RUTEO DE VEHÍCULOS VRPTWPD PARA INFORMACIÓN DE VALOR MEDIANTE EL USO DE ALGORITMOS GENÉTICOS

AUTOR

DAVID YOAN GÓMEZ VELOZA

POLITÉCNICO GRANCOLOMBIANO

ESPECIALIZACIÓN EN LOGÍSTICA Y CADENA DE ABASTECIMIENTO

BOGOTÁ 2022

RUTEO DE VEHÍCULOS VRPTWPD PARA INFORMACIÓN DE VALOR MEDIANTE EL USO DE ALGORITMOS GENÉTICOS

PRESENTADO POR:

DAVID YOAN GÓMEZ VELOZA

TRABAJO FINAL DE ESPECIALIZACIÓN

DIRECTOR: JAIRO ENRIQUE PARRA HERRERA

POLITÉCNICO GRANCOLOMBIANO

ESPECIALIZACIÓN EN LOGÍSTICA Y CADENA DE ABASTECIMIENTO

BOGOTÁ 2022

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

INTRODUCCIÓN

La distribución de última milla se considera dentro de la cadena de abastecimiento como un proceso ineficiente y costoso, según menciona Mckinsey Company en su estudio Parcel Delivery The Future Of Last Mile, puesto que, representa hasta el 50% del costo final del bien. Por ser un proceso costoso es necesario realizar una asignación eficiente de recursos en el sistema de distribución, sin embargo, en la práctica existen diferentes condiciones que no permiten efectuar la planeación de forma correcta. Para dar solución a esta problemática surgen los modelos matemáticos para optimización de ruteo de vehículos.

En empresas de transporte con sistemas de distribución difíciles de resolver y condiciones especiales como demandas estocásticas, ventanas de tiempo para atención, flota y capacidad limitada, se debe considerar una herramienta tecnológica que facilite la asignación de rutas en función de la demanda. Puesto que, al realizar un ruteo de forma manual no se contemplan el 100% de las restricciones, generando un sobre costo para el sistema e incumplimiento en la atención de solicitudes de acuerdo con las necesidades de cada cliente.

El problema de estudio corresponde al caso de una empresa de seguridad de información que presta servicio de recolección, custodia y entrega de documentos valorados en cajas con un peso promedio de 13 Kilos, para diferentes entidades ubicadas en la ciudad de Bogotá. Los clientes de la compañía están dispuestos a pagar un valor adicional en el transporte para atender su solicitud dentro de ventanas de tiempo definidas conforme a su urgencia, el incumplimiento significa el no pago o pago parcial por parte del cliente.

El desafío logístico es optimizar los recursos en el sistema de distribución cumpliendo siempre las necesidades del cliente y respetando las restricciones del sistema, no obstante, como muchas empresas la asignación se realiza de forma manual acorde a la experiencia de un programador que, por medio de zonas fijas ya demarcadas por calles y carreras, realiza la planeación de forma empírica, lo que conlleva a fallas en su distribución de rutas e incumplimientos.

El programador realiza una asignación de clientes a cada ruta sin contemplar restricciones como volumen, ventanas de atención, cantidad de paradas, distancias y secuencia lógica de clientes a visitar, debido a la dificultad para relacionar estas variables. Por este motivo es decisión del conductor establecer el orden de la ruta de acuerdo con su experiencia e instrucciones de horario en las ordenes de trabajo, la practica indica que, si bien los conductores toman buenas decisiones, muchas veces incurren en afectación de los clientes.

La empresa, aunque tiene recursos suficientes para cumplir con la demanda, no logra satisfacer las necesidades de los clientes y cumplir los tiempos que ellos exigen para atender sus solicitudes, debido a una mala asignación de recursos por parte del programador de rutas. Por este motivo, se ve necesario aplicar una técnica de optimización, que permita solucionar el problema de ruteo, facilitando al programador y conductores tomar mejores decisiones para lograr cumplir todos los requerimientos.

Este documento final de especialización presenta el proceso de construcción de un Algoritmo genético para la asignación de rutas considerando, ventanas de tiempo, tiempo de servicios, distancias, flota homogénea en un modelo operativo de recolecciones, que permitirá realizar a la empresa una mejor asignación de recursos y lograr satisfacer las necesidades de tiempos de cada cliente.

Primero se realizó un diagnóstico de la operación actual y se identificó la problemática y causa raíz de la misma, posterior se hizo un análisis en la literatura de las principales técnicas y estrategias para dar solución y se seleccionó la metaheurística algoritmos genéticos, por último, se diseñó el algoritmo con lenguaje de programación en Python el cual permitió encontrar una solución para el problema, tomando como rutas iniciales las propuestas por el profesor Günes Erdogan con su herramienta VRP en Excel.

Después de aplicar la metodología cuantitativa propuesta se encontró que, si es posible solucionar el problema de ruteo de vehículos VRPTW por medio de un algoritmo genético, el cual determina una disminución del 54% de los kilómetros recorridos por el conjunto de rutas respecto a la herramienta de optimización "*vrp_spreadsheet_solver_v3.72*", además se creó un código en Python que permite realizar el ruteo de vehículos contemplando todas las restricciones y variables del sistema.

1. CAPÍTULO 1: CARACTERÍSTICAS Y PLANTEAMIENTO DEL PROBLEMA

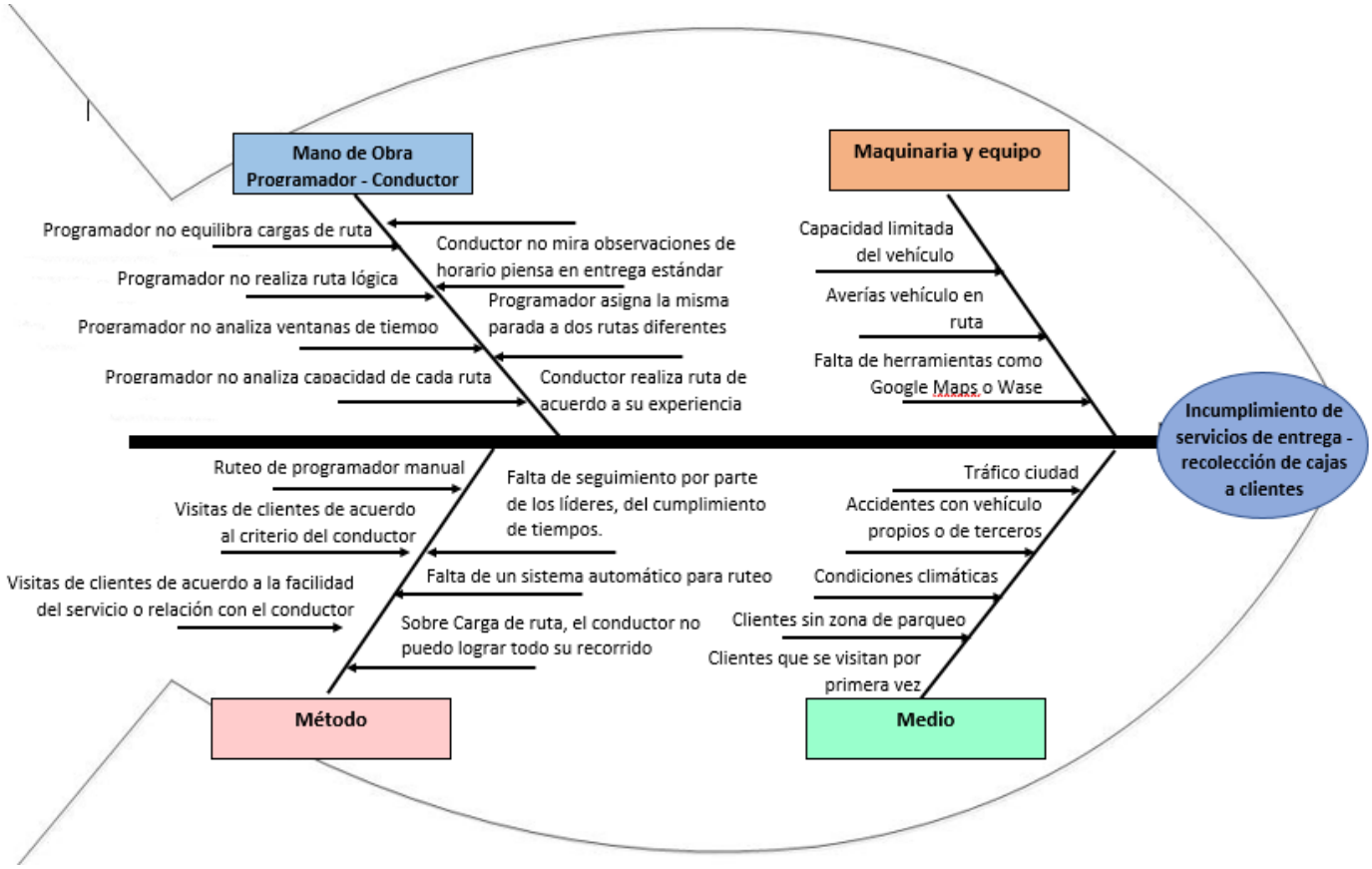
En esta sección se presenta una una descripción del contexto general de la empresa, por medio de datos y diagramas de causa que permiten realizar la caracterización del sistema en estudio e identificar la problemática y causa raíz de esta, además se plantea la formulación y delimitación del problema.

En la ***Figura 1*** se utiliza la herramienta de análisis conocida como diagrama de causa y efecto para mostrar los principales factores de incumplimiento de servicio de transporte, entre los cuales se pueden observar algunos externos, que no pueden ser controlados como es el medio y sus variables como tráfico, accidentes, condiciones en la vía, sin embargo, en su gran mayoría las causas son ocasionadas por factores internos, como el método y mano de obra, con causas como, ruteo manual, sobrecarga de rutas, decisiones erradas del programador y conductor.

Dentro de las causas de incumplimiento, el método y mano de obra pueden ser controladas, puesto que, son variables que dependen de la forma en la cual se desarrollan los procesos dentro de la operación, el método de enrutamiento no determina el orden lógico en el cual se debe desarrollar la ruta para no afectar al cliente y muchas veces el conductor no toman las mejores decisiones ocasionando incumplimiento.

Figura 1

Causas de incumplimiento de servicios en ruta



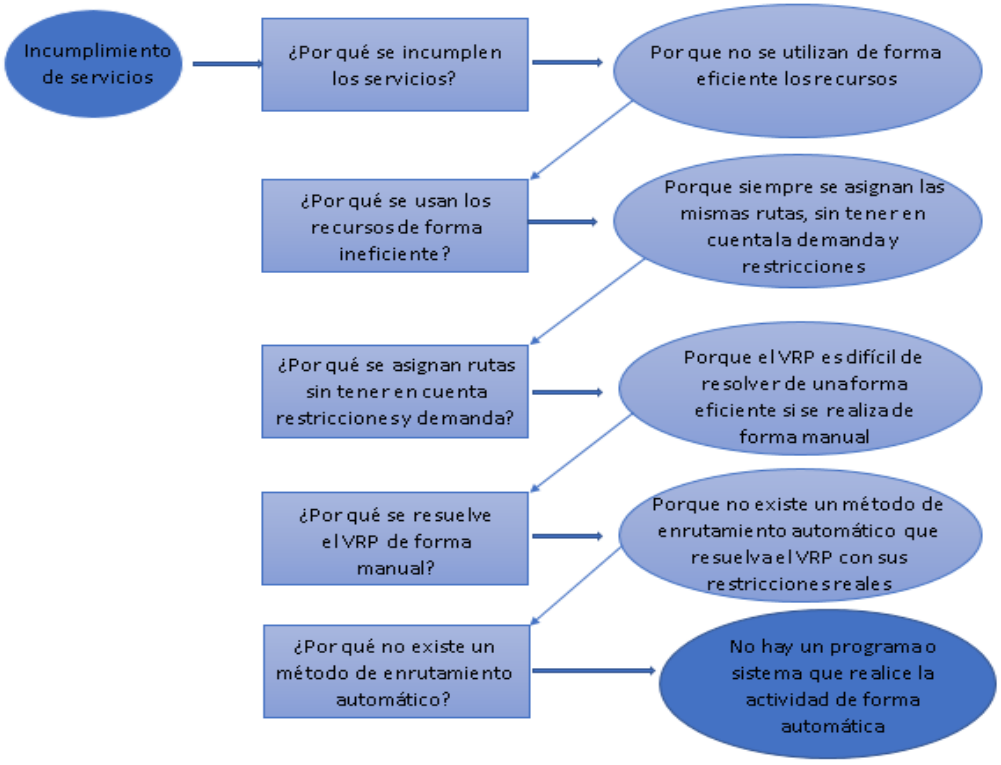
Nota: Fuente El autor.

Dentro del análisis de la ***Figura 1***, se resalta que la mayoría de las causas de incumplimiento están asociadas a factores internos, como el método en el desarrollo del proceso de programación de rutas y mano de obra, con sub-causas como; ruteo de programador manual, visita a clientes de acuerdo a la experiencia, criterio de cada ruta y relación del conductor con el usuario o facilidad para ejecución del servicio y falta de seguimiento al cumplimiento de las ventanas de tiempo por parte de los líderes de transportes.

Como herramienta complementaria para el diagrama de causa y efecto se utiliza la técnica sistemática conocida como los 5 ¿Por qué?, donde por medio de la misma pregunta en diferentes niveles se puede llegar a identificar la causa más probable que ocasiona o es responsable del efecto, para este caso se analiza nuevamente el problema de incumplimientos de servicio en 5 niveles para llegar a su causa raíz.

Figura 2

Causas de incumplimiento, 5 ¿Por qué?



Nota: Fuente El autor.

De acuerdo con la información proporcionada en la ***Figura 1***, y ***Figura 2*** se resalta que el proceso para ruteo de vehículos en el cual no se contemplan las restricciones del sistema y a su vez no se optimizan los recursos en función de la demanda, es el causa de incumplimiento de la compañía hacia los clientes, por este motivo se ve la necesidad de establecer un método para la asignación correcta de rutas y clientes a cada vehículo.

Tabla 1

Porcentaje de cumplimiento de servicios

Mes	Total, órdenes de trabajo	Total, órdenes de trabajo atendidas a tiempo	% De cumplimiento órdenes (% On Time)
Enero	3478	3280	94,3%
Febrero	3093	2858	92,4%
Marzo	3148	2817	89,5%
Abril	3348	3134	93,6%
Mayo	3064	2788	91,0%
Junio	3235	2941	90,9%
Julio	3324	2982	89,7%

Nota: Fuente El autor.

La ***Tabla 1*** muestra en un periodo de 7 meses el total de órdenes de trabajo realizadas atendidas a tiempo contra el total de órdenes atendidas, considerando que una orden realizada a tiempo es aquella que fue entrega por la ruta dentro de los horarios establecidos por los clientes. Se tiene un porcentaje de cumplimiento promedio de 91,6% y 270 servicios realizados tarde cada mes, lo que representa afectación en la promesa de valor de la compañía, no siendo los guardianes de confianza de cada cliente afectado.

Tabla 2

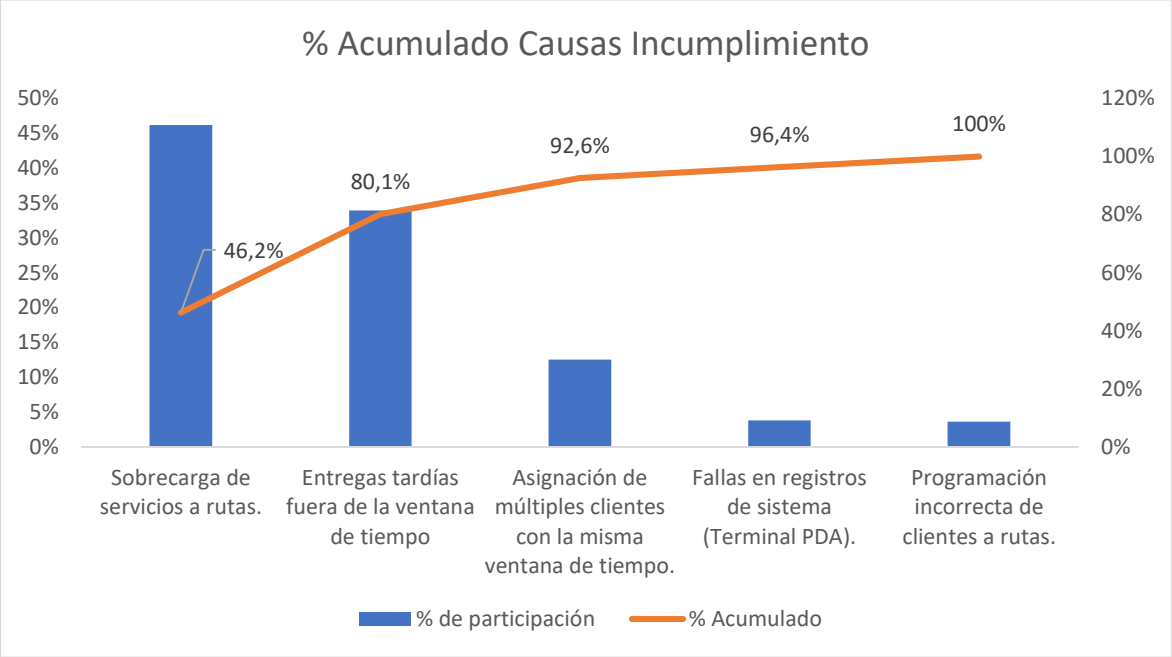
Causas de incumplimiento de servicios a clientes

Causas de incumplimiento en ruta	% de participación
Sobrecarga de servicios a rutas, no se logran completar.	46,20%
Entregas tardías por parte del conductor fuera de la ventana de tiempo	33,90%
Asignación de múltiples clientes con la misma ventana de tiempo.	12,50%
Fallas en registros de sistema (Terminal PDA).	3,80%
Programación incorrecta de clientes a rutas.	3,60%

Nota: Fuente El autor, tomado de reuniones de seguimiento de indicadores.

La ***Tabla 2*** muestra que el 46,20% de conductores justifican el incumplimiento de los servicios asignados se deben a sobrecarga en la ruta y no logran completar todos sus recorridos, el 33,9% realizan los servicios fuera de la ventana de tiempo, estos datos son obtenidos de las reuniones del comité primario de transporte todas las semanas y donde se analizan resultados de diferentes indicadores del área.

Figura 3
Diagrama de Pareto, incumplimiento de servicios en ruta



Nota: Fuente El autor.

En la ***Tabla 2*** y ***Figura 3*** se puede observar cuales son las principales causas del incumplimiento de servicios, donde por sobrecarga no logran cumplir el 100% de las entregas asignadas, en algunas oportunidades se realizan fuera de las ventanas de tiempo y en otras no se completan el día que el cliente los solicito, sumando un 80% del total de las causas.

De acuerdo con los elementos mencionados en el diagrama de causa y efecto, técnica de los 5 ¿Por qué? y tabla de Pareto, es necesario definir una técnica o método para realizar el ruteo de vehículos teniendo cuenta las restricciones del sistema y poder atacar directamente las principales causas de incumplimientos detectadas. Cumpliendo con la promesa de valor de la compañía hacia todos los clientes afectados.

1.1 Formulación del Problema

¿Cómo resolver el problema de asignación de vehículos para la operación de recolección, teniendo en cuentas restricciones de capacidad y ventanas de tiempo de acuerdo con las necesidades de cada cliente?

1.2 Delimitación del Problema

- El caso en estudio se centrará en la línea de negocio de vehículos dedicados a las recolecciones de información de valor, puesto que, representan el 70% de la operación diaria y un 85% del volumen de unidades totales.
- Se realizará el estudio y aplicación del VRPTWPD en la ciudad de Bogotá. D.C, donde el 95% de los clientes se encuentra ubicados geográficamente en un polígono demarcado por calles y carreras, desde la calle 1 hasta la calle 170 y entre la carrera 7 hasta la autopista norte.
- Dentro de las restricciones del problema se tendrán en cuenta, capacidad limitada de flota homogénea, ventanas de tiempo de atención duras, cada cliente será visitado una sola vez por un único vehículo y todas las rutas se despachan y retornar a un único depósito sin posibilidad de regresar en medio para realizar cargues o descargues.

- De acuerdo al último informe de la secretaria de movilidad y gracias al nuevo pico y placa extendido de Bogotá la velocidad media se tendrá en cuenta en 37.5km/h.

1.3 Objetivos

1.3.1 Objetivo General

Proponer una solución para el ruteo de vehículos VRPTWDP para el transporte de información de valor mediante el uso de algoritmos genéticos.

1.3.2 Objetivos Específicos

1. Diagnosticar el modelo de distribución del sistema tomado en el caso en estudio y sus elementos de interés.
2. Analizar los modelos matemáticos, aproximaciones heurísticas y metaheurísticas que se adapten a las características del problema VRPTWPD.
3. Establecer soluciones para el ruteo de vehículos con restricciones de ventanas de tiempo a partir del uso de algoritmos genéticos.

2. CAPÍTULO 2: MARCO DE REFERENCIA

El problema de ruteo de vehículos (VRP, por sus siglas en inglés Vehicle Routing Problem), en adelante VRP es uno de los problemas habituales que conforman la investigación de operaciones y la gestión logística, se realiza el primer registro en la literatura por Dantzig, Flukerson y Jhonson en 1954 con el modelo planteado (TSP, Traveling Salesman Problem), llamado así debido a la naturaleza de este, donde un viajante debe visitar un conjunto de ciudades y regresar al punto de partida. La solución de esta problemática consiste en reducir la distancia recorrida visitando una vez cada ciudad y se vuelve muy compleja su solución cuando el grupo de nodos aumenta, puesto que, el modelo necesita grandes recursos computacionales en la búsqueda exhaustiva de su solución. (Eksioglu et al., 2009).

A partir de la publicación de Dantzig, Flukerson y Jhonson en 1954, se empiezan a publicar diferentes artículos que se enfocan en la resolución de problemas de ruteo de vehículos, como; “Transportation Network Desing” (O´Connor & De Wald, 1970), “Fleet Routing” (Levin, 1971), “Vehicle Routing” (Golden, Magnati Y Nguyan, 1972), en el año 1978 se introducen diferentes variables y restricciones que hacen que los algoritmos para su solución se vuelven más complejos, por ejemplo Solomon en 1983 introduce los modelos VRP con ventanas de tiempo, con un conjunto de instancias conocidas ahora como las instancias de Solomon, en el año 1990 gracias a los avances tecnológicos y disponibilidad de nuevos microprocesadores se presentan estudios con variantes del VRP debido a la practicidad en la modelación de los mismos, mostrando diferentes heurísticas y metaheurísticas aplicadas.(Eksioglu et al., 2009)

El VRP surge de forma natural en las compañías de transportes y es uno de los problemas más complejos en la programación lineal entera, se clasifica como NP Hard, puesto que, por su gran complejidad no se pueden resolver en tiempo polinomial en función al tamaño de las

entradas, además necesita una gran cantidad de recursos computacionales y el tiempo de su solución aumenta exponencialmente respecto al tamaño del problema. Este tipo de problema necesitan soluciones rápidas y aceptables, para tomar decisiones suficientemente buenas, es por esto por lo que los métodos exactos no son usados comúnmente en su solución y se aplican diferentes técnicas que mejoren los tiempos de procesamiento del modelo, aunque no garanticen optimalidad de este. (Lüer et al., 2009)

Un ejemplo de VRP es la problemática planteada en el TSP (Traveling Salesman Problem) clasificado como NP, donde por medio de permutaciones un viajante debe visitar un número de ciudades minimizando la distancia total recorrida, este problema es sumamente difícil de resolver de acuerdo a la cantidad de nodos a visitar, el mejor algoritmo conocido del problema es $\mathbf{On^2 2^n}$ el cual su tamaño de espacio de solución crece conforme a la ecuación $\frac{\mathbf{(n-1)!}}{\mathbf{2}}$, se presentan algunos ejemplos para entendimiento de la ecuación.

- Para n = 10, hay 181.000 soluciones posibles.
- Para n = 20, hay $1x10^{16}$ soluciones posibles.
- Para n = 50, hay $1x10^{59}$ soluciones posibles.

Para tener idea de estas magnitudes, basta decir que solo hay $\mathbf{1x10^{21}}$ litros de agua en el planeta. (Coello, 2006)

Técnicas de solución para este tipo de problemas sin tener que usar métodos de búsqueda exhaustiva son conocidas como heurísticas, algoritmos diseñados a la medida de cada problema que permiten encontrar una solución aceptable en tiempos muy reducidos utilizando menores recursos computacionales, sus aplicaciones varían de acuerdo con el modelo planteado, puesto que, cada autor acota su problema y lo soluciona conforme a sus supuestos.

Posterior se dan a conocer nuevos algoritmos generalizados que eran capaz de resolver estos problemas complejos inspirados en el comportamiento de la naturaleza, llamados Metaheurísticos, estas técnicas tienen un gran impacto en la investigación de operaciones, debido a que gracias a la combinación de diferentes algoritmos se puede llegar a una solución muy cercana a la que proporciona un método exacto, dentro de las metaheurísticas más conocidas se encuentran, recocido simulado, búsqueda tabú, algoritmos genéticos, búsqueda de vecindarios variables y redes neuronales (Lüer et al., 2009)

2.1 Problema de Ruteo de Vehículos

De acuerdo con la definición de Toth & Vigo (1998) el VRP clásico se define en su forma teórica de la siguiente manera:

Sea g = (v, a) un grafo completo donde v = {0, 1, ..., n} es el conjunto de vértices y a es el conjunto de arcos, los vértices j = {1, ..., n} corresponden a un conjunto de clientes, cada uno con una demanda conocida no negativa $\mathbf{D_j}$, mientras que el vértice 0 corresponde a un depósito. existe un costo $\mathbf{C_{ij}}$, asociado a cada arco (ij) ea y representa el costo de viajar del vértice i al vértice j. Si los valores de costo satisfacen $\mathbf{C_{ij}}$, = $\mathbf{C_{ij}}$, para todo i, jev, entonces se dice que el problema es un VRP simétrico; de lo contrario, se denomina VRP asimétrico.

En varios casos prácticos, la matriz de costos satisface la desigualdad del triángulo, de manera que $\mathbf{C_{ik}}$, + $\mathbf{C_{jk}}$, = $\mathbf{C_{ij}}$, para cualquier i, j, k ev. el VRP consiste en encontrar un conjunto de k circuitos simples que corresponden a una ruta vehicular con costo mínimo, definido como la suma de los costos de los arcos de los circuitos tal que:

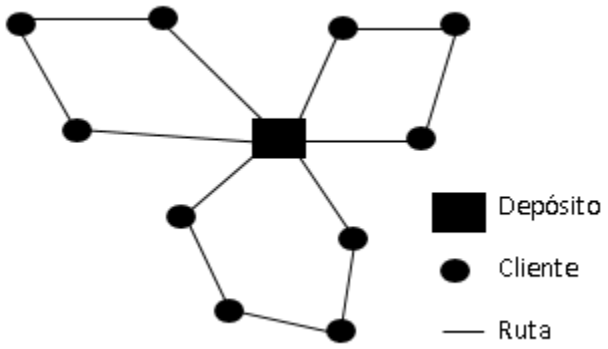
1. Cada circuito k, empieza en el vértice 0, ósea en el depósito.
2. Cada vértice j ev, es visitado por un único circuito k.

3. La suma de la demanda de los vértices no puede exceder la capacidad del vehículo en el circuito k.

Como existe un limitante de capacidad en el VRP, donde la suma de la demanda de los vértices no puede exceder la capacidad del vehículo en el circuito k, también se denomina a este modelo como VRP capacitado, (CVRP, por sus siglas en inglés Capacited VRP), otra particularidad de estos problemas es la distancia recorrida del vértice i hasta el vértice j, o de un cliente a otro, si la distancia $\mathbf{D_{ij}}$, es igual a la distancia $\mathbf{D_{ji}}$, se denomina como CVRP simétrico (SCVRP, por sus siglas en inglés Symmetric Capacited VRP), sin embargo, si esta restricción no se cumple es un (ACVRP, por sus siglas en inglés Asymetric Capacited VRP)

Figura 4

Ejemplo solución VRP



Nota: Fuente El autor.

En la ***Figura 4*** se muestra la representación de una solución al VRP, con su respectivo conjunto de vértices (clientes), vértice 0 o deposito inicial y sus arcos direccionados en secuencia

lógica, se determinan 3 rutas para la totalidad de k circuitos con 3 vehículos, teniendo como variables representativas de este modelo, vehículos, deposito, cliente y distancias entre clientes.

En el marco taxonómico literario se encuentran variantes del VRP de acuerdo al sistema en estudio, algunos casos altamente teóricos con restricciones académicas y otro grupo de estudios aplicados, en la práctica y llevada la problemática a la realidad existen variables que son difíciles de controlar, por este motivo en la búsqueda de soluciones para esta problemática nacen diferentes modelos matemáticos donde se contemplan restricciones como, tipo de operación, entregas y recolecciones, capacidad de vehículos, clientes, demandas, tráfico y ventanas horarias, a continuación se exponen algunos modelos que se han creado para VRP con restricciones.

Dentro de los VRP más destacados se pueden mencionar:

- CVRP (Capacitated VRP)
- MDVRP (Multi Depost VRP)
- VRPPD (VRP With Pickup and Delivery)
- VRPTW (VRP With Time Windows)

2.2 Variantes del VRP

2.2.1 Problemas con Restricciones de Capacidad (CVRP)

El CVRP es una variante del VRP en la cual se tiene en cuenta la capacidad del vehículo con flota homogénea, se debe satisfacer la demanda de un conjunto de clientes k, con un grupo de flota con capacidad limitada c, el objetivo de este modelo es minimizar la cantidad de vehículos, tiempos y costos empleando una distancia mínima, para poder realizar este modelo la demanda no puede exceder la sumatoria de capacidad de la flota y cada vehículo no puede exceder su capacidad en la ruta asignada.

Características:

- Demanda de cada cliente conocida.
- Capacidad de cada vehículo conocida.
- Ubicación geográfica de cada cliente conocida.
- Objetivo: Disminuir la cantidad de vehículos utilizados con sus respectivas distancias recorridas.

Modelación matemática: de acuerdo a la definición de (Yepes Cañada, 2014), se puede resumir el modelo matemático de la siguiente manera:

- Función objetivo: minimizar costos totales, realizando la sumatoria de costos de cada arco asignado $\mathbf{X_{ij}}$
- CVRP: tiene una demanda $\mathbf{d_j}$ asociada a cada cliente i ϵ v
- Cada cliente o nodo visitado y, debe ser asignado a un único vehículo k.
- Cada camión k, debe salir y regresar al depósito 0.
- Ninguna ruta debe superar la capacidad de cada vehículo k

2.2.2 Problema con Múltiples Depósitos (MDVRP)

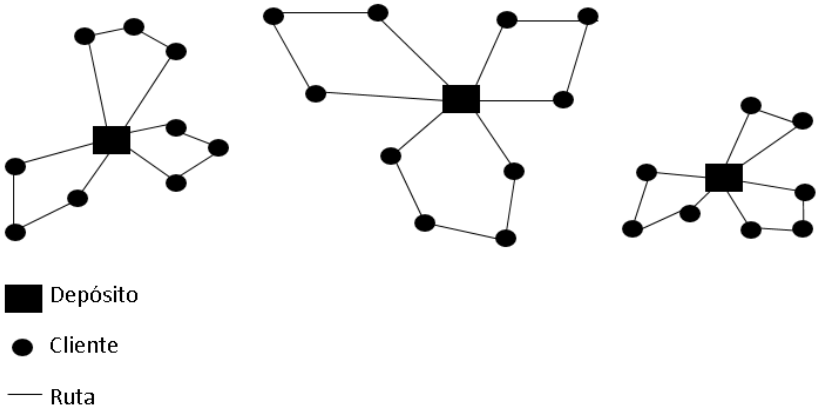
El MDVRP es una extensión del VRP, donde la empresa tiene al menos dos depósitos para atender clientes, este problema de asignación y ruteo se pueden dividir en sub-problemas independientes y realizar una asignación de rutas para cada uno, es necesario que los clientes estén agrupados de acuerdo con un depósito y de no ser así teniendo en cuenta las restricciones de capacidad, distancia, demanda, deben ser agrupados antes de resolver el modelo. (Yepes Cañada, 2014)

Características:

- Demandas conocidas.
- Cada ruta inicia y finaliza recorrido en un mismo depósito.
- Cada cliente es visitado por una sola ruta.
- La demanda asignada no puede superar la capacidad de cada vehículo.
- Cada cliente es agrupado a un solo depósito.

Figura 5

VRP con múltiples depósitos



Nota: Fuente El autor.

En la ***Figura 5*** se muestra la representación gráfica del MDVRP con 3 depósitos de los cuales se despachan en cada uno 3 vehículos para atención de 9 clientes diferentes.

2.2.3 Problemas con Entregas y Devoluciones (VRPPD)

De acuerdo a la definición de (Yepes Cañada, 2014), se puede precisar el VRPPD de la siguiente manera:

Modelo de ruteo de vehículos que pretende generar rutas teniendo en cuenta que cada cliente i tiene asociado un volumen $\mathbf{d_i}$ y $\mathbf{P_i}$ que representan para cada nodo una cantidad de entrega y otra para recoger respectivamente, para programar este modelo siempre se determina la demanda como d= $\mathbf{d_i}$ - $\mathbf{P_i}$, así se asegura que la demanda no pueda ser negativa y que primero el vehículo entregara para luego recoger, en este modelo la capacidad del vehículo juega un papel fundamental puesto que esta determinara la cantidad de nodos a visitar.

Características:

- Capacidad de cada vehículo conocida.
- Cada ruta empieza y finaliza en el depósito inicial.
- Los clientes son visitados por una única ruta.
- Carga del vehículo debe ser positiva.
- Demanda de cada cliente es conocida.

2.2.4 Problemas con Ventanas de Tiempo (VRPTW)

El VRPTW se ha convertido en un problema matemático muy famoso que se aproxima a los modelos de distribución reales, este problema de decisión consiste en crear diferentes rutas para un conjunto de flota k, teniendo en cuenta que deben satisfacer restricciones de ventanas de tiempo con horarios fijos por cada cliente y no se debe exceder la capacidad de carga para cada

vehículo, la solución no será factible si las rutas sobre pasan el tiempo de llegada a cada cliente, en este caso existirá un factor de penalización la cual se verá reflejado en la función objetivo. (Ricardo Díaz Lozada, 2012)

Para este modelo una restricción dura, exige que siempre se tenga que cumplir la ventana de tiempo, mientras que se llama blandas si puede no tenerse en cuenta, sin embargo, la violación de una restricción blanda deberá ser penalizada en la función objetivo, el VRP con restricciones de ventana de tiempo estrictas se abrevia como VRPHTW, generalmente en estos modelos matemáticos no se restringe el sistema a ventanas de tiempo estrictas por la complejidad en su solución, además, en situaciones del mundo real, la ventana de tiempo y las limitaciones de capacidad a menudo se pueden violar hasta cierto punto (hashimoto et al., 2008)

A continuación, se relatan algunos de los artículos y trabajos que cuentan con gran relevancia en el tema de VRPTW, el cual es el enfoque de esta monografía.

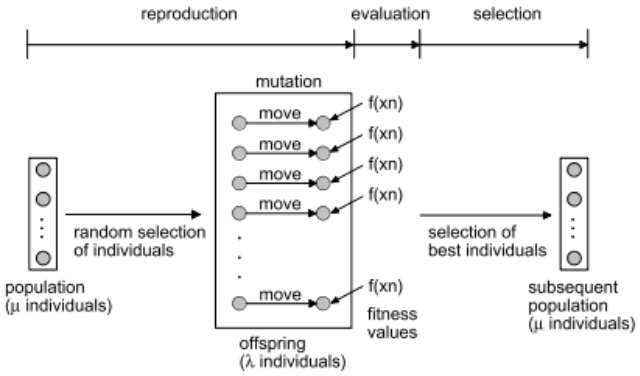
- El autor (Solomon, 1987) estudia las heurísticas aplicadas al VRPTW y realiza un análisis comparativo en cuanto a su rendimiento computacional, analiza las heurísticas de ahorros, heurísticas orientadas al tiempo, del vecino más cercano, inserción, barrido, encontrando las heurísticas de inserción como buenas soluciones para esta variante del problema.
- (Bonostro, 1995) realiza un resumen de técnicas de solución que se adaptan a problemas reales de VRPTW, concretamente desarrolla un conjunto de métodos exactos y heurísticos para el problema de ruteo de vehículos con ventanas de tiempo, inicia sus modelos de VRP con un TSP clásico y finaliza con el TSPTW y VRPTW para ser programados en lenguajes de alto nivel como C++ O C, sus algoritmos los diseña en un computador con características: AT I486 DX2 a 50 MHZ.

- (Chiang & Russell, 1996) resuelven el VRPTW mediante la metaheurística de recocido simulado, en este artículo, la heurística de mejora del recocido simulado se invoca periódicamente durante el proceso de construcción de la ruta paralela, basada en la heurística de construcción de Solomon, se implementan tres recocidos simulados, con dos estructuras de vecindades diferentes, una mejora usando una memoria tabú a corto plazo, sus algoritmos son programa en un computador con características 486dx2 / 66.
- (Fisher et al., 1997) describen dos procesos para la resolución del VRPTW con ventanas de tiempo duras, el primero un método k- árbol de relajación como extensión del TSP original, el segundo una descomposición Lagrangiana de división de variables, donde se suaviza el problema original dividiéndolo en dos sub-problemas, para dar solución óptima a problemas con hasta 100 clientes, el problema se resume en una semi asignación y sub-problemas de VRPTW con restricciones de capacidad.
- (Cordone & Calvo, 2001) propone una solución heurística alternativa para el VRPTW, por medio de la combinación de 3 procesos, en el primero se utilizan intercambios clásicos K-OPT para mejorar las rutas, un procedimiento ad hoc reduce el número de vehículos y una segunda función objetivo impulsa la búsqueda fuera de los óptimos locales, con este trabajo el objetivo es demostrar que, incluso si no se explotan metaheurísticas de moda como la búsqueda tabú, el recocido simulado o los algoritmos genéticos, se pueden obtener resultados competitivos en poco tiempo, inclusive mejores.
- (Gehring, 2004) resuelve el VRPTW con un solo deposito mediante una metaheurística hibrida de dos fases, con el objetivo de minimizar la cantidad de vehículos asignados para el enrutamiento y en un segundo plano la distancia recorrida por cada uno, en la primera fase por medio de una metaheurística evolutiva se minimizan la cantidad de vehículo y en

la segunda fase por medio del algoritmo de búsqueda tabú se minimizan las distancias recorridas, se presenta una comparación de resultados en problemas de 100 a 1000 nodos.

Figura 6

Estrategia de evolución para el VRPTW



Nota: Tomada de *(Gehring, 2004)*

En la ***Figura 6*** se muestra la estrategia evolutiva para la generación y evaluación de descendientes aplicada por Gerhring, para la selección de individuos con alta relación $\mu = \lambda$ y poder controlar el carácter de la búsqueda global exploratorio o explotadora.

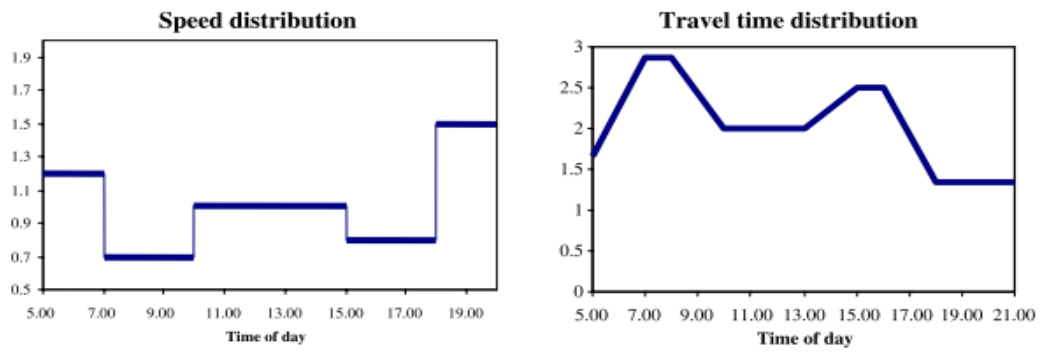
- (Hashimoto et al., 2008) en este artículo se utiliza la búsqueda local para determinar las rutas asignadas, calculado los horarios óptimos por ruta evaluando las soluciones de cada vecindad, a diferencia de otros autores Hashimoto et al., generaliza el VRPTW permitiendo que los tiempos de viaje y costos asociados al viaje sean funciones dependientes del tiempo y puedan ser aplicados en horas punta, el vecindario de la búsqueda local consiste en ligeras modificaciones de los vecindarios estándar llamados 2-OPT, intercambio cruzado y OR-OPT. Los experimentos computacionales para evaluar el

algoritmo ILS propuesto fueron codificados en lenguaje c ejecutados en un pc (INTEL PENTIUM 4, 2,8 GHZ, 1 GB de memoria).

- (Donati et al., 2008) el tráfico juega un papel fundamental cuando se realizan modelos VRPTW y no pueden ser ignoradas para encontrar soluciones viables que se ajusten a la realidad, este artículo muestra que cuando se trabaja con ventanas de tiempo con problemas de entrega difíciles para los clientes, las soluciones al problema clásico se vuelven sub-óptimas si no se considera la restricción del tráfico en diferentes rangos de tiempo.

Figura 7

Distribución de velocidad y tiempo de viaje



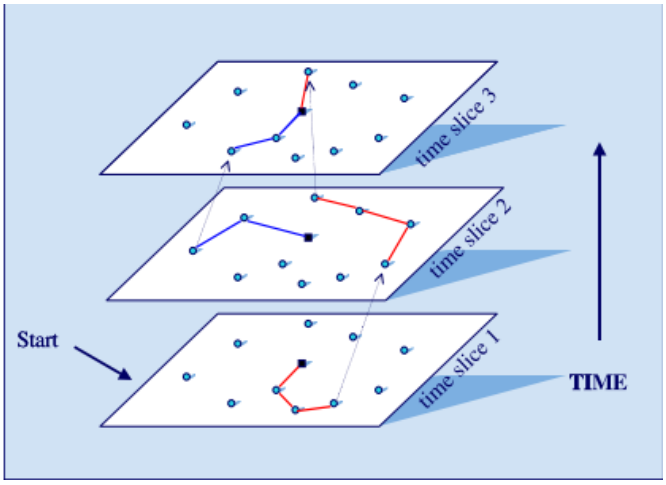
Nota: Tomada de (Donati et al., 2008)

En la ***Figura 7*** se puede observar la velocidad de distribución en diferentes ventanas de tiempo con su respectivo tiempo de viaje cuando se depende del tráfico, entre mayor sea la velocidad de la distribución menor será el tiempo de viaje de un nodo a otro, el tráfico se convierte en una variable que es necesaria tener en cuenta cuando se resuelve un VRPTW, así el modelo se puede ajustar más a la realidad.

Donati, utiliza el algoritmo llamado Multi Ants Colony System (MACS-VRPTW) con una jerarquía de dos colonias de hormigas artificiales, las cuales se ocupan de uno de los objetivos de la optimización: la primera colonia se ocupa de la minimización del recorrido mientras que la otra minimiza la distancia. Las dos colonias cooperan intercambiando información mediante la actualización de feromonas.

Figura 8

Representación de ruteo con m periodos de tiempo



Nota: Tomada de (Donati et al., 2008)

En la ***Figura 8***, se puede observar tres niveles de tiempo diferentes que representan el ruteo de vehículos con colonias de hormigas y como en cada iteración se mejora la solución inicial.

- (El-Sherbeny, 2010) presenta una revisión con algunos limitantes para el VRPTW, en métodos exactos heurísticos y Metaheurísticos, dentro de los modelos exactos explica métodos basados en la relajación de Lagrange, generación de columnas, programación

dinámica, dentro de las heurísticas, modelos contractivos y de mejora, inteligencia artificial y metaheurísticas, recocido simulado, observación y algoritmos genéticos.

- (Pureza et al., 2012) resolución del modelo VRPTWMD (múltiples repartidores) utilizando dos enfoques Metaheurísticos, en una primera fase resuelve el modelo utilizando el algoritmo de búsqueda tabú y en una segunda parte utiliza la optimización de colonia de hormigas para encontrar una solución alternativa, para la primera fase se utiliza una heurística constructiva basada en el TSP con múltiples repartidores y posterior se utiliza la búsqueda tabú para mejorar la solución inicial. Los experimentos se realizaron en un ordenador Intel Core2 I7 A 2,67 GHz Con 12 Gb De RAM.

En la segunda fase, la optimización de colonia de hormigas, se aplica la siguiente lógica:

1. Leer los datos de entrada.
2. Iniciar los parámetros y la matriz de la feromona.
3. Repetir hasta que se cumpla un criterio de parada preestablecido.
    a. Construir una solución factible.
    b. Aplicar la búsqueda local.
    c. Actualizar la matriz de feromona.
4. Devolver la mejor solución encontrada.

- Los autores (Lozada Díaz, Ricardo, 2012) resuelven el VRPTW, para una instancia de 20 clientes, en el aplicativo Matlab versión R2010A, utilizando el algoritmo de ahorros de Clarke and Wright durante tres corridas, algoritmo del vecino más cercano en cuatro corridas, heurística de inserción I1 y I2 consignado los tiempos promedios en cuatro corridas y la heurística de inserción I3, para esta última heurística se utilizan 3 criterios como inicio de ruta: cliente más lejano del depósito, cliente con ventana de tiempo que se

termina primero que las demás, cliente que minimice la distancia y el tiempo, esa solución fue ejecutada en un equipo con Procesador Intel Core I3 y 3 Gb de memoria RAM instalada.

- (Montes Orozco, 2017) trabajo realizado para obtener el título de master en optimización, se presentan 7 técnicas metaheurísticas hibridas para resolver el VRPTW programadas en el lenguaje de C, entre ellas el "Sistema de hormigas (AS), búsqueda armónica (HS), algoritmo genético (GA), búsqueda local iterada (ILS), algoritmo primal dual (PDA) y método dual simplex (DSM)" (Montes Orozco, 2017,p.7).

  Sus métodos de solución fueron probados en todos los casos para instancias de 12 clientes, dentro de sus algoritmos híbridos se encuentran AS-GA, AS-HS, DSM-AS-PDA, AS-ILS trabajando de forma entre lazada, por ejemplo el AS – GA: se utiliza el algoritmo genético, para encontrar una solución de población inicial, el AS se utiliza para mejorar la solución de esta población inicial y así actualizar la matriz de feromonas del siguiente ciclo, esto ayuda a no converger prematuramente y explorar otros espacios de búsqueda.

2.3 Métodos para Abordar el Problema

De acuerdo con la complejidad en la resolución del VRP se han desarrollado diferentes métodos de solución los cuales se clasifican en exactos y aproximados, los exactos buscan llegar a una solución óptima por medio de programación lineal entera y los aproximados buscan llegar a una solución confiable y aceptable por medio de heurísticas y metaheurísticas.

2.4 Métodos Exactos

Conforme a la definición de (Lüer et al., 2009) los métodos exactos son aquellos que se presentan por medio de programación lineal entera, los cuales llegan a una solución óptima y

factible por medio de un conjunto de acotaciones del problema, estos problemas se vuelven complejos de resolver cuando el grupo de entradas y restricciones aumenta en el modelo matemático, se considera una técnica apropiada cuando la cantidad de nodos es inferior a 50, algunos modelos conocidos son; programación lineal entera, técnicas de Branch & Bound, técnicas de Branch & Cut y técnicas de relajación.

2.4.1 Programación Lineal Entera

Los problemas de programación lineal entera normalmente requieren gran cantidad de recursos computacionales en la búsqueda exhaustiva de la mejor solución, puesto que, este método garantiza optimalidad en su respuesta, existen algunas técnicas que permiten dividir el modelo en sub-problemas y mejorar los tiempos de procesamiento encontrando la mejor respuesta.

2.4.2 Algoritmo de Ramificación y Acotamiento (Branch & Bound)

Es un método propuesto por Little en 1998, diseñado para la solución de problemas de programación lineal entera por medio de algoritmos, también conocido como método de ramificación y acotamiento, su algoritmo consiste en crear soluciones factibles recorriendo cada nodo del árbol desde el nivel superior hasta las hojas, se van creando cotas y soluciones las cuales son guardadas, el algoritmo finaliza cuando se visitaron todos los nodos y la solución óptima es la de la acotación inferior, este problema tiene un tiempo de solución exponencial y no debe usarse cuando la cantidad de nodos es alto, además se debe determinar que si un nodo inicial no es factible, todas sus ramificaciones tampoco lo serán.

2.4.3 Algoritmo de Ramificación y Corte (Branch & Cut)

Es un método propuesto por Gemory en 1958 también conocido como técnica de ramificar y cortar, consiste en dividir el problema por medio de planos para resolver los sub problemas

generados en cada rama, a diferencia de Branch& Bound no se trabaja con restricciones de números enteros y se puede solucionar inclusive con métodos simplex convencionales, cuando se tiene una solución óptima no entera se utiliza los planos cortantes para crear restricciones que satisfagan el modelo y los puntos enteros factibles en la primera solución encontrada.

2.4.4 Técnicas de Relajación

Son métodos matemáticos asociados a la programación lineal entera, donde se descompone el modelo en restricciones complejas difíciles de resolver y otras restricciones más sencillas, obteniendo una relajación lineal, el objetivo es relajar las restricciones difíciles para combinarlas con la función objetivo por medio de un multiplicador de penalidad, la técnica de relajación más conocida es la introducida por Held y Karp en 1970, relajación Lagrangiana la cual consiste en penalizar la función objetivo cuando se violen las restricciones que fueron relajadas a partir de las restricciones más complejas del modelo, la penalidad se efectúa con pesos determinados y se obtiene un problema más sencillo de resolver, a continuación se muestran las mejores técnicas de relajación.

1. Relajación Lagrangeana. fue introducida por Held y Karp (1970).
2. Descomposición de Dantzig-Wolfe. fue introducida por Dantzig y Wolfe (1960)
3. Generación de planos de Cort. (Montes Orozco, 2017)

2.5 Métodos Heurísticos

A diferencia de los métodos exactos las heurísticas no proporcionan una solución óptima y factible en la resolución del modelo, exploran buenas soluciones en tiempos razonables basadas en el sentido común, estas técnicas son usadas cuando es necesario tomar decisiones en sistemas complejos, o cuando se tiene poco tiempo para ponerlos en marcha, en problemas de mayor complejidad permite tener tiempos de ejecución menores para la obtención de soluciones

razonables y aceptables, de acuerdo a cada modelo y cada acotamiento las heurísticas se clasifican de la siguiente manera:

- Constructivas: No parten de una solución factible, estas se van encontrando a medida que el algoritmo progresa, por ejemplo, una muy conocida es la heurística de ahorros donde mediante la creación de n rutas factibles, se empiezan a agregar arcos de un nodo i a un nodo j calculando los ahorros en cada uno de estos movimientos, otro ejemplo es la heurística de pétalos, donde por medio de un conjunto de nodos ubicados geográficamente se realiza un barrido de acuerdo al ángulo que forman con el depósito y regresan al mismo cuando se cumple su tiempo de viaje o capacidad del vehículo.
- De Mejora: El modelo parte de una solución factible y encuentran diferentes soluciones cuando el algoritmo progresa, algunas heurísticas conocidas son: 2- Opt, 3 – Opt, Link-Kernighan y 2 – Swap.

2.5.1 Heurísticas Constructivas

Son algoritmo que van creando soluciones factibles paso a paso a partir de una solución vacía, o no factible, dentro de este grupo se encuentran los algoritmos de ahorro, a continuación, se describen las principales heurísticas constructivas:

- Heurística de ahorro: Planteado por Clarke and Wright en 1964, permite generar rutas de una forma más eficiente optimizando los recursos asignados en la distribución (cliente – deposito), (deposito – cliente).
- Mejoras del algoritmo de ahorros: Propuestas por Gaskell en 1967, se establece la relación a menor distancia entre nodos mayor ahorro en el modelo.

- Algoritmo de ahorros basado en acoplamientos: Propuesto por Desrochers y Verhoog en 1989, se debe tener en cuenta una gráfica y matriz de ahorros que considera como la unión afectara las próximas iteraciones.

2.5.2 Heurística del Vecino más Cercano

Es un método que permite crear una ruta para cada vehículo realizando un análisis de distancias y puntos de clientes próximos a visitar, el algoritmo evalúa todos los nodos y selecciona el más próximo teniendo en cuenta el criterio de la distancia, este criterio se repite n – veces hasta completar la restricción de capacidad ya sea de volumen o de tiempo de viaje por cada vehículo k, sin embargo este método puede llegar a la construcción de rutas que muchas veces no siguen una secuencia lógica y puede aumentar el costo de viaje, esta técnica fue propuesta por Rosenkrantz en 1974 y su algoritmo consta de los siguientes pasos:

1. Determinar nodo inicial de la ruta.
2. Encontrar el nodo más cercano al último añadido.
3. Repetir el paso dos hasta que todos los nodos estén contenidos (Bonostro, 1995)

2.5.3 Heurística de Dos Fases

Esta heurística consiste en dividir el modelo VRP en dos fases, la primera realiza un agrupamiento de nodos y la segunda asigna las rutas a cada vehículo, esta técnica permite encontrar soluciones factibles que normalmente no resultan ser las óptimas, en esta heurística los métodos más utilizados para resolver los VRP son los siguientes:

2.5.4 Método de Rutear Primero y Asignar Después

Técnica propuesta por Beasley en 1983 la cual se divide en dos fases, en la primera partiendo de un único vehículo se resuelve el problema de VRP utilizando el algoritmo TSP, en esta

primera fase no se tiene en cuenta la restricción de capacidad del vehículo y todo el sistema es resuelto por un solo agente, en la segunda fase esta ruta se descompone en varias rutas factibles, de acuerdo a la solución de la primera fase, se determina la mejor partición la cual cumple la restricción de capacidad del vehículo.(Montes Orozco, 2017)

2.5.5 Método de Asignar Primero y Rutear Después

Esta técnica fue propuesta por Christofides, Mingozzi y Toth en 1979, consiste en ser resuelta mediante dos fases, en la primera se realiza una división de clientes en grupos (Clusters) teniendo en cuenta que el grupo de clientes será visitado por una única ruta la cual no podrá exceder su restricción de capacidad, ya sea de volumen o tiempo, la segunda fase consiste en construir una ruta dentro del Cluster por medio de técnicas exactas o aproximadas y que se ajuste a las restricciones del sistema. Algunas heurísticas conocidas para asignar y luego rutear son: barrido, asignación generalizada y de locación.(Comercial & Pesqueros, 2014)

2.5.6 Algoritmo de Barrido

Técnica propuesta por Wren en 1975, consiste en realizar un barrido de clientes con el fin de generar rutas optimas cumpliendo con todas las restricciones del modelo y en particular la de capacidad del vehículo, esta heurística consiste en trazar un línea recta desde el depósito e ir girando hasta agrupar un conjunto de nodos y ocupar el espacio asignado en cada ruta, luego cada ruta generada se mejora por medio otra técnica de ruteo, generalmente se soluciona por medio del TSP, esta técnica necesita que la ubicación geográfica de cada nodo esté representada por medio de coordenadas polares o angulares y se divide de la siguiente manera:

1. Agrupar clientes en rutas.
2. Definir k=2, nr=1
3. Si el vehículo tiene capacidad agregar k a la ruta nr.

4. Resolver el TSP para cada grupo formado. (Bonostro, 1995)

2.5.7 Heurística de Pétalos

Propuesta por Ryan, Hjorring y Gloveren en 1993, esta técnica surge como extensión del algoritmo de barrido donde se realiza una asignación partiendo del depósito y teniendo en cuenta la capacidad del vehículo, los nodos deben estar ubicados geográficamente mediante coordenadas polares con depósito en (0,0) para realizar un barrido desde cada nodo i e ir agregando más clientes conforme el algoritmo se desarrolla, este proceso se repite n veces hasta que el vehículo cumpla su capacidad y tenga que regresar al punto de donde partió (0,0), una vez se asignan todas las rutas o pétalos se puede mejorar la solución por medio del TSP en cada circuito **K**.

2.5.8 Heurística de Mejora

Este método conocido como técnica de búsqueda local se utiliza para encontrar una mejor solución a la inicialmente calculada en el VRP propuesto, se emplea para optimizar una o más rutas modificando el orden de visitas de cada nodo sin afectación de la totalidad de nodos que debe visitar, esta técnica de búsqueda local se desarrolla partiendo de una solución K la cual es reemplazada por K', el objetivo es encontrar soluciones vecinas que tengan asociado un menor costo, este procedimiento se repite n-veces hasta que no se encuentre una mejor solución, a continuación se describen los principales métodos de búsqueda local:

1. Operador de intercambio: λ: Propuesto por lin en 1965.
2. El algoritmo de Lin-Kernighan: Propuesto en 1973, intercambios de subconjuntos.
3. El operador OR-OPT: Propuesto por or en 1976, eliminación de secuencias de clientes.
4. Geni y Genius: Propuesto por Gendreau et al. en 1992.

5. Algoritmos de transferencias cíclicas: Propuesto por Thompson y Psaraftis en 1993, asignación de clientes de forma cíclica.
6. Operadores de Van Breedam: Propuestos en 1995, intercambio de clientes entre rutas.

2.6 Métodos Metaheurísticos

Los métodos metaheurísticos son una estrategia tipo heurística diseñada para solucionar problemas de difícil solución, fueron desarrollados a partir del año 1990 y se caracterizan por resolver modelos donde las heurísticas clásicas no son efectivas, estas técnicas pueden ser usadas cuando se requiere una búsqueda más exacta puesto que contempla más soluciones que un método heurístico.(Montes Orozco, 2017)

Las técnicas metaheurísticas están por encima de las técnicas heurísticas y permiten resolver problemas de optimización combinatoria muy complejos, a un coste computacional razonable de manera autónoma.

A continuación, se presenta la lista de metaheurísticas aplicadas al VRP

2.6.1 Algoritmos Genéticos

Propuesto por Holland en 1975, esta técnica estudia el comportamiento de la naturaleza y fue inspirada en la teoría de la evolución Darwiniana, este modelo parte de una población inicial de individuos como posible solución del problema, posteriormente esta población evoluciona por medio de operaciones creando una población nueva, normalmente se trabajan con técnicas de cruce, selección y mutación. “El algoritmo opera sobre una población p de soluciones codificadas, llamadas individuos. para cada individuo i ∈ p se define una función de adaptación f(i) de modo que cuando mayor sea el valor de adaptación de un individuo, mejor es la solución”. (Olivera, 2004, p.36).

Proceso evaluado en algoritmos genéticos:

1. Población Inicial: De forma aleatoria o utilizando técnicas exactas o heurísticas se crea una población la cual será la primera solución y la base para seguir generando poblaciones.
2. Evaluación de los Individuos: Se crea una población de individuos más adaptados, mediante la evaluación de la solución o el valor de la adaptación de cada individuo, se deben tener en cuenta los aspectos por los cuales se define que es el mejor.
3. Selección de padres: " Hay varias maneras de seleccionar los padres (estos crearán los cruces), entre las más destacadas están: el modelo ruleta en el cual se tiene en cuenta la probabilidad de cada individuo, y el modelo elitista, donde se escoge siempre al mejor individuo" (Gomez David, 2019, p.37).
4. Operadores genéticos: Se deben crear mejores individuos después de haber seleccionado la población P de padres, posteriormente a estos nuevos individuos se les aplica un proceso de mutación, este principio "Se basa en la modificación de la introducción de información aleatoria del código genético del individuo, se resalta que esta mutación tiene una probabilidad de suceder pequeña, pero es importante que exista, puesto que de esta manera ayuda a la introducción de nuevo material genético y al estudio de cualquier espacio de zona de soluciones". (Castañeda, 2014, p.57).

   Finalmente se genera un proceso de terminación del algoritmo después de n-repeticiones, el fin del proceso lo determina el programador por medio de estabilidad del algoritmo. se pueden representar por medio grafico las diferentes soluciones y ver visualmente cuando este se encuentra estable y sus soluciones no mejoran de forma notoria.

2.6.1 Búsqueda en Vecindades Variables

De acuerdo con la definición de K.H. Hansen y J. Krarup (1974) la búsqueda en vecindades variables se puede definir como:

"Técnica que realiza una búsqueda local, en la cual, se lleva a cabo distintos cambios de vecindarios; aquí, generalmente se realizan vecindarios con cambios pequeños al inicio y conforme al paso de la búsqueda, se utilizan vecindarios con cambios más fuertes o viceversa. por otro lado, para evitar que la búsqueda se vuelva aleatoria es recomendable utilizar distintos tamaños de vecindario al inicio de la búsqueda y cuando esta se encuentra avanzada". (Montes Orozco, 2017, p.38).

2.6.2 Recocido Simulado

El recocido simulado o templado simulado es una técnica que busca escapar el óptimo local al optimo global y se basa en el comportamiento físico de los metales y algunas cerámicas cuando se someten a temperaturas muy altas en su proceso de templado y enfriamiento, para obtener de forma gradual propiedades específicas de cada material, este proceso es aplicado en problemas de programación combinatoria donde en la primera fase de calentamiento los átomos de los materiales aumentan su energía y pueden tomar diferentes estructuras, después se enfrían controlando la energía y obteniendo estabilidad en el proceso.

2.6.3 Redes Neuronales

Propuestos por Hopfield y Tank en 1985, es una técnica inspirada en el comportamiento del sistema nervioso en el cual se define una técnica de auto aprendizaje, se trata de un sistema de interconexión de sistemas que funcionan y operan entre sí para producir un estímulo de salida o de posibles soluciones, en el 2010 Asensio et al, presentó el trabajo "Predecir el riesgo de

padecer trastornos músculo- esqueléticos asociado a puestos de trabajo con tareas de levantamiento de carga" por medio de redes neuronales artificiales.

2.6.4 Búsqueda Tabú

Propuesta por Hopfield y Tank en 1985, es una técnica de optimización la cual consiste en generar una solución inicial aleatoria y realizar una búsqueda local para mejorar su solución en diferentes vecindades en cada iteración, hasta alcanzar un óptimo local por medio de la búsqueda por proximidad, de tal manera que en cada iteración se desplaza de una solución (KT) a otra mejor (KT+1) dentro del sub conjunto de posibles soluciones cercanas, como (KT+1) no es necesariamente la mejor solución se utiliza una memoria de corto plazo que restringe soluciones ya visitadas, "este método de mejora local puede recalar en un óptimo local que no es mejor que un óptimo global, pero que no pudo ser explorado en el vecindario de soluciones. para ello, se permiten movimientos que no tengan mejora alguna, lo que permite hallar nuevos óptimos locales y, finalmente, el óptimo global" (Velazco, 2019, p.38).

2.6.5 Algoritmo Colonia de Hormigas

Propuesta por Dorigo et al., en 1992, esta técnica se inspira en cómo se mueven las hormigas para encontrar su alimento, en este proceso las hormigas recorren diferentes caminos hasta encontrar comida, una vez es encontrada la comida las hormigas regresan a la colonia dejando un rastro llamado feromona, la cual guiara a otras hormigas a seguirlo hasta volver, mientras más hormigas sigan este rastro y se libere más feromona se fortalecerá el camino y la feromona se acumulara en los caminos más cortos para regresar.

De forma analógica cada hormiga al regresar a la colonia establecerá una posible solución (k) "cada hormiga construye soluciones parciales, cada vez más completas, mediante procesos constructivos, pasando de unos estados a otros. en este caso se dice que el rastro de feromonas

dejada por la hormiga es la probabilidad de pasar de un estado a otro (solución parcial) determinado. cuanto mayor sea el rastro de feromonas, mayor será la probabilidad de elegir ese cambio de estado". (Yepes Cañada, 2014).

Por analogía con la realidad (Yepes Cañada, 2014) define que:

Este método emplea hormigas que crean soluciones. En cada iteración una colonia construye un conjunto de soluciones, cada hormiga construye soluciones parciales, cada vez más completas, mediante procesos constructivos, pasando de unos estados a otros. En este caso se dice que el rastro de feromonas dejada por las hormigas es la probabilidad de pasar de un estado (solución parcial) a otro determinado, cuanto mayor sea el rastro de feromonas, mayor será la probabilidad de elegir ese cambio de estados. (p.42)

A continuación, se presenta la modelación matemática del algoritmo de colonia de hormigas:

- Aporte de feromonas de cada hormiga (h) a cada arco de camino recorrido $\mathbf{S}(\boldsymbol{\tau}_{\mathbf{rs}})$:

$$\boldsymbol{\tau}_{\mathbf{rs}} = \boldsymbol{\tau}_{\mathbf{rs}} + \Delta\boldsymbol{\tau}_{\mathbf{rs}}^{\mathbf{h}} \; \forall \; \mathbf{a}_{\mathbf{rs}} \in \mathbf{s}^{\mathbf{h}}$$

- Evaporación de feromonas evitando óptimos locales:

$$\boldsymbol{\tau}_{\mathbf{ij}} = (\mathbf{1} - \mathbf{p}) * \boldsymbol{\tau}_{\mathbf{ij}}$$

- Probabilidad de que una hormiga (h), recorra el nodo (i) al nodo (j)

$$\mathbf{p}_{\mathbf{ij}}^{\mathbf{h}} = \left\langle \frac{\left(\boldsymbol{\tau}_{\mathbf{ij}}^{\boldsymbol{\alpha}}\right) * \left(\mathbf{n}_{\mathbf{ij}}^{\boldsymbol{\beta}}\right)}{\sum\left(\boldsymbol{\tau}_{\mathbf{iu}}{}^{\boldsymbol{\alpha}}\right) * \left(\mathbf{n}_{\mathbf{iu}}{}^{\boldsymbol{\beta}}\right)} \right\rangle$$

0 en otros casos

2.6.6 Algoritmo colonia de hormigas en el VRPTW

De acuerdo al trabajo realizado por (Montes Orozco, 2017), se puede abordar el VRPTW con sistemas de hormigas de la siguiente manera, generalidades y modelo matemático.

Generalidades:

- Cada iteración a, se lanza un conjunto de hormigas (n) la cual construye una solución guiada por la probabilidad de visitar un cliente j, luego de salir de un cliente i, se utiliza la función de iniciación de feromonas y actualización de la probabilidad de elección antes de ejecutar el primer ciclo de n hormigas y se elige la mejor solución y se actualiza la matriz de feromonas.
- Para construir soluciones a cada hormiga se le asocia un conjunto de datos, entre los cuales están las ventanas de tiempo de cada cliente, una matriz de distancias d_{ij}, una matriz de feromonas, una matriz de probabilidad de elección de un camino, adaptación de la fórmula del cálculo de la probabilidad de elección para VRPTW – ACO (p.55).

Modelo matemático:

Iniciación de feromonas en términos del VRPTW:

$$\boldsymbol{\tau_{ij}^{a+1} = \frac{1}{l_j}, con\ a = 0} \quad (1)$$

Ecuación 1: donde: $\boldsymbol{\tau_{ij}^{1}}$ es el nivel de feromonas que tiene el camino de un cliente i a un cliente j y $\mathbf{l_j}$ último momento en el cual se puede atender a un cliente j

Actualización de feromonas en términos de VRPTW:

$$\boldsymbol{\tau_{ij}^{a+1} = \rho * (\tau_{ij}^{a}) + \Delta_{ij}} \quad (2)$$

Ecuación 2: Donde: $\boldsymbol{\rho}$ es el coeficiente de evaporación, $\boldsymbol{\tau_{ij}^{a}}$ es el nivel de feromonas que tiene el camino de un cliente i a un cliente j, $\boldsymbol{\tau_{ij}^{a+1}}$ es el nivel de feromonas que guiara el siguiente ciclo y Δ_{ij} la cantidad de feromonas depositadas por la mejor hormiga del ciclo.

Cálculo del Δ_{ij} en términos del VRPTW:

$$\Delta_{ij} = \frac{\mathbf{sol}[a] - \mathbf{sol}[a-1]}{\mathbf{sol}[a-1]} \tag{3}$$

Ecuación 3. donde $\mathbf{sol}[a]$, el valor de la solución entregada por la mejor hormiga del ciclo y $\mathbf{sol}[a-1]$, el valor entregado por la mejor hormiga del ciclo (a-1)

Cálculo de probabilidad en términos de VRPTW:

$$\mathbf{p_{ij}^{a+1}} = \frac{(\boldsymbol{\tau_{ij}^{a+1}})^{\alpha}(\mathbf{n_{ij}})^{\beta}(\mathbf{v_{ij}})^{\gamma}}{\sum(\boldsymbol{\tau_{ij}^{a+1}})^{\alpha}(\mathbf{n_{ij}})^{\beta}(\mathbf{v_{ij}})^{\gamma}} \tag{4}$$

Ecuación 4. Por cada ciclo a, de n hormigas, o se agrega un cliente a la solución, se actualiza la probabilidad de escoger un camino de un cliente i a un cliente j, mediante la ecuación 4, donde: $\boldsymbol{\tau_{ij}^{a}}$ es el nivel de feromona, $\mathbf{n_{ij}} = \frac{\mathbf{1}}{\mathbf{d_{ij}}}$ regula la conveniencia de viajar entre los clientes i y j, $\mathbf{v_{ij}} = \frac{\mathbf{1}}{\mathbf{l_j}}$ denota que entre menor sea el valor del último momento en que se puede visitar, la probabilidad de visitarlo al inicio será mayor, además α se utiliza para controlar la influencia del nivel de feromona, β es un parámetro para controlar la influencia de la cercanía entre clientes y γ controla la influencia del valor de l_j.

Normalizan de probabilidades VRPTW:

$$\mathbf{Pn_{ij}^{a+1} = Pn_{ij-1}^{a+1} + P_{ij}^{a+1}} \quad (5)$$

Tabla 3

Resumen de Técnicas de Solución para VRP

M	Técnica de Solución	Características	Recomendación
Métodos Exactos	Programación Lineal Entera	Requiere grandes recursos computacionales en la búsqueda exhaustiva.	Se vuelve ineficiente y su tiempo de solución es Exponencial cuando supera los 50 nodos.
	Algoritmos de Ramificación y Acotamiento	Se tiene el riesgo de encontrar una solución no factible.	El tiempo de solución es exponencial, se debe usar en problemas de hasta 50 nodos.
	Algoritmo de Ramificación y Corte	Divide el problema en sub-problemas para encontrar diferentes soluciones.	El tiempo de solución es exponencial, se debe usar en problemas de hasta 50 nodos
	Técnicas de Relajación	Se descomponen las restricciones difíciles del sistema, en restricciones más sencillas con penalidades en la función objetivo.	Se utilizan en problemas difíciles, su tiempo de solución es exponencial y no se recomienda para problemas con más de 50 nodos.
Métodos Heurísticos	Constructivos	Encuentra soluciones factibles en cada iteración del problema, calculando los ahorros con respecto a la solución anterior.	No parten de una solución factible, se debe usar en problemas de hasta 200 nodos.
	Vecino más cercano	Construye rutas de acuerdo con el cliente más próximo, teniendo en cuenta como criterio la distancia y capacidad del vehículo.	Se puede llegar aumentar el costo debido a que la técnica no realiza siempre secuencias lógicas para optimizar la distancia final.

	Rutear Primero y Asignar Después	Técnica utilizada para resolver el problema en dos fases, primero encontrar una solución y dividirla en más soluciones.	Es una técnica general que se puede aplicar para iniciar diferentes métodos de solución del VRP.
	Asignar Primero y Rutear Después	Técnica utilizada para resolver el problema en dos fases, combina estrategias de los métodos exactos.	Al utilizar técnicas de los métodos exactos, se vuelve ineficiente con problemas de más de 60 nodos.
	Algoritmo de Barrido	Soluciona el problema de forma angular teniendo en cuenta la capacidad del vehículo.	Se utiliza esta técnica para dar soluciones iniciales para otro tipo de estrategias, como AG.
	Algoritmo de Pétalos	Soluciona el problema de forma angular, teniendo en cuenta un único depósito en el vértice 0,0.	Se recomienda utilizar esta estrategia cuando solo existe restricción de capacidad.
	Algoritmo de Mejoras	Método para encontrar una solución mejor sobre una ya encontrada que tenga un menor costo.	Es una técnica que se puede aplicar en cualquier método heurístico o metaheurístico del VRP para mejorar la solución.
Métodos Metaheurísticos	Algoritmo Genético	Estudia el comportamiento de la naturaliza y fue inspirado en la teoría de la evolución de Darwin.	Se debe tener una población inicial de posibles soluciones para operar los AG, el desarrollador del algoritmo puede determinar en qué momento finalizar su algoritmo al evidenciar que no se encuentran mejores soluciones.

Recocido Simulado	Tánica basada en el comportamiento físico de los materiales en su enfriamiento cuando se someten a temperaturas altas.	Técnica utilizada para escapar de óptimos locales a óptimos globales, de acuerdo a trabajos realizados se recomienda en problemas complejos con gran cantidad de nodos en el sistema, sin embargo, se pueden encontrar soluciones de mayor costo al explorar otros espacios de búsqueda.
Búsqueda Tabú	Técnica que almacena soluciones a corto plazo y permite guardar soluciones que no son mejores a las anteriores encontradas, con el objetivo de explorar otras vecindades y escapar de óptimos locales.	Método utilizado en redes de gran cantidad de nodos, permiten encontrar una solución muy cercana a la óptima, se puede usar en problemas de restricciones difíciles de resolver.
Colonia de Hormigas	Técnica inspirada en el comportamiento del movimiento de las hormigas cuando buscan su alimento y regresan a la colonia, fortaleciendo los caminos y minimizando las distancias recorridas.	Las restricciones propias del modelo se dejan adaptar de acuerdo al tipo de VRP a solucionar, se utiliza normalmente en problemas con gran cantidad de nodos y sus tiempos de soluciones no son tan altos como otras metaheurísticas.

Nota: Fuente El autor.

En la ***Tabla 3*** se muestra un breve resumen de las técnicas de solución exactas, heurísticas y metaheurísticas para los problemas de VRP.

3. CAPÍTULO 3: METODOLOGÍA

Dado que el objeto de estudio es proponer una solución para el VRPTW, se utilizó una metodología con enfoque cuantitativo y alcance descriptivo, puesto que es la que mejor se adapta a las características del proyecto. Esta metodología con diseño experimental se desarrolla dentro de un proceso secuencial iniciando desde el planteamiento del problema, revisión de la literatura, alcance de la investigación, desarrollo, recolección y análisis de datos, este enfoque permite delimitar y plantear el problema de estudio de forma concreta.

3.1 Población de Estudio

Se estableció como población objeto los problemas relacionados con el VRPTW que se han expuesto desde 1987 con las instancias de Solomon, como referencia se tendrá en cuenta el modelo matemático expuesto por Montes, Orozco en el año 2017 en su tesis “Metaheurísticas para el problema de ruteo de vehículos con ventanas de tiempo (VRP-TW)”, donde se tienen en cuenta todas las restricciones que se asemejan al problema y el cual permitirá obtener un resultado más acorde a la realidad de la compañía.

Una vez definido el método de solución del problema se establece que la población de estudio será conformada por el conjunto de 610 clientes de la compañía en la ciudad de Bogotá, así como sus ubicaciones geográficas, demandas y condiciones especiales de atención. Para el cálculo de coordenadas y distancias entre nodos, se utilizará la herramienta VRP en Excel la cual tiene enlace directo con la aplicación web de mapas creada por Microsoft Bing Maps.

3.2 Muestra y Muestreo

Se utilizó el método de muestreo no probabilístico intencional donde se escogió como muestra los datos más representativos de la población, en este caso los clientes con mayor frecuencia de solicitudes y mayor demanda en unidades en cada solicitud, con ayuda del

Software EasyFit se realizó un análisis estadístico de datos de la compañía, ajuste de parámetro y distribución de probabilidad. Finalmente se estableció por medio de un análisis de Pareto que el 80% de las solicitudes pertenecían a un grupo de clientes que ingresaban dentro de los parámetros de la distribución, para obtener así la muestra final de 140 clientes diarios para calcular el conjunto de rutas en el VRPTW.

3.3 Técnicas e instrumentos de recolección de datos

Para la recolección de datos, se utilizó la técnica de análisis de datos secundarios, puesto que estos fueron suministrados y recolectados por la compañía del caso en estudio, estos datos son generados directamente de las aplicaciones internas de la organización y pueden ser consultados en tiempo real y ser descargados en cualquier rango o periodo de tiempo.

3.4 Técnicas de análisis de datos

La técnica que se utilizó para el procesamiento y análisis de datos fue la estadística descriptiva, la cual se aplicó con ayuda del software EasyFit para los datos de la compañía, inicialmente se analizaron descriptivamente los datos, se evaluó la confiabilidad de los resultados encontrados y se realizó los ajustes a las distribuciones encontradas, para finalmente presentar los resultados en tablas y figuras y poder dar inicio al desarrollo del AG.

3.5 Procedimiento para el Desarrollo

El procedimiento que se elaboró en este proyecto para encontrar una solución del VRPTW consistió:

- Realizar una descripción del contexto general de la empresa y caracterizar el sistema en estudio, identificando la problemática y causas fundamentales por medio de análisis estadístico de datos históricos.

- Describir y recopilar información referente a los VRP, métodos de solución y realizar un enfoque en los VRPTW para encontrar el método de solución que se adapte a las condiciones reales del problema en estudio.
- Definir el modelo matemático para el VRPTW, variables, parámetros, función objetivo y restricciones del sistema. Se utilizó la herramienta VRP de Excel con la cual se construyó una solución inicial y fue tomada como padres o rutas iniciales, teniendo en cuentas las condiciones de factibilidad definidas.
- Desarrollar por medio de lenguaje de programación en Python, un código que permite validar el algoritmo por medio de datos simulados de acuerdo a la distribución de datos reales, dentro de la construcción del código se realizaron operaciones de selección, cruza, mutación y reparación.

CAPÍTULO 4: SOLUCIÓN CASO VRPTW

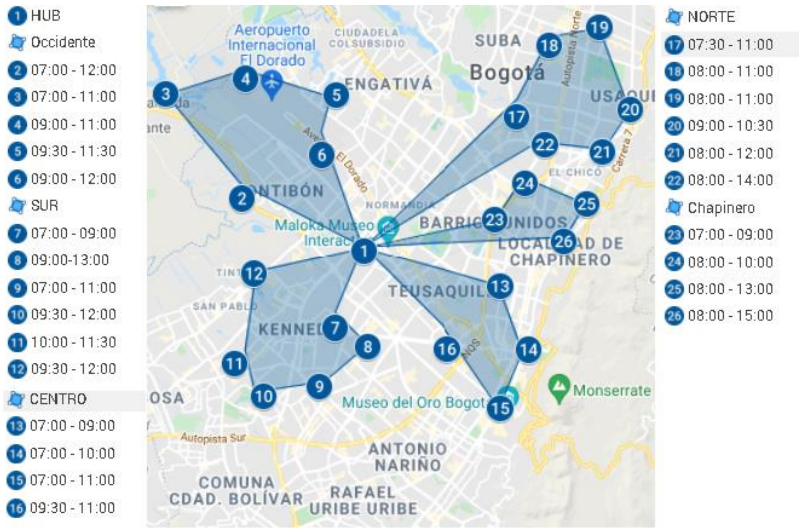
En esta sección se realiza una descripción detallada de la operación actual de la compañía, el modelo matemático que se tendrá en cuenta en el caso en estudio para el VRPTWPD y la selección del método metaheurístico para solución del problema, seguido del análisis estadístico para obtener las soluciones iniciales.

4.1 Descripción de la Operación

La operación logística se realiza en horario de 07:00 a 18:00 de lunes a viernes, puesto que, el 95% de los clientes trabaja en horario de oficina y en sus contratos se establece que el servicio de transporte se ofrece dentro de estas jornadas, ocasionalmente los fines de semana con una tarifa diferencial, el ruteo de vehículos se realiza de forma empírica sin tener en cuenta las restricciones del sistema, únicamente se establece dentro de polígonos en la ciudad de Bogotá, como limites demarcados por calles y carreras el recorrido a realizar por cada ruta.

Figura 9

Ejemplo modelo VRP por zonas con ventanas de tiempo



Nota: Fuente El autor.

En la ***Figura 9*** se pueden ver 5 zonas en la ciudad de Bogotá que agrupan un total de 26 clientes o nodos con sus respectivas ventanas de tiempo, también se puede observar el depósito de origen y llegada para cada ruta, este sistema de distribución describe la operación actual de la empresa. Un solo depósito en la zona industrial de Montevideo y un conjunto de 610 clientes ubicados en la ciudad de Bogotá que se deben atender de acuerdo con sus necesidades.

Los vehículos utilizados para la operación de Pickup and Delivery son vehículos tipo Carry, tipo Van y Camionetas con diferentes características en sus dimensiones y capacidades de carga, sin embargo, por la proyección de le empresa de renovar el parque automotor en el 2030 se tendrá en cuenta como un único vehículo la Renault Trafic – techo bajo, con una capacidad de carga de 1300 kg o 93 cajas de peso estándar, con una ocupación del 80% en su contenido.

Figura 10

Vehículo modelo de ruteo VRPTW Renault Trafic

Nota: Tomado de “Carro ya, Renault”

Cada vehículo es conducido por un único conductor, el cual es acompañado por su respectivo auxiliar, quien es el encargado de dar un orden a la ruta, verificar horarios de servicios y realizar gestión con clientes para agilizar el tiempo de ejecución de cada entrega o recolección, además es la persona responsable de ejecutar los servicios de transporte donde el cliente y quien captura a conformidad las firmas de aceptación, en esta operación logística el conductor y su auxiliar no tienen que manejar dinero por recolecciones de mercancía.

Para realiza el proceso de transportes de entrega o recolección de cajas la única guía que posee el auxiliar de la ruta es el manifiesto con todas las ordenes de servicio, donde en un apartado de instrucciones se escribe el horario en el cual el cliente solicita la recolección o si es necesario tener en cuenta alguna consideración de tiempo adicional, al auxiliar a su vez es el encargado de enrutar todos los servicios en su ruta de acuerdo con su experiencia.

4.2 Definición del modelo VRPTW

De acuerdo a la definición de (Montes Orozco, 2017) el modelo matemático del VRPTW se define de la siguiente manera:

a) Parámetros:

G = (Cl, A); Gráfica completa no dirigida.

Cl = $\{\mathbf{Cl_0}\ \mathbf{Cl_1}, \mathbf{Cl_2}, ..., \mathbf{Cl_n}\}$; Conjunto de nodos (clientes), donde $\mathbf{Cl_0}$ es el nodo depósito.

A = (i, j): i, j $\in$ cl, i $\neq$ j; Conjunto de aristas que unen a todos los clientes i y j.

C = $(\mathbf{C_{ij}})$: Costo de moverse desde el cliente i hacia el cliente j.

$\mathbf{D_i}$ = Demanda del cliente i.

K = Número de vehículos pertenecientes a la flota.

Q = Capacidad de los vehículos.

$\mathbf{Y_{iv}}$ = Cantidad recogida por el vehículo v al cliente i.

$\mathbf{B_{iv}}$ = Hora de llegada del vehículo v al cliente i.

b) Variables de decisión:

$$\mathbf{X_{ij}^{v}} \begin{cases} \mathbf{1} \\ \mathbf{0} \end{cases} \begin{matrix} \mathbf{Si\ un\ vehículo\ v\ recorre\ el\ arco\ (i,j)} \\ \mathbf{En\ caso\ contrario} \end{matrix}$$

Función objetivo:

Representa la distancia asociada a los procesos de recolección.

$$\mathbf{Min\ Z} = \sum_{\mathbf{i=0}}^{\mathbf{n}} \sum_{\mathbf{j=0}}^{\mathbf{n}} \sum_{\mathbf{v=1}}^{\mathbf{k}} \mathbf{C_{ij} X_{ij}^{v}} \tag{1}$$

Restricciones:

Cada cliente debe ser visitado por un único vehículo:

$$\sum_{\mathbf{i=0}}^{\mathbf{n}} \sum_{\mathbf{v=1}}^{\mathbf{k}} \mathbf{X_{ij}^{v} = 1;\ \forall\ j = 1, \dots, n} \tag{2}$$

Cada vehículo siempre inicia su ruta en el depósito:

$$\sum_{\mathbf{j=1}}^{\mathbf{n}} \mathbf{X_{0j}^{v} = 1;\ \forall\ v = 1, \dots, K} \tag{3}$$

Cada vehículo finaliza recorrido en el depósito:

$$\sum_{\mathbf{i=1}}^{\mathbf{n}} \mathbf{X_{i0}^{v} = 1;\ \forall\ v = 1, \dots, K} \tag{4}$$

Cada vehículo debe salir del cliente al cual ingreso:

$$\sum_{i=0}^{n} X_{ip}^{v} - \sum_{j=o}^{n} X_{pj}^{v} = 0;\ \forall\, p = 0, \dots, n; v = 1, \dots, k \tag{5}$$

La cantidad entregada al cliente i por el vehículo v sea satisfecha en su totalidad:

$$Y_{iv} = d_i \sum_{i=0}^{n} X_{ji}^{v};\ \forall\, i = 1, \dots, n; v = 1, \dots, k \tag{6}$$

Cantidad recogida en cada ruta no exceda la capacidad del vehículo:

$$\sum_{i=0}^{n} Y_{iv \le Q}\ ;\ \forall\, V = 1, \dots, K \tag{7}$$

Las ventanas de tiempo de cada cliente son respetadas:

$$e_i \le b_i^v \le l_i\ \forall\, i, j = 0, \dots, n; v = 1, \dots, k \tag{8}$$

Asegurar que cada vehículo v no pueda iniciar el servicio a cierto cliente j, si la suma del tiempo de transporte entre los clientes i y j, la duración del servicio para el cliente i y el tiempo de llegada al cliente i, es mayor que la ventana de tiempo para el cliente j.

$$X_{ij}^{v}\,(b_i^v + s_i + t_{ij} - b_j^v \le 0\ \forall\, i = 1, 2, \dots, n; v = 1, 2, \dots, k \tag{9}$$

Define el tipo de las variables utilizada

$$X_{ij}^{v} \in \{0.1\} y\, Y_{iv} + \ge 0; \forall i = 1, \dots, n; v = 1, \dots, k \tag{10}$$

En esta variante del problema, como se detalla en la sección anterior, además de capacidades sobre los vehículos, cada cliente $Cl_i \in cl \backslash Cl_0$ tiene asociada una ventana de tiempo $[e_i, l_i]$ que establece un horario de servicio permitido para que un vehículo llegue a él y un tiempo de servicio o demora, por ejemplo, si el camino entre los clientes (i, j) es parte de una solución, b_i y b_j son las horas de llegada a los clientes i y j respectivamente, entonces las ventanas de tiempo implican que debe cumplirse $b_i \le l_i$ y $b_j \le l_j$. además, se consideran las

constantes $\mathbf{t_{ij}}$, que tienen como valor, la distancia existente entre los clientes i y j.(Montes Orozco, 2017)

4.3 Selección de Método de Solución Algoritmo Genético

De acuerdo con la revisión de la literatura los problemas de ruteo de vehículos son considerados como problemas de difícil solución puesto que requiere altos recursos computacionales para encontrar una respuesta, es por este motivo que la mayoría de los autores utilizan técnicas heurísticas y metaheurísticas para encontrar soluciones en tiempos más cortos que las que ofrece un método exacto. Por lo anterior y fundamentado en la cantidad de recursos computacionales y la magnitud de este problema, se toma la decisión de utilizar la metaheurística un algoritmo genético, por su practicidad en los VRPTW y su fácil entendimiento al momento de su desarrollo.

Los algoritmos genéticos son técnicas de optimización basadas en procesos evolutivos y cambios en las estructuras genéticas de seres vivos, en la naturaleza se almacena información genética de un ser en un cromosoma por medio de genes y estos genes se heredan a su descendencia. Los individuos con buenos genes sobrevivirán con el tiempo y los individuos con malos genes morirán, lo que conlleva a que cada generación sea más fuerte y se elimine una generación anterior. Este concepto es trasladado al VRP como una estrategia para encontrar un conjunto de soluciones mientras se borrar soluciones existentes que no son tan fuertes de acuerdo con el objetivo del problema. (Kurnia et al., 2018).

Tabla 4

Referencias trabajaos VRP- Tiempo

Autor	Proyecto	Método de Solución
Hideki Hashimoto *et al*	En iterated local search algorithm for the time-dependent vehicle routing problem with time windows	2 Opr Or Opt
Segura Peñuela Ángel Padua Dueñas Andrea	Modelo para la solución de un problema de ruteo de vehículos con capacidad y ventanas de tiempo, en el servicio de transporte de canje y correo bancario	Algoritmo de Barrido
Daniel Quagliaroli	Desarrollo de un Algoritmo Genético Culturar para el Problema de Enrutamiento de Vehículos con Ventanas de Tiempo.	Algoritmo Genético
Catalina Segura Londoño	Análisis y Prototipado de un Algoritmo Genético Modificado para Solucionar el Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRPTW), Prioridad de Metas Económicas Y Componente Medio Ambiental	Algoritmo Genético Modificado
Fredy Guasmayan	Solución del Problema de Ruteo De Vehículos Dependientes del Tiempo Utilizando un Algoritmo Genético Modificado	Algoritmo Genético Modificado
Brian Velazco	Diseño de Un Modelo de Ruteo de Vehículos Dependiente del Tiempo en una Zona Urbana de Bogotá	Algoritmo Genético Modificado
Wen-Chyuan Chiang and Robert A. Russell	Simulated annealing metaheuristics for the vehicle routing problem with time windows	Búsqueda Tabú
Jorg Homberger Hermann Gehring	A two-phase hybrid metaheuristic for the vehicle routing problem with time windows	Búsqueda Tabú
Nasser A. El-Sherbeny	Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods	Búsqueda Tabú Algoritmos Genéticos

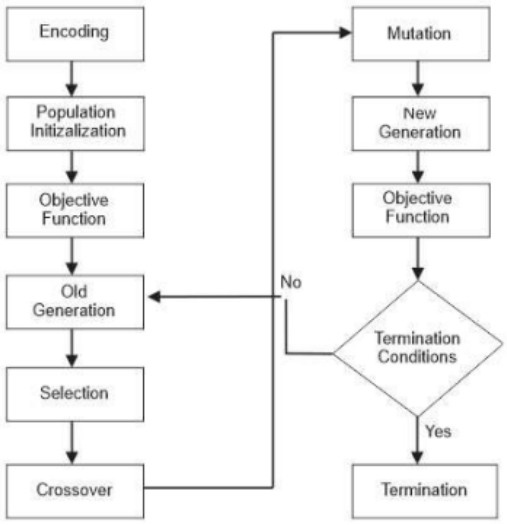
Roberto Cordone	A Heuristic for The Vehicle Routing Problem With Time Windows	Búsqueda Tabú Algoritmos Genéticos
Vitória Pureza, *et al*	Vehicle routing with multiple deliverymen: Modeling and heuristic approaches for the VRPTW	Búsqueda Tabú Colonia de Hormigas
Alberto Donati, *et al*	Time dependent vehicle routing problem with a multi ant colony system	Colonias de Hormigas
Jorge Enrique Restrepo	A logistic case of programming vehicle routing problem with time windows	Heuristica R
Marshall L Fisher	Vehicle Routing with Time Windows: Two Optimization Algorithms	Técnica de Relajación

Nota: Fuente El autor

La ***Tabla 4*** se presenta un breve resumen de trabajos realizados para resolver los problemas de VRP relacionados con el tiempo y donde se puede observar que la mayoría de estos autores utilizan técnicas metaheurísticas para la solución de este problema, teniendo como referencia estos trabajos, por la practicidad y facilidad de entendimiento del algoritmo, se desarrollada un Algoritmo Genético para el VRPTW.

La ***Figura 11*** describe de forma general los procesos que ocurren en algoritmos genéticos, comenzando con una etapa de codificación e iniciación de una población y estrategias evolutivas como: selección, cruce, mutación y en algunos casos reparación, en la etapa de selección se determinan que individuos sobrevivirán, el cruce produce nuevos individuos que reemplazan los muertos y permite la aparición de nuevos individuos para encontrar una nueva solución.

Figura 11
Flujo del algoritmo genético



Nota: Tomado de Hari Kurnia et al (2018), Genetic Algorithm with Multi Compartment

En la ***Figura 11*** se puede ver el flujo para el desarrollo de un algoritmo genético, partiendo de una población inicial y una función objetivo luego se realizan diferentes operaciones genéticas para terminar el proceso si mejora la aptitud de los individuos.

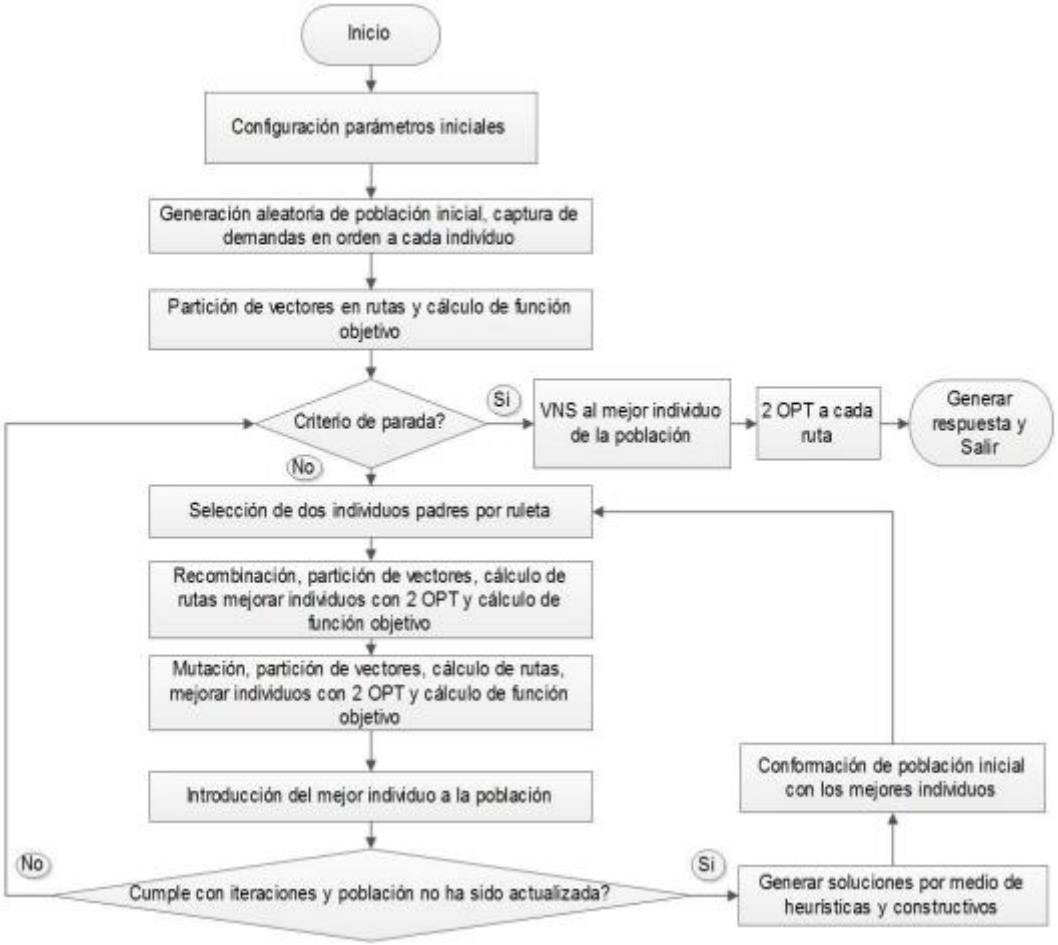
De acuerdo al trabajo realizado por (Gusmayan Fredy, 2014), se puede abordar el AG para el VRPTW de una forma más detallada con la siguiente secuencia de pasos:

- Paso 1: Generación de población inicial.
- Paso 2: Partición en vectores o rutas de acuerdo con las restricciones del problema.
- Paso 3: Cálculo de la función objetivo.
- Paso 4: Seleccionar padres.
- Paso 5: Realizar recombinaciones de padres (Cruce)
- Paso 6: Aplicar procedimiento de mutación y calcular la F.O

- Paso 7: Decidir que individuo ingresa y cual sale de la población.
- Paso 8: Reiniciar interacciones y reemplazar individuos.
- Paso 9: Elegir la mejor respuesta
- Paso 10: Realiza un proceso de reparación en caso de que alguna solución no sea factible.

Figura 12

Diagrama de flujo detallado, algoritmo genético para el VRPTW



Nota: Tomado de Fredy Alexander Guasmaya (2014), Problema De Ruteo De Vehículos Dependientes Del Tiempo Utilizando Un Algoritmo Genético Modificado

En ***Figura 12*** se puede observar el diagrama de flujo detallado del procedimiento para el desarrollo de un algoritmo genético para dar solución a un VRPTW, partiendo de una población inicial la cual va mejorando de acuerdo con diferentes operaciones e iteraciones, este procedimiento será utilizado como ejemplo para el desarrollo del AG propuesto en este proyecto.

Para la construcción de la población inicial se realizó un análisis estadístico de la demanda de la empresa, específicamente cantidad de clientes visitados diariamente con sus respectivos volúmenes de trabajo para determinar por medió de distribuciones de probabilidad cual iba a ser el tamaño de la población o nodos del sistema de distribución.

4.4 Análisis Estadístico de Datos Históricos

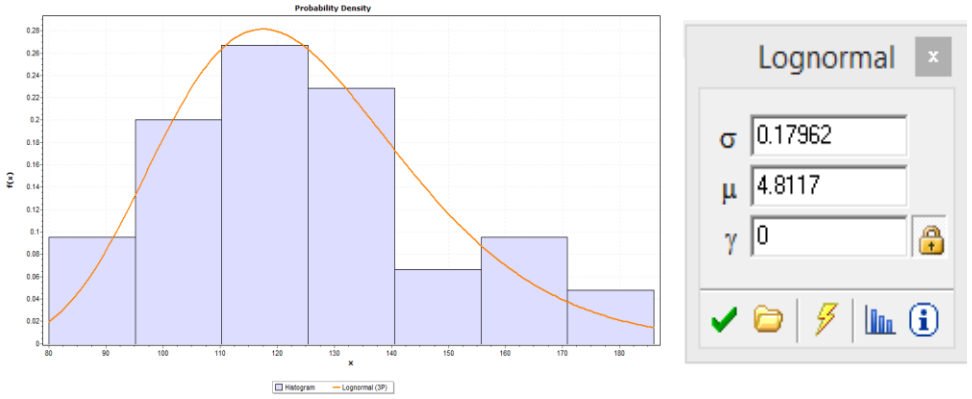
Se proporcionó por parte de la empresa una base de datos que contiene registro de cinco meses desde mayo de 2021 hasta septiembre de 2021 con información de órdenes de trabajo de recolección y entrega de material suficiente para empezar a entender el comportamiento de la demanda. desde que inicio la emergencia sanitaria del COVID19 en Colombia en marzo de 2020, solo hasta finales de abril de 2021 la empresa evidencio una recuperación de volumen de trabajo del 95% y por este motivo se seleccionó este periodo de tiempo, a continuación, se describen los datos suministrados.

- Dirección de cada cliente, con la cual se calculan las coordenadas geográficas, distancias y tiempo de viaje entre ellos con el archivo “herramienta VRP en Excel”.
- Ventanas de atención de cada cliente.
- Demanda de cada cliente
- Datos de órdenes de trabajo, frecuencias de solicitud.

Se realizó una limpieza inicial de datos en Excel y posterior se realizaron pruebas de bondad de ajuste en el software Easyfit para determinar la distribución de probabilidad de la variable paradas dentro de las 23 distribuciones que permite evaluar el software, encontrando como resultado la distribución log normal de acuerdo con las características de la demanda como se evidencia en ***Figura 13***.

Figura 13

Distribución y parámetros de probabilidad para variables paradas

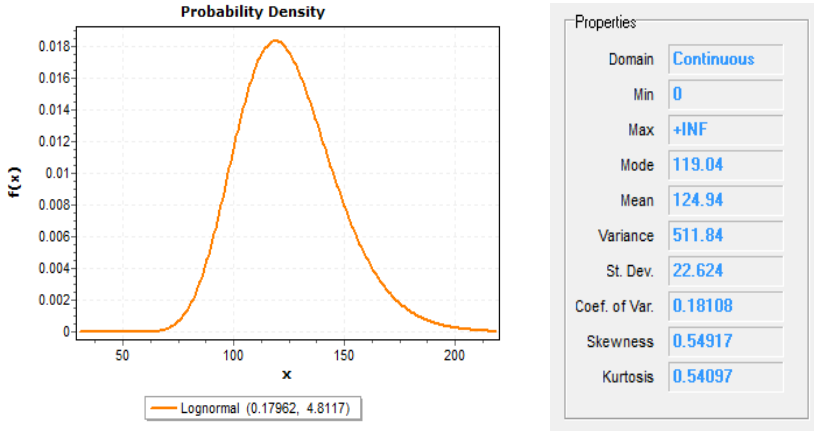


Nota: Fuente El autor, realizado en EasyFit

Por medio del software Easyfit posterior a la prueba de bondad de ajuste se realizaron tres pruebas estadísticas no paramétricas para determinar si los datos corresponden a la distribución de probabilidad específica, encontrando los resultados presentados en él **Anexo 1** donde se puede determinar de acuerdo con la prueba estadística de Kolmogorov, Anderson Darling y Chi cuadrado la distribución de probabilidad log – normal se acepta por el comportamiento de los datos.

Figura 14

Distribución de probabilidad y parámetros de paradas diarias ajustada



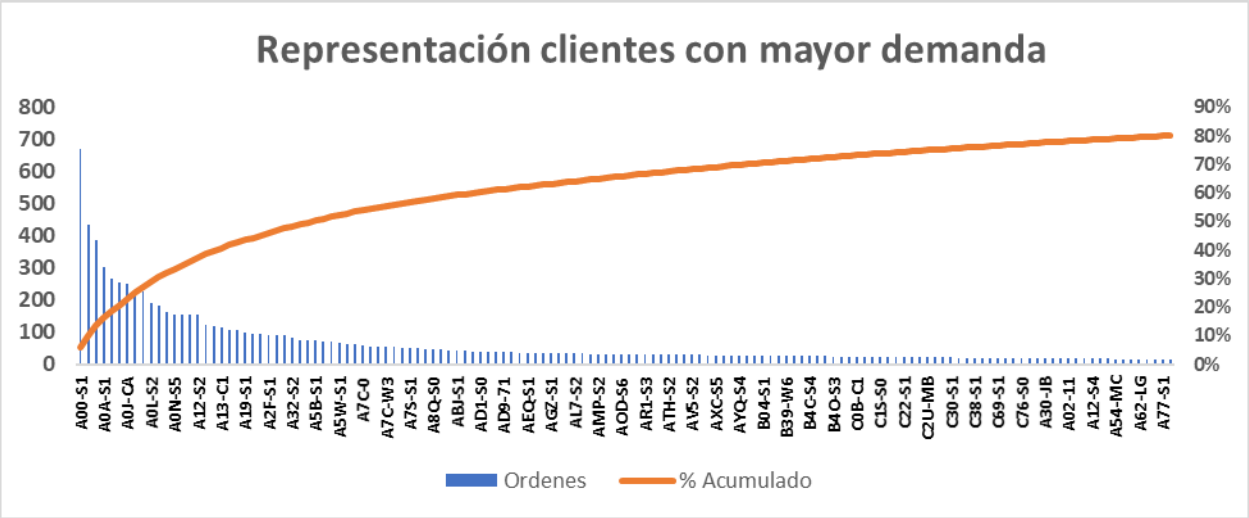
Nota: Fuente El autor, realizado en EasyFit

En la ***Figura 14*** se muestran los parámetros suministrados por Easyfit luego de realizar el ajuste de la probabilidad y se determina de acuerdo con la distribución de datos y parámetros encontrados, que la media de paradas será de 125 cliente, con una desviación de 23 clientes.

1. Para determinar la cantidad de clientes o paradas a simular dentro del modelo del caso en estudio, se realizó un análisis por medio de diagrama de Pareto con 610 clientes, donde se determinó que 140 clientes representan el 80% de las solicitudes diarias, en la ***Figura 15*** se puede ver en el eje X, los ID o código de identificación de cada cliente y en el eje Y, la cantidad de órdenes de trabajo generadas para cada uno de ellos.

Figura 15

Análisis de Pareto clientes



Nota: Fuente El autor.

En la ***Figura 15*** se observa una muestra de 48 clientes dentro del análisis de Pareto para un total de 140 clientes seleccionados en una base de 610 clientes.

Tabla 5

Cantidad de clientes a simular

Clientes	Total	Pareto 80% Solicitudes	Pareto 20% Solicitudes
	610	140	470

Nota: Fuente El autor.

Se encontró dentro del análisis de Pareto que 140 clientes representan el 80% del volumen de trabajo, por este motivo se toma dentro de este proyecto este dato de clientes como base para la simulación del programa, además considerando la distribución de probabilidad de paradas y sus

parámetros con media de 125 y una desviación de 22, el valor ingresa dentro de los parámetros establecidos.

4.5 Generación de Rutas Iniciales Herramienta "VRP en Excel"

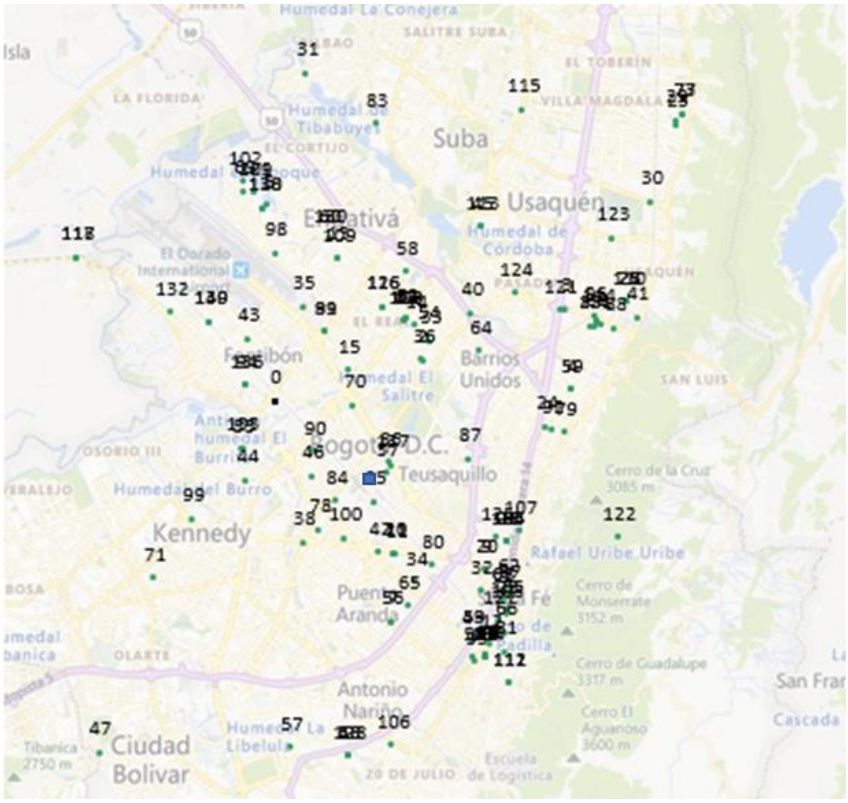
El VRPTW es considerado dentro de la familia de VRP un problema NP- Hard de difícil solución, por esto es necesario desarrollar una metodología que permita llegar a una solución eficaz del mismo, en este caso se construirá un algoritmo genético como herramienta metaheurística para encontrar un conjunto de rutas, sin embargo, se establece como solución inicial la encontrada con la herramienta "vrp_spreadsheet_solver_v3.67" de Günes Erdogan luego de realizar 10 simulaciones con demandas aleatorias que cumplen una distribución de probabilidad para cada cliente, los cuales se tomarán como padres para el inicio del algoritmo genético.

1. Simulación inicial archivo: "herramienta VRP en Excel"

Se realizó un análisis de distribuciones de probabilidad en Excel, con la demanda de cada uno de los 140 clientes seleccionados en el periodo de 4 meses, encontrando los parámetros de la distribución para cada uno, posterior se realizó una simulación para 10 escenarios con los parámetros encontrados y números aleatorios para tener el insumo y proceder con la ejecución del VRP de Excel, en el ***Anexo 2*** se evidencia un ejemplo de los datos a simular en el archivo "herramienta VRP en Excel"

Figura 16

Conjunto de cliente, Bing Maps



Nota: Fuente: Extraído de *VRP_Spreadsheet_Solver_v3.72*

En la ***Figura 16*** se puede observar la distribución del clientes en la ciudad de Bogota de acuerdo a sus coordenadas geograficas utilizando la herramienta de VRP en Excel que serán objeto de estudio, como codigo de identificaciòn se deja el número del cliente en el orden por filas dentro de la tabla de simulación.

2. El archivo "*vrp_spreadsheet_solver_v3.72*", se establecen los de forma inicial los parámetros descritos en la ***Tabla 6***

Tabla 6

Parámetros iniciales VRPTW

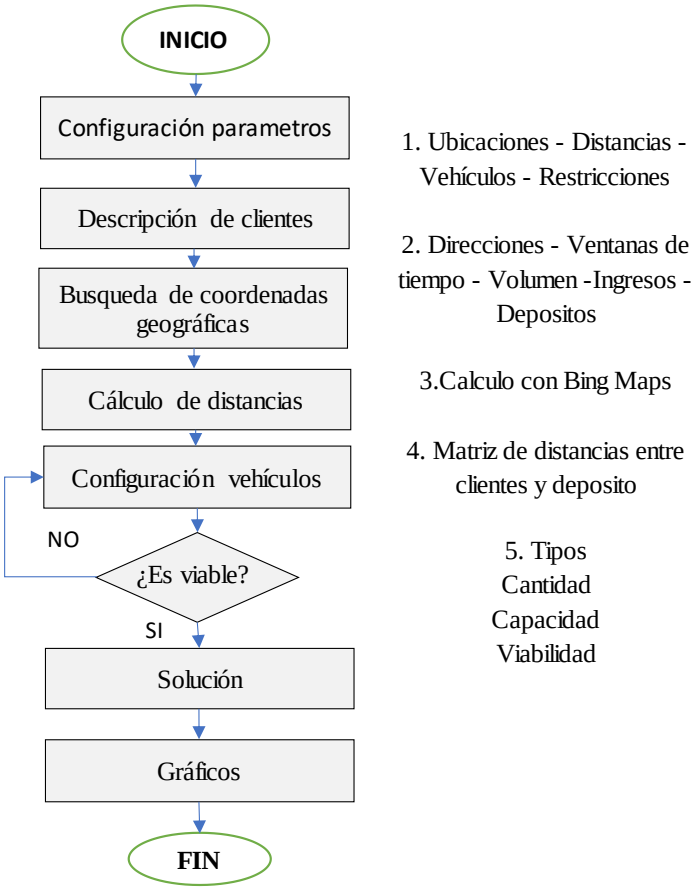
Parámetro	Valor	Observaciones
Número de depósitos	1	En zona industrial de Montevideo
Número de clientes	140	Distribución de probabilidad, Pareto
Método de cálculo de distancia	Bing Maps	Calculo automático
Velocidad media del vehículo	37.5 km/h	De acuerdo con el ministerio de transporte
Tipo de vehículos	1	Renault Trafic techo bajo
¿Los vehículos regresan al depósito?	sí	Solo una vez al finalizar el día y la ruta
Tipo de ventana de tiempo	Difícil	No se pueden atender servicios fuera
Límite de tiempo de la CPU (segundos)	1680	Cálculo matemático por cantidad de nodos

Nota: Fuente El autor.

3. Representación del flujo del proceso en el archivo "*vrp_spreadsheet_solver_v3.72*", para cálculo de rutas, se configuran parámetros iniciales como, cantidad de clientes, métodos de cálculo de distancias, depósitos y restricciones, posterior se realiza una descripción de clientes con sus respectivas direcciones, ventanas de horario de atención, importes de unidades a entregar y/o a recoger y el valor del ingreso de visitar al cliente para prestar el servicio, se realiza un cálculo de coordenadas con ayuda de la herramienta Bing Maps para luego elaborar una matriz de distancias entre todos los nodos de la red. para finalizar se parametrizan las capacidades de los vehículos y se soluciona el problema.

Figura 17

Flujo macro funcionamiento "vrp_spreadsheet_solver_v3.72"



Nota: Fuente El autor.

4.6 Resultados Iniciales Utilizando Archivo "*vrp_spreadsheet_solver_v3.72*"

Luego de realizar la simulación 10 veces con grupos de datos aleatorias para 140 clientes, se utilizaron en promedio 12 vehículos para satisfacer la demanda total de la red de nodos para este problema, por lo tanto, la solución estará establecida por este número de vehículos y cada uno realizará una ruta especifica sin repetir clientes, iniciando el depósito y regresando a él una sola vez finalizando la ruta.

Las rutas asignadas son:

Tabla 7
Simulación 1, solución inicial VRPTW

Vehículo	V1	V2	V3	V4	V5	V6	V7	V8	V9	V10	V11	V12
Paradas	14	12	14	14	14	11	6	16	10	16	11	14
Cantidad Und	88	80	85	80	80	86	30	69	67	83	88	86
Distancia recorrida (Km)	68,43	104,95	197,06	137,24	119,79	87,18	27,3	132,49	146,85	161,25	143,48	238,85
Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito	Depósito
Parada 1	56	52	47	20	94	137	53	99	85	51	110	104
Parada 2	31	130	127	140	117	128	97	132	86	25	22	75
Parada 3	95	81	109	69	134	71	80	41	79	88	24	90
Parada 4	62	27	67	14	123	48	135	138	38	7	30	126
Parada 5	60	17	1	13	70	98	64	129	108	68	83	57
Parada 6	5	49	82	34	115	18	Depósito	72	36	131	28	63
Parada 7	35	116	39	112	44	2		26	58	91	100	55
Parada 8	124	29	105	102	19	10		136	125	84	120	119
Parada 9	87	3	12	121	133	15		66	78	92	11	9
Parada 10	45	42	74	93	21	114		50	Depósito	8	46	43
Parada 11	23	96	40	106	37	Depósito		33		32	Depósito	6
Parada 12	4	Depósito	103	61	101			65		118		73
Parada 13	113		139	107	76			89		54		122
Parada 14	Depósito		Depósito	Depósito	Depósito			111		59		Depósito
Parada 15								77		16		
Parada 16								Depósito		Depósito		

Nota: Fuente El autor.

En la ***Tabla 7*** se puede observar la cantidad de clientes asignados a cada ruta para un total de 140 paradas, cada ruta empieza en el depósito y finaliza en él, en total se obtiene una solución con 1566,8 km recorridos con aplicación de ventanas de tiempo en Excel, la cual será mejorada aplicando algoritmos genéticos para dar un orden lógico a cada vehículo dentro de su ruta especifica.

Tabla 8
Resumen de 10 soluciones simulados

Simulación	1	2	3	4	5	6	7	8	9	10
Paradas	140	140	140	140	140	140	140	140	140	140
Cantidad Und	912	922	917	911	922	912	929	896	899	922
Distancia recorrida (Km)	1566,81	2158,80	1542,13	1505,54	1532,63	1598,79	1494,66	1622,81	1462,04	1586,89
Cantidad Rutas	12	11	16	13	13	14	13	14	15	14

Nota: Fuente El autor

En la ***Tabla 8*** se muestra el resumen de los datos encontrados en las 10 simulaciones realizadas con demandas aleatorias, cantidad de rutas y kilómetros recorridos totales, obteniendo como mejor resultado la novena simulación con distancia recorrida de 1462,04 Kilómetros y una utilización de 15 vehículos, como peor solución se encontró la simulación número dos con un total de 11 rutas y una distancia de 2158 kilómetros.

Con base en los resultados encontrados durante la simulación de datos, se selecciona la solución con la menor distancia recorrida para aplicar algoritmos genéticos, las cuales se tomarán como posible solución del problema y serán la base para generar hijos y ser evaluados.

4.7 Procedimiento para Generación de Población Inicial en AG

Para la creación de individuos o población inicial, se utilizó el archivo de Excel VRP, en una red de 141 nodos, un depósito desde el cual salen y retornan todos los vehículos y un total de 140 clientes, se encontraron 15 rutas o individuos con diferentes clientes de forma constructiva. Con base en lo anterior se determina que un individuo estará compuesto por un conjunto de genes o clientes. La ***Tabla 9*** muestra los quince individuos generados los cuales cumplen las restricciones del sistema, sin violar las ventanas de tiempo o capacidad de los vehículos

Tabla 9

Representación de individuos – Rutas

I1	I2	I3	I4	I5	I6	I7	I8	I9	I10	I11	I12	I13	I14	I15
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
47	91	41	99	56	6	137	20	51	94	52	109	129	34	23
127	117	62	9	31	12	66	84	25	68	118	67	74	130	33
36	39	134	116	95	60	120	92	1	132	89	63	105	125	54
83	108	14	10	88	5	53	7	35	126	55	87	13	78	75
28	85	22	15	104	128	97	140	124	0	119	26	18	0	0
100	44	49	43	4	71	80	86	40		37	93	21		
101	57	17	76	114	48	107	2	27		98	19	73		
131	50	110	69	70	139	0	0	58		11	133	24		
61	46	45	103	90	0			121		123	82	136		
29	0	138	115	135				38		0	30	79		
106		72	64	65				111			112	8		
59		0	0	3				77			102	32		
16				42				0			81	113		
0				96							122	0		
				0							0			

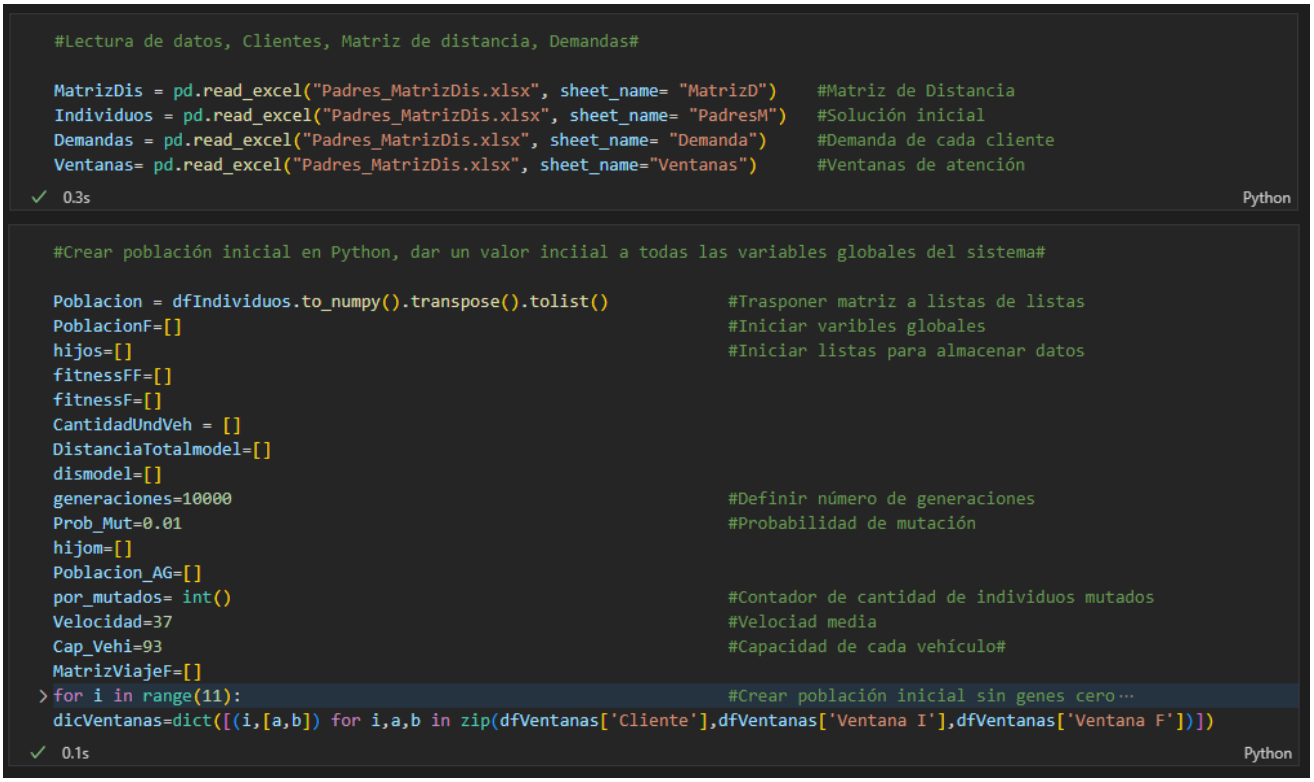
Nota: Fuente El autor

En la ***Tabla 9*** se puede observar un grupo de 15 individuos o rutas los cuales deben visitar un grupo de clientes, el individuo uno (I1) visita un total de 13 clientes, el individuo 7 (I7) visita 7 clientes y así sucesivamente hasta visitar los 140 clientes con las rutas generadas, este grupo de 15 individuos se conoce como población inicial dentro del AG.

Por medio del lenguaje de programación Python en el editor de código Visual Studio Code, se convierte el archivo de Excel con la población inicial en una lista compuesta por 15 sub listas, para luego ser transformadas en la población final e iniciar los procesos de selección, cruce, mutación y reparación. A continuación, se expone una parte del código realizado.

Figura 18

Lectura de datos en Visual Code Python



```
#Lectura de datos, Clientes, Matriz de distancia, Demandas#

MatrizDis = pd.read_excel("Padres_MatrizDis.xlsx", sheet_name= "MatrizD")    #Matriz de Distancia
Individuos = pd.read_excel("Padres_MatrizDis.xlsx", sheet_name= "PadresM")   #Solución inicial
Demandas = pd.read_excel("Padres_MatrizDis.xlsx", sheet_name= "Demanda")     #Demanda de cada cliente
Ventanas= pd.read_excel("Padres_MatrizDis.xlsx", sheet_name="Ventanas")      #Ventanas de atención
```

✓ 0.3s Python

```
#Crear población inicial en Python, dar un valor inciial a todas las variables globales del sistema#

Poblacion = dfIndividuos.to_numpy().transpose().tolist()          #Trasponer matriz a listas de listas
PoblacionF=[]                                                     #Iniciar varibles globales
hijos=[]                                                          #Iniciar listas para almacenar datos
fitnessFF=[]
fitnessF=[]
CantidadUndVeh = []
DistanciaTotalmodel=[]
dismodel=[]
generaciones=10000                                                #Definir número de generaciones
Prob_Mut=0.01                                                     #Probabilidad de mutación
hijom=[]
Poblacion_AG=[]
por_mutados= int()                                                #Contador de cantidad de individuos mutados
Velocidad=37                                                      #Velociad media
Cap_Vehi=93                                                       #Capacidad de cada vehículo#
MatrizViajeF=[]
for i in range(11):                                               #Crear población inicial sin genes cero···
dicVentanas=dict([(i,[a,b]) for i,a,b in zip(dfVentanas['Cliente'],dfVentanas['Ventana I'],dfVentanas['Ventana F'])])
```

✓ 0.1s Python

Nota: Fuente El autor, generados en Visual Studio Code

La ***Figura 18*** muestra el procedimiento para realizar lectura de datos y generar una población inicial teniendo como base la solución proporcionada por la herramienta VRP en Excel.

Figura 19
Población inicial generada en Visual Studio Code

```
[[0.0, 47.0, 127.0, 36.0, 83.0, 28.0, 100.0, 101.0, 131.0, 61.0, 29.0, 106.0, 59.0, 16.0, 0.0], [0.0, 91.0, 117.0, 39.0, 108.0,
85.0, 44.0, 57.0, 50.0, 46.0, 0.0], [0.0, 41.0, 62.0, 134.0, 14.0, 22.0, 49.0, 17.0, 110.0, 45.0, 138.0, 72.0, 0.0], [0.0, 99.0,
9.0, 116.0, 10.0, 15.0, 43.0, 76.0, 69.0, 103.0, 115.0, 64.0, 0.0], [0.0, 56.0, 31.0, 95.0, 88.0, 104.0, 4.0, 114.0, 70.0, 90.0,
135.0, 65.0, 3.0, 42.0, 96.0, 0.0], [0.0, 6.0, 12.0, 60.0, 5.0, 128.0, 71.0, 48.0, 139.0, 0.0], [0.0, 137.0, 66.0, 120.0, 53.0,
97.0, 80.0, 107.0, 0.0], [0.0, 20.0, 84.0, 92.0, 7.0, 140.0, 86.0, 2.0, 0.0], [0.0, 51.0, 25.0, 1.0, 35.0, 124.0, 40.0, 27.0,
58.0, 121.0, 38.0, 111.0, 77.0, 0.0], [0.0, 94.0, 68.0, 132.0, 126.0, 0.0], [0.0, 52.0, 118.0, 89.0, 55.0, 119.0, 37.0, 98.0,
11.0, 123.0, 0.0], [0.0, 109.0, 67.0, 63.0, 87.0, 26.0, 93.0, 19.0, 133.0, 82.0, 30.0, 112.0, 102.0, 81.0, 122.0, 0.0], [0.0,
129.0, 74.0, 105.0, 13.0, 18.0, 21.0, 73.0, 24.0, 136.0, 79.0, 8.0, 32.0, 113.0, 0.0], [0.0, 34.0, 130.0, 125.0, 78.0, 0.0],
[0.0, 23.0, 33.0, 54.0, 75.0, 0.0]]
```

Nota: Fuente El autor, generados Visual Studio Code

La ***Figura 19*** muestra una lista compuesta por 15 individuos generados con sus respectivos genes, los cuales en sus alelos o posición dentro de la cadena tienen un valor único, el cual identifica el número o ID de cada cliente.

Figura 20

Representación de un individuo.

0	56	31	95	62	60	5	35	124	87	45	23	4	113	0

Nota: Fuente El autor

La ***Figura 20*** indica que el individuo o vehículo número 1, parte del nodo 1, siendo este el depósito hacia el cliente número 56, luego parte hacia el cliente número 31, repitiendo este procedimiento, hasta llegar al cliente número 113, o gen número 14, cuyo alelo tiene valor 113, para finalmente retornar al depósito.

4.8 Selección de individuos en AG

Dentro del desarrollo de algoritmos genéticos es fundamental realizar un proceso probabilístico de selección de individuos para determinar cuales se reproducirán, de tal forma que todos tengan oportunidad de reproducirse inclusive siendo los menos aptos. Una de la técnica más conocida es la **selección por ruleta,** en la cual se escogen diferentes individuos de acuerdo con su contribución de aptitud sobre el total de la población, con esto se garantiza que puedan ser seleccionados inclusive los menos aptos y evitar convergencia prematura, sin embargo, el mayor problema de esta técnica es que el peor individuo puede ser seleccionado varias veces.

Para mejorar el desempeño del AG se tendrá en cuenta bajo un criterio elitista la selección de individuos a reproducirse, donde se pueden segregar individuos con valores altos de aptitud para permitir descendencia, “La técnica permite trasladar el mejor individuo de la población inicial a la siguiente generación, así que si bien esta no mejora, por lo menos mantiene el mismo nivel de aptitud que la anterior” (Velazco, 2019, p.74).

Para el desarrollo de este AG se define que los individuos de la población inicial o también conocidos padres serán seleccionados de forma aleatoria para para realizar las operaciones de reproducción llamadas cruce y mutación

Figura 21

Familia número 1, familia número 2

I5	52	47	104	78	140	17	110	58	125	98	2
I6	129	75	100	120	13	119	81	30	48	135	116

I7	85	55	27	6	21	57	126	63	130	31	115	5	86	76
I9	127	4	19	133	20	124	44	37	69	132	40	134	123	77

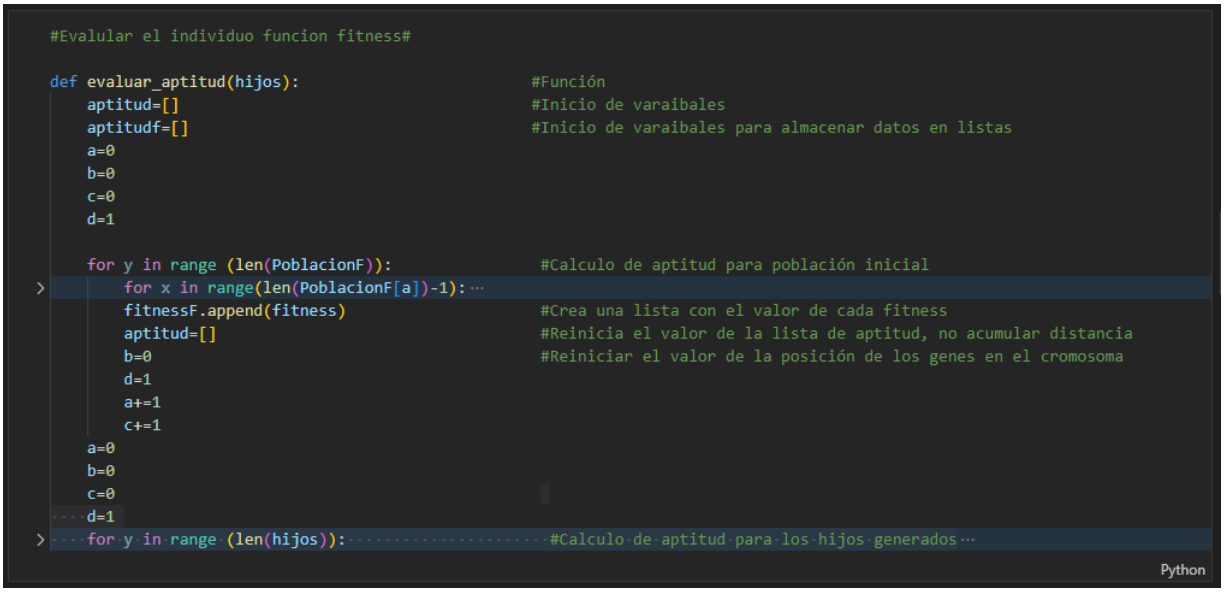
Nota: Fuente El autor

En la ***Figura 21*** se puede observar la familia número uno y número dos cada una con dos padres la primera familia con 11 genes y la familia número dos, con un total de 14 genes los cuales representan la ruta a seguir para cada uno.

A continuación, se expone una parte del código desarrollado para la función que evalúa la aptitud de cada individuo seleccionado en cada generación:

Figura 22

Función para evaluar aptitud de cada individuo



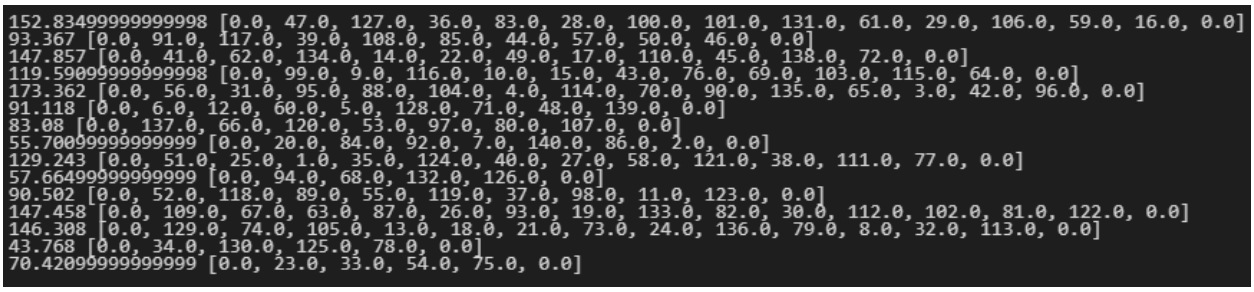
```python
#Evalular el individuo funcion fitness#

def evaluar_aptitud(hijos):                          #Función
    aptitud=[]                                       #Inicio de varaibales
    aptitudf=[]                                      #Inicio de varaibales para almacenar datos en listas
    a=0
    b=0
    c=0
    d=1

    for y in range (len(PoblacionF)):                #Calculo de aptitud para población inicial
        for x in range(len(PoblacionF[a])-1):···
        fitnessF.append(fitness)                     #Crea una lista con el valor de cada fitness
        aptitud=[]                                   #Reinicia el valor de la lista de aptitud, no acumular distancia
        b=0                                          #Reiniciar el valor de la posición de los genes en el cromosoma
        d=1
        a+=1
        c+=1
    a=0
    b=0
    c=0
    d=1
    for y in range (len(hijos)):·····················#Calculo de aptitud para los hijos generados···
```

Python

Nota: Fuente El autor, generados Visual Studio Code

En la ***Figura 22*** se calcula la aptitud de cada individuo teniendo como criterio la distancia recorrida de cada ruta , los individuos con mejor aptitud serán aquellos que tengan la menor distancia recorrida ya que lo que busca el desarrollo del AG es minimizar esta variable en cada ruta para obtener un beneficio económico, a su vez la función de avaluación de aptitud, evalúa el fitness de cada hijo generado para determinar si son mejores soluciones para ser almacenadas en la matriz de población final.

Figura 23
Aptitud calculada de cada individuo



Nota: Fuente El autor, generados Visual Studio Code

En la ***Figura 23*** se muestra el valor de la aptitud para cada individuo, teniendo como menor distancia recorrida 43,76 kilómetros la cual corresponde al individuo número 14 quien tiene 4 genes o clientes, este valor de aptitud es obtenido de la solución inicial de individuos y recalculado en cada generación.

4.9 Función de adaptación para AG

Para la función de adaptación se tendrá en cuenta la demanda de cada cliente y la capacidad de cada vehículo, donde la demanda del total de clientes visitados debe ser menor a la capacidad del vehículo, si el individuo generado, cumple con la función de adaptación será evaluado en la función fitness, para determinar si da una mejor solución.

A continuación, se expone una parte del código desarrollado para la función de adaptación la cual permite evaluar los individuos seleccionados en cada generación:

Figura 24
Función de adaptación vector hijos generados

```python
#Generar función de adaptación#
def adaptacion(hijos):                                  #Función
    DemandaT_Clientes=[]                                #Inicio de varaibales
    CapacidadT_veh=[]                                   #Inicio de lista para almacenar datos
    a=0
    b=0
    m=0
    for y in range (len(hijos)):                        #Evaluar hijos
        for x in range(len(hijos[a])):···
        CapacidadT_veh = sum(DemandaT_Clientes)         #Demanda total a atender por vehículo
        CantidadUndVeh.append(CapacidadT_veh)           #Lista demandas totales para cada vehiculo
        DemandaT_Clientes=[]                            #Reinicio de variables
        a+=1
        b=0
```

Nota: Fuente El autor, generados Visual Studio Code

En la ***Figura 24*** se muestra la función de adaptación, para evaluar si cada hijo generado, generación tras generación cumple con la restricción de capacidad del sistema, de no ser así el hijo no desciende y se sigue repitiendo la operación hasta obtener un hijo que cumpla esta función y genere una mejor aptitud a la de los padres.

Figura 25
Adaptación Población Inicial

```
72 [10, 5, 10, 2, 4, 4, 14, 4, 3, 3, 4, 6, 3]
72 [3, 10, 5, 18, 9, 11, 4, 3, 9]
74 [4, 2, 5, 4, 5, 9, 13, 11, 8, 10, 3]
74 [6, 15, 5, 4, 16, 6, 3, 2, 9, 3, 5]
92 [10, 6, 6, 3, 6, 16, 11, 3, 8, 4, 4, 5, 2, 8]
40 [3, 3, 4, 7, 9, 6, 4, 4]
54 [3, 2, 21, 6, 10, 5, 7]
49 [9, 12, 5, 2, 3, 10, 8]
67 [3, 10, 4, 9, 6, 8, 2, 3, 2, 7, 8, 5]
33 [7, 7, 4, 15]
77 [12, 7, 5, 3, 10, 5, 17, 14, 4]
90 [11, 6, 3, 2, 3, 8, 9, 2, 5, 9, 9, 13, 5, 5]
77 [3, 5, 10, 5, 8, 4, 6, 9, 3, 2, 6, 9, 7]
35 [11, 16, 6, 2]
16 [5, 6, 3, 2]
```

Nota: Fuente El autor, generados Visual Studio Code

En la ***Figura 25*** se muestra que los 15 individuos generados inicialmente de forma constructiva cumplen la restricción de capacidad de cada ruta, la cual se estableció en 93, en la parte izquierda de cada lista la demanda a satisfacer por cada ruta y en la parte derecha la cantidad de unidades que debe recolectar en cada cliente.

4.10 Técnica de cruza para AG Order Crossover (OX)

El procedimiento de cruza para realizar operaciones de permutaciones es conocido como Order Crossover (OX) y habitualmente utilizado en problemas de operaciones combinatorias, este proceso se puede realizar en uno o más puntos de corte, para el desarrollo del AG el proceso de cruza de padres se realizó con la técnica en dos puntos:

1. Seleccionar dos padres dentro de la población inicial, por técnica de selección o aleatoriamente.
2. Crear dos hijos, por medio de dos vectores los cuales tendrán copia de los genes de los padres.
3. Determinar dos puntos en la cadena del padre1, de forma aleatoria los cuales corresponderán al límite inferior y superior para realizar permutación con el padre2.
4. Realizar permutación de los genes del padre 1 y padre 2 que serán trasladados a los nuevos hijos.

En la ***Figura 26***, sea el padre 1 de color azul el individuo 1 de la población y el padre 2 de color gris el individuo 2 de la población, se realiza el procedimiento de cruza en dos puntos de la siguiente manera:

Figura 26

Representación de dos padres seleccionados, proceso OX

I1	41	62	60	29	3	9	61	71	93	106	59	50	33	65	49	16	122
I2	94	84	36	128	137	66	45	109	67	1	79	8	64				

I1	41	62	60	29	3	9	61	71	93	106	59	50	33	65	49	16	122
I2	94	84	36	128	137	66	45	109	67	1	79	8	64				

Nota: Fuente El autor

Después de realizar el proceso de cruza a dos puntos, los individuos se cruzarán en los genes 1 y 8, en la ***Figura 27*** se muestra los dos puntos de cruza para los padres, los cuales trasladarán las sub-cadenas a sus respectivos hijos.

Figura 27

Sub cadenas generadas de padres a hijos

41								93	106	59	50	33	65	49	16	122
	62	60	29	3	9	61	71									
94								67	1	79	8	64				
	84	36	128	137	66	45	109									

Nota: Fuente El autor

La ***Figura 27*** muestra que el hijo 1 e hijo 2 generados de los dos padres se unen con las sub cadenas en los dos puntos de cruza para ser permutadas y dar origen a un nuevo individuo, el cual será evaluado en la función de adaptación y aptitud para determinar si pueden se almacenado en la matriz de la población final y luego ser candidato para mutación, en la ***Figura 28*** se muestran los hijos generados de la primera iteración de padres.

Figura 28
Hijos generados padre 1 y 2

H1	41	84	36	128	137	66	45	109	93	106	59	50	33	65	49	16	122
H2	94	62	60	29	3	9	61	71	67	1	79	8	64				

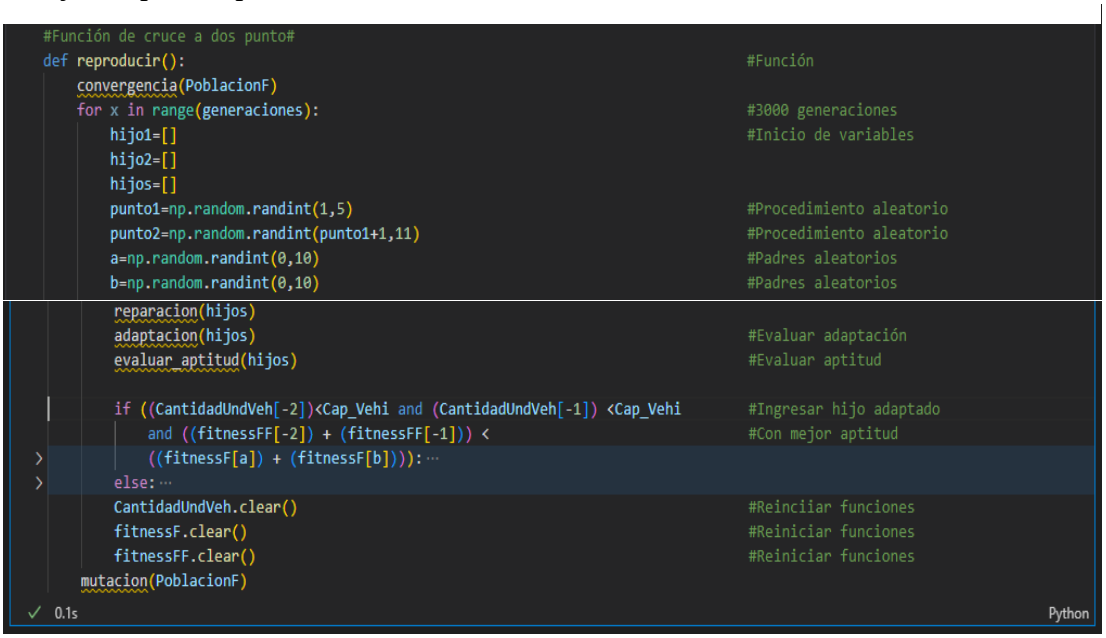
Nota: Fuente El autor

Luego de repetir n veces este procedimiento con todos los individuos de la población sin descartar ninguna sub-cadena de cada padre, los hijos o valores resultantes se almacenan en una nueva matriz denominada “PoblacionF” la cual únicamente almacena datos de los hijos que son aptos y cumplen los criterios para ser sometidos a un proceso de mutación.

A continuación, se expone una parte del código desarrollado para el procedimiento de generación de hijos y copias de sub-cadenas de genes para almacenar los valores en una nueva matriz de individuos, este procedimiento se realizó para un total de 300000 generaciones.

Figura 29

Función Python para reproducir individuos



```python
#Función de cruce a dos punto#
def reproducir():                                                    #Función
    convergencia(PoblacionF)
    for x in range(generaciones):                                    #3000 generaciones
        hijo1=[]                                                     #Inicio de variables
        hijo2=[]
        hijos=[]
        punto1=np.random.randint(1,5)                                #Procedimiento aleatorio
        punto2=np.random.randint(punto1+1,11)                        #Procedimiento aleatorio
        a=np.random.randint(0,10)                                    #Padres aleatorios
        b=np.random.randint(0,10)                                    #Padres aleatorios
        reparacion(hijos)
        adaptacion(hijos)                                            #Evaluar adaptación
        evaluar_aptitud(hijos)                                       #Evaluar aptitud

        if ((CantidadUndVeh[-2])<Cap_Vehi and (CantidadUndVeh[-1]) <Cap_Vehi      #Ingresar hijo adaptado
            and ((fitnessFF[-2]) + (fitnessFF[-1])) <                #Con mejor aptitud
            ((fitnessF[a]) + (fitnessF[b]))):···
        else:···
        CantidadUndVeh.clear()                                       #Reinciiar funciones
        fitnessF.clear()                                             #Reiniciar funciones
        fitnessFF.clear()                                            #Reiniciar funciones
    mutacion(PoblacionF)
```

✓ 0.1s Python

Nota: Fuente El autor, generados Visual Studio Code

La ***Figura 29*** muestra la función para reproducir hijos de la población inicial, teniendo en cuenta en la línea “Evaluar adaptación” si el hijo generado se adapta a la restricción de capacidad del vehículo y a su vez en la línea “Evaluar aptitud” se determina si el hijo generado tiene mejor fitness que los padres para ser almacenados en la nueva matriz de población.

4.11 Técnica de Mutación por Intercambio Reciproco en AG

El concepto de mutación se dio a conocer por primera vez en el año 1900 por un botánico inglés, el cual encontró una flor roja entre un cultivo de solo flores amarillas, desde este momento se empezó hablar de la mutación en las leyes de la herencia, en AG se adoptó esta técnica en el año 1964 como una estrategia evolutiva, sin embargo, se considera como un operador secundario siendo la cruza el operador principal. De acuerdo con las observaciones realizadas por Eduardo Grefenstette en su libro “Optimization of Control Parameters for Genetic Algorithms” se establece que los porcentajes de mutación por encima del 0.05 tienen a ser

perjudiciales en el desempeño del AG, sin embargo, para porcentajes <=0.01 se pueden conseguir mejores resultados de forma aleatoria.

Para el desarrollo del AG propuesto se establece un porcentaje de mutación del 0.01 y se adopta la técnica de mutación por intercambio reciproco, en la cual se establecen dos puntos al azar del individuo y se intercambian estos dos valores en cada posición. Al obtener el individuo o hijo generado este se evalúa si debe ser mutado mediante una función de mutación que genera un número aleatorio entre 0 y 1, si el valor generado es menor al porcentaje de mutación el hijo muta de lo contrario el individuo inicial se almacena en la matriz de la población de soluciones.

Figura 30

Técnica de Mutación por Intercambio Recíproco

I9	127	4	19	133	20	124	44	37	69	132	40	134	123	77
I9	127	4	19	69	20	124	44	37	133	132	40	134	123	77

I7	85	55	27	6	21	57	126	63	130	31	115	5	86	76
I7	85	55	27	6	21	86	126	63	130	31	115	5	57	76

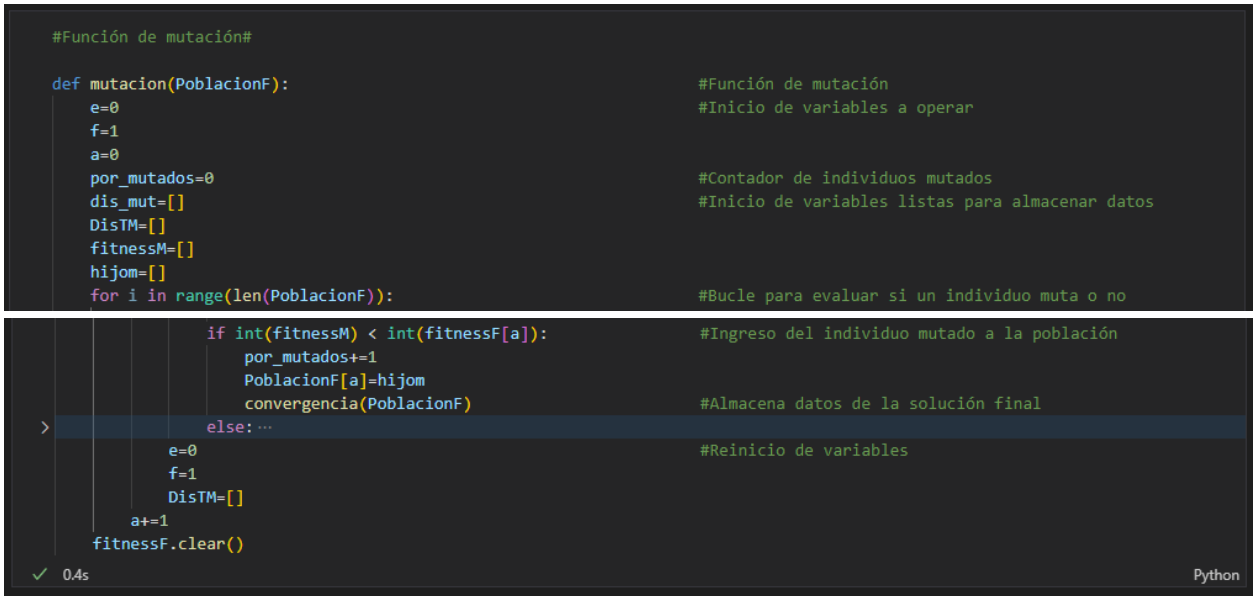
Nota: Fuente El autor

La ***Figura 30*** muestra el proceso de mutación en los individuos I7 e I9, donde para el individuo 9 se intercambian los genes de la población 4 y 9 con valores en sus alelos de 133 y 69, para el individuo 7, utilizando la misma metodología se intercambian los valores 57 y 86 entre sí, generando individuos mutados.

A continuación, se expone una parte del código desarrollado para realizar el procedimiento de mutación de individuos, en caso de que el valor generado de forma aleatoria sea menor al porcentaje de mutación.

Figura 31

Función para procedimiento de mutación



```
#Función de mutación#

def mutacion(PoblacionF):                                   #Función de mutación
    e=0                                                     #Inicio de variables a operar
    f=1
    a=0
    por_mutados=0                                           #Contador de individuos mutados
    dis_mut=[]                                              #Inicio de variables listas para almacenar datos
    DisTM=[]
    fitnessM=[]
    hijom=[]
    for i in range(len(PoblacionF)):                        #Bucle para evaluar si un individuo muta o no
                if int(fitnessM) < int(fitnessF[a]):        #Ingreso del individuo mutado a la población
                    por_mutados+=1
                    PoblacionF[a]=hijom
                    convergencia(PoblacionF)                #Almacena datos de la solución final
                else:···
            e=0                                             #Reinicio de variables
            f=1
            DisTM=[]
        a+=1
    fitnessF.clear()
```

0.4s Python

Nota: Fuente El autor, generados Visual Studio Code

4.12 Reparación de individuos

Para evitar posibles soluciones no factibles que puedan aparecer en los cromosomas o individuos generados dentro del proceso de cruza y mutación, interviene en las funciones “Reproducir”, “Mutación”, “Adaptación” y la función especial “Reaparición”, el cual realiza un procedimiento de evaluación en cada individuo determinando si este cumple o no las restricciones de factibilidad del sistema, en caso de que alguna restricción no se cumpla, el individuo no ingresa a la matriz de soluciones.

La función de reparación evalúa si el individuo viola las restricciones de ventanas de tiempo, en caso de que algún cliente este fuera de su rango de atención se realiza un proceso de reparación en el cual el individuo reordena los valores de su cadena de tal modo que se cumpla esta restricción. Este proceso se realiza teniendo en cuenta el tiempo de viaje de un cliente i a un

cliente j de acuerdo a la secuencia de la ruta generada. En caso de que algún individuo no se pueda reparar se procede a realizar un proceso de eliminación y no se almacena el resultado en la población final de individuos.

Figura 32

Rutas Infactibles Ventanas de tiempo

Cliente	1	2	3	4	5	6	7	8	9	10
Ventana de Inicio	8	8	8	8	8	8	11	8	8	8
Ventana de Salida	15,0	12,0	16,5	11,0	12,0	12,0	17,0	16,5	11,0	12,0

Nota: Fuente el Autor.

Figura 33

Rutas Reparadas VRPTW

Cliente	9	10	1	2	3	4	5	6	7	8
Ventana de Inicio	8	8	8	8	8	8	8	8	11	8
Ventana de Salida	11,0	12,0	15,0	12,0	16,5	11,0	12,0	12,0	17,0	16,5

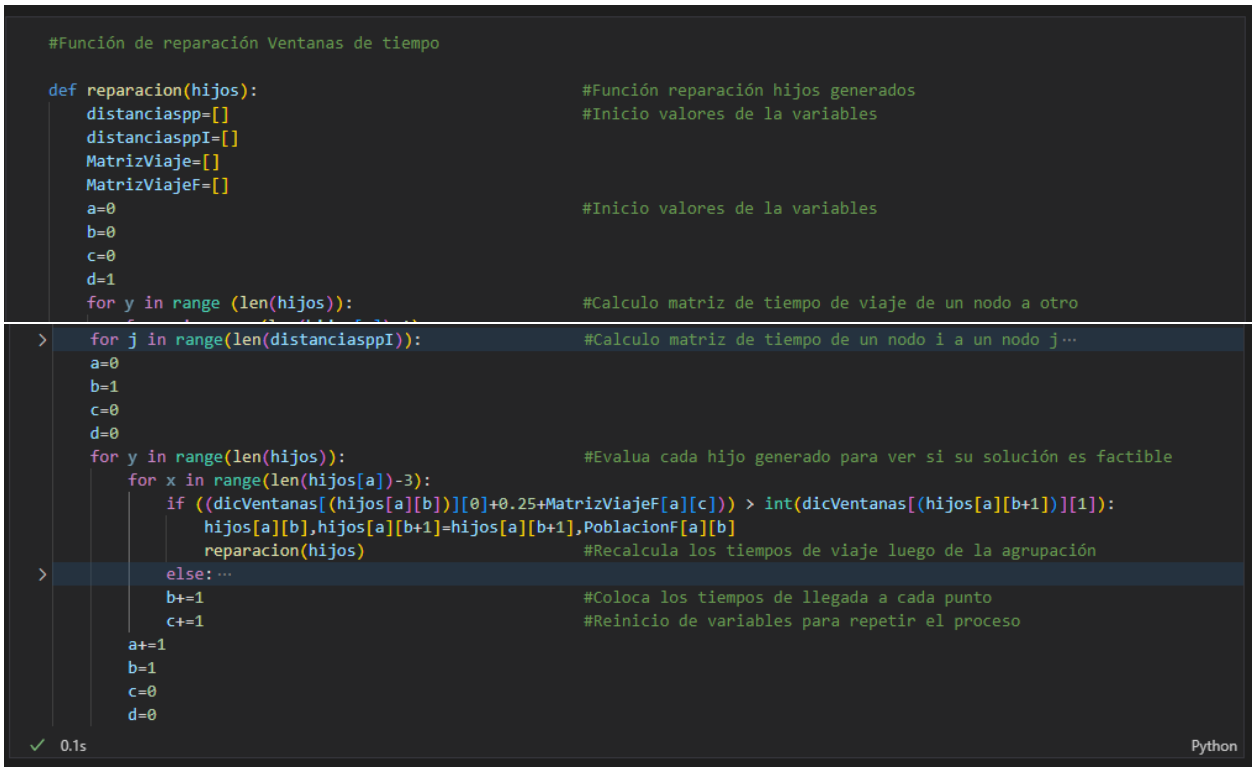
Nota: Fuente el Autor.

En la ***Figura 32*** y ***Figura 33*** se puede observar el proceso de reparación de rutas con el objetivo que se cumplan las restricciones de ventanas de tiempo, se puede observar que los clientes 9 y 10 no cumplen esta restricción y es necesario repararlos para ser agregados como una solución factible.

A continuación, se expone una parte el código desarrollado para el procedimiento de reparación de individuos, en caso de que la solución no sea factible por violar restricciones de ventanas de tiempo.

Figura 34

Función para procedimiento de Reparación



```
#Función de reparación Ventanas de tiempo

def reparacion(hijos):                                  #Función reparación hijos generados
    distanciaspp=[]                                     #Inicio valores de la variables
    distanciasppI=[]
    MatrizViaje=[]
    MatrizViajeF=[]
    a=0                                                 #Inicio valores de la variables
    b=0
    c=0
    d=1
    for y in range (len(hijos)):                        #Calculo matriz de tiempo de viaje de un nodo a otro
        for j in range(len(distanciasppI)):             #Calculo matriz de tiempo de un nodo i a un nodo j···
        a=0
        b=1
        c=0
        d=0
        for y in range(len(hijos)):                     #Evalua cada hijo generado para ver si su solución es factible
            for x in range(len(hijos[a])-3):
                if ((dicVentanas[(hijos[a][b])][0]+0.25+MatrizViajeF[a][c])) > int(dicVentanas[(hijos[a][b+1])][1]):
                    hijos[a][b],hijos[a][b+1]=hijos[a][b+1],PoblacionF[a][b]
                    reparacion(hijos)                   #Recalcula los tiempos de viaje luego de la agrupación
                else:···
                    b+=1                                #Coloca los tiempos de llegada a cada punto
                    c+=1                                #Reinicio de variables para repetir el proceso
            a+=1
            b=1
            c=0
            d=0
```

0.1s Python

Nota: Fuente El autor, generados Visual Studio Code

5. RESULTADOS

Se construyó un procedimiento para dar solución a un problema de ruteo de vehículos con restricciones de capacidad y ventanas de tiempo, partiendo de una solución inicial con ayuda de la herramienta de Excel “vrp_spreadsheet_solver_v3.72” proporcionada de manera libre por el profesor Günes Erdogan, la cual consiste mediante lenguaje de programación en Visual Basic para aplicaciones, encontrar una solución o un conjunto de rutas de manera constructiva que cumplen las restricciones del sistema, esta herramienta permite calcular valores muy útiles como lo son coordenadas geográficas teniendo como base las direcciones de cada cliente, posterior se desarrolló un algoritmo genético en el lenguaje de programación de Python para mejorar la solución inicial.

El AG se diseñó en el editor de código Visual Studio Code para el lenguaje de programación de Python, en una computadora personal Lenovo 81B0 con procesador Intel(R) Core(TM) i5-8250U CPU @ 1.60GHz, 1801 Mhz y memoria RAM 12,0GB, el tiempo de solución de cada corrida tuvo un tiempo promedio de 25,88 minutos y en cada una se realizaron 300.000 operaciones de cruce y cada 10.000 operaciones de cruce se realizaron 200 operaciones de mutación en cada hijo generado dentro de la población cruzada. Para cada corrida del AG se realizó el grafico de convergencia y se determinó que los datos convergen en un promedio de 300.000 generaciones.

Se implementó una función de adaptación en la cual se tuvo en cuenta únicamente la capacidad de cada vehículo, sin embargo, por medio de una función de reparación se ordenador los clientes de cada ruta en cada una de las 300.000 operaciones para satisfacer la restricción de ventanas de tiempo y lograr que solo se almacenaran en la matriz de la población final, soluciones factibles del modelo.

Tabla 10

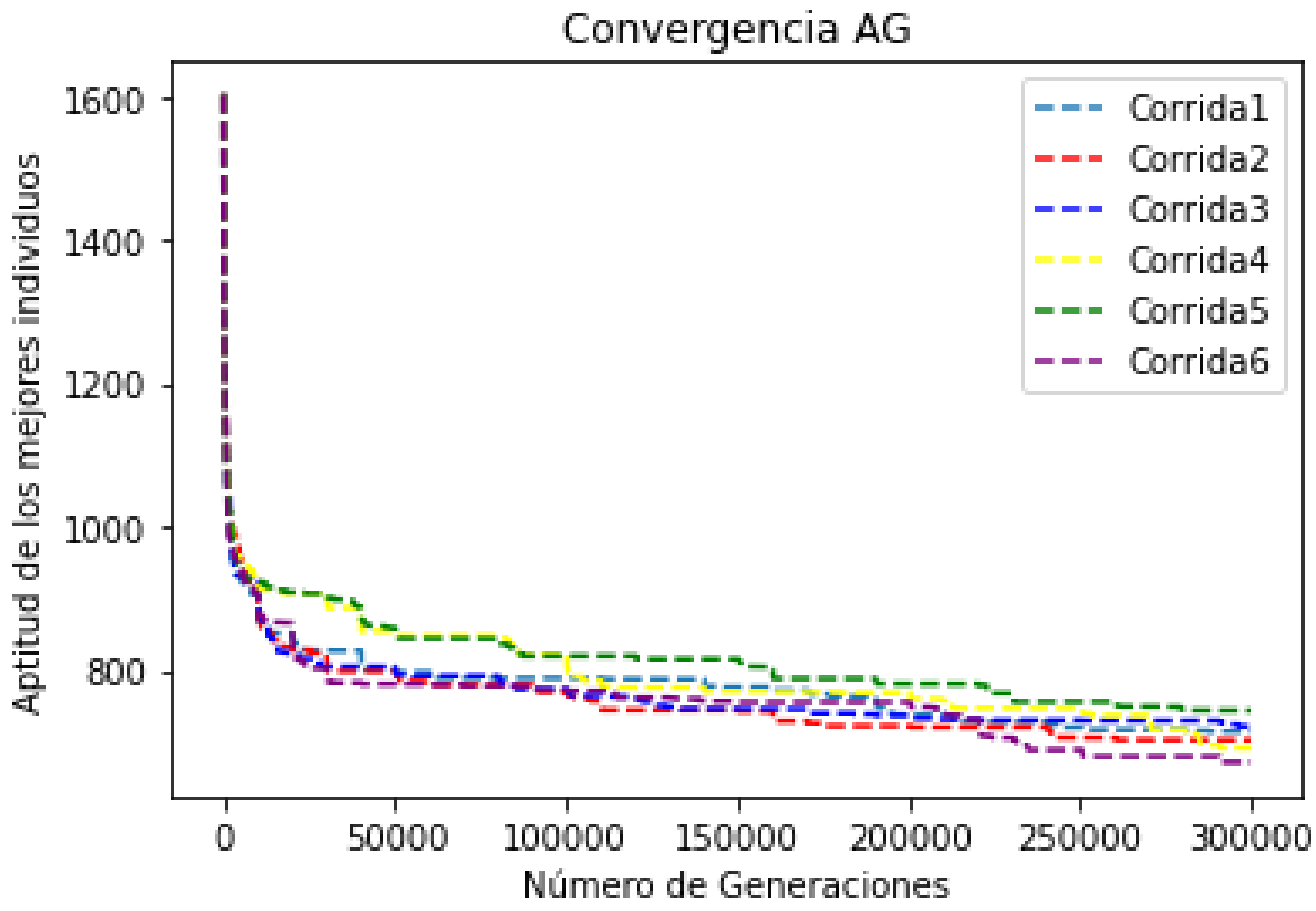
Tiempo de solución para AG

Corrida	Tiempo de Solución (minutos)
1	26,1
2	24,2
3	26,4
4	26,8
5	25,4
6	26,4
Promedio	27,61

Nota: Fuente El autor

En la ***Tabla 10*** se puede observar el tiempo de solución de cada corrida del AG, donde se encontró un promedio de 27,61 minutos y una desviación de 0,94 minutos para corridas de 300.000 iteraciones.

Figura 35
Convergencia del AG



Nota: Fuente El autor, generados en Python

En la ***Figura 35*** se puede analizar que aproximadamente en la iteración número 300.000 el AG converge en un total de seis experimentos ejecutados, esto quiere decir que, realizando las operaciones de cruza y mutación en los mejores individuos seleccionados, no se evidencia una mejoría notoria en la aptitud, el AG converge en un promedio de 705 Kilómetros o aptitud del individuo.

Tabla 11

Resultados Población Inicial

Rutas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Total
Clientes	15	11	13	13	16	10	9	9	14	6	11	16	15	6	6	170
Demanda	72	72	74	74	92	40	54	49	67	33	77	90	77	35	16	922
Distancia	205	108	138	33	92	76	43	66	148	15	76	204	171	38	42	1462

Nota: Fuente El autor.

En la ***Tabla 11*** se puede observar el resumen de la población inicial generada por la herramienta de Excel VRP donde se encuentra una distancia total de 1462 Km para todo el sistema, un total de 922 unidades a transportes para 140 clientes, teniendo en cuenta que se hace la resta del depósito inicial y final para cada ruta.

Tabla 12

Resultados Población final

Rutas	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	Total
Clientes	15	11	13	13	16	10	9	9	14	6	11	16	15	6	6	170
Demanda	88	69	91	69	71	59	56	35	85	28	73	85	71	26	16	922
Distancia	72	39	32	53	54	36	29	28	49	26	52	67	65	23	45	671

Nota: Fuente El autor.

En la ***Tabla 12*** se puede observar el resumen del resultado obtenido luego de realizar el AG, se tuvieron en cuenta los mismos parámetros de demanda, total clientes, capacidad del vehículo, ventanas de atención, obteniendo una distancia total de 671,26 Kilómetros lo que representan una reducción del **54,1%** del kilometraje de la población inicial.

Tabla 13
Soluciones VRPTW- Resumen

Ruta	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
C1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C2	6	47	35	51	129	20	116	91	56	117	112	89	96	99	54
C3	9	25	130	74	67	12	66	127	109	62	94	68	31	84	33
C4	139	1	34	36	134	126	120	72	39	95	132	125	63	92	23
C5	88	83	53	14	10	13	7	55	118	105	5	140	87	137	75
C6	15	124	22	64	81	28	18	41	119	0	97	128	113	0	0
C7	86	93	71	21	100	80	44	49	40		4	43	37		
C8	76	19	48	98	135	11	107	2	103		17	101	73		
C9	58	110	57	50	131	60	0	0	65		70	133	24		
C10	82	46	136	45	121	0			114		123	90	61		
C11	27	0	30	38	79				115		0	138	29		
C12	111		108	16	8				69			122	106		
C13	102		0	0	3				77			59	32		
C14	26				85				0			42	104		
C15	0				52							78	0		
C16					0							0			
Ci	15	11	13	13	16	10	9	9	14	6	11	16	15	6	6
Di	88	69	91	69	71	59	56	35	85	28	73	85	71	26	16
Disi	72	39	32	53	54	36	29	28	49	26	52	67	65	23	45

Nota: Fuente el Autor.

En ***Tabla 13*** se presenta la solución final para el VRPTWPD, para un conjunto de 15 rutas, cada ruta presenta en la parte final, la cantidad de clientes que va a visitar, la cantidad de información que recogerá de acuerdo con la capacidad del vehículo y la distancia total recorrida.

Tabla 14
Kilómetros de ahorro AG

Rutas	Distancia PI (Km)	Distancia AG (Km)	Ahorro (Km)
R1	205,43	71,79	133,64
R2	108,23	38,635	69,595
R3	138,76	31,942	106,818
R4	33,62	52,925	-19,305
R5	92,42	54,287	38,133
R6	76,59	36,22	40,37
R7	43,56	29,441	14,119
R8	66,6	28,467	38,133
R9	148,19	49,416	98,774
R10	15,86	26,185	-10,325
R11	76,13	52,446	23,684
R12	204,71	67,216	137,494
R13	171,32	65,194	106,126
R14	38,34	22,539	15,801
R15	42,29	44,557	-2,267
Total	1462,05	671,26	790,79

La ***Tabla 14*** muestra los Kilómetros ahorrados para cada ruta después de aplicar el Algoritmo genético, se puede observar que las rutas 4, 10 y 15 superaron su Kilometraje inicial, esto sucede porque a pesar de que el individuo tiene una mejor aptitud fue necesario aplicar la función de reparación para que cumpliera todas las ventanas de tiempo dentro de la ruta, el restante de individuos mejoró su aptitud la cual era disminuir la distancia recorrida.

6. CONCLUSIONES

De acuerdo con el trabajo realizado y los diferentes componentes integrados para su desarrollo, se identifica un conjunto de soluciones factibles para el modelo VRPTWPD a partir de la implementación de un Algoritmo Genético, integrando como soluciones padres la solución propuesta con la herramienta de Excel VRP del profesor Günes Erdogan.

En este proceso se destacan las siguientes condiciones resultado del ejercicio de este trabajo de especialización:

Se desarrolla el diagnóstico del modelo de distribución de la organización en estudio en donde para las características de sus recursos, operación y clientes se realiza la descripción de sus por menores, de acuerdo con las consideraciones necesarias para construir el modelo.

Dentro del desarrollo del documento se presenta la revisión de literatura sobre la cual se soporta el análisis, los modelos matemáticos, aproximaciones heurísticas y metaheurísticas que se adapten a las características del problema VRPTWPD, y su relación para la evaluación del caso de estudio, teniendo en cuenta todas las consideraciones propuestas del ejercicio.

Las soluciones para el ruteo se realizaron teniendo en cuenta diferentes condiciones para la definición de los experimentos, en donde se encontraron soluciones usando el Algoritmo Genético para el rutero VRPTWPD con un promedio de 705 kilómetros recorridos por el total de la flota vehicular, obteniendo una mejora del 54,1% de Kilómetros con respecto a la población inicial.

7. RECOMENDACIONES Y TRABAJOS FUTUROS

A continuación, se presentan las recomendaciones procedentes luego del desarrollo del trabajo de investigación.

De acuerdo con el trabajo de investigación realizado, se encontró dentro de la revisión de la literatura que la combinación de diferentes técnicas metaheurísticas mejoran los resultados en el VRP, por esta razón, a futuros investigadores se recomienda tomar este trabajo como punto de partida para realizar técnicas hibridas que permitan mejorar la solución de cada problema.

Con respecto al modelo matemático propuesto y todas sus restricciones de factibilidad se recomienda dentro del AG.

Considerar el VRP desde el enfoque mixto con operación de entregas y recolección para cualquier tipo de material a transportar en un mismo vehículo, con el fin de optimizar el total de rutas utilizadas.

Utilizar flota heterogénea para asignación de rutas, puesto que con el AG desarrollado se identificó que algunos individuos tenían mejor aptitud, sin embargo, no ingresaron a la solución final, por no cumplir el criterio de adaptación de la capacidad del vehículo.

Se recomienda a la empresa formular un modelo económico y de emisión de CO_2 en el cual se pueda cuantificar el valor del ahorro que representa la reducción de kilómetros recorridos con el AG desarrollado, a su vez las emisiones de gases que se dejan de producir y como están contribuyen a la conservación del ambiente, esto le permite a la empresa tener mayor control sobre sus recursos, ahorros económicos y aportar a la diminución de huella de carbono.

8. REFERENCIAS BIBLIOGRÁFICAS

Alvez, D., Chalupa, J. P., & Correa, D. (2019). *Problema de Ruteo con Multiples Ventanas de Tiempo para la Recolección de Leche (Tesis de Pregrado).* Montevideo.

Arévalo Alarcón, C., & Rojas Romero, R. A. (2019). *Modelo de Ruteo de Vehículos para Diminuir Emisión de Material Particulado Generado por Transporte de Carga: Empresa del Sector Retail en Bogotá (Tesis de Pregrado).* Universidad de la Salle, Bogotá D,C.

Augerat, P., Belenguer b, J., Benavent, E., Corberin, A., & Naddef, D. (1998). Separating Capacity Constraints in the CVRP Using Tabu Search. *European Journal of Operational Research, 106*, 547-557.

Bakker, S., Wang, A., & Gounaris, C. (2021). Vehicle Routing With Endogenous Learning: Application to Offshore. *European Journal of Operational Research, 289*, 93-106.

Cañada, A. (2014). *Diseño de Itinerarios para el Reparto de Gases Embotellados Mediante una Flota Heterogenea de Camiones.*

Castañeda Jimenez, J., & Cardona Arias, J. A. (2014). *Implementación del Modelo de Ahorro para Resolver el VRP Aplicado al Diseño de una Red de Logistica Inversa para la Recolección de Aceite Vehícular Usado Generado en los Puntos de Acopio Ubicado en Pereia (Tesis de Pregrado).* Universidad Tecnológica de Pereira, Pereira.

Chiang, W.-C., & Russell , R. (1996). Simulated Annealing Metaheuristics for the Vehicle Routing Problem With Time Windows. *Annals of Operations Research, 63*, 3-27.

Coello Coello, C. A. (2020). *Introduccion a la Computación Evolutiva.* México: Investav AV.

Cordone, R. (2001). A Heuristic for the Vehícle Routing Problem With Time Windows. *Kluwer Academic Publishers, 7*, 107-129.

Denisova, A., Alekseytsev, D., Meshcheryakov, V., & Denisov, O. (2021). Optimization of the Control System Parameters Using the Genetic Algorithm. *Journal of Physics: Conference Series, 01095*, 1742-1791.

Donati, A., Montemanni, N. C., Rizzoli, A., & Gambardella, L. (2008). Time Dependent Vehicle Routing Problem. *European Journal of Operational Research, 185*, 1174-1191.

Dr. P. V. Ingole, & Mr. Mangesh K Nichat . (2013). Landmark Based Shortest Path Detection by Using Dijkestra. *International Journal of Engineering Research and Applications (IJERA), 3*, 162-165.

Eksioglu, B., Bural, A., & Reisman, A. (2009). The Vehicle Routing Problem: A Taxonomic Review. *Computers & Industrial EngineerinG, 57*, 1472-1483.

El-Sherbeny, N. (2010). Vehicle Routing With Time Windows: An Overview. *Journal of King Saud University, 22*, 123-131.

Gómez Veloza, D. Y., & Gonzalez Restrepo, D. A. (2019). *Solución al Problema de Ruteo de Vehículos con Entregas y Recogidas Aplicando el Algoritmo de Petalos y el Algoritmo del Vecino más Cercano (Tesis de Pregrado).* Universidad Distrital Francisco Jose de Caldas, Bogotá D,C.

Guasmayan Guasmayan, F. A. (2014). *Solución del Problema de Rueteo de Vehículos Dependientes del Tiempo Utilizandoun Algoritmo Genetico Modificado (Tesis de Maestria).* Universidad Tecnológica de Pereria, Pereira.

Hashimotoa, H., Mutsunori, Y., & Toshihide, I. (2008). An Iterated Local Search Algorithm For The Time-Dependent Vehicle. *Discrete Optimization, 5*, 434-456.

Henández Sampieri, R. (2014). *Metodología de la investigación.* (6, Ed.) México D.F: McGrawhill.

Hernan Restrepo, J., Medina, P. D., & Arturo Cruz, E. (2008). A Logistic Case of Programming Vehicle Routing Problem With The Time Windows. *Scientia et Technica, 39*, 0122-1701.

Hiniestroza Marquinez, C. A. (2019). *Modelo de Ruteo para la Recolección de Tapas Plasticas Apoyo a la Labrol Sosicla de la Fundación Carlos Portela (Tesis de Pregrado).* Escuela de Ingenieria Informatica, Santiago de Cali.

Homberger, J., & Gehring, H. (2005). A Two-Phase Hybrid Metaheuristic For The Vehicle Routing Problem With Time Windows. *European Journal of Operational Research, 162*, 220-238.

Joerss, M., Jürgen, S., Neuhaus, F., Klink, C., & Mann, F. (2016). Parcel Delivery The Future of Last Mile: Travel, Transport and Logistics. 25.

Jourdan, L., Basseur, M., & Talb, E.-G. (2005). Hybridizing Exact Methods and Metaheuristics: A Taxonomy. *European Journal of Operational Research, 199*, 620-629.

Kurnia, H., Wahyuni, E. G., Pembrani, E. C., Gardini, S. T., & Aditya, S. K. (2017). Vehicle Routing Problem Using Genetic Algorithm with Multi Compartment on Vegetable Distribution. *IOP Conf. Series: Materials Science and Engineering, 325*.

Londoño, C. S., & Arias Hernandez, C. P. (2018). *Análisis y Prototipado de una Algoritmo Genético Modificado para Solucionar el Problema de Ruteo de Vehículos con Ventanas de Tiempo(VRPTW), Prioridad de Metas Económicas y Componente Medio Ambiental (Tesis de Maestria).* Universidad Tecnológica de Pereira, Pereira.

López, E., Salas, Ó., & Murillo, Á. (2014). The Travelin Salesman Problem A Deterministic Algorithm Using Tabu Search. *Reviste de Matemática: Teoria y Aplicaciones, 21*, 127-144.

Lozada Diaz, A., & Cadena Gonzalez, R. A. (2012). *Solución del Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRPTW) Mediante Métodos Heuristicos (Tesis de Pregrado).* Universidad Industrial de Santader, Bucaramanga.

Lüer, A., Benavente, M., Bustos, J., & Venegas, B. (2009). El Problema de Rutas de Vehículos: Extensiones y Métodos de Resolución, Estado del Arte. *Workshop Internacional*.

Márquez Gómez, M. (2014). Las metaheurísticas: Tendencias Actuales y su Aplicabilidad en la Ergonomía. *Ingeniería Industrial. Actualidad y Nuevas Tendencias, 4*, 108-120.

Marshall, L. F., Madsen, O. B., & Jornsten, k. O. (1997). Vehicle Routing With Time Windows Two Optimization Algorithms. *Institute for Operations Research and the Management Sciences, 45*, 488-492.

Mediorreal Carrillo, A. F. (2014). *Modelo de Ruteo de Vehículos para la Distribución de las Empresas Laboratorios Veterland, Laboratorios Callbest y Cosméticos Marlioü París (Tesis de Pregrado).* Pontifica Universidad Javeriana, Bogotá D,C.

Orozco, E. M. (2017). *Metaheurísticas para el Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRP-TW) (Tesis de Maestria).* Universidad Autónoma Metropolitana Azapotzal Co, Ciudad de México.

Pacheco Bonostro, J. A. (2000). *Problemas de Rutas con Ventanas de Tiempo.* uniyersidad Complutense de Madrid, Madrid.

Pureza, V., Morabito, R., & Reimann, M. (2012). Vehicle Routing With Multiple Deliverymen: Modeling and Heuristic Approaches. *European Journal of Operational Research, 218*, 636-647.

Quagliaroli Zapata, D. R. (2007). *Desarrollo de un Algoritmo Genetico-Cultiral para el Problema de Enrutamiento de Vehículos con Ventanas de Tiempo (Tesis de Pregrado).* Pontificia Universidad Católica de Valparaíso, Valparaiso.

Segura Peñuela, Á. D., & Padua Dueñas, A. K. (2018). *Modelo para la Solución de un Problema de Ruteo de Vehículos con Capacidad y Ventanas de Tiempo, en el Servicio de Transporte de Canje y Correo Bancario (Tesis de Pregrado).* Universitaria Agustiniana, Bogotá D,C.

Segura, V., Fuster , A., Antolín, F., Casellas, C., Payno, M., Grandío, A., . . . Muelas, M. (2020). Logística de Última Milla: Retos y soluciones en España . 65.

Solomon, M. (1987). Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Operations Research, 35*, 254-265.

Subramanian, A., Uchoa, E., & Ochi, L. S. (2013). A Hybrid Algorithm For a Class of Vehicle Routing Problems. *Computers & Operations Research, 40*, 2519-2531.

Szeto, W., Yongzhong, W., & Sin C, H. (2011). An Artificial Bee Colony Algorithm For The Capacitated Vehicle Routing Problem. *European Journal of Operational Research, 215*, 126-135.

Velasco Linares, B. S. (2019). *Diseño de un Modelo de Ruteo de Vehículos Dependientes del Tiempo en una Zona Urbana de Bogotá D,C (Tesis de Pregrado).* Universidad Distrital Francisco Jose de Caldas, Bogotá D,C.

9. ANEXOS

Anexo 1 *Pruebas estadísticas, distribución log normal*

Goodness of Fit - Details [hide]

[#13] **Lognormal**

Kolmogorov-Smirnov					
Sample Size Statistic Rank	105 0.04759 4				
α	0.2	0.15	0.1	0.05	0.01
Critical Value	0.10442	0.11125	0.11906	0.13272	0.15907
Reject?	No	No	No	No	No
Anderson-Darling					
Sample Size Statistic Rank	105 0.33405 4				
α	0.2	0.15	0.1	0.05	0.01
Critical Value	1.3749	1.6024	1.9286	2.5018	3.9074
Reject?	No	No	No	No	No
Chi-Squared					
Deg. of freedom Statistic Rank	6 3.7894 4				
α	0.2	0.15	0.1	0.05	0.01
Critical Value	8.5581	9.4461	10.645	12.592	16.812
Reject?	No	No	No	No	No

Anexo 2 *Matriz de Distancias*

Anexo *2* se presenta en el archivo de Excel “Padres_MatrizDis” hoja “MatrizD”

Anexo 3 *Lista de ubicaciones Georreferenciadas con su Ventana de Atención*

El ***Anexo 3*** se presenta en el archivo de Excel “Padres_MatrizDis” hoja “Simulaciones”

Anexo 4 Ejemplo Base, Datos a Simular

Anexo *4* se presenta en el archivo de Excel “Padres_MatrizDis” hoja “Simulaciones”

Anexo 5 *Código y funciones Python VRPTW – AG – Visual Code Studio*

```
#Importar librerías para operar datos#
#Lectura de datos, Clientes, Matriz de distancia, Demandas#
#Convertir hojas del Excel en Data Frame"
#Crear población inicial en Python, dar un valor inicial a todas las variables
globales del sistema#
#Función de cruce a dos puntos#
def reproducir():
      convergencia():
      reparacion():
      adaptacion():
      convergencia():
      Convergencia():
#Generar función de adaptación#
def adaptacion(hijos):
#Evalular el individuo funcion fitness#
def evaluar_aptitud(hijos):
#Función para guardar distancia total del modelo#
def convergencia(PoblacionF)
#Función de mutación#
def mutacion(PoblacionF):
#Función de reparación Ventanas de tiempo hijos no factibles#
def reparacion(hijos):
#Función para graficar los resultados en una grafica de convergencia#
def grafico():
for x in range(30):
    reproducir()
grafico()
```

Por cada función la cual se reconoce por iniciar con las letras def se tiene una secuencia de códigos, bucles, condiciones y funciones de reemplazos, además se combinan las funciones para operar y obtener resultados factibles