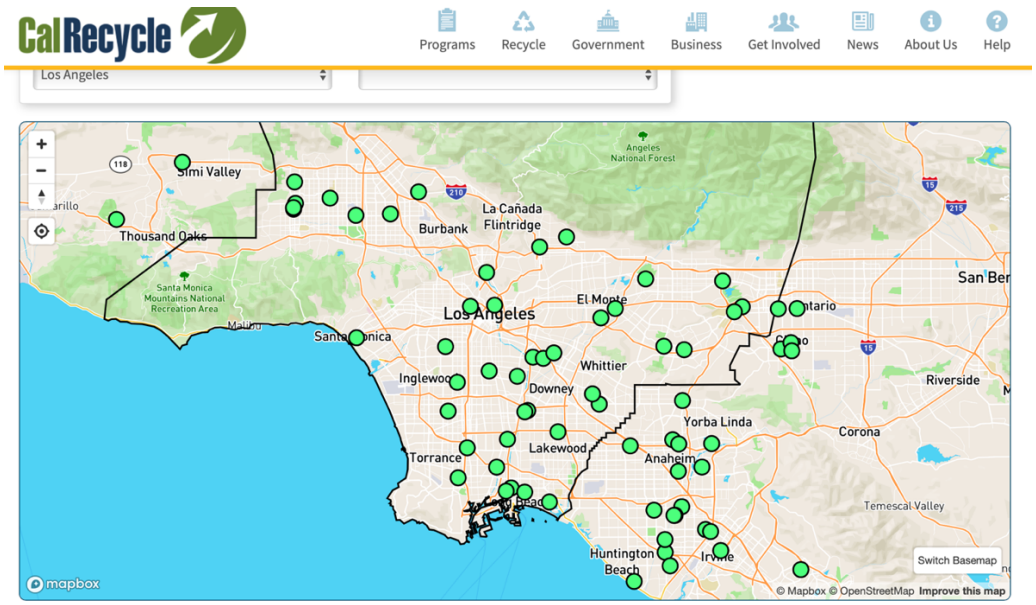

Figura 4.7 - Mapa de Los Angeles, Estados Unidos con puntos limpios

Fuente: Hurst & CalRecycle (2023)

El *Covered Electronic Waste Recycling Program*, administrado por *CalRecycle*, despliega esfuerzos significativos para facilitar el reciclaje de desechos eléctricos y electrónicos en el estado de California. *CalRecycle* proporciona una aplicación que permite a los ciudadanos encontrar el centro de reciclaje más cercano para estos materiales. La aplicación, accesible en https://www2.calrecycle.ca.gov/electronics/eRecycle/, es una herramienta fundamental para resolver el problema de recolección de los residuos electrónicos en California. En la Figura 4.7, se ve la interfaz de usuario de la aplicación.

En el caso de estudio, se utilizan los algoritmos propuestos para optimizar la ruta de vehículos encargados de la recolección de residuos electrónicos en el condado de Los Ángeles. La Figura 4.9 exhibe la distribución geográfica de los puntos de recolección en la mencionada ciudad, según los datos proporcionados por CalRecycle. Este mapa detalla los

diversos lugares donde los ciudadanos pueden depositar sus desechos electrónicos para su posterior reciclaje. En amarillo se pueden observar los puntos que son "*Collector/Recycler*" y en verde los que son "*Collector*".

Para aplicar el problema a estos datos reales, se filtraron los datos, para únicamente enfocarse en los datos del condado de Los Ángeles. Además, se consideró un centro "*Collector/Recycler*" como depósito. El criterio para su selección fue su ubicación, pues se encontraba relativamente céntrico en la distribución geográfica relativa a los otros puntos y junto a una carretera de gran envergadura. Así, este caso de estudio se trabaja con una instancia formada por los puntos de recolección y depósito presentados en la Figura 4.9.

Los datos geográficos no proveen información sobre: tiempo de apertura, tiempo de cierre, tiempo de servicio y demanda (ver Anexo, Tabla A.1). Estos parámetros son esenciales para las instancias del modelo. Para abordar esta carencia, se optó por modelar estos aspectos utilizando variables aleatorias. Posteriormente, se empleó un generador de números aleatorios junto con el método de la transformada inversa para generar valores basados en las funciones de densidad de probabilidad correspondientes a cada variable aleatoria.

Se utilizan las variables aleatorias exponencial truncada y normal con las funciones de densidad de probabilidad presentadas en la Tabla 4.3.

Variable aleatoria	**Notación**	**Función de densidad de probabilidad**
Exponencial truncada	$X \sim TEXP(\lambda, b)$	$f(x,b) = \frac{\exp(-x)}{1-\exp(-b)}\ ;\ 0 < x < b$
Normal	$X \sim N(\mu, \sigma^2)$	$f(x,\mu,\sigma^2) = \frac{1}{\sigma\sqrt{\{2\pi\}}}\, e^{-\frac{(x-\mu)^2}{2\sigma^2}}$

Tabla 4.2 - Distribuciones de probabilidad utilizadas en caso de estudio

Fuente: Elaboración propia

A continuación, se detalla cada variable aleatoria utilizada.

- Tiempo de apertura (a_i): se asume que los centros de reciclaje abren sólo entre las 9am y las 12pm, con mayor tendencia a abrir a las 9am. Se utiliza una función de distribución de probabilidad exponencial truncada, tal que $a_i \sim TEXP(9{,}12)$.

- Tiempo de cierre (b_i): se asume que los centros de reciclaje tienden a cerrar desde las 18pm, hasta las 21pm. Se utiliza una función distribución de probabilidad exponencial truncada, tal que $b_i \sim TEXP(18,21)$.
- Demanda (q_i): se asume que se distribuye normal con media 150kg y desviación estándar 25kg, tal que $q_i \sim N(150,25)$.
- Tiempo de servicio (s_i): se asume fijo de 60 minutos por nodo.

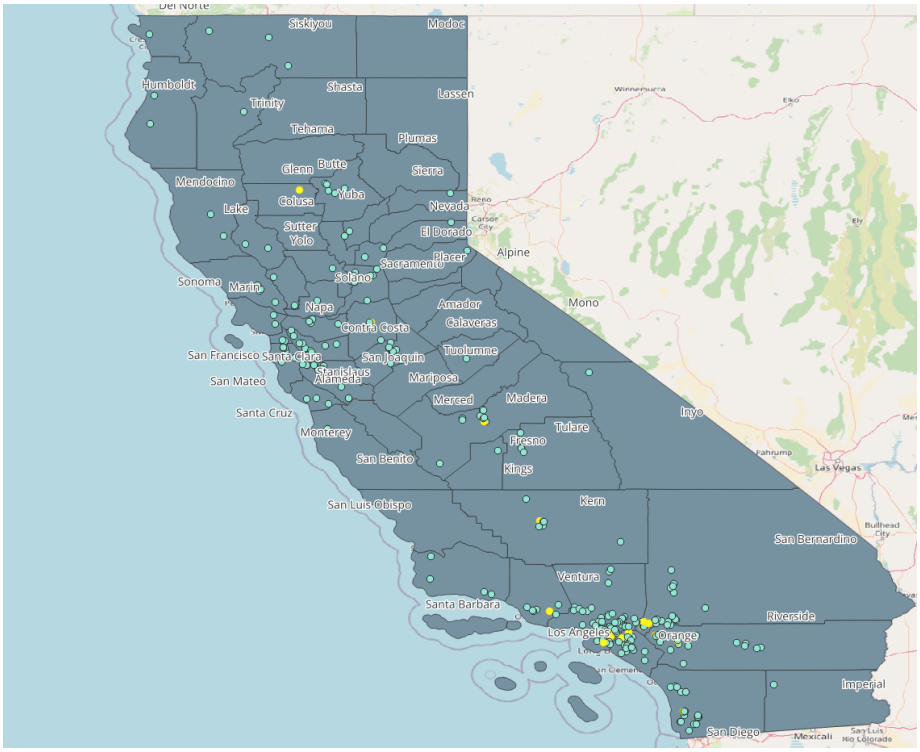

Figura 4.8 - Distribución de Puntos de Recolección en el estado de California

Fuente: Elaboración propia sobre datos obtenidos por OpenStreetMap Contributors (2015) y Hurst & CalRecycle (2023)

Para el modelo se necesitan los datos de distancia c_{ij} (km). Los datos de Hurst & CalRecycle, (2023) entregan las coordenadas geográficas en grados decimales (ϕ, λ) (latitud y longitud). Existen varios métodos para el cálculo de distancia a partir de grados decimales. La proyección de Mercator asume que la Tierra es una esfera y la proyecta sobre un cilindro de coordenadas cartesianas (x, y). La fórmula de Haversine (Sinnott, 1984) permite calcular la distancia sobre una esfera a partir de coordenadas geográficas. Algoritmos de ruta mínima como Djikstra ó A*, sobre datos geográficos permiten calcular la distancia entre dos puntos

en una ciudad (Mehlhom & Sanders, 2008). *Google Maps Distance Matrix API* es una aplicación de pago de Google, que permite calcular distancias entre coordenadas geográficas.

Se descarta el uso de algoritmos de ruta mínima para el cálculo de distancia, pues implica un mayor costo computacional. También se descarta el uso de *Google Maps Distance Matrix API,* debido a los costos asociados al cálculo. El cálculo de distancias usando la proyección de Mercator no resulta ser tan precisa como Haversine para distancias de más de 20km. La fórmula de Haversine es el método más preciso y eficiente, para calcular distancias en la superficie de la Tierra (Chamberlain, 2019). Así, se usa la fórmula de Haversine.

En la ecuación (15) se presenta la fórmula de Haversine (Sinnott, 1984), donde r es el radio promedio de la tierra (6371 km), ϕ es la latitud y λ es la longitud en grados decimales.

$$c_{ij} = 2r \arcsin \sqrt{\sin^2\left(\frac{\phi_i - \phi_j}{2}\right) + \cos(\phi_i)\cos(\phi_j)\sin^2\left(\frac{\lambda_i - \lambda_j}{2}\right)} \quad \textbf{(14)}$$

Para la evaluación económica bajo supuestos del caso de estudio, se considera que todos los vehículos son iguales, usan Diésel, tienen un rendimiento de 8km/L y pueden cargar hasta 1600kg. Un ejemplo de este vehículo es el *Ford Transit*, que pertenece a un segmento que facilita el transporte de carga en zonas urbanas. En la Tabla 4.3 se presentan los valores de referencia utilizados para el caso de estudio en USD. Entonces, se calcula el costo total de recolección en función de los kilómetros recorridos y de la cantidad de vehículos usando la ecuación (15), donde: $z_c$es el costo final; Z_s el costo de la función objetivo de la solución (en km); r_d el rendimiento del vehículo en km/l; c_d el costo del Diésel en $\$/l$; n_v el número de vehículos en la solución y c_v el precio de un vehículo.

$$z_c = \frac{Z_s[km]}{r_d[km/l]} * c_d\left[\frac{\$}{l}\right] + n_v * c_v[\$] \quad \textbf{(15)}$$

Item	Precio	Unidad de medida
Diésel	$1.05	USD/l
Ford Transit	$42,985	USD/vehículo

Tabla 4.3 - Valores de referencia usados para caso de estudio (USD)

Fuente: Precio de diésel obtenido el 23 de noviembre del 2023 de US Energy Administraiton Information https://www.eia.gov. Precio del vehículo obtenido el 23 de noviembre del 2023 de Ford https://www.ford.com/commercial-trucks/transit-chassis/

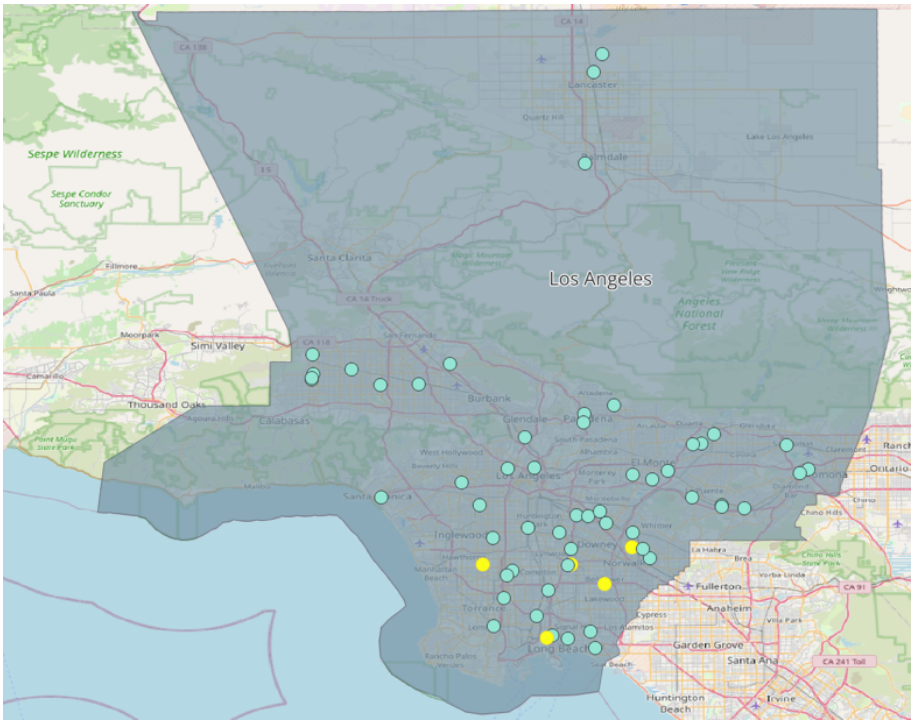

Figura 4.9 - Distribución de Puntos de Recolección en el condado de Los Ángeles

Fuente: Elaboración propia sobre datos obtenidos por OpenStreetMap Contribuitors (2015) y Hurst & CalRecycle (2023)

4.7.Experimentos computacionales

Se realizaron experimentos computacionales para resolver tanto el modelo de programación matemática con el uso de un *solver*, como la metaheurística con el uso de un lenguaje de programación.

Los experimentos fueron ejecutados en un computador Apple M1 Pro con 16 GB de memoria RAM. El sistema operativo corresponde a macOS Sonoma 14.0. El *solver* utilizado para el modelo de programación matemática es CPLEX 22.10 y la metaheurística fue implementada en Python 3.7.9. Otras herramientas computacionales que fueron utilizadas son NumPy, Pandas, DoCPLEX, Matplotlib y QGIS.

En los experimentos, se definió un máximo de 3600 segundos para la resolución del modelo de programación matemática, y 300 segundos para la metaheurística *ILS*.

En la Salida de Texto 1 se presentan los requisitos de formato de las instancias. En la línea 1 se indica el nombre de la instancia. En la línea 5 la cantidad de vehículos disponibles y la capacidad. De las líneas 10 a 13 se detallan los datos de cada punto de recolección; ID; coordenada X, coordenada Y, demanda, tiempo de apertura, tiempo de cierre y tiempo de servicio.

```
1 <Instance name>
2 <empty line>
3 VEHICLE
4 NUMBER     CAPACITY
5   K           Q
6 <empty line>
7 CUSTOMER
8 CUST NO.  XCOORD.   YCOORD.    DEMAND   READY TIME  DUE DATE   SERVICE TIME
9  <empty line>
10     0       x0        y1         q0         a0          b0            s0
11    1       x1        y2         q1         a1          b1            s1
12  ...     ...       ...        ...        ...         ...           ...
13  100     x100      y100       q100       a100        b100          s100
```

Salida de Texto 1 – Formato estándar de instancias para *benchmark* del VRPTW

Fuente: Tomado de Transport Optimization Portal (2023)

5. Resultados

En el presente capítulo se describe el sistema logístico, las instancias de prueba utilizadas, los resultados obtenidos con el modelo de programación matemática y la metaheurística *ILS*. Finalmente, se analizan los resultados obtenidos sobre el caso de estudio con datos geográficos.

5.1.Sistema logístico

Se utilizó el "*Mind Mapping"* para generar el diagrama que compone el sistema logístico presentado en la Figura 5.1. El sistema obtenido permite identificar en óvalos los distintos actores y lugares que componen el sistema. Las flechas indican el flujo de materiales (en negro) e información (en rojo).

Los proveedores le entregan las materias primas a los fabricantes, que siguen los diseños de los diseñadores para fabricar AEE. Estos productos van a los distribuidores, que almacenan y venden AEE. Una vez el usuario final adquiere el producto, inicia un ciclo de uso, que incluye mantenimiento, inspección y reparación. Cuando el AEE es desechado, termina su primer ciclo de vida. Este RAEE o *e-waste* es recibido en puntos limpios, donde son recolectados y transportados a depósitos de tratamiento de *e-waste*. Aquí, pueden ser reparados, reciclados o se puede recuperar sus materias primas, para volver a ingresar al sistema mediante los proveedores o los distribuidores secundarios. Por otra parte, se describe un flujo de información desde el tratamiento de *e-waste* a diseñadores y fabricantes, para tomar nota de fallas comunes u otros.

Destaca la importancia de los óvalos en verde. La presencia y actividades de los puntos limpios, los recolectores y los depósitos de tratamiento permiten la retroalimentación de materiales e información a otros actores. Sin estos elementos en el sistema, la cadena de suministro no permitiría iniciar un ciclo de economía circular. Luego, se destaca la importancia de la planificación de rutas para vehículos recolectores, utilizando herramientas de optimización.

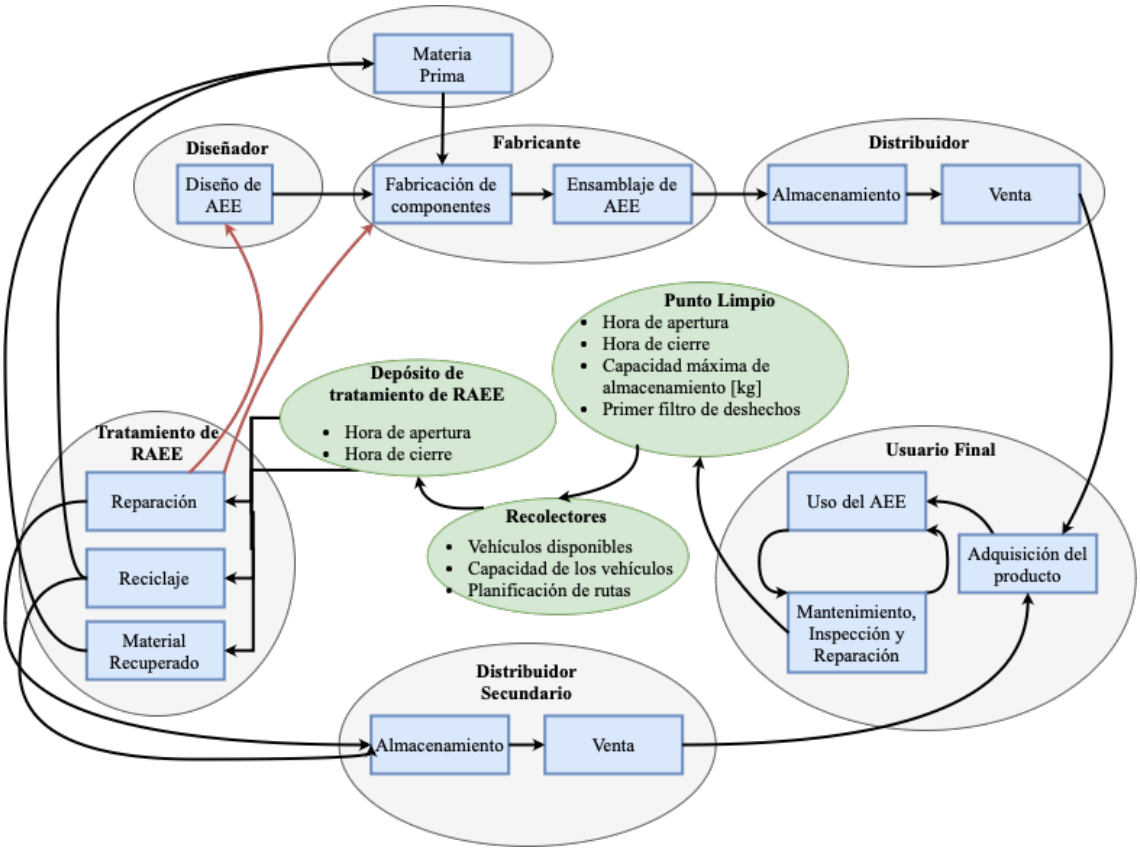

Figura 5.1 - Sistema logístico para la recolección de *e-waste*

Fuente: Elaboración propia

5.2.Validación con instancias de prueba

Con el objetivo de validar tanto la implementación del modelo de programación matemática, como del algoritmo metaheurístico propuesto, se utilizaron las instancias de Solomon (1987).

Para el modelo de programación matemática, se realizaron ejecuciones sobre problemas más pequeños, de máximo 25 puntos de recolección, dados los elevados tiempos computacionales que supone la resolución del problema NP-Completo.

En la Figura 5.2 se observa el resultado obtenido con el modelo de programación matemática para la instancia c101-25 de Solomon (1987) y en la Salida de Texto 2 se puede ver el resultado del programa. Para resolver este problema, el computador demoró un total de 0.62 s, llegando rápidamente a la solución óptima utilizando las herramientas del *solver*. Dado que el resultado coincide con lo planteado por otros autores, la implementación del modelo de programación matemática del *CVRPTW* queda validado.

```
Instance name : c101-25
Authors       : Nicolás Netz
Date          : 04.12.2023 20:34:39
Reference       : Ruteo de vehículos para la recolección de residuos
electrónicos. Un caso de estudio utilizando datos geográficos.
Solution

Route  1 : 13 17 18 19 15 16 14 12
Route  2 : 5 3 7 8 10 11 9 6 4 2 1
Route  3 : 20 24 25 23 22 21

Cost: 191.30
```

Salida de Texto 2 - Resultados instancia c101-25

Fuente: Elaboración propia

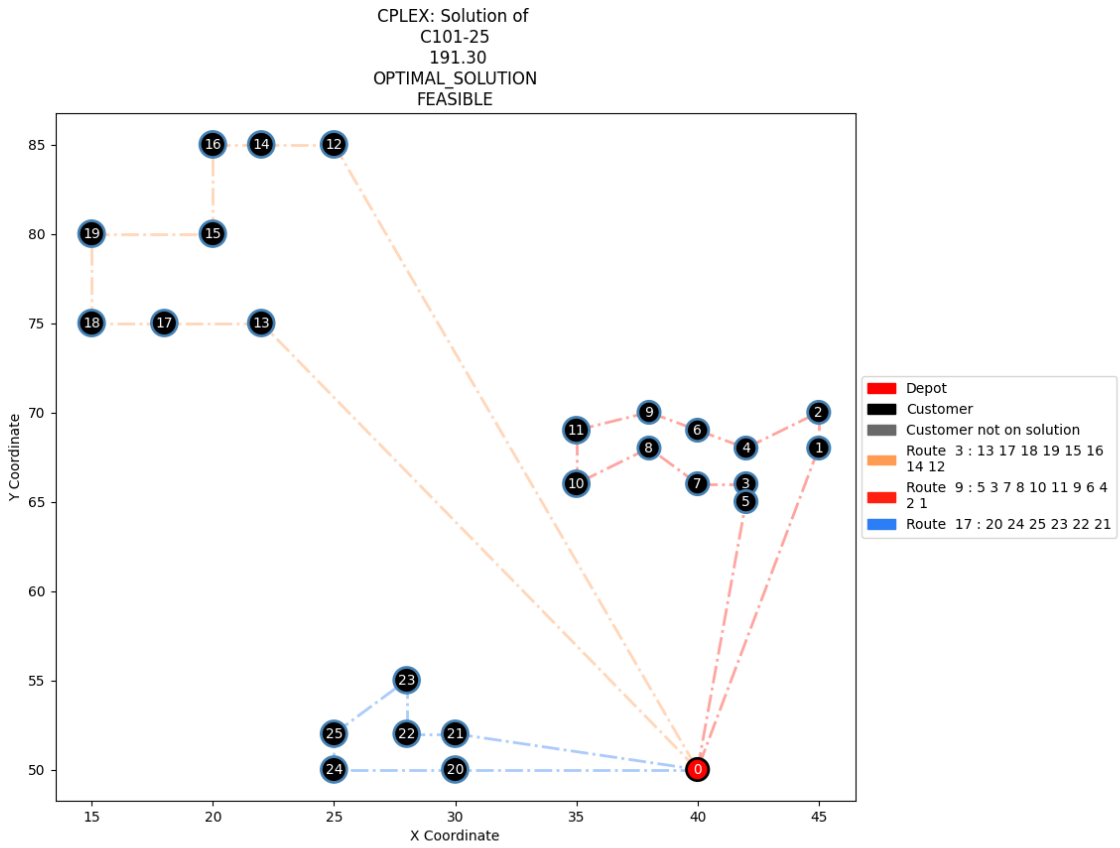

Figura 5.2 - Resultados instancia c101-25

Fuente: Elaboración propia

Se realizaron otros experimentos, donde se dejó al computador resolver mediante el modelo de programación matemática y el uso de CPLEX con un límite de tiempo de 3600 segundos todas las instancias de Solomon (1987) con el total de nodos, sin embargo, debido a la complejidad y la capacidad computacionales, sólo se logró resolver una reducida cantidad de instancias con factibilidad. El resto de las instancias no se logró resolver con el modelo de programación matemática. En la Tabla 5.1 - Resultados factibles de *solver* en instancias de se presentan los resultados para las instancias que terminaron en un resultado factible.

Instancia	Nodos	Vehículos	Vehículos	Objetivo	Tipo de Resultado	gap (%)
C202-101	100	25	7	961,2	Factible	46,1
RC101-101	100	25	21	2111,9	Factible	46,6

Tabla 5.1 - Resultados factibles de *solver* en instancias de Solomon (1987)

Fuente: Elaboración propia

En la Tabla 5.2 se presentan los resultados obtenidos mediante el algoritmo *Iterated Local Search*, que se aplicó a las instancias de Solomon (1987). Se limitó a un tiempo de ejecución de 300 segundos y se utilizaron todos los nodos.

En la columna *ILS* se presenta el número de vehículos utilizado y el valor de la función objetivo, mientras que en la columna Literatura, se presenta el número de vehículos utilizado, el valor de la función objetivo y la referencia a la publicación donde se encontró ese valor óptimo mediante el modelo de programación matemática del *CVRPTW*. Finalmente, en la columna de comparación, se detalla el porcentaje de variación entre el objetivo encontrado por *ILS* y el óptimo, además de la diferencia en número de vehículos utilizados para obtener el resultado.

Además, en la Figura 5.3 y la Figura 5.4 se presenta la convergencia de los costos de la función objetivo mediante las iteraciones de *ILS*, para dos instancias. En estas figuras se pueden ver dos líneas, una con el mejor costo actual en la iteración de *ILS*, en azul, y otra con el costo evaluado en la iteración, tras las perturbaciones y búsquedas locales. Notar que la metaheurística logra fuertes mejoras al inicio, sin embargo, queda estancada en óptimos locales hasta que logra salir de ellos, optimizando aún más la función objetivo en algunos casos.

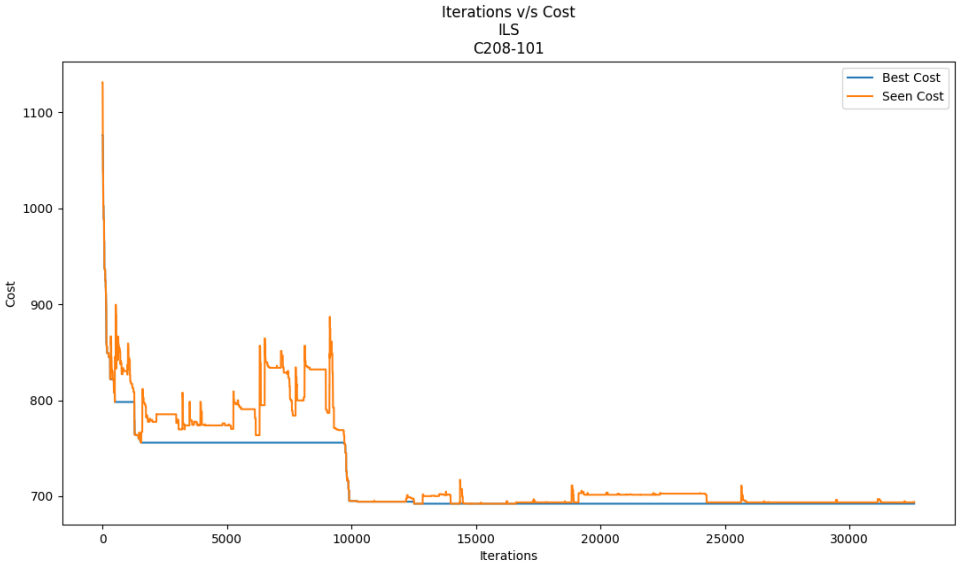

Figura 5.3 - Convergencia de costos de función objetivo en ILS para c208-101

Fuente: Elaboración propia

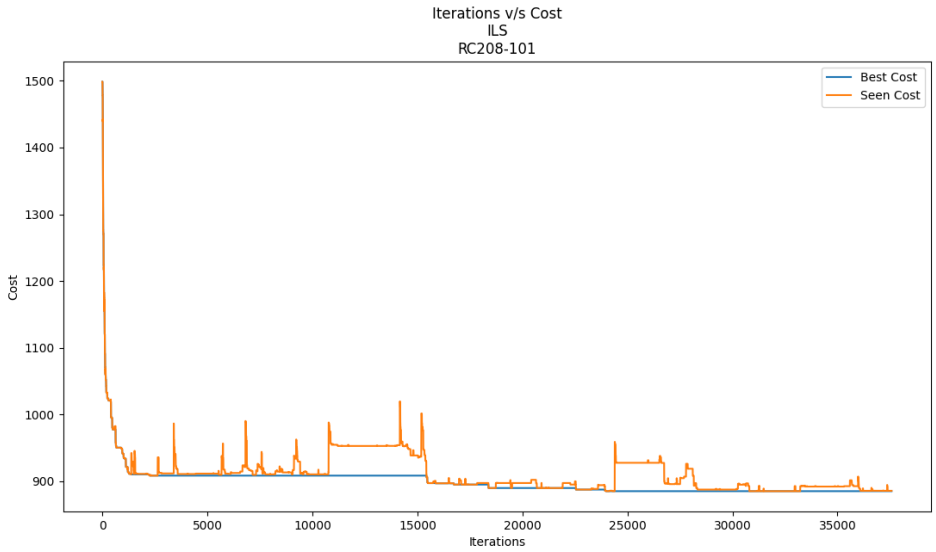

Figura 5.4 - Convergencia de costos de función objetivo en ILS para RC208-101

Fuente: Elaboración propia

En promedio, el porcentaje (%) de diferencia entre el óptimo y el valor obtenido por la metaheurística es de 6,1%, con una desviación estándar de 5%. Además, las soluciones de la metaheurística utilizan en promedio 1,9 vehículos más que el óptimo, con una desviación estándar de 1,2 vehículos.

Inst-ancia	ILS		Literatura			Comparación	
	Vehículos	Objetivo	Vehículos	Objetivo	Referencia	% dif	delta vehículo
c101	11	894,4	10	828,94	Rochat et. al (1995)	7,3%	1
c102	11	840,5	10	828,94	Rochat et. al (1995)	1,4%	1
c108	11	856,9	10	828,94	Rochat et. al (1995)	3,3%	1
c109	10	849,4	10	828,94	Rochat et. al (1995)	2,4%	0
c201	4	648,2	3	591,56	Rochat et. al (1995)	8,7%	1
c203	5	690,2	3	591,17	Rochat et. al (1995)	14,3%	2
c204	4	711,7	3	590,6	Rochat et. al (1995)	17,0%	1
c205	4	660,8	3	588,88	Rochat et. al (1995)	10,9%	1
c206	4	727,3	3	588,49	Rochat et. al (1995)	19,1%	1
c207	4	678,8	3	588,29	Rochat et. al (1995)	13,3%	1
c208	4	694	3	588,32	Rochat et. al (1995)	15,2%	1
r101	21	1739,7	19	1650,8	Homberger (2000)	5,1%	2
r102	19	1571	17	1486,12	Rochat et. al (1995)	5,4%	2
r104	13	1031	9	1007,31	Mester et. al (2005)	2,3%	4
r105	17	1458,2	14	1377,11	Rochat et. al (1995	5,6%	3
r106	15	1295,1	12	1252,03	Mester et. al (2005)	3,3%	3
r107	13	1129,4	10	1104,66	Shaw (1997)	2,2%	3
r108	11	984,5	9	960,88	Berger (2001)	2,4%	2
r109	14	1214,1	11	1194,73	Homberger et. al (1999)	1,6%	3
r110	13	1127,3	10	1118,84	Mester et. al (2005)	0,8%	3
r203	5	968,2	3	939,5	Woch et. al (2009)	3,0%	2
r206	4	1020,9	3	906,14	Schrimpf et. al (2000)	11,2%	1
r208	3	756,6	2	726,82	Mester et. al (2005)	3,9%	1
r209	5	910,3	3	909,16	Homberger (2000)	0,1%	2
r210	5	962,4	3	939,37	Mester et. al (2005)	2,4%	2
rc103	13	1374,9	11	1261,67	Shaw (1998)	8,2%	2
rc104	12	1199,2	10	1135,48	Cordeau et. al (2000)	5,3%	2
rc105	18	1672,3	13	1629,44	Berger (2001)	2,6%	5
rc106	16	1496,4	11	1424,73	Berger (2001)	4,8%	5
rc108	11	1172,7	10	1139,82	Taillard et. al (1997)	2,8%	1
rc204	5	885,7	3	798,46	Mester et. al (2005)	9,8%	2
rc206	4	1151,3	3	1146,32	Homberger (2000)	0,4%	1
rc208	4	885,9	3	828,14	Ibaraki et. al (2001)	6,5%	1

Tabla 5.2 - Resultados ILS en instancias de Solomon (1987)

Fuente: Elaboración propia en base a lo presentado enTransport Optimization Portal (2023)

5.3. Caso de estudio con datos geográficos

Se utilizó la instancia con datos geográficos para el condado de Los Ángeles, California, Estados Unidos.

La instancia, compuesta por un total de 59 puntos de recolección y 1 depósito fue evaluada tanto en el modelo de programación matemática, utilizando el *solver* de CPLEX, como con la metaheurística implementada en Python.

Se hizo el experimento de resolver el modelo de programación matemática, el cual, tras ejecutarse por 3600 segundos, no llegó a encontrar ninguna solución factible.

Por otro lado, se utilizó la metaheurística ILS implementada en Python, donde se permitió un tiempo de ejecución de 300 segundos. El problema fue resuelto logrando la obtención de una solución factible con un costo de 688 unidades, tal como se presenta en la Figura 5.5. Además, al analizar la Figura 5.6, podemos ver el progreso de la minimización del costo, donde la solución converge rápidamente al inicio, hasta quedarse en un sector de mínimo local.

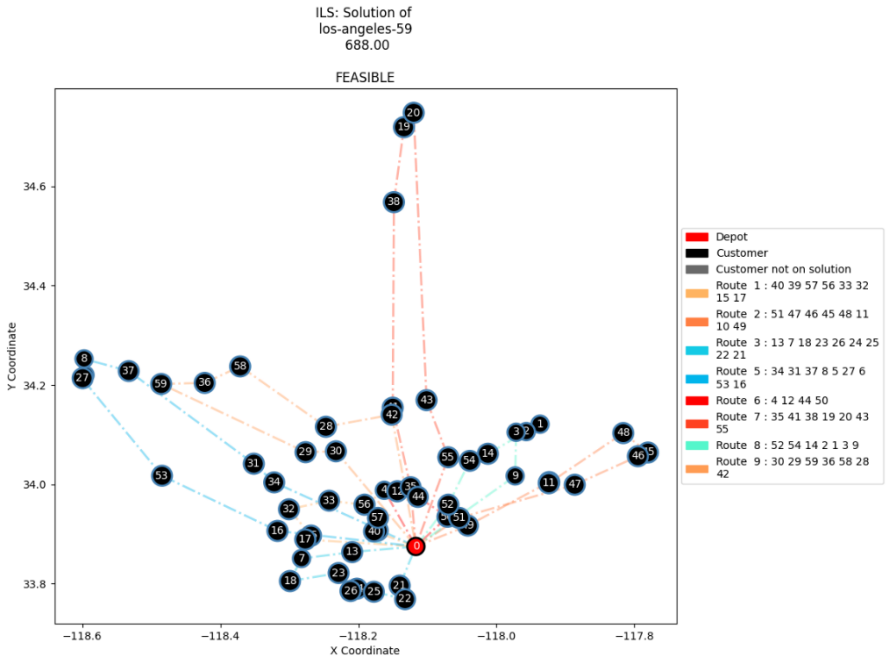

Figura 5.5 - Resultados ILS para caso de estudio (59 puntos limpios)

Fuente: Elaboración propia

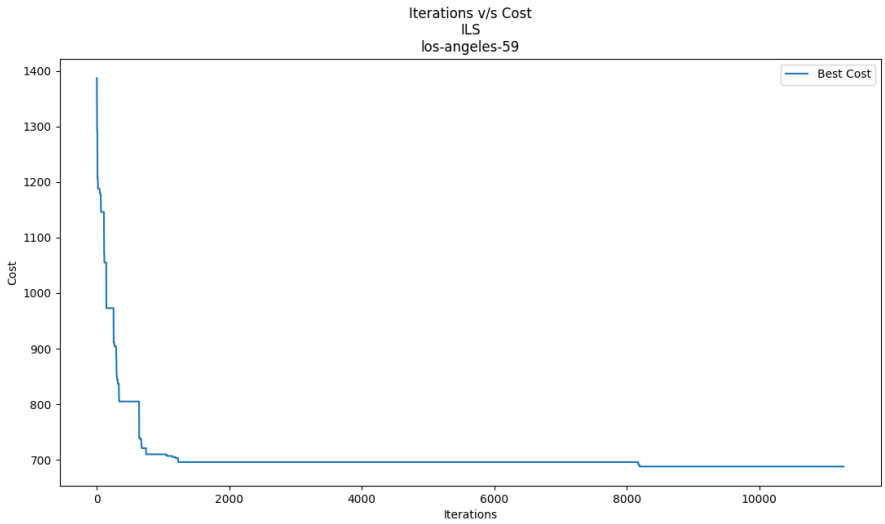

Figura 5.6 - Convergencia de costo de la función objetivo en ILS para caso de estudio

Fuente: Elaboración propia

Adicionalmente, se utilizó la información geográfica de las instancias para visualizar las rutas en un mapa. Se utilizó la herramienta QGIS para visualizar las rutas y los nodos sobre un mapa de Los Ángeles, que se presenta en la Figura 5.7.

Bajo los costos supuestos presentados en la metodología y la ecuación (15), se obtiene que el costo de la ruta es de $378.090, incluyendo la inversión en furgones y otros costos operativos. Sólo en Diésel, el costo de la ruta es de $90 USD.

Se destaca que el costo obtenido de la ruta logra un valor relativamente bajo en Diesel, debido al método implementado, que minimiza las distancias totales de la ruta. No obstante, la inversión en 9 camiones es elevada.

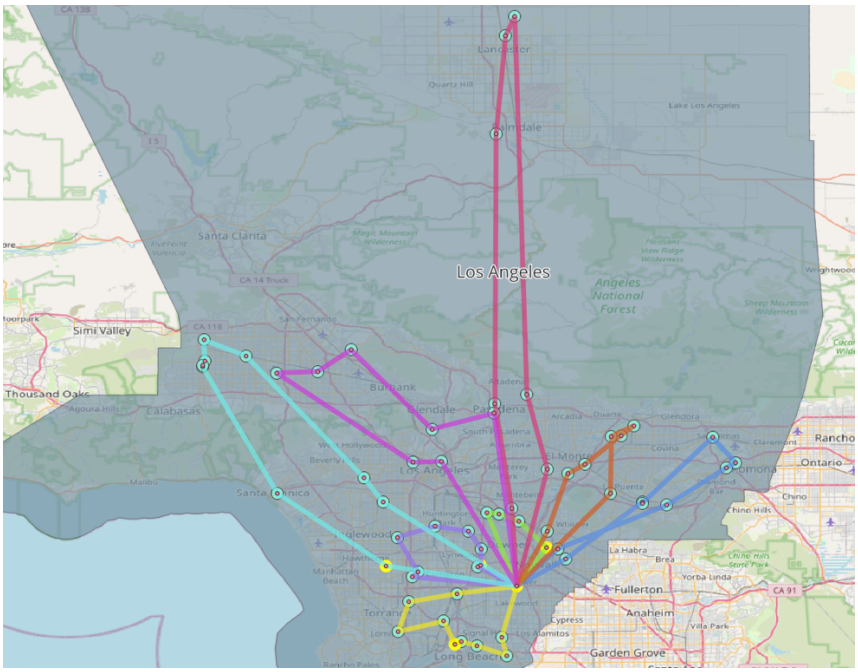

Figura 5.7 – Solución propuesta sobre mapa de Los Ángeles, Estados Unidos

Fuente: Elaboración propia sobre datos obtenidos por OpenStreetMap Contributors (2015) y Hurst & CalRecycle (2023)

Finalmente, en la Salida de Texto 3 se presentan los resultados obtenidos mediante el programa, indicando los nodos que visita cada ruta y el costo obtenido.

```
Instance name : los-angeles-59
Authors       : Nicolás Netz
Date          : 07.12.2023 14:50:08
Reference       : Ruteo de vehículos para la recolección de residuos
electrónicos. Un caso de estudio utilizando datos geográficos.
Solution

Route  1 : 40 39 57 56 33 32 15 17
Route  2 : 51 47 46 45 48 11 10 49
Route  3 : 13 7 18 23 26 24 25 22 21
Route  4 : 34 31 37 8 5 27 6 53 16
Route  5 : 4 12 44 50
Route  6 : 35 41 38 19 20 43 55
Route  7 : 52 54 14 2 1 3 9
Route  8 : 30 29 59 36 58 28 42

Cost: 688.00
Feasible: FEASIBLE
Unassigneds:
```

Salida de Texto 3 - Resultados caso de estudio

Fuente: Elaboración propia

6. Conclusiones

La creciente adopción de dispositivos tecnológicos ha llevado a un aumento significativo en la generación de residuos electrónicos, presentando desafíos ambientales y de salud pública en todo el mundo. Además, cambios legislativos originados en tratados internacionales como el Acuerdo de Basel, generan una tendencia que hace atractivo resolver este problema con un incentivo económico. Debido a lo anterior, resulta sustancial el uso de herramientas de la ingeniería para enfrentar eficientemente estas problemáticas.

En el presente estudio se utilizó un modelo de programación matemática como herramienta para abordar el problema de la recolección de desechos. El modelo utilizado corresponde a el *Capacitated Vehicle Routing Problem with Time Windows*. Debido a la complejidad computacional que este modelo supone, se planteó un algoritmo metaheurístico para resolver el problema, siguiendo una estructura de *Iterated Local Search*.

Se identificaron y caracterizaron los elementos claves del sistema logístico, cuya formulación identifica con claridad los diferentes componentes de la cadena de suministro inversa que lo representa.

Con el objetivo de validar los métodos, se llevaron a cabo experimentos computacionales sobre las instancias de Solomon (1987). Se resolvieron exitosamente las instancias de prueba con la metaheurística, encontrando eficientemente soluciones factibles cercanas a los valores óptimos presentados por otros autores. Con el modelo de programación matemática, no se logró obtener resultados para todas las instancias, puesto que la complejidad computacional del modelo era muy elevada para los tiempos y recursos asignados para resolverlo.

Una vez validado el de programación matemática y la eficacia de las metaheurísticas, se utilizaron datos geográficos reales de los centros de recolección de residuos electrónicos en el condado de Los Ángeles, Estados Unidos, facilitados por *CalRecycle*. En este experimento, se logró resolver un conjunto de rutas factible utilizando la metaheurística, sin embargo, el experimento no fue exitoso para la resolución con el modelo de programación matemática, pues tras el tiempo definido, no se encontró una solución.

Para futuros estudios, se recomienda modelar otras características, como penalizar la función objetivo por nodos no visitados, la posibilidad de recolectar residuos de manera parcial o la distribución de la carga de recolección entre múltiples vehículos. Asimismo, se recomienda considerar la existencia de más de un depósito, ya que, al analizar datos geográficos reales, se observó que en ciudades grandes suelen existir múltiples puntos de depósito.

En conjunto, el estudio entrega un marco integral para abordar la problemática de recolección de residuos, incluyendo los componentes de la cadena de suministro y la optimización de las rutas de recolección.

Glosario

AEE: Aparatos eléctricos y electrónicos.

EEE: Electric and Electronic Equipment.

RAEE: Residuos de aparatos eléctricos y electrónicos

E-Waste: Electronic waste

REP: Responsabilidad Extendida del Productor

OECD: Organisation for Economic Co-operation and Development

PIB: Producto Interno Bruto.

ILS: Iterated Local Search

SA: Simulated Annealing

GA: Genetic Algorithms

TS: Tabu Search

LNS: Large Neighborhood Search

RSC: Reverse Supply Chain

VRP: Vehicle Routing Problem

HVRP: Heterogeneous Vehicle Routing Problem

CVRPTW: Capacitated Vehicle Routing Problem with Time Windows

VRPSPD: Vehicle Routing Problem with Simultaneous Pickup and Delivery.

CCCVRPTWSD: Chance-constrained Capacited Vehicle Routing Problem with Time Windows and Stochastic Demand.

VRPTWMD: Vehicle Routing Problem with Time Windows and Multiple Deliverymen

Referencias

Raggio, R. (2021). *Un modelo y método metaheurístico, considerando elementos estocásticos para la recolección de residuos electrónicos en ciudades.* Concepción: DIRECCIÓN DE POSTGRADO UNIVERSIDAD DE CONCEPCIÓN.

Baldé, C. F. (2017). *The Global E-waste Monitor.* Bonn/Geneva/Vienna: The Global E-waste Monitor.

Forti, V. B. (2020). *The Global E-waste Monitor 2020,Quantities, flows, and the circular economy potential.* Bonn/Geneva/Rotterdam: United Nations University (UNU),.

Kaza, S. Y.-T. (2018). *What a waste 2.0: A global snapshot of solid waste managment to 2050.* World Bank.

Buhrkal, K. L. (2012). The Waste Collection Vehicle Routing Problem with Time Windows in a City Logistics Context. *Procedia - Social and Behavioral Sciences,*, 241-254.

Zhang, W. G. (2020). Multi-Depot Green Vehicle Routing Problem to Minimize Carbon Emissions. *Sustainability*, 12-15.

Yu, H. S. (2020). A stochastic network design problem for hazardous waste management. *Journal of Cleaner Production*, 277.

Tirkolaee, E. B. (2019). Developing an applied algorithm for multi-trip vehicle routing problem with time windows in urban waste collection: A case study. *Waste Management & Research*.

Expósito-Márquez, A. E.-I.-S.-P. (2019). Greedy randomized adaptive search procedure to design waste collection routes in La Palma. *Computers & Industrial Engineering*, 137.

Amphos 21 Consulting Chile Ltda. (2015). EVALUACIÓN DE LOS IMPACTOS AMBIENTALES, SOCIALES Y ECONÓMICOS DE LA IMPLEMENTACIÓN DE LA RESPONSABILIDAD EXTENDIDA DEL PRODUCTOR EN CHILE APLICADA A LOS APARATOS ELÉCTRICOS.

Iniciativa StEP. (June de 2014). *Iniciativa Step, "Solving the E-Waste Problem (Step) White Paper: One Global Definition of E-waste" (Libro blanco sobre la resolución del problema de los residuos electrónicos (Step): una definición mundial de los residuos electrónicos).* Obtenido de United Nations University: https://collections.unu.edu/eserv/UNU:6120/step_one_global_definition_amended.pdf.

European Union. (24 de June de 2012). *DIRECTIVE 2012/19/EU OF THE EUROPEAN PARLIAMENT AND OF THE COUNCIL.* Obtenido de https://eur-lex.europa.eu/LexUriServ/LexUriServ.do?uri=OJ:L:2012:197:0038:0071:en:PDF

Oguchi, M. M. (2011). A preliminary categorization of end-of-life electrical and electronic equipment as secondary metal resources. *Elsevier BV*, https://doi.org/10.1016/j.wasman.2011.05.009.

M. Wagner, C. B. (2022). *Monitoreo regional de los residuos electrónicos para América Latina: resultados de los trece países participantes en el proyecto UNIDO-GEF 5554.* Bonn, Alemania: United Nations University.

OECD. (2023). *Waste from electrical and electronic equipment(WEEE-e-waste).* Obtenido de OECD Stat: https://stats.oecd.org/Index.aspx?DataSetCode=EWASTE

CCL North Ltd. (2023). *CCL North Ltd.* Obtenido de Homepage: https://www.cclnorth.com

Montoya T, e. a. (2015). A literature review on the vehicle routing problem with multiple depots. *Computers & Industrial Engineering*.

G. Gutiérrez-Jarpa, e. a. (2010). A branch-and-price algorithm for vehicle routing problem with deliveries, selective pickups and time windows. *European Journal on Operations Research*.

Toth, P., & Vigo, D. (2014). The Vehicle Routing Problem with Time Windows. En P. Toth, & D. Vigo, *Vehicle Routing Problems, Methods and Applications.* Bolognia, Italy: Society for Industrial and Applied Mathematics and the Mathematical Optimization Society.

Lourenço, H., Martin, O., & Stützle, T. (2010). Iterated local search: Framework and applications. En G. M., & J.-Y. Potvin, *Handbook of Methaeuristics* (págs. 363-397). New York: Springer.

Cote, J.-F. (2012). VRP. *cotejean@iro.umontreal.ca*.

Bräysy, O., & Gendreau, M. (2005). Vehicle routing problem with time windows, part I: Route construction and local search algorithms. *Transportation Science, 39*, 104, 119.

Solomon, M. M. (1987). Algorithms for the vehicle routing and scheduling problem with time window constraints. *Operations Research, 35*, 254-265.

OpenStreetMap Contribuitors. (2015). Mapa de Los Angeles California. https://planet.openstreetmap.org/.

Savelsbergh, M. (1985). Local search in routing problems with time windows. *Annals of Operations Research*, 285-305.

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability A guide to the theory of NP-Completeness.* Bell Telephone Laboratories.

Linh, D., Amer, Y., Lee, S.-H., Phuc, P., & Dat, L. (2019). E-Waste Reverse Supply Chain: A Review and Future Perspectives. *Applied Sciences*.

Ballou, R. H. (2004). *Logística. Admiministración de la cadena de suministro. Quinta edición* . Mexico: Pearson.

University of Adelaide. (5 de December de 2023). *Mind Mapping.* Obtenido de Mind Mapping: https://www.adelaide.edu.au/writingcentre/sites/default/files/docs/learningguide-mindmapping.pdf

Bowersox, D. J., Closs, D. J., & Cooper, M. B. (2007). *Administración y Logística en la cadena de suministros.* Mexico: McGraw-Hill Interamericana.

Sar, K., & Ghadimi, P. (2023). A systematic literature review of the vehicle routing problem in reverse logistics operations. *Computers & Industrial Engineering*.

Dantzig, G., & Rmaser, J. (1959). The Truck Dispatching Problem. *Management Science*.

Han, H., & Ponce-Cueto, E. (2015). Waste Collection Vehicle Routing Problem: A Literature Review. *Traffic & Transportation*.

Golden, B., Assad, A., & Wasil, E. (2014). Routing Vehicles in the Real World: Applications in the Solid Waste, Beverage, Food, Dairy, and Newspaper Industries. En P. Toth, & D. Vigo, *The Vehicle Routing Problem* (págs. 245-286). Milan: Society for Industrial and Applied MAthematics.

Kassem, S., & Chen, M. (2013). Solving reverse logistics vehicle routing problems with time windows. *The international Journal of Advanced Manufacturing Technology*, 57-68.

Kim, B., Kim, S., & Sahoo, S. (2006). Waste Collection Vehicle Routing Problem with Time Windows. *Computers & Operations Research*, 3624-3642.

Mohammadi, M., Rahmanifar, G., Hajiaghaei-Keshteli, M., Fusco, G., & Colombaroni, C. (2023). Industry 4.0 in waste management: An integrated IoT-based approach for facility location and green vehicle routing. *Journal of Industrial Information Integration*.

Rahmanifar, G., Mohammadi, M., Sherafat, A., Hajiaghaei-Keshteli, M., Fusco, G., & Colombarini, C. (2023). Heuristic approaches to address vehicle routing problem in the Iot-based waste management system. *Expert Systems With Applications*.

Kapadia, N., & Metha, R. (2023). Dynamic route optimization for IoT based intelligent waste collection vehicle routing system. *Intelligent Decision Technologies*.

Wan, H., Ma, J., Yu, Q., Sun, G., He, H., & Li, H. (2023). Modeling and Optimization of Multi-Model Waste Vehicle Routing Problem Based on the Time Window. *Journal of Database Management*.

Hashemi-Amiri, O., Mohammadi, M., Rahmanifar, G., Hajiaghaei-Keshteli, M., Fusco, G., & Colombaroni, C. (2023). An allocation-routing optimization model for integrated solid waste management. *Expert Systems With Applications*.

Hurst, A., & CalRecycle. (15 de Diciembre de 2023). *CalRecycle.* Obtenido de CalRecycle: https://calrecycle.ca.gov

Desaulniers, G., Madsen, O. B., & Ropke, S. (2014). The Vehicle Routing Problem with Time Windows. En P. Toth, & D. Vigo, *Vehicle Routing Problems, Methods, and Applications* (págs. 130-159). Milan: Society for Industrial and Applied Mathematics and the Mathematical Optimization Society.

Hashimoto, H., Yagiura, M., & Ibaraki, T. (2008). An iterated local search algo- rithm for the time-dependent vehicle routing problem with time windows. *Discrete Optimization*, 434-456.

Öztaş, T., & Tuş, A. (2022). A hybrid metaheuristic algorithm based on iterated local search for vehicle routing problem with simultaneous pickup and delivery. *Expert Systems with Applications*.

Máximo, V., Cordeau, J.-F., & Nascimento, M. C. (2022). An adaptive iterated local search heuristic for the Heterogeneous Fleet Vehicle Routing Problem. *Computers & Operations Research*.

Transport Optimization Portal. (28 de October de 2023). *Transport Optimzation Portal - VRPTW*. Obtenido de SINTEF: https://www.sintef.no/projectweb/top/vrptw/100-customers/

Rochat, Y., & Taillard, E. D. (1995). Probabilistic Diversification and Intensification in Local Search for Vehicle Routing. *Journal of Heuristics 1*, 147-167.

Homberger, J. (2000). *Verteilt-parallele Metaheuristiken zur Tourenplanung.* Wiesbaden: Gaber.

Mester, D., Bräysy, O., & Dullaert, W. (2005). *A Multi-parametric Evolution Strategies Algorithm for Vehicle Routing Problems.* Israel: Institute of Evolution, University of Haifa.

Shaw. (1997). *A New Local Search Algorithm Providing High Quality Solutions to Vehicle Routing Problems.* University of Strathclyde, Glasgow, Scotland.

Berger, J., Barkaoui, M., & Bräysy, O. (2001). *A Parallel Hybrid Genetic Algorithm for the Vehicle Routing Problem with Time Windows.* Defense Research Establishment Valcartier, Canada.

Ibaraki, T., Kubo, M., Masuda, T., Uno, T., & Yagiura, M. (2001). *Effective Local Search Algorithms for the Vehicle Routing Problem with General Time Windows.* Department of Applied Mathematics and Physics, Kyoto University, Japan.

Taillard, E., Badeau, P., Gendreau, M., Geurtin, F., & Potvin, J. (1997). A Tabu Search Heuristic for the Vehicle Routing Problem with Time Windows. *Transportation Science*, 170-186.

Homberger, J., & Gehring, H. (1999). Two Evolutionary Metaheuristics for the Vehicle Routing Problem with Time Windows. *INFOR*, 297-318.

Woch, M., & Lebkowski, P. (2009). Sequential Simulated Annealing for the Vehicle Routing Problem with Time Windows. *Decision Making in Manufacturing and Services*, 87-100.

Schrimpf, G., Schneider, J., Stamm-Wilbrandt, H., & Dueck, G. (2000). Record Breaking Optimization Results Using the Ruin and Recreate Principle. *Journal of Computational Physics*, 139-171.

Shaw, P. (1998). Using Constraint Programming and Local Search Methods to Solve Vehicle Routing Problems. *Principles and Practice of Constraint Programming*, 417-431.

Cordeau, J., Laporte, G., & Mercier, A. (2000). A Unified Tabu Search Heuristic for Vehicle Routing Problems with Time Windows. *Centre for Research on Transportation*.

Chamberlain, B. (2019). *U. S. Census Bureau GIS FAQ.* Obtenido de U.S. Census: http://www.census.gov/cgi-bin/geo/gisfaq?Q5.1

Sinnott, R. (1984). Virtues of the Haversine. *Sky and Telescope*, 159.

Gehring, & Homberger. (1999). A parallel hybrid evolutionary metaheuristic for the vehicle routing problem with time windows.

Mehlhom, K., & Sanders, P. (2008). *Algorithms and Data Structures. The Basic Toolbox.* Springer.

A. Anexo

Participant Name	Role	County	Latitude	Longitude
FRG Waste Resources, Inc.	Collector	Napa	38,19484	-122,2583
Napa Recycling & Waste Services, LLC	Collector	Napa	38,20947	-122,2645
A D S Gold, Inc.	Collector	Orange	33,85718	-117,8362
CVT Recycling	Collector	Orange	33,84903	-117,8698
Orange County Conservation Corps	Collector	Orange	33,86296	-117,9074
Salvation Army Adult Rehabilitation Center (Anaheim)	Collector	Orange	33,81541	-117,897
Silicon Salvage, Inc.	Collector/Recycler	Orange	33,8537	-117,9848
Recology Auburn Placer	Collector	Placer	38,95571	-121,0939
Veolia ES Technical Solutions, LLC	Collector	Los Angeles	34,12103	-117,9367

Tabla A.1 - Muestra de datos geográficos facilitados por CalRecycle

Fuente: Andrew Hurst (2023), CalRecycle (Hurst & CalRecycle, 2023)

B. Anexo

En la Figura B.1 se presenta una solución al problema y la respectiva representación en arreglos. Los arreglos *next, prev, assign_to* y *unassigneds* tienen $n + 2v$ elementos (n: Número de puntos de recolección y v: Número de vehículos). Los primeros n representan los puntos de recolección y los últimos $2v$ representan las salidas y llegadas de cada vehículo ordenados, tal que los elementos $n + k$ son la salida del vehículo y los $n + k + 1$ son las llegadas. Notar que tanto la salida como la llegada representan el depósito 0.

En la Figura B.1 $n = 7$ y $v = 2$. Para construir la ruta del vehículo $k = 1$ usando *next*, se ve el elemento en la posición $n + k = 7 + 1 = 8$. En la posición 8, se tiene el valor 5, por lo que a continuación del depósito, le sigue el punto de recolección 5 ($Ruta\ k_1 = 8{,}5$). Luego, se revisa el elemento en la posición 5, que es 6 ($Ruta\ k_1 = 8{,}5{,}6$). Luego, se revisa el elemento en la posición 6, que es 7 ($Ruta\ k_1 = 8{,}5{,}6{,}7$). Luego, se revisa el elemento en la posición 7, que es 9 ($Ruta\ k_1 = 8{,}5{,}6{,}7{,}9$). Luego, al revisar el elemento en la posición 9 y notar que está vacío, se tiene que ha finalizado la ruta.

Finalmente, la ruta del vehículo 1 es 0,5,6,7,0, pues 8 y 9 son el depósito y se reemplazan por él.

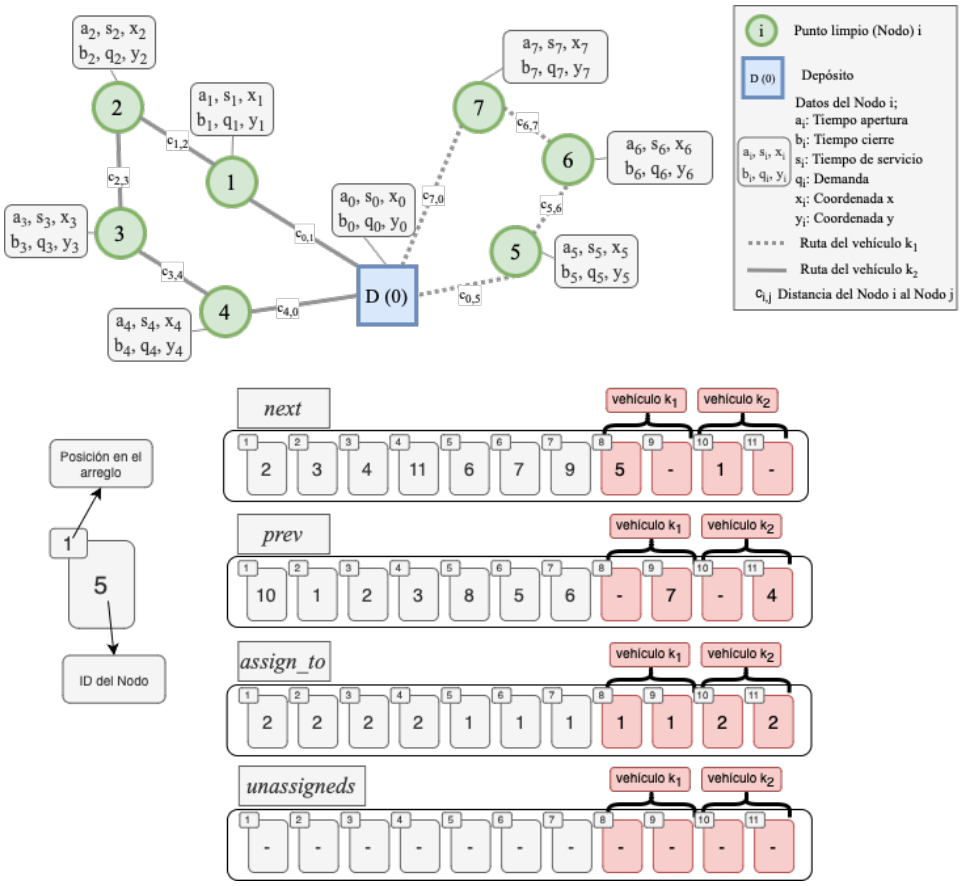

Figura B.1 - Ejemplo de representación de la solución con arreglos

Fuente: Elaboración propia