

PROGRAMA PERIÓDICO DE RUTAS DE RECOLECCIÓN DE MATERIAL
RECICLABLE EN GUADALAJARA DE BUGA, PARA UNA EMPRESA
RECUPERADORA DE MATERIAL INDUSTRIAL

DANIELA GALINDO
JEAN CARLOS SÁNCHEZ GALLEGO

UNIVERSIDAD DEL VALLE SEDE BUGA
FACULTAD DE INGENIERÍA - ESCUELA DE INGENIERÍA INDUSTRIAL
GUADALAJARA DE BUGA
2021

PROGRAMA PERIÓDICO DE RUTAS DE RECOLECCIÓN DE MATERIAL
RECICLABLE EN GUADALAJARA DE BUGA, PARA UNA EMPRESA
RECUPERADORA DE MATERIAL INDUSTRIAL

DANIELA GALINDO
JEAN CARLOS SÁNCHEZ GALLEGO

Trabajo de grado presentado como requisito para optar al título de Ingeniero Industrial

Director: Julio Cesar Londoño Ph.D.

UNIVERSIDAD DEL VALLE SEDE BUGA
FACULTAD DE INGENIERÍA - ESCUELA DE INGENIERÍA INDUSTRIAL
GUADALAJARA DE BUGA
2021

AGRADECIMIENTOS

Agradecemos en primera instancia a Dios, quien ha guiado nuestro camino a lo largo de la carrera y nos ha permitido llegar hasta este punto. A nuestros padres José Arbey Sánchez Valencia, María Nubia Gallego Tabares, Jhon Jairo Muñoz y Sandra Muñoz por el amor, apoyo incondicional, por creer en nosotros y brindarnos todas las herramientas para hacer posible todos nuestros sueños.

De manera muy especial agradecemos a nuestro director Julio César Londoño por su compromiso, dedicación, apoyo y disposición para guiarnos en todo este proceso de nuestro trabajo de grado.

Agradecemos a todos nuestros docentes por las enseñanzas y a nuestros compañeros de carrera, porque de alguna u otra manera son parte de este resultado.

TABLA DE CONTENIDO

Pág.

1. INTRODUCCIÓN	7
2. PLANTEAMIENTO DEL PROBLEMA	8
3. OBJETIVOS	10
3.1 OBJETIVO GENERAL	10
3.2 OBJETIVOS ESPECÍFICOS	10
4. REVISIÓN DE LITERATURA	11
5. CARACTERIZACIÓN DEL PROCESO DE RECOLECCIÓN DE MATERIAL RECICLABLE	12
6. FORMULACIÓN MATEMÁTICA PARA EL PROBLEMA CARP	16
7. HEURÍSTICA PROPUESTA PARA EL PCARP.....	18
7.1 DESARROLLO DE LA HEURÍSTICA	19
7.1.1 Definición de los clústeres	19
7.1.2 Diseño de rutas	20
8. VALIDACIÓN DE LA HEURÍSTICA Y ANÁLISIS DE RESULTADOS	23
9. COMPARACIÓN ENTRE EL MODELO EXACTO CARP Y LA HEURÍSTICA PROPUESTA.....	30
10. CONCLUSIONES Y RECOMENDACIONES	32
11. BIBLIOGRAFÍA	33
12. ANEXOS	35

LISTA DE TABLAS

Tabla 1. Características de la flota de vehículos.....	13
Tabla 2. Barrios de la zona sur.....	14
Tabla 3. Rutas de recolección actuales.....	15
Tabla 4. Clusters generados por KMeans.....	24
Tabla 5. Distancia recorrida para salir y llegar a la empresa desde el inicio del cluster.....	25
Tabla 6. Recorrido, distancias y tiempos de ejecución de cada cluster.....	26
Tabla 7. Programación propuesta para la recolección.....	29
Tabla 8. Distancias recorridas actuales vs propuestas.....	29
Tabla 9. Comparación de la distancia recorrida actual y la solución propuesta.....	30
Tabla 10. CARP Vs Heurística (GAP).....	30
Tabla 11. Tiempos de ejecución en segundos.....	31
Tabla 12. Resultados del CARP y la Heurística.....	31
Tabla 13. Resultados de la heurística.....	31

LISTA DE ILUSTRACIONES

Ilustración 1. Representación del PCARP.....	9
Ilustración 2. Zonificación de la ciudad.....	13
Ilustración 3. Rutas actuales de recolección.....	15
Ilustración 4. División del problema PCARP en fases, estrategia de solución.....	19
Ilustración 5. Pseudocódigo de la heurística propuesta.....	22
Ilustración 6. Clusters generados con KMeans.....	23
Ilustración 7. Recorrido actual de la ruta 2.....	27
Ilustración 8. Recorrido propuesto para el cluster 2.....	28

LISTA DE ANEXOS

Anexo A. Recorridos actuales de la empresa.....	35
Anexo B. Coordenadas de las calles halladas por medio de Google Maps.....	40
Anexo C. Algoritmo KMeans.....	43
Anexo D. Algoritmo de Dijkstra.....	43
Anexo E. Algoritmo de Fleury.....	44
Anexo F. Visualización del código desarrollado en Google colab.....	44
Anexo G. Resultados en Python de las rutas de entrada y salida a cada cluster.....	45
Anexo H. Resultados en Python de los recorridos de cada cluster.....	46
Anexo I. Recorridos de la solución propuesta.....	48
Anexo J. Código en AMPL para el CARP.....	53
Anexo K. Recorrido arrojado por el modelo CARP y la heurística propuesta.....	54

RESUMEN

El objetivo de este trabajo es diseñar un programa periódico de rutas para la recolección de material reciclable que permita la disminución de la distancia recorrida lo que corresponde a un problema periódico de enrutamiento de arco capacitado PCARP.

Para el desarrollo del trabajo se establecieron 3 fases: la primera consiste en la caracterización del proceso, donde se identificaron los materiales recolectados, los vehículos utilizados, los recorridos, la demanda por calle y la frecuencia de recolección en un horizonte de tiempo de una semana. En la segunda fase se simplificó el problema PCARP al problema de enrutamiento de vehículos capacitados sobre arcos para un periodo CARP, debido a que la frecuencia de recolección en el horizonte de una semana es igual a uno para cada ruta, siendo así, se formuló el modelo matemático con base al realizado por Golden (1981) que nos da como resultado los nodos a recorrer buscando minimizar la distancia total recorrida por los dos vehículos teniendo en cuenta que se debe salir y regresar al depósito y no se debe exceder la capacidad del vehículo.

Sin embargo, como el tamaño del problema caso de estudio lo hace intratable, se diseñó una heurística en el lenguaje de programación Python, como estrategia de solución que divide el problema en dos fases: la primera fase con el propósito de abordar las limitaciones de capacidad de los vehículos y la periodicidad en la recolección por medio del algoritmo KMeans, y la segunda fase con el fin de encontrar los recorridos para cada ruta a través del algoritmo del cartero chino CPP el cual a su vez se basa en dos algoritmos como Dijkstra y Fleury para determinar la secuencia de los nodos que conforman el circuito euleriano con sus respectivas distancia. Por último, para demostrar que tan efectivos eran los resultados de la heurística propuesta a comparación de cómo la empresa lo realiza se hizo una comparación de las distancias recorridas logrando una disminución de 5,65 kilómetros por semana.

Para efectos comparativos entre el modelo exacto del CARP y la fase 2 de la heurística propuesta con respecto a la secuencia del recorrido y la distancia total obtenida, se resolvieron ambos modelos en una instancia pequeña del problema abordado obteniendo un porcentaje del 0% de desviación (GAP) de la solución obtenida por la heurística con respecto a la obtenida por el CARP.

Palabras clave: PCARP, CARP, Dijkstra, Fleury, Kmeans, Cartero Chino, Python, Ruteo, Euleriano, Optimización.

1. INTRODUCCIÓN

La generación de residuos siempre ha estado presente en toda la actividad humana, pero la creciente industrialización, el crecimiento acelerado de la población en los últimos años y el estilo de vida de las personas actualmente han ocasionado un incremento gigantesco de los residuos sólidos, por lo cual este problema ha generado la oportunidad de que surjan nuevos emprendimientos dedicados al reciclaje, donde su actividad principal es recuperar los materiales originados del posconsumo y ser convertidos en materia prima para la fabricación de nuevos productos. Sin embargo, estas empresas tienen consigo problemas complejos de logística de recolección, pues en la mayoría de los casos no se cuenta con conocimiento técnico ni con un software que realice el diseño de las rutas, lo cual conlleva a que estas empresas sobre todo de menor tamaño realicen la recolección de manera intuitiva, generando que el tiempo empleado y los gastos, como, por ejemplo, el combustible se incrementen debido a que en el recorrido se transitan las mismas vías más de una vez haciendo que este sea mucho más extenso.

En la recolección de los residuos sólidos, es indispensable tener en cuenta ciertos parámetros como la cantidad de residuos, el ancho, sentido y estado de las vías dado el tamaño del vehículo recolector, los costos de transporte, la distancia a recorrer, la capacidad de carga de los camiones, la frecuencia de recolección, etc; así que para las empresas que se dedican a esta actividad es supremamente importante definir un programa de rutas de recolección que permita pasar por todas aquellas calles donde exista una demanda. Este problema en investigación de operaciones corresponde al problema periódico de enrutamiento de arco capacitado PCARP.

El problema PCARP es un problema de optimización donde en un horizonte de tiempo dividido en periodos (días) se deben asignar arcos o calles y así mismo establecer una ruta para el vehículo en ese periodo teniendo en cuenta que cada recorrido debe empezar y terminar en el depósito, cada calle debe ser recorrida por un solo vehículo y la demanda total atendida por la ruta no debe exceder la capacidad del camión recolector.

No obstante, en el presente trabajo se desarrolla una heurística basada en un modelo matemático que permite resolver la problemática del PCARP en una empresa caso de estudio dedicada a la recolección de material reciclable en Guadalajara de Buga, y de este modo realizar el diseño de un programa periódico de rutas de recolección de material reciclable, donde se busca determinar la ruta que debe seguir el vehículo para cada día de recolección cumpliendo con cada tarea, utilizando una distancia mínima.

2. PLANTEAMIENTO DEL PROBLEMA

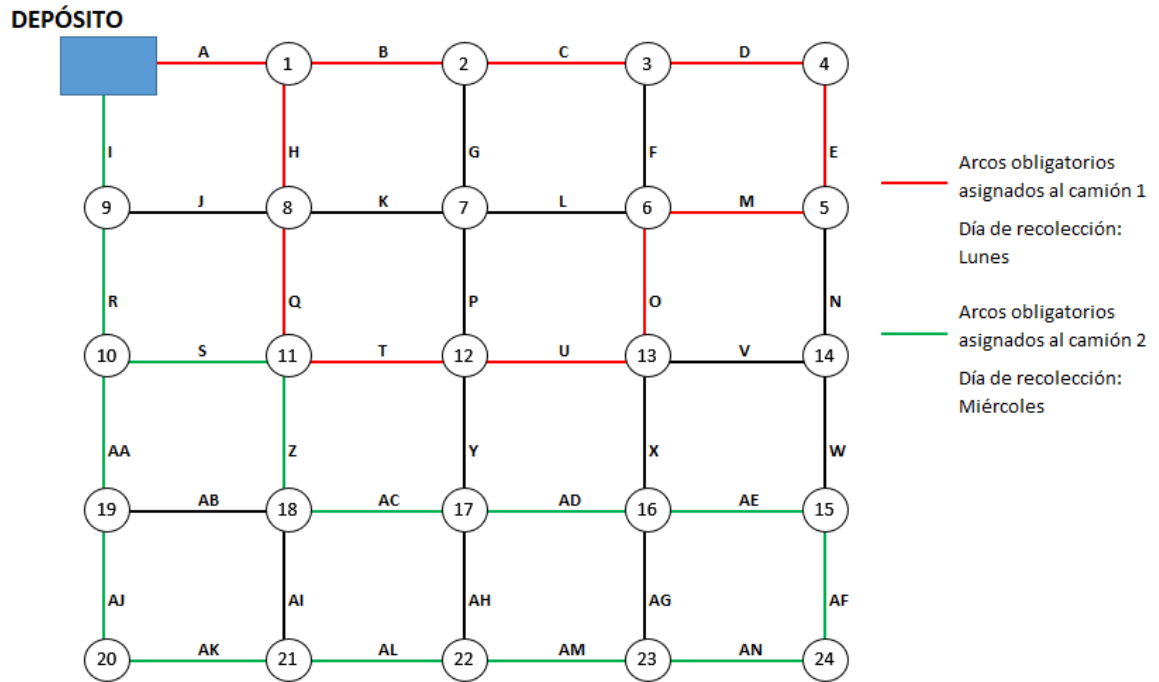
Hoy en día, la generación de diferentes tipos de desechos sólidos y la manifestación de sus inconsistencias sociales, económicas y ambientales asociadas crean numerosas complicaciones para la gestión del servicio urbano con respecto a la recolección, transporte, procesamiento y eliminación de dichos desechos. Una planificación óptima en la recolección, gestión y disposición de los desechos urbanos es una de las estrategias más importantes que pueden producir múltiples mejoras tanto en el costo total como en la salud (Babaei Tirkolaei, Mahdavi & Seyyed Esfahani, 2018). Sin embargo, el diseño de rutas de vehículos es una de las funciones operativas más críticas para las empresas a nivel mundial, debido a que, durante las operaciones de transporte de bienes y servicios, los costos asociados pueden representar entre uno y dos tercios del costo logístico total (Ballou, 2004).

Un inadecuado diseño de las rutas ocasiona un incremento en la distancia total recorrida y por ende en los costos asociados al transporte. En el caso de las empresas dedicadas a la recolección de residuos sólidos domiciliarios, el problema toma mayor relevancia debido a la complejidad que implica diseñar diferentes rutas de recolección periódicas para ciudades con características diferentes, y esto se ve reflejado en los costos pues en este sector tan solo la fase de recolección puede llegar a representar entre 70 y el 85% de los costos totales de la gestión de los residuos (Bentazo, Torres, Romero & Obregón, 2016); además las condiciones económicas y logísticas bajo las cuales operan las empresas en Colombia, han hecho que muy pocas cuenten con herramientas de software para el diseño de rutas debido al costo que representa para la empresa, por lo cual el uso de herramientas de la investigación de operaciones como el ruteo e incluso herramientas de otras ramas diferentes a la ingeniería industrial como los lenguajes de programación, son una oportunidad para reducir los costos fijos y operativos, y aumentar los niveles de servicio del sistema de transporte (Gutiérrez, Palacio & Villegas, 2012).

En la empresa caso de estudio ubicada en la ciudad de Buga, las actividades relacionadas con la recolección de materiales reciclables se realizan totalmente de manera empírica, pues la empresa no cuenta con un sistema o software para el diseño de rutas y por el contrario, utiliza el método empírico teniendo en cuenta sólo una regla, que los conductores deben procurar siempre girar a la derecha en las esquinas, por tanto, se presentan cruces en las rutas, es decir, el vehículo recorre rutas que ya había transitado, además, el conductor es quien decide por donde inicia la ruta y muchas veces según información suministrada por la empresa, se recorren calles donde no se recolecta ningún tipo de material, lo cual genera recorridos de largas distancias y el desperdicio de la capacidad de los camiones, situación que puede tener un alto impacto en los costos para la empresa.

Analizando la situación de la empresa, la problemática corresponde al problema periódico de enrutamiento de arco capacitado (PCARP: Periodic Capacitated Arc Routing Problem), la cual se ilustra en la siguiente figura:

Ilustración 1. Representación del PCARP



Fuente: Elaboración propia

El PCARP resulta ser una versión periódica del CARP, enfocado en la modelación de importantes aplicaciones como la recolección de residuos municipales, donde los productos o desechos de un conjunto de calles y en un día (período) pueden ser recogidos por vehículos (Chu, Labadi & Prins, 2004), tal como sucede en la empresa caso de estudio donde se busca diseñar un programa de rutas para la zona sur de la ciudad, en un horizonte de tiempo (una semana) que defina qué calles se deben recorrer en los distintos días de la semana para realizar la recolección y determinar la ruta que debe seguir cada vehículo en cada periodo de tiempo con el fin de disminuir la distancia recorrida, teniendo en cuenta que cada recorrido debe comenzar y terminar en las instalaciones de la empresa, y para lograrlo es importante tener en cuenta el contexto de la organización y la complejidad que requieren este tipo de problemas para su solución por el requerimiento computacional, no obstante, el PCARP es considerado un problema NP-HARD, y por lo tanto, es importante definir una correcta estrategia a través de una combinación de herramientas de la ingeniería industrial y otras ramas por el tamaño del problema abordado. Con base a lo anterior se plantea la siguiente pregunta:

¿CÓMO DISEÑAR UN PROGRAMA PERIÓDICO DE RUTAS DE VEHÍCULOS PARA LA RECOLECCIÓN DE MATERIAL RECICLABLE, QUE PERMITA MINIMIZAR LA DISTANCIA RECORRIDA?

3. OBJETIVOS

3.1 OBJETIVO GENERAL

Diseñar un programa periódico de rutas para la recolección de material reciclable, para una empresa recuperadora de material industrial ubicada en Guadalajara de Buga que permita la disminución de las distancias recorridas.

3.2 OBJETIVOS ESPECÍFICOS

- Caracterizar el proceso de recolección de material reciclable, para la identificación de factores a tener en cuenta en el diseño del programa de rutas.
- Formular un modelo matemático para el establecimiento de un programa de rutas.
- Validar la solución del modelo propuesto para el caso de estudio mediante el experimento computacional.

4. REVISIÓN DE LITERATURA

El problema de ruteo de vehículo ha sido ampliamente estudiado desde hace décadas, y esto se debe esencialmente a la cantidad de variantes que tienen este tipo de problemas por el número de casos de aplicación específica que existen en la industria. Basado en lo anterior, la teoría en el tema de ruteo ha sido muy amplia y diversa, por lo que es importante saber identificar la versión del tipo de problema en la literatura para abordar el problema de la mejor manera. En el presente caso, la problemática abordada resulta estar representada por el problema PCARP (Periodic Capacitated Arc Routing Problem), el cual resulta ser una variante del problema CARP originalmente propuesto por Golden y Wong (1981), en donde se busca determinar un conjunto de viajes de vehículos que minimice el costo total de los bordes atravesados, teniendo en cuenta que cada viaje comienza y termina en el depósito; cada arco o calle debe ser atendida por un solo vehículo y la demanda total atendida por cualquier viaje no debe exceder la capacidad del vehículo. Desde un punto de vista más general, el CARP resulta ser a su vez una variante del CPP (Chinese Postman Problem), en el cual la capacidad no está asociada al problema, pues solo interesa realizar un recorrido por todos los arcos del grafo con una distancia mínima, partiendo y finalizando en un único punto y por supuesto, teniendo en cuenta la configuración del grafo (Dirigido, Mixto, No dirigido). Por lo anterior, podríamos determinar que el PCARP resulta ser un problema CPP asociado a una capacidad y a una periodicidad o frecuencia en el servicio.

En el caso del PCARP, el primer modelo de programación lineal para la solución de este tipo de variantes del CARP, fue propuesto por Chu, et al., (2004), en el cual se proporciona un conjunto de posibles períodos de servicio para cada calle o arco con demanda, también denominadas *tareas*, es decir, en un horizonte de tiempo de una semana como sucede por ejemplo, en el caso de la recolección de las basuras, los recorridos por una misma ruta se realizan más de una vez por semana, para lo cual los autores desarrollaron tres heurísticas diferentes en base a dos conjuntos de instancias, a través de una estrategia de solución conocida como búsqueda de solución (SS) propuesta por Glover (1977), la cual es utilizada comúnmente en la programación entera para generar soluciones a partir de la combinación de un conjunto de soluciones utilizadas inicialmente como referencia.

Posteriormente, los mismos autores Chu et al. (2004) proponen un nuevo modelo de programación lineal más general, en el cual se tienen en cuenta límites inferiores preliminares como estrategia para acotar las dimensiones del problema por su complejidad, teniendo en cuenta que, por ejemplo, en este caso los autores no consideran restricciones como la variación de la demanda entre dos servicios prestados a la semana, y tampoco se conoce previamente un conjunto de posibles periodos de servicio prestado en el horizonte de tiempo, como si lo hacen en su primera publicación, donde al final proponen dos conjuntos para nuevos investigadores. Como resultados finales, los autores determinan que la solución propuesta es satisfactoria, aunque recomiendan técnicas más avanzadas como relajación lagrangiana o el enfoque de generación de columnas.

Aunque este tipo de casos como el CARP y su variante PCARP suelen ser menos estudiados en comparación a sus variantes anteriores como el CPP, lo cierto es que todas las variantes a medida que avanzan resultan ser más desafiantes por el nivel combinatorio que se aborda según el tamaño del problema; de por sí el CPP para instancias muy grandes resulta ser ya intratable o imposibles de resolver desde modelos exactos (Yordá, 2015), por lo cual, en la actualidad algoritmos como Dijkstra, Fleury, Floyd o Hierholzer propuestos hace décadas, aún siguen siendo parte indispensables para la resolución no solo del CARP sino también de otras variaciones, tal como lo demuestran Mei, Tang & Yao (2011) con la incorporación de Dijkstra para el desarrollo de su algoritmo memético para el PCARP.

Durante los últimos años, la investigación en estos tipos de problemas ha ido aumentando, pues ha surgido la necesidad de abordar cada vez más problemas de mayor tamaño como los que se presentan actualmente, tal como lo plantearon Chen y Hao (2016), quienes presentaron un método metaheurístico (HMA) para CARP, el cual fue capaz de resolver las instancias de este tipo de problemas en un tiempo razonable y mejoró la solución para 6 instancias clásicas y 9 instancias de gran tamaño. De igual manera Babaei, Mahdavi y Seyyed (2016), estudian el PCARP planteando como objetivo minimizar la distancia recorrida de acuerdo con la incertidumbre de la demanda, y plantean un algoritmo metaheurístico híbrido con técnicas de recocido simulado y otras heurísticas con tiempos aceptables en comparación a los modelos exactos.

Todas las anteriores investigaciones fueron importantes por las estrategias abordadas, aunque inicialmente las instancias de los problemas son pequeñas en comparación al problema abordado en este trabajo.

5. CARACTERIZACIÓN DEL PROCESO DE RECOLECCIÓN DE MATERIAL RECICLABLE

La empresa caso de estudio recolecta varios tipos de materiales en la ciudad, los cuales se separan y se transforman en materias primas para diferentes empresas que los incorporan en sus procesos productivos. Dentro de los materiales que hacen parte del proceso de recolección en la ciudad, se encuentran:

- | | | |
|----------|------------|------------|
| - Pasta | - Papel | - Plástico |
| - Vidrio | - Aluminio | - PET |
| - Cartón | - Chatarra | - PVC |

Estos materiales provienen de varias fuentes como hogares, colegios, empresas y entidades públicas de la ciudad. Es importante mencionar que estos materiales son recolectados, siempre y cuando estén secos y no estén mezclados con comida o residuos orgánicos.

El proceso de recolección que realiza actualmente la empresa se desarrolla por medio de una flota de vehículos heterogénea cuyas características se encuentran descritas en la [tabla 1](#).

Tabla 1. Características de la flota de vehículos.

Tipo de vehículo	Cantidad	Capacidad (Kg)
Camioneta	1	1000
Camión sencillo	1	3000

Fuente: Elaboración propia en base a información suministrada por la empresa.

La empresa actualmente cuenta con una cobertura para la recolección de los residuos aprovechables en la mayor parte de la ciudad de Buga, donde la operación se enfoca principalmente en los barrios y urbanizaciones de la zona centro, sur, oriente y occidente de Buga como se detalla en la [ilustración 2](#).

Ilustración 2. Zonificación de la ciudad.



Fuente: Elaboración propia y el mapa de la ciudad extraído de Google Maps.

Cada una de las zonas anteriormente descritas, tienen características determinadas en cuanto al sentido vial, cantidad de material generado, etc., tal como se describe a continuación:

- **Zona Centro:** esta zona se caracteriza por tener en mayor proporción vías con único sentido y genera un volumen importante de material, pues son generados en su mayoría por establecimientos comerciales con los que la empresa ya tiene una relación establecida.
- **Zona Sur:** particularmente en esta zona de la ciudad, la mayoría de los barrios y urbanizaciones que comprende se caracterizan por ser vías dobles sentido. Esta zona se caracteriza por ser una de las que mayor volumen de material reciclable genera, ya que los recorridos iniciaron en esta zona y se han venido realizando jornadas de sensibilización.
- **Zona Oriente:** esta zona se distingue por tener en mayor medida vías con único sentido y el volumen generado en esta zona suele ser alto, pero en menor proporción al de la zona sur y centro.
- **Zona Occidente:** esta zona tiene una particularidad de acuerdo con la distribución de la ciudad, pues comprende en gran medida las vías de ingreso a la ciudad, la zona industrial de la ciudad y el barrio Palo Blanco, sin embargo, el volumen de material generado no suele ser tan importante como otras zonas, pues las empresas del sector no hacen parte de la ruta de recolección.

Como se mencionó anteriormente el presente trabajo se acotó a la zona sur de la ciudad, por la importancia que representa para la empresa en cuanto a volumen de material recolectado y de mayor complejidad para realizar el ruteo por la distribución de los barrios. La zona sur está conformada por un total de 7 barrios, los cuales son visitados una vez por semana. En la siguiente tabla se podrá apreciar el número de nodos, calles que componen cada barrio y las respectivas demandas de cada calle, las cuales fueron calculadas con base al histórico suministrado por la empresa del peso del material recolectado por ruta, durante los últimos dos años.

Tabla 2. Barrios de la zona sur.

N° Barrio	N° Nodos	N° Calles	Demanda prom. Anual (Kg)	Demanda prom. Semanal (Kg)	Demanda prom. Por calle (Kg)
Barrio 1	46	56	39,167	739.00	13.20
Barrio 2	3	3	1,633	30.80	10.27
Barrio 3	15	19	4,544	85.73	4.51
Barrio 4	30	34	43,786	826.15	24.30
Barrio 5	53	60	26,063	491.75	8.20
Barrio 6	96	122	41,893	790.44	6.48
Barrio 7	5	6	1,282	24.18	4.03
Total	248	300	158,366.05	2,988.04	70.98

Fuente: Elaboración propia en base a los datos suministrados por la empresa.

Cada uno de los barrios descritos anteriormente, tiene un día de recolección durante la semana, debido a que actualmente la generación de material reciclable no es lo suficientemente considerable para realizar la recolección con una frecuencia mayor a 1 día por semana. Por lo anterior, la empresa ha elaborado un plan de recolección para 5 rutas, las cuales pueden contener recorridos de 1 o más barrios, y estas son asignadas a uno de los 3 días planificados por la empresa para la recolección como lo son el lunes, miércoles y viernes en jornadas distintas, tal como se detalla en la [tabla 3](#):

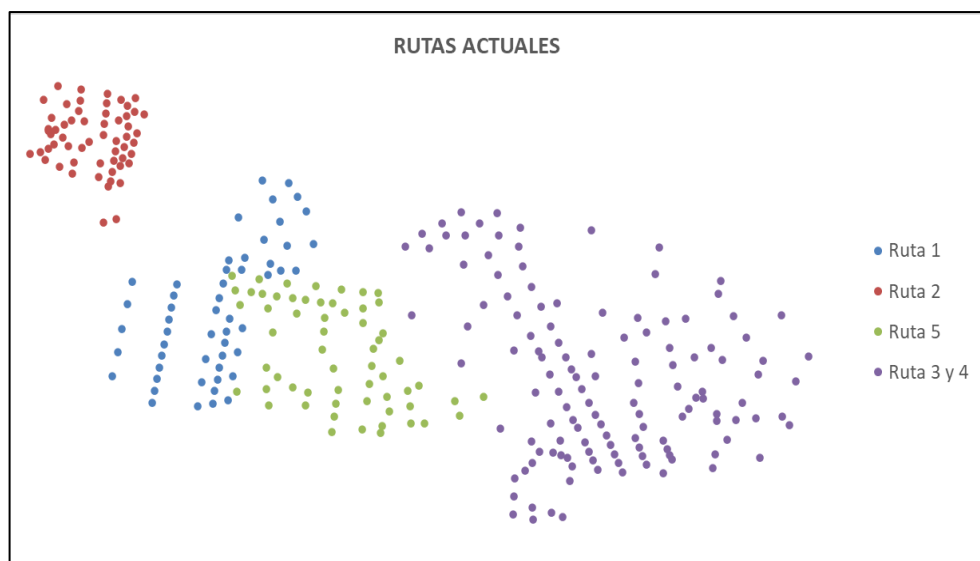
Tabla 3. Rutas de recolección actuales.

Nº Ruta	Barrios	Nº Calles	Día de recolección	Jornada	Vehículo
1	Barrio 3, Barrio 4	53	Lunes	Noche	Camión sencillo
2	Barrio 2, Barrio 1	59	Miércoles	Noche	Camioneta
3	Barrio 6, Barrio 7	128	Viernes	Mañana	Camioneta
4	Barrio 6		Viernes	Tarde	Camioneta
5	Barrio 5	60	Viernes	Noche	Camión sencillo

Fuente: Elaboración propia en base a los datos suministrados por la empresa.

Cada una de las rutas mencionadas anteriormente, fueron recorridas junto al respectivo vehículo con el fin de identificar factores claves a tener en cuenta durante el diseño de las nuevas rutas, como posibles alternativas para los recorridos, condiciones de cada barrio y características de cada sector. A continuación, se muestran la agrupación actual y para observar cada recorrido consultar el [anexo A](#).

Ilustración 3. Rutas actuales de recolección.



Fuente: Elaboración propia por medio de Excel.

6. FORMULACIÓN MATEMÁTICA PARA EL PROBLEMA CARP

De acuerdo con las condiciones actuales de la empresa donde la frecuencia de recolección en el horizonte de una semana es igual a uno para cada ruta puesto que el volumen del material recolectado no es lo suficiente considerable, el problema del PCARP se reduce al problema CARP, además el problema PCARP es considerado como un problema NP-Hard, por ende, no se puede pretender resolver un problema real como el del caso que se aborda en este trabajo debido a que en instancias reales el problema se vuelve intratable computacionalmente.

En ese sentido, en esta sección se presenta la formulación matemática que representa el problema de enrutamiento de vehículos capacitados sobre arcos para un periodo, cuyo objetivo es minimizar la distancia recorrida, el cual tiene como base el modelo realizado por Golden (1981).

Veamos ahora la formulación como un problema de programación lineal. Sea:

Conjuntos

$$\begin{aligned} N &= \{1 \dots n\} \text{ Conjunto de nodos} \\ K &= \{1 \dots k\} \text{ Conjunto de vehículos} \\ A &= \{1 \dots m\} \text{ Conjunto de arcos} \end{aligned}$$

Parámetros

D_{ij} : Distancia del nodo i al nodo j

Q_{ij} : Demanda de la tarea $[i, j]$

W_k : Capacidad del vehículo k

Variables

$X_{ijk} = \{1, \text{ si el arco } [i, j] \text{ es atravesado de } i \text{ a } j \text{ por un vehículo } k$
 $0, \text{ de lo contrario}$

$L_{ijk} = \{1, \text{ si el vehículo } k \text{ atiende la tarea } [i, j]$ $0, \text{ de lo contrario}$

$F_{ijk} \geq 0$, Variable de flujo, que toma un valor positivo si $X_{ijk} = 1$

Función Objetivo

$$\text{Minimizar Distancia: } \sum_{[i,j] \in N} \sum_{k=1}^K D_{ij} X_{ijk} \quad (1)$$

Restricciones

Continuidad del viaje

$$\sum_{[f,i] \in N, i \neq f}^n X_{fik} = \sum_{[i,f] \in N, i \neq f}^n X_{ifk} \quad \forall k = 1 \dots K \quad (2)$$

Cada tarea se debe atender exactamente una vez.

$$\sum_{k=1}^k (L_{ijk} + L_{jik}) = \left\lceil \frac{Q_{ij}}{W_k} \right\rceil \quad \forall [i,j] \in N \quad (3)$$

Una tarea no puede recibir servicio si no se atraviesa.

$$\sum_{k=1}^k X_{ijk} \geq \sum_{k=1}^k L_{ijk} \quad \forall [i,j] \in N \quad (4)$$

Se debe respetar la capacidad del vehículo k.

$$\sum_{[i,j] \in N}^k L_{ijk} * Q_{ij} \leq W_k \quad \forall k \quad (5)$$

Eliminación de subtour.

$$\sum_{f=1, i \neq f}^n F_{ifk} - \sum_{f=1, i \neq f}^n F_{fik} = \sum_{j=1}^n L_{ijk} \quad \forall i \in N, i > 1, \forall k \quad (6)$$

$$F_{ijk} \leq n^2 * X_{ijk} \quad \forall (i,j) \in N \quad (7)$$

La función objetivo (1) utilizada en este modelo busca minimizar la distancia total recorrida por los 2 vehículos, saliendo del depósito y regresando al mismo cuando se finaliza cada ruta.

En la restricción (2) se asegura la continuidad del viaje. La restricción (3) garantiza que cada arco con demanda positiva (tarea) debe ser recogida solo una vez. La ecuación (4) avala que un arco (i,j) no puede recibir servicio si no se atraviesa por el vehículo k. La restricción (5) asegura que el material recolectado no supere la capacidad del vehículo k. El conjunto de ecuaciones (6) y (7) prohíben la formación de sub-ciclos o sub-tours.

7. HEURÍSTICA PROPUESTA PARA EL PCARP

Como el PCARP es un problema NP – hard se convierte en un desafío computacional puesto que es un problema de optimización combinatoria y el tamaño del problema caso de estudio lo hace intratable, por lo cual se hace necesario establecer una estrategia de solución que pudiese ser resuelta en un tiempo razonable. Aunque actualmente existen diversas estrategias, algunas de estas se siguen quedando cortas para el tamaño de algunos problemas que se enfrentan actualmente.

El modelo exacto normalmente en algunos referentes resuelve en tiempo razonable para instancias pequeñas del problema (no más de 30 nodos o 60 arcos), tal como lo presentan Chu, Labadi & Prins, 2004; por esta razón, se ha desarrollado una heurística basada en algoritmos como Dijkstra, Fleury y KMeans, tal como se detallará a continuación.

Considerando que el problema caso de estudio resulta computacionalmente intratable, se propone una heurística que divide el problema en dos fases; la primera fase con el propósito de abordar las limitaciones de capacidad de los vehículos y la periodicidad en la recolección, y la segunda fase con el fin de encontrar los recorridos para cada ruta, tal como como se puede observar en la [ilustración 4](#).

- a) **Primera fase:** Definición de clusters. Para realizar la asignación de un día a cada tarea y recoger teniendo en cuenta la capacidad de los vehículos se utilizó una estrategia de agrupación (Clusterización) conocido como KMeans, el cual es un algoritmo utilizado para agrupar un número n de datos en k clusters. El valor de k es previamente definido por la persona de acuerdo a su necesidad, e inicialmente cada punto se asignará a un cluster teniendo en cuenta la distancia hasta el centroide más cercano; posteriormente se recalcularán los centroides en base a las asignaciones realizadas en el paso anterior, y de manera iterativa se repetirán estos pasos hasta minimizar la distancia o hasta que las asignaciones de cada punto no cambien.

En el presente caso, el algoritmo KMeans tendrá como datos de entrada las coordenadas de latitud y longitud de las calles de cada barrio, posteriormente, cada cluster representa un grafo el cual será asignado al vehículo que mejor se adapte teniendo en cuenta las restricciones de capacidad de ambos vehículos (1,000 y 3,000 kg), también será asignado a uno de los 3 días de la semana disponibles y a una jornada del día (mañana, tarde o noche) dependiendo del tamaño del cluster (número de tareas a recorrer).

- b. **Segunda fase:** Programa de rutas por cluster. Una vez generados los clusters, para establecer cómo debe realizar el recorrido cada vehículo, se trató cada uno de ellos como el problema del cartero chino (CPP), utilizando los algoritmos Dijkstra, el cual es utilizado para encontrar el camino más corto de un punto i a un punto j , y el algoritmo de Fleury que se utiliza para encontrar circuitos eulerianos en un grafo, para finalmente determinar el orden en que se deben visitar cada una de las tareas minimizando la distancia recorrida.

Ilustración 4. División del problema PCARP en fases, estrategia de solución.



Fuente: Elaboración propia.

7.1 DESARROLLO DE LA HEURÍSTICA

A partir de la heurística propuesta, a continuación, se describen los pasos propuestos para llevar a cabo cada una de las fases las cuales fueron desarrolladas en el lenguaje de programación Python, por medio de ambientes como Jupyter Notebook y Pycharm, los cuales sirvieron como herramientas para encontrar la solución del problema:

7.1.1 Definición de los clústeres

Como se explicó anteriormente, se ha utilizado el algoritmo K-means, el cual agrupa cada objeto, en este caso calles, en k grupos, llamados clusters, que son definidos por el usuario y así, simplificar el problema con lo que respecta a las tareas a visitar en cada periodo y el vehículo asignado, visto que no se puede superar la capacidad de este en el recorrido.

Pasos:

1. Definir los nodos y calles que pertenecen a cada barrio de la zona sur, los cuales harán parte de algún cluster, las demás calles formarán parte de las vías de acceso a cada cluster (Vías de entrada y salida a cada barrio).
2. Enumerar los nodos (esquinas) de las rutas que realiza la empresa caso de estudio.
3. Hallar las coordenadas (latitud y longitud) intermedias entre cada nodo (i,j), es decir, las coordenadas de los centros de cada una de las calles ([Anexo B](#)).
4. Asignar a cada arco (calle) un valor de demanda de acuerdo con el barrio al que pertenece, y a los datos mostrados en la tabla 2.

5. Con ayuda de la librería *Sklearn.cluster* del lenguaje de programación Python se importó el algoritmo K-means y se ajustó para que recibiera como datos de entrada las coordenadas de los centros de cada una de las calles, y las agrupara para minimizar la suma de las distancias que hay entre cada centro de las calles y el centroide de cada cluster, es importante mencionar que no se tuvieron en cuenta para la agrupación, las calles de acceso por ser vías comunes, por lo cual no se consideraron dentro de los datos de entrada al algoritmo. El algoritmo de KMeans se presenta en el [anexo C](#).
6. Una vez obtenidos los clusters se revisó la demanda acumulada de cada uno, de acuerdo con la demanda promedio obtenida por calle en el paso 4, con el fin de garantizar que no se exceda la capacidad de los vehículos y así asignar cada cluster a un vehículo.
7. Evaluar la posibilidad de incluir arcos o calles adicionales dentro de cada cluster que actualmente no son recorridas por la empresa y pueden ser incluidas de manera estratégica dentro de la ruta, tal como se evidenció durante los recorridos realizados con la empresa, y así generar posibles nuevas soluciones. Así mismo, cuando se observan las fronteras de cada cluster y se evidencia que existen dos calles que convergen en un mismo nodo, se evalúa la viabilidad de dejar ambas en un solo cluster o en ambos de ser necesario, por medio de la segunda fase.

7.1.2 Diseño de rutas

Al obtener los clusters conformados, se han utilizado los algoritmos de Dijkstra y Fleury para evaluar las rutas de entrada y salida de cada cluster y trabajar cada uno como el CPP respectivamente, y así solucionar cómo se debe realizar el recorrido teniendo como objetivo disminuir la distancia transitada. Para consultar la forma matemática de estos algoritmos lo invitamos a consultar el [anexo D](#) y el [anexo E](#) respectivamente.

Pasos:

8. Determinar la distancia de cada uno de los arcos (i,j) que conforman cada cluster, estableciendo como inicio el nodo i y como final el nodo j mediante la herramienta Google Maps.
9. Escoger un nodo de inicio y fin para cada cluster, a partir del cual se buscará la menor distancia de entrada (empresa – nodo inicial) y de salida (nodo final – empresa).
10. Evaluar mediante el algoritmo Dijkstra las rutas de entrada y salida para cada clúster.
11. Se determina si el grafo (cluster) es euleriano, mediante la librería Network en Python, es decir, se evalúa si todos sus nodos son de grado, es decir, cada nodo debe

tener un número par de arcos o calles para garantizar un inicio y fin en un mismo punto.

12. Si el grafo es euleriano, se procede a resolverlo mediante el algoritmo de Fleury, el cual permite encontrar el circuito euleriano. En caso contrario, para convertir cada cluster en un grafo euleriano, se deben eliminar los nodos impares (los nodos que tienen un número impar de arcos o calles).
13. Para eliminar un nodo impar debe agregarse otro arco (calle) sin embargo, y como es evidente, esto tiene un costo asociado, por lo cual, se plantea como objetivo determinar qué arcos repetir para que se eliminen todos los nodos impares, con el mínimo coste.
14. Posteriormente, se encuentran todas las combinaciones posibles de la paridad de nodos impares, por medio del algoritmo Dijkstra y se encuentra el costo de la ruta mínima entre esos pares.
15. Se procede a modificar el grafo con estos nuevos bordes paralelos, para tener un nuevo grafo con nodos pares en su totalidad y garantizar la condición de Euleriano.
16. Finalmente, se tiene un grafo en el cual se podrá encontrar un ciclo Euleriano con solo nodos pares, para lo cual se utilizó el algoritmo de Fleury.

Nota: Se repite del paso 4 al 9 de acuerdo con las consideraciones realizadas en el paso 7 de la fase 1.

Para facilitar la visualización del código desarrollado en Python para abordar la heurística propuesta, se ha consolidado en Google Colab donde se podrá interactuar con el código ([Ver anexo F](#)).

A continuación, se presenta el pseudocódigo del algoritmo utilizado para la heurística propuesta, en el cual están presentes las dos fases mencionadas anteriormente, siendo de la línea 1 a la 20 los pasos de la fase 1, y a partir de la línea 21 se aborda la fase 2:

Ilustración 5. Pseudocódigo de la heurística propuesta.

ALGORITMO: HEURÍSTICA PROPUESTA PARA EL PCARP

INICIO:

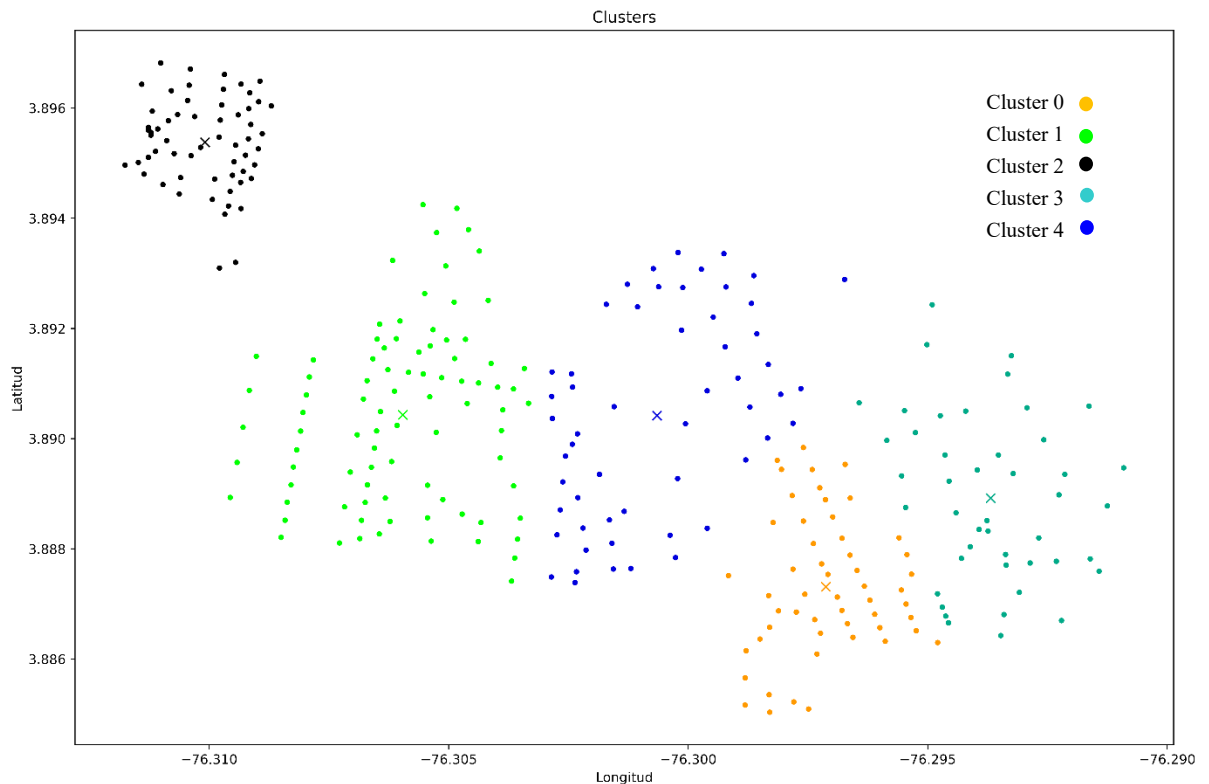
```
1: Coordenadas de las calles  $\rightarrow (i,j)$ 
2: Número de clúster  $(n) = 5$ 
3: Algoritmo KMeans  $\leftarrow \{ \text{Coordenadas\_calles } (i,j), \text{ Número de clúster} \}$ 
4: Clúster_5 /Archivo con las calles asignadas a un clúster
5: For  $n \rightarrow \text{Número de clúster do}$ :
6:   For  $k \rightarrow \text{Clúster\_5 do}$ :
7:     If  $n == \text{Clúster\_5 } (k)$ :
8:       Dem_cluster  $(n) = \text{Dem\_cluster } (n) + \text{Clúster\_5 } (k)$ 
9:     End If
10:   End For
11: End For
12: Programación
13: For  $n \rightarrow N\_clúster do$ :
14:   For  $l \rightarrow \text{Vehículo do}$ :
15:     If Dem_acum  $(n) \leq \text{Vehículo } (l)$ 
16:       Programación  $(n,l,d,j)$ 
17:     Next  $n$ 
18:   End If
19: End For
20: End For
21: entradas_posibles
22: salidas_posibles
23: For  $n \rightarrow N\_cluster do$ :
24:   Determinar nodo de inicio y fin  $(p), \{n\}$ 
25:   For  $p \rightarrow \text{Nodo do}$ :
26:     Algoritmo de Dijkstra  $\{ \text{entradas posibles, empresa, } p \}$ 
27:     Algoritmo de Dijkstra  $\{ \text{salidas posibles, } p, \text{ empresa} \}$ 
28:     Rutas  $(n, \text{entrada, salida})$ 
29:   Next  $n$ 
30: Rutas /Archivo de rutas
31: For  $n \rightarrow N\_cluster do$ :
32:    $n \leftarrow \text{Test para determinar si es euleriano}$ 
33:   If Test == True:
34:     Algoritmo de Fleury  $\rightarrow \{n\}$ 
35:     Rutas  $(n, \text{entrada, secuencia, salida})$ 
36:   Else:
37:     Eliminar imparidad en nodos  $\rightarrow \{n\}$ 
38:     Evaluar combinaciones de mínimo costo por Dijkstra  $\rightarrow \{n\}$ 
39:     Agregar nuevos arcos al grafo  $\rightarrow \{n\}$ 
40:     Algoritmo de Fleury  $\rightarrow \{n\}$ 
41:     Rutas  $(n, \text{entrada, secuencia, salida})$ 
42:   End If
43: End For
FIN
```

Fuente: Elaboración propia.

8. VALIDACIÓN DE LA HEURÍSTICA Y ANÁLISIS DE RESULTADOS

Para el problema se trabajó con un total de 5 cluster, en vista de que la empresa tiene planificado trabajar 3 días a la semana en diferentes jornadas: lunes en la noche, miércoles en la noche y el día viernes en las tres jornadas. Pero lo más importante aún, es que establecer un número de clusters inferior generaría recorridos demasiado extensos, y en el caso contrario, un número mayor de clusters generaría recorridos demasiado cortos como se pudo evidenciar durante la fase experimental del algoritmo KMeans. El desarrollo del algoritmo tuvo como inputs las coordenadas de las calles de cada barrio, y el resultado obtenido se muestra en la siguiente ilustración, donde cada color representa un cluster.

Ilustración 6. Clusters generados con KMeans.



Fuente: Elaboración propia por medio de Python.

Después de obtener los resultados de KMeans, se pudo observar que la ruta 2 se mantiene igual, es decir, el algoritmo agrupa las calles de los barrios 1 y 2 en un mismo cluster, distinguidos en la anterior ilustración con el color negro; esto se debe principalmente a que la localización de estas calles es relativamente lejana con respecto a los demás barrios que se encuentran más cercanos entre sí. Para las demás rutas hubo cambios significativos, pues el

algoritmo dividió a barrios como el 5 y el 6 en tres partes y asignó cada una a un clúster diferente, quedando las nuevas rutas con las características que se muestran en la [tabla 4](#).

Tabla 4. Clusters generados por KMeans

Nº Cluster	Nº Barrios	Nº Calles	Demanda Total
0	Barrio 5, Barrio 6 y Barrio 7	60	375.8
1	Barrio 3, Barrio 4 y Barrio 5	80	1,133.2
2	Barrio 1 y 2	59	769.8
3	Barrio 6	46	298.0
4	Barrio 5 y Barrio 6	55	411.3
Total		300	2,988.0

Fuente: Elaboración propia.

A partir de lo anterior, se puede observar como el cluster 1 resulta ser el de mayor tamaño por el volumen de calles que agrupa el algoritmo KMeans, y esto se refleja en la demanda estimada que se debe considerar para dicho clúster. Para los demás clusters, el número de tareas consideradas se mantienen cercanas a las rutas actuales, por lo cual, se podría estimar un tamaño similar del cluster, sin embargo, las rutas de recolección evidentemente serán distintas. Según los resultados, en cuanto a la demanda, no se tiene inconvenientes inicialmente, puesto que cualquier clúster se puede asignar a cualquier vehículo, a excepción del 1, el cual según su demanda estimada será necesariamente obligatorio asignarlo al camión, y aunque para los demás no existe este problema como ya se mencionó anteriormente, se decide asignar los dos clústeres de mayor demanda (1 y 2) al camión y los demás a la camioneta, esto con el fin de evitar posibles inconvenientes de capacidad por variaciones inesperadas en la demanda.

Una vez definidos los cluster a visitar por día, la jornada y el vehículo encargado de realizar el trayecto, se seleccionaron las posibles calles adicionales a incluir en los recorridos de cada cluster, para ser evaluadas en la última fase dentro de la solución mediante n iteraciones en Python hasta obtener la mejor solución.

Con el KMeans ya resuelto y toda la fase 1 finalizada, se consideraron varias vías de entrada y salida a cada cluster y fueron evaluadas por medio del algoritmo Dijkstra, utilizado para encontrar el camino más corto desde un nodo de origen a uno de destino, y de ese modo tomar un criterio para seleccionar el nodo de inicio y fin de cada ruta y así mismo cómo llegar a cada cluster partiendo de las instalaciones de la empresa; cabe destacar que el camino de llegada al nodo de inicio de cada cluster no necesariamente es el mismo de salida ya que hay un factor fundamental en esta decisión que es el sentido vial de cada calle. A continuación, se observa la solución obtenida donde la ruta indica la secuencia en que se pasa por cada nodo; para visualizar el resultado obtenido en Python [Ver anexo G](#).

Tabla 5. Distancia recorrida para salir y llegar a la empresa desde el inicio del cluster.

Nº Cluster	Ruta de entrada	Distancia (Metros)	Ruta de salida	Distancia (Metros)
0	1-2-5-18-37-75-82-81-100-372-374-316	4,329.32	316-374-378-379-377-102-103-81-82-75-37-18-5-2-1	4,557.42
1	1-2-5-18-37-75-82-81-100-367-370-371-363-366-261-163	3,853.69	163-366-363-371-364,367-101-102-103-81-82-75-37-18-5-2-1	3,852.17
2	1-2-5-18-37-75-82-81-100-367-368-369-260-152	3,933.17	152-260-369-364-367-101-102-103-81-82-75-37-18-5-2-1	3,970.25
3	1-2-5-18-37-75-82-81-100-372-375-374-316	4,329.32	316-374-378-379-377-102-103-81-82-75-37-18-5-2-1	4,557.42
4	1-2-5-18-37-75-82-81-100-367-370-371-363-366-261-202-203-266-265	4,383.34	265-266-203-202-366-363-371-364-367-101-102-103-81-82-75-37-18-5-2-1	4,268.49
Total		20,828.84		21,205.75

Fuente: Elaboración propia.

En este caso, los resultados obtenidos mostraron que la totalidad de las rutas tanto de entrada como de salida a cada cluster, deben hacerse a través de la ciudad y no de vías intermunicipales como la variante, debido a que factores como la distancia de los retornos puede generar recorridos muy extensos e innecesarios. Adicionalmente, la distribución de los clusters ayudó a la determinación de entradas y salidas mucho más eficientes debido a que se ubicaron en puntos clave de ingreso a los barrios.

Al llegar a este punto, resta resolver el problema del CPP teniendo en cuenta las calles adicionales incluidas, lo cual se evaluó a través de los pasos 4,5,6,7,8 y 9 de la fase 2 descritos con anterioridad, obteniendo finalmente la secuencia de los nodos que conforman el circuito euleriano con sus respectivas distancias y tiempos de solución, como se puede evidenciar en la [tabla 6](#):

Tabla 6. Recorrido, distancias y tiempos de ejecución de cada cluster.

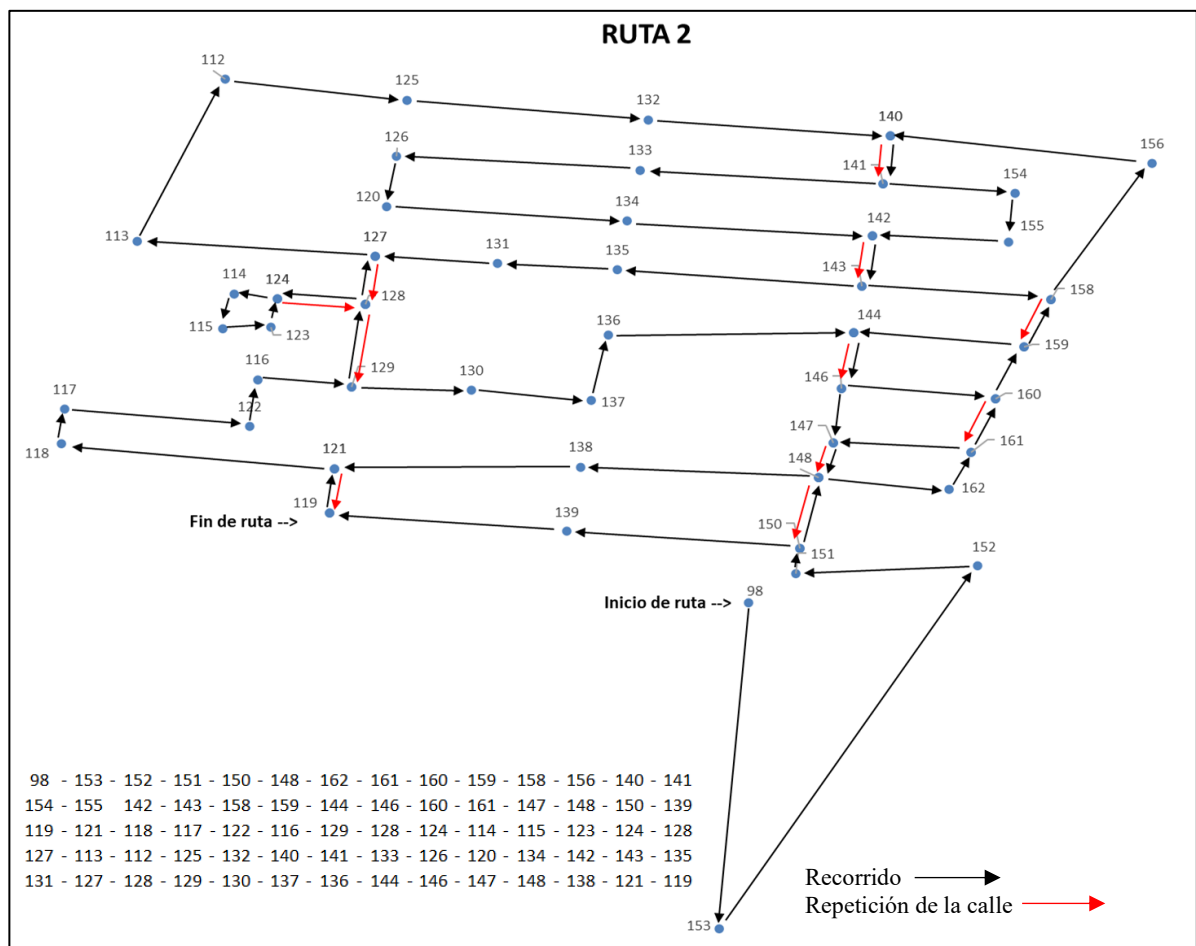
N° Cluster	Secuencia de los nodos	Distancia (Metros)	Tiempo computacional (Seg)
0	316-309-308-307-274-275-307-306-357-305-304-305-340-305-357-276-277-304-303-278-279-302-303-349-362-349-301-300-362-300-299-298-291-290-289-288-287-283-284-285-294-358-359-358-286-285-284-295-296-360-296-297-296-293-292-291-292-297-298-351-353-297-353-354-353-351-350-299-281-280-301-302-279-282-258-259-275-306-330-328-329-309-316	7,947.79	2
1	163-164-200-164-165-166-167-168-169-170-167-170-171-172-173-174-175-176-177-178-179-192-191-182-181-180-179-178-193-194-175-174-195-196-171-196-232-262-233-234-238-236-235-237-189-186-187-188-187-186-185-184-183-182-183-190-189-237-241-237-235-236-238-234-233-262-232-222-221-220-217-216-215-214-213-207-199-165-166-198-197-209-208-211-210-212-213-207-206-218-219-204-203-202-201-205-206-200-201-202-163	9,315.93	154,260
2	152-153-98-151-150-139-119-121-118-117-122-116-129-121-138-148-147-161-160-159-144-136-137-130-129-128-124-123-115-114-124-128-127-131-135-143-158-159-158-156-140-141-140-132-125-112-113-127-131-135-143-142-155-154-141-133-126-120-134-142-143-144-146-147-146-160-161-162-148-150-151-152	3,798.27	2
3	316-317-318-321-322-323-324-325-326-327-328-327-331-332-338-339-330-340-341-342-348-346-348-343-344-345-352-351-353-354-355-356-380-295-380-360-354-353-351-350-351-362-349-348-347-346-344-345-361-337-336-335-334-333-332-331-327-322-321-320-319-318-317-316	7,784.54	1
4	265-268-267-314-315-314-313-312-311-310-309-272-273-263-365-225-224-223-222-223-224-225-226-227-228-229-230-231-232-262-243-231-243-244-252-244-233-244-245-239-240-241-247-246-245-251-250-246-240-241-242-248-249-247-248-249-254-255-258-259-257-254-253-256-257-259-263-273-308-309-310-271-270-311-312-313-269-268-269-264-265	8,363.83	336
Total		37,210.36	154,601

Fuente: Elaboración propia.

Como se evidencia en la anterior tabla, el cluster 1 es el de mayor distancia y significativamente mayor a los demás cluster, pero el factor más importante resulta ser el tiempo de ejecución para encontrar el menor recorrido sobre todo en el cluster 1, donde Python se demoró aproximadamente 43 horas en encontrar la solución, pues al evaluar las distintas alternativas sobre las calles a agregar al recorrido, las distancias mejoraron, pero el tiempo de solución fue significativamente mayor, por ejemplo, al incluir tan solo una calle que unía al barrio 3 con el barrio 5, la solución mejoró en aproximadamente 500 metros que anualmente representa un ahorro de 26 km en tan solo una ruta, sin embargo, el tiempo de solución pasó de 2 segundos a casi 330 segundos. Así, el tiempo de solución obtenido corrobora también la complejidad del problema, y que abordar el problema en una instancia tan grande lo hace imposible de resolver computacionalmente.

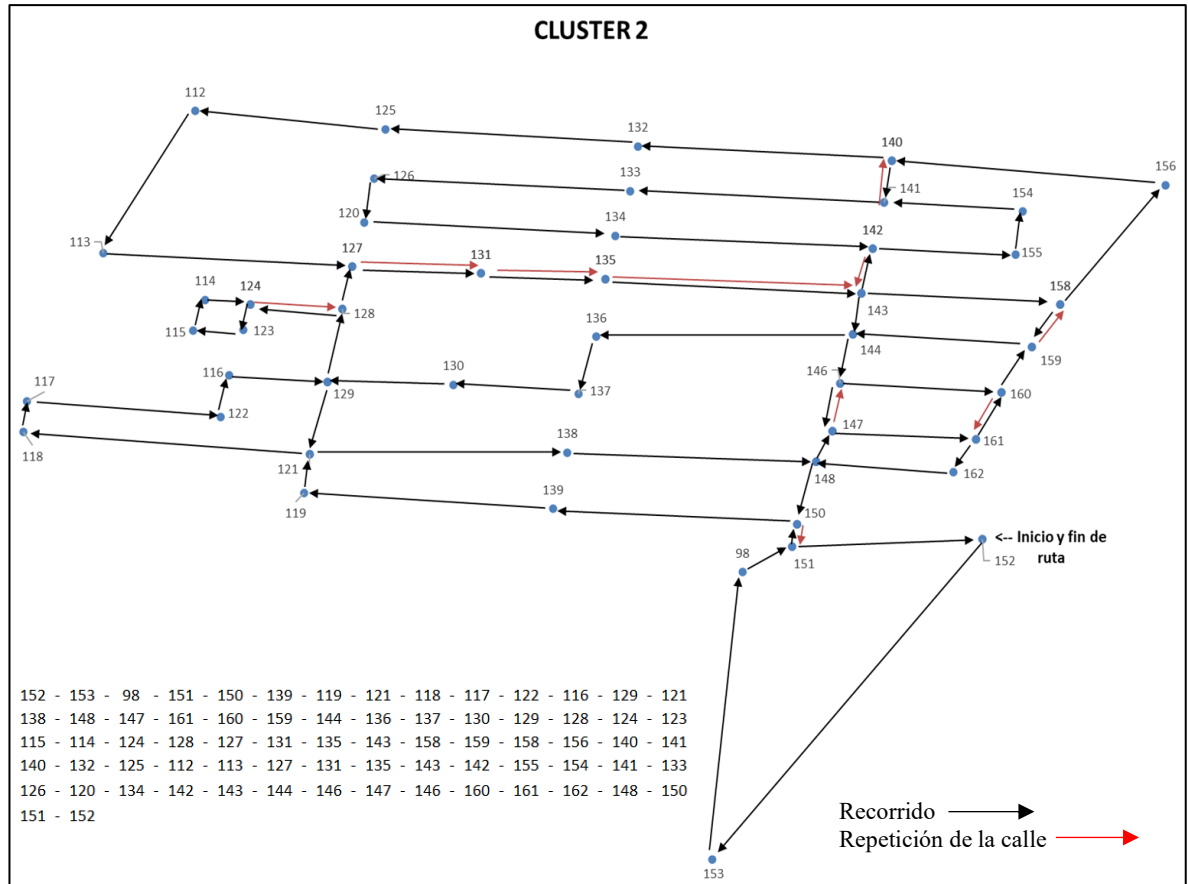
A continuación, para efectos comparativos, se muestran las ilustraciones de la ruta 2 actual y la del cluster 2 propuesta, las cuales comparten los mismos barrios y calles a recorrer:

Ilustración 7. Recorrido actual de la ruta 2.



Fuente: Elaboración propia.

Ilustración 8. Recorrido propuesto para el cluster 2.



Fuente: Elaboración propia.

Las soluciones mostradas en Python podrán ser consultadas en el [anexo H](#), y las demás gráficas de los recorridos en el [anexo I](#).

Aunque el tiempo de solución más largo que se obtuvo parece ser muy alto, teniendo en cuenta que actualmente se recorren casi la totalidad de las calles, y que, según lo evaluado con la empresa, la dinámica del reciclaje en la sociedad por lo menos en la ciudad en un corto o mediano plazo no parece tener la tendencia a cambiar significativamente, los tiempos de solución de la heurística no representan un retraso en su planeación si la empresa decide implementarla.

Después de culminar la fase 2 de la heurística, con los clusters y secuencias ya definidas, la programación propuesta para la recolección teniendo en cuenta las capacidades de los vehículos, los días y jornadas disponibles para realizar el proceso de recolección, se presenta en la tabla 7:

Tabla 7. Programación propuesta para la recolección.

N° Cluster	Barrios	# Calles	Demanda Total	Día asignado	Jornada	Vehículo
0	Barrio 5, Barrio 6 y Barrio 7	67	388.7	Viernes	Mañana	Camioneta
1	Barrio 3, Barrio 4 y Barrio 5	80	1,145.6	Lunes	Tarde	Camión sencillo
2	Barrio 1 y 2	61	796.2	Miércoles	Noche	Camión sencillo
3	Barrio 6	51	330.4	Viernes	Tarde	Camioneta
4	Barrio 5 y Barrio 6	65	470.4	Viernes	Noche	Camioneta
Total		324	3,131.3			

Fuente: Elaboración propia.

Para la asignación de la jornada, se tuvo en cuenta como principal factor la distancia a recorrer por cada cluster, es decir, para los recorridos más extensos como en el caso de los clusters 1 y 4, se eligieron las jornadas de la mañana y de la tarde, y para los recorridos de menor distancia, la jornada de la noche.

Para demostrar qué tan buenos eran los resultados de la heurística propuesta a comparación del método empírico utilizado actualmente por la empresa, a continuación, se realiza una comparación en términos de distancia total entre los recorridos actuales y las distancias que generaría la programación propuesta anteriormente, para la recolección del material reciclable:

Tabla 8. Distancias recorridas actuales vs propuestas.

Recorrido actual		Recorrido de la solución propuesta	
N° de Ruta	Distancia total recorrida (Km)	N° Cluster	Distancia total recorrida (Km)
1	16,054.55	0	16,834.53
2	13,684.90	1	17,021.79
3	21,082.27	2	11,701.69
4	17,425.56	3	16,671.28
5	16,650.51	4	17,015.66
Total	84,897.79		79,244.95

Fuente: Elaboración propia

Al comparar los recorridos actuales con la solución propuesta obtenida a través de la heurística, teniendo en cuenta las distancias de las entradas y salidas en ambos escenarios, se puede comprobar que se logra una mejora significativa, además de lograr que los recorridos sean más homogéneos en distancia, lo cual favorece evitando jornadas muy largas de trabajo como sucede con la ruta 3 en comparación a las demás.

Ahora, se comparan los resultados de manera generalizada para detallar la distancia total ahorrada en metros y de manera porcentual:

Tabla 9. Comparación de la distancia recorrida actual y la solución propuesta.

Distancia total actual recorrida (Metros)	Distancia solución propuesta (Metros)	Diferencia (Metros)	Variación %
84,897.79	79,244.95	-5,652.84	-6.66%

Fuente: Elaboración propia

Como se evidencia en la tabla anterior, se logra una disminución de 5,652.84 metros (5.65 km) que representa porcentualmente un ahorro de la distancia recorrida del 6.66%, lo cual resulta ser muy importante teniendo en cuenta que este valor representa el ahorro de tan solo una semana, lo que implicaría un ahorro anual de aproximadamente 293 kilómetros.

9. COMPARACIÓN ENTRE EL MODELO EXACTO CARP Y LA HEURÍSTICA PROPUESTA

Para comparar el modelo exacto del CARP y la fase 2 de la heurística propuesta con respecto a la secuencia del recorrido y la distancia total obtenida, se resolvieron ambos modelos en una instancia pequeña del problema abordado, con el cluster 2 conformado por los barrios 1 y 2, compuesto por un total de 49 nodos y 59 calles o tareas con un promedio de demanda de 13.2 kg; se incluyó 1 vehículo con capacidad de 1000 kg, dado que la demanda total del cluster es 796.2 kg, y definiendo como punto de inicio el nodo 98. Una vez resueltas ambas instancias, con los resultados obtenidos se analizó el GAP o porcentaje de desviación de la solución obtenida por la heurística con respecto a la obtenida por el CARP, siendo este último modelo abordado en AMPLIDE ([Ver anexo J](#)). A continuación, se presentan los análisis, junto al porcentaje de error el cual es obtenido con la relación solución heurística - CARP) / CARP:

Tabla 10. CARP Vs Heurística (GAP).

Grafo	Distancia total en metros		GAP (Metros)	GAP %
	CARP	Heurística		
Cluster 2	3,798.27	3,798.27	0	0%

Fuente: Elaboración propia.

Como se pudo observar, la distancia obtenida por medio del modelo exacto es la misma a la obtenida por la heurística implementada en la fase 2, obteniendo así un GAP del 0%, es decir, obteniendo la misma distancia en ambos modelos, lo cual resulta ser bastante beneficioso para la estrategia empleada. En cuanto al tiempo de solución, se observó que para la heurística es mucho más corto que para el modelo exacto, con 34 segundos menos, tal como se muestra en la siguiente tabla:

Tabla 11. Tiempos de ejecución en segundos.

Grafo	Tiempo de solución en segundos		Diferencia (Segundos)
	CARP	Heurística	
Cluster 2	2.67	1.5	-1.17

Fuente: Elaboración propia.

Respecto a los resultados de ambos modelos, a continuación, se muestra también un resumen de la asignación del vehículo, el recorrido, la distancia, y la demanda por ruta o cluster:

Tabla 12. Resultados del CARP y la Heurística.

Nº Cluster	Barrios	Demanda acumulada (kg)	Distancia (metros)	Vehículo	Holgura (kg)
2	Barrio 1 y Barrio 2	796.2	3,798.27	Camión sencillo	203.8

Fuente: Elaboración propia.

Finalmente, se muestran los recorridos para detallar las diferencias en la secuencia de nodos a visitar, pues aún obteniendo la misma distancia, los recorridos resultan ser diferentes:

Tabla 13. Resultados de la heurística.

Modelo	Secuencia de los nodos
Exacto	98-153-152-151-150-139-119-121-118-117-122-116-123-128-124-114-115-123-124-128-127-131-127-113-112-125-132-140-141-140-156-158-143-142-134-120-126-133-141-154-155-142-143-135-131-135-143-144-136-137-130-129-121-138-148-162-161-147-146-144-159-158-159-160-146-147-148-150-151-98
Heurística	98-151-150-139-119-121-129-130-137-136-144-159-160-161-160-146-144-143-142-155-154-141-140-132-125-112-113-127-131-135-143-142-134-120-126-133-141-140-156-158-159-158-143-135-131-127-128-124-123-115-114-124-128-129-116-122-117-118-121-138-148-147-146-147-161-162-148-150-151-152-153-98

Fuente: Elaboración propia.

Las ilustraciones de los recorridos tanto del CARP como de la heurística podrán ser consultadas en el en el [anexo K](#).

10. CONCLUSIONES Y RECOMENDACIONES

Al realizar la revisión de la literatura para el problema PCARP se encontró que está catalogado como un problema NP-hard debido al problema combinatorio que presenta y así mismo el costo y el tiempo computacional que requiere. Por lo tanto- el problema del sistema de recolección se redujo al problema de ruteo de vehículo capacitados sobre arcos (CARP) como se recomienda en algunas fuentes, sin embargo, se ha encontrado que el tiempo de solución para instancias consideradas grandes- es excesivo- y en nuestro caso se contaba con 300 nodos- por lo cual- se decidió como estrategia de solución abordar una heurística basada en algoritmos como Dijkstra- Fleury y KMeans.

Realizar la clusterización por KMeans simplificó en gran parte el problema de la capacidad y periodicidad- sin embargo- al igual que sucede con el modelo exacto revisado- cuando se generaron los clusters- en las fronteras algunas vías quedaban sin salida obligando a repetir esa misma vía para continuar el recorrido- lo cual se trató realizando varias consideraciones de agregar o compartir calles de otros clusters para generar un menor recorrido- lo cual desde la herramienta Python no resultó ser difícil.

Para realizar el diseño del programa periódico de rutas para la empresa caso de estudio se formuló una heurística desarrollada en el lenguaje de programación Python- donde los resultados evidenciaron una disminución significativa en la distancia recorrida obtenida por la herramienta propuesta en relación con la situación real de la empresa- lo cual estaría generando un ahorro importante semanalmente del 6.66% en la distancia recorrida.

Los resultados de la heurística vs el primer modelo exacto propuesto por Golden- demostró bajo las condiciones del problema actual- obtener una buena solución- pero el factor más importante es el tiempo para la resolución del problema- pues en este caso- abordar el problema desde un lenguaje de programación contribuyó a obtener una buena solución de acuerdo a las condiciones en un tiempo muy razonable- teniendo en cuenta el horizonte de planeación.

La solución obtenida se propone para un horizonte de seis (6) aproximadamente, a partir de allí si la demanda aumenta la empresa debe realizar un análisis de la dinámica y de ser necesario aplicar la heurística nuevamente.

Para trabajos futuros- se recomienda abordar el problema desde un enfoque de cartero rural- donde se pueda construir el grafo de todas las calles- incluidas las que no se recorren y resolver el problema utilizando algunas heurísticas propuestas en la literatura- con el fin de comparar resultados. También, como algoritmo de solución al CPP para encontrar caminos eulerianos- se recomienda en lugar de Fleury utilizar Hierholzer para comparar cuál algoritmo arroja una mejor solución en términos de distancia.

Finalmente se recomienda en aras de abordar un problema más robusto- incluir otras consideraciones como el consumo de gasolina lo cual podría generar distintas soluciones ya que al recorrer menor distancia no necesariamente implica un menor consumo de

combustible. También se pueden considerar ventanas de tiempo con los horarios de los operarios y los costos asociados a horas extras.

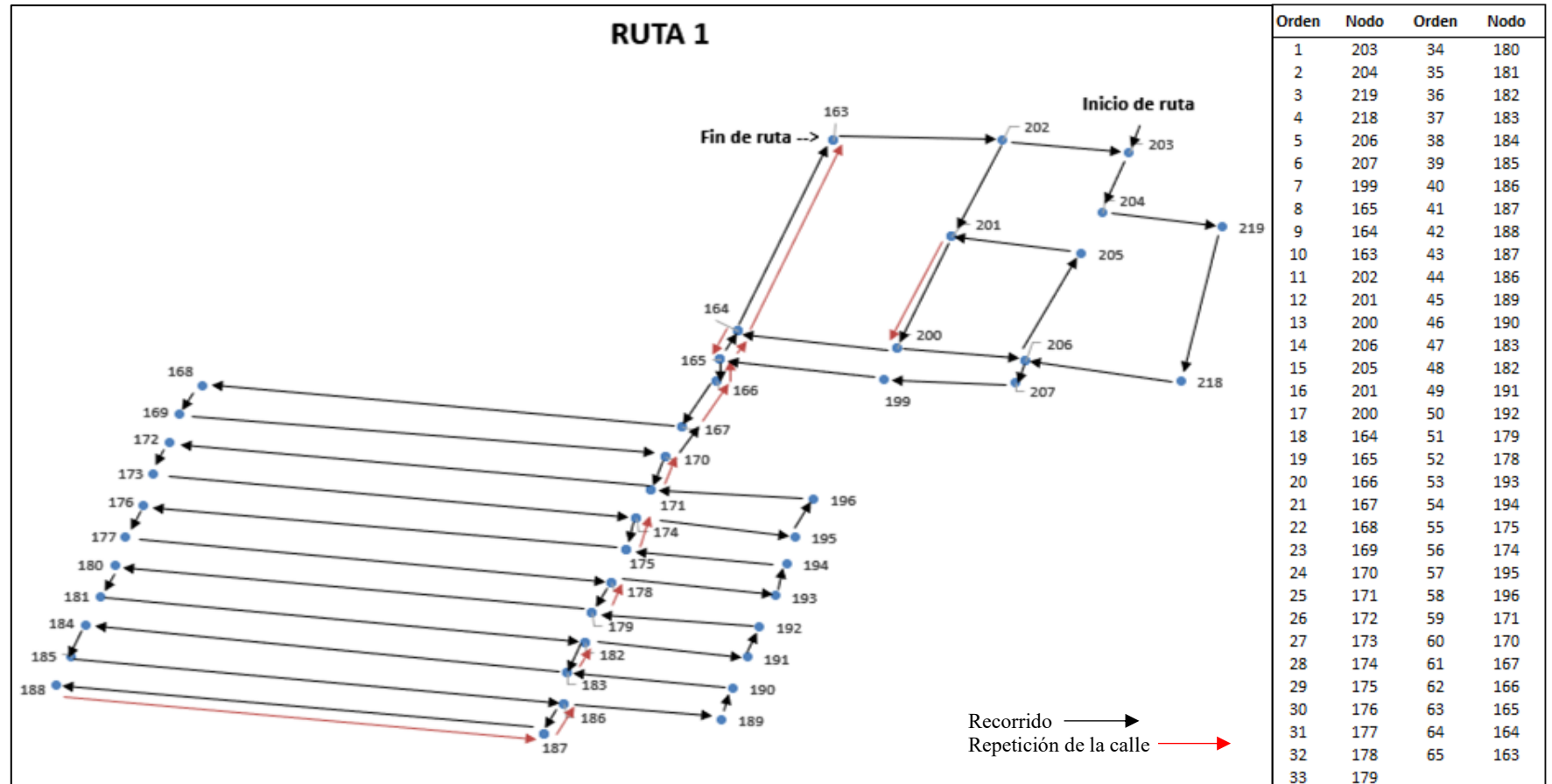
11. BIBLIOGRAFÍA

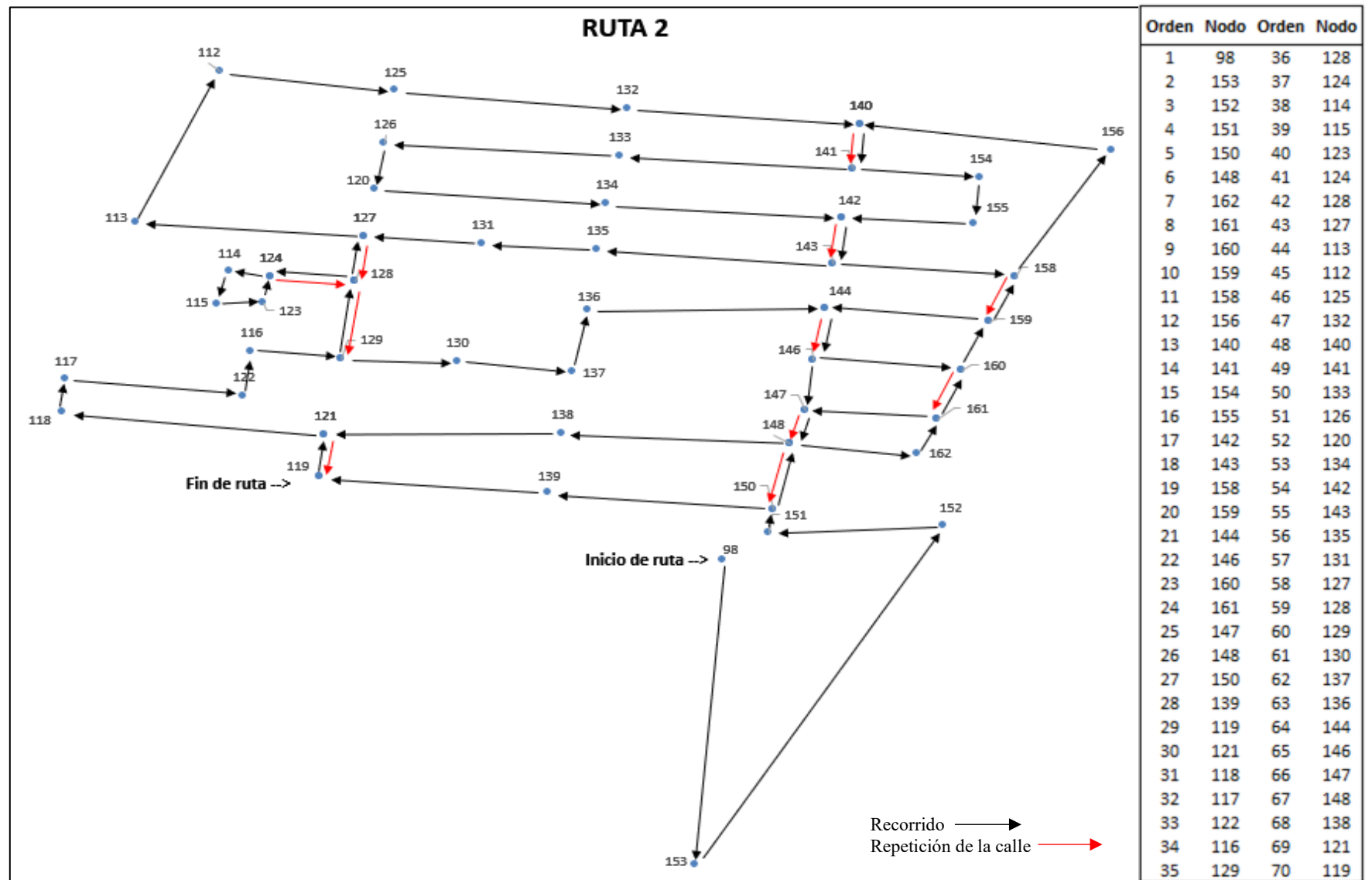
- Ballou- R. H. (2004). Logística: Administración de la cadena de suministro. 5ta ed. México: Prentice Hall.
- Babae Tirkolaee- E.- Alinaghian- M.- Bakhshi Sasi- M.- & Seyyed Esfahani- M. (2016). Solving a robust capacitated arc routing problem using a hybrid simulated annealing algorithm: a waste collection application. *Journal of Industrial Engineering and Management Studies*- 3(1)- 61-76.
- Babae Tirkolaee- E.- Alinaghian- M.- Bakhshi Sasi- M.- & Seyyed Esfahani- M. (2018). A robust periodic capacitated arc routing problem for urban waste collection considering drivers and crew's working time. *Waste Management*- 76- 138-146.
- Bentazo Quezada- E.- Torres Gurrola- M.- Romero Navarrete- J.- & Obregón Biosca S. (2016). Evaluación de rutas de recolección de residuos sólidos urbanos con apoyo de dispositivos de rastreo satelital: análisis e implicaciones. *Revista Internacional de Contaminación Ambiental*- 32(3)- 323-337.
- Calvo- M. (2018). Problemas de rutas de vehículos por arcos. Tesis de fin de grado. Universidad de Sevilla- España.
- Chu- F.- Labadi N.- & Prins C. (2004). A scatter search for the periodic capacitated arc routing problem. *Journal of Systems Science and Systems Engineering*- 13(4)- 423-435.
- Chu- F.- Labadi N.- & Prins C. (2004). The periodic capacitated arc routing problem linear programming model- metaheuristic and lower bounds. *European Journal of Operation Research*- 169- 586-605.
- Golden- L. Bruce.- & Wong R. (1981). Capacitated arc routing problems. *Networks*- vol 11- 306-308.
- Chen- Y.- Hao J. (2017). Two phased hybrid local search for the periodic capacitated arc routing problem. *European Journal of Operation Research*- 264- 55-65.
- Gutiérrez- V.- Palacio- J.- & Villegas- J. (2012). Reseña del Software disponible en Colombia para el diseño de rutas de distribución y servicios. *Revista Universidad EAFIT*- 43(145)- 60-80.

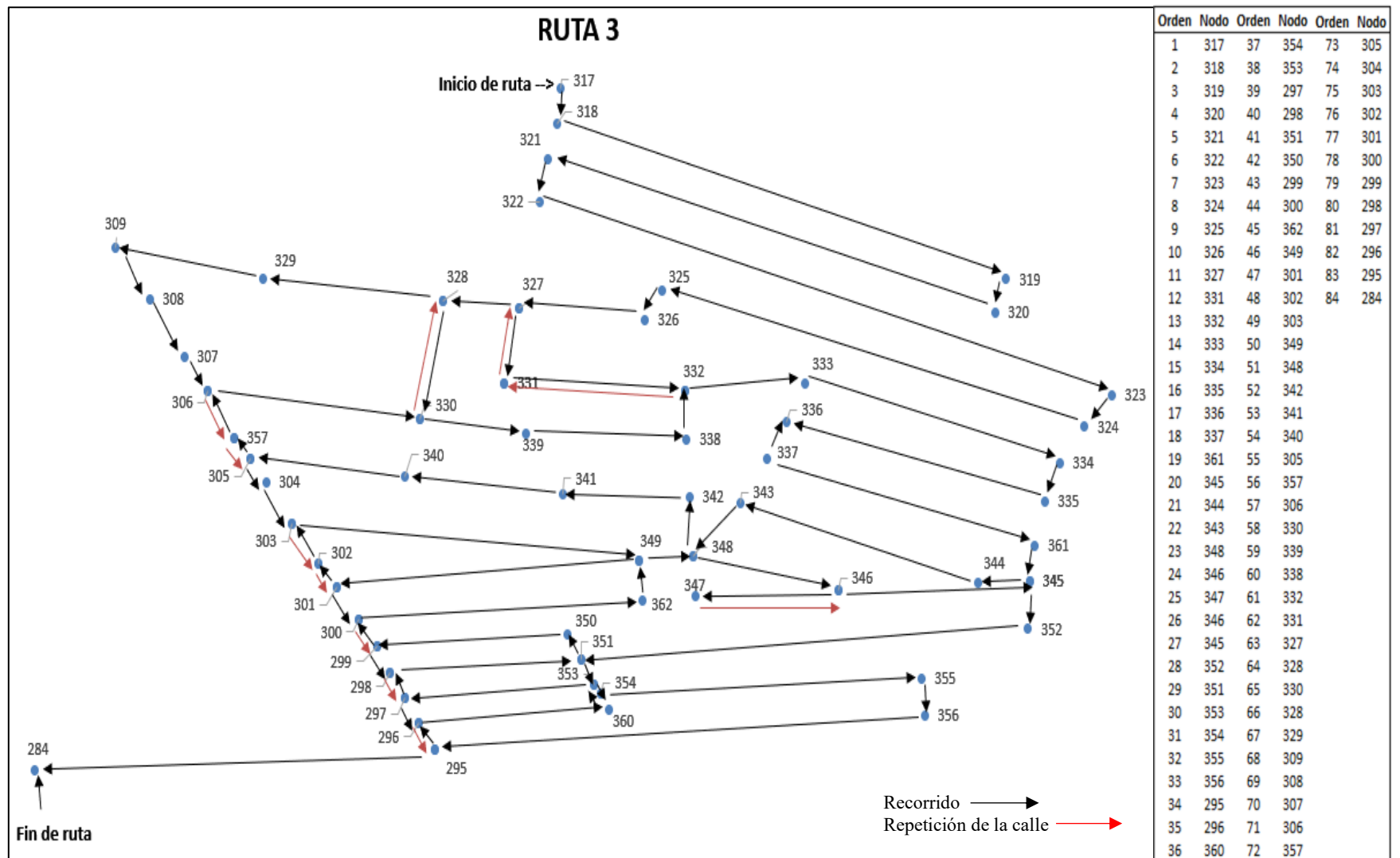
- Golden B.- & Wong R. (1981). Capacitated arc routing problems. *Networks: An international Journal*- 11(3)- 305-315.
- Leblanc- M.- Feng- Y.- & Coombe- G.- (2019). Chinese-Postman. GitHub. <https://github.com/supermitch/Chinese-Postman>
- Martínez- C. (2011). Metaheurísticas híbridas aplicadas al problema de ruteo de arcos capacitados. Tesis doctoral en Ciencias de la computación. Universidad de Buenos Aires- Argentina.
- Martínez- F. (2018). Propuesta de rediseño de macro y micro rutas del sistema de recolección de residuos sólidos de la ciudad de Tulcán. Tesis para el título de ingeniería ambiental. Escuela Politécnica Nacional- Ecuador.
- Pineda- Y. (2015). Rediseño de rutas de recolección de residuos sólidos mediante el problema de Ruteo de Vehículos Capacitados sobre Arcos. Tesis de maestría en Ingeniería. Universidad de Nacional Autónoma de México- México- D.F.
- Yordá- J. (2015). El problema del cartero chino. Máster Interuniversitario en técnicas estadísticas. Universidad de Santiago de Compostela- Universidad de Vigo & Universidad de la Coruña- España.
- Zhang Y.- Mei Y.- Tang K.- & Jiang K. (2016). Memetic algorithm with route decomposing for periodic capacitated arc routing problem. *Applied Soft Computing*- 52- 1130-114

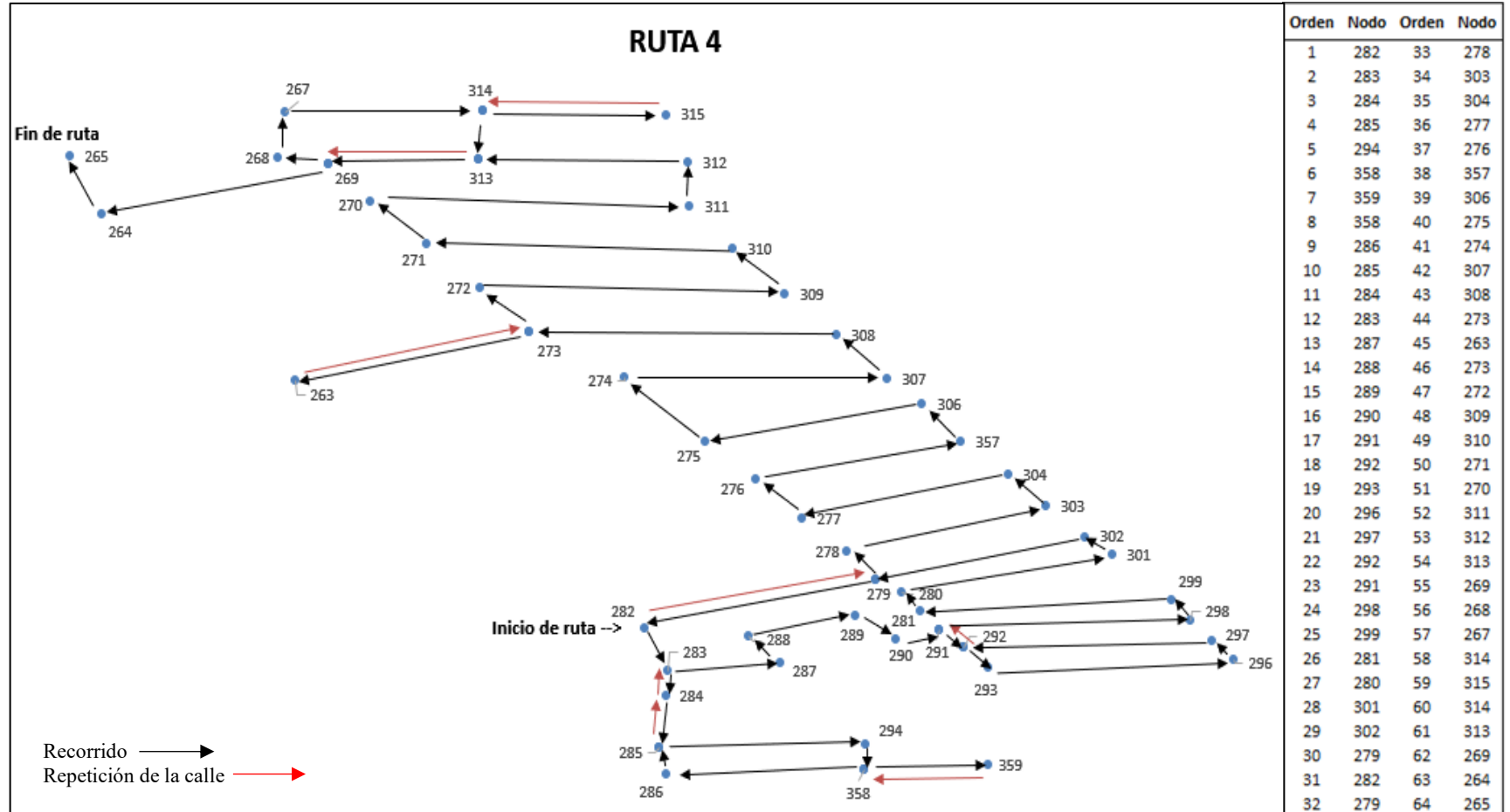
12. ANEXOS

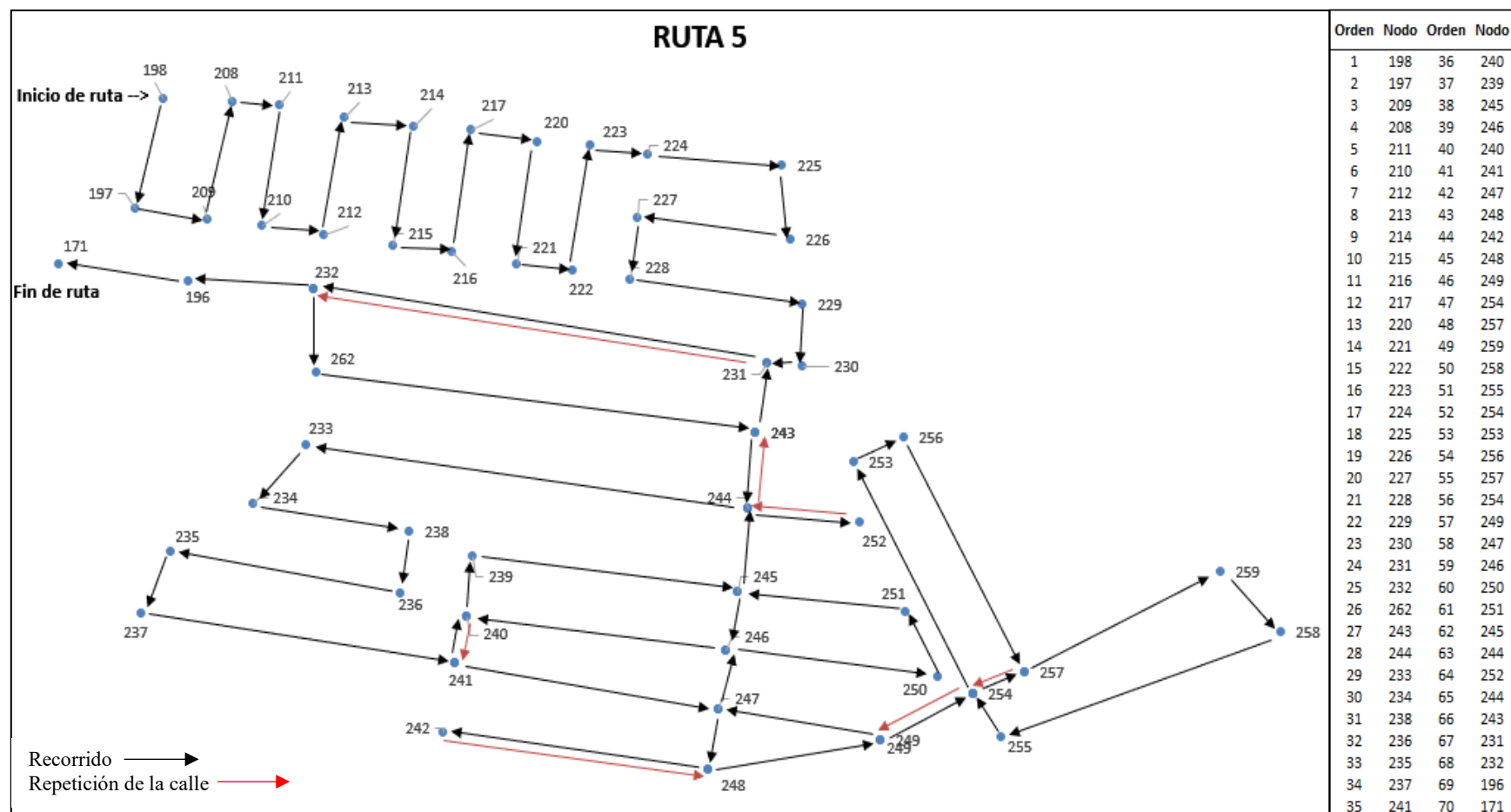
Anexo A. Recorridos actuales de la empresa











Anexo B. *Coordenadas de las calles halladas por medio de Google Maps.*

Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio
1-2	3.919172	-76.314464	Acceso	260-366	3.894576	-76.307132	Acceso	267-268	3.893085	-76.300728	Barrio 6	303-304	3.888579	-76.296983	Barrio 6
2-5	3.914671	-76.308833	Acceso	95-96	3.901793	-76.308398	Acceso	267-314	3.893377	-76.300212	Barrio 6	278-303	3.888095	-76.297386	Barrio 6
5-18	3.914064	-76.308066	Acceso	96-98	3.892985	-76.310743	Acceso	314-315	3.893357	-76.299255	Barrio 6	278-279	3.887631	-76.297812	Barrio 6
94-95	3.911612	-76.305198	Acceso	119-97	3.894611	-76.311541	Acceso	313-314	3.893075	-76.299726	Barrio 6	279-302	3.887727	-76.297215	Barrio 6
83-94	3.91181	-76.305627	Acceso	97-99	3.903142	-76.308525	Acceso	268-269	3.892756	-76.300618	Barrio 6	302-303	3.888191	-76.296789	Barrio 6
75-82	3.912454	-76.305899	Acceso	99-100	3.91076	-76.303405	Acceso	269-313	3.892742	-76.300113	Barrio 6	301-302	3.887886	-76.296623	Barrio 6
82-83	3.91217	-76.305817	Acceso	260-369	3.895805	-76.307957	Acceso	312-313	3.892753	-76.299211	Barrio 6	301-280	3.887536	-76.297083	Barrio 6
18-37	3.913486	-76.307317	Acceso	369-364	3.897044	-76.307011	Acceso	311-312	3.892454	-76.29868	Barrio 6	280-281	3.887176	-76.297565	Barrio 6
81-82	3.912424	-76.305622	Acceso	368-369	3.899495	-76.30676	Acceso	270-311	3.892205	-76.299477	Barrio 6	281-299	3.887125	-76.296887	Barrio 6
81-100	3.911242	-76.303694	Acceso	367-368	3.902561	-76.305123	Acceso	270-271	3.891969	-76.300138	Barrio 6	299-300	3.887322	-76.296323	Barrio 6
100-101	3.909897	-76.301834	Acceso	367-370	3.902177	-76.304235	Acceso	271-310	3.891667	-76.299228	Barrio 6	300-301	3.887608	-76.296474	Barrio 6
101-102	3.909982	-76.301637	Acceso	370-371	3.899171	-76.304775	Acceso	310-311	3.891903	-76.298567	Barrio 6	298-299	3.887067	-76.296207	Barrio 6
102-103	3.910172	-76.301569	Acceso	100-372	3.909124	-76.300566	Acceso	309-310	3.891346	-76.298328	Barrio 6	291-298	3.886883	-76.296791	Barrio 6
103-81	3.911361	-76.30359	Acceso	372-373	3.901878	-76.301877	Acceso	272-309	3.891099	-76.298962	Barrio 6	291-292	3.886715	-76.297359	Barrio 6
152-260	3.894251	-76.308941	Acceso	373-376	3.894855	-76.302744	Acceso	272-273	3.890868	-76.299604	Barrio 6	292-297	3.886643	-76.296674	Barrio 6
260-261	3.89443	-76.307513	Acceso	376-374	3.894735	-76.299667	Acceso	273-308	3.890573	-76.298711	Barrio 6	298-297	3.886812	-76.296106	Barrio 6
261-163	3.894398	-76.306078	Acceso	374-316	3.894112	-76.298541	Acceso	308-309	3.890804	-76.298069	Barrio 6	296-297	3.886569	-76.295998	Barrio 6
163-366	3.894544	-76.305697	Acceso	372-375	3.907109	-76.296605	Acceso	307-308	3.890278	-76.297813	Barrio 6	293-296	3.886394	-76.296561	Barrio 6
261-366	3.894703	-76.305833	Acceso	375-374	3.901486	-76.296014	Acceso	274-307	3.890011	-76.298343	Barrio 6	292-293	3.886468	-76.297237	Barrio 6
363-364	3.896191	-76.305818	Acceso	266-376	3.894001	-76.301604	Acceso	274-275	3.889614	-76.298799	Barrio 6	295-296	3.886322	-76.295887	Barrio 6
364-367	3.899924	-76.305529	Acceso	373-363	3.895288	-76.304629	Acceso	275-307	3.889606	-76.298142	Barrio 6	284-295	3.886091	-76.297312	Barrio 6
37-75	3.912893	-76.306523	Acceso	376-377	3.901323	-76.29914	Acceso	275-306	3.889441	-76.298053	Barrio 6	283-284	3.886151	-76.29879	Barrio 6
261-202	3.894405	-76.305673	Acceso	377-102	3.909081	-76.298677	Acceso	306-307	3.889838	-76.297596	Barrio 6	283-287	3.886363	-76.298501	Barrio 6
203-266	3.893694	-76.303152	Acceso	374-378	3.895405	-76.297765	Acceso	306-357	3.889439	-76.297411	Barrio 6	287-288	3.886575	-76.2983	Barrio 6
265-266	3.893045	-76.301784	Acceso	378-379	3.901155	-76.295169	Acceso	357-305	3.889107	-76.297253	Barrio 6	288-289	3.886874	-76.298115	Barrio 6
202-366	3.894551	-76.305292	Acceso	379-377	3.907508	-76.294954	Acceso	276-357	3.888966	-76.297829	Barrio 6	289-290	3.886852	-76.297743	Barrio 6
261-366	3.894703	-76.305833	Acceso	264-265	3.892437	-76.301709	Barrio 6	276-277	3.888477	-76.298228	Barrio 6	290-291	3.886775	-76.297546	Barrio 6
366-363	3.89517	-76.305254	Acceso	264-269	3.892392	-76.301058	Barrio 6	277-304	3.888504	-76.297595	Barrio 6	279-282	3.887152	-76.298321	Barrio 6
364-366	3.89587	-76.306016	Acceso	265-268	3.892801	-76.301269	Barrio 6	304-305	3.888893	-76.297135	Barrio 6	356-380	3.886426	-76.293474	Barrio 6

Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio
295-380	3.886299	-76.294792	Barrio 6	332-338	3.889432	-76.293964	Barrio 6	259-263	3.889275	-76.300221	Barrio 6	144-146	3.895326	-76.30945	Barrio 1
355-356	3.886698	-76.292207	Barrio 6	331-332	3.889704	-76.294639	Barrio 6	258-282	3.887514	-76.299161	Barrio 6	146-147	3.895025	-76.30948	Barrio 1
354-355	3.886807	-76.293408	Barrio 6	327-331	3.890109	-76.295257	Barrio 6	273-263	3.890281	-76.300038	Barrio 6	125-132	3.896708	-76.310391	Barrio 1
354-360	3.886657	-76.294567	Barrio 6	327-328	3.890508	-76.295485	Barrio 6	151-152	3.894176	-76.309336	Barrio 1	112-125	3.896819	-76.311012	Barrio 1
296-360	3.886513	-76.29524	Barrio 6	328-329	3.890651	-76.296432	Barrio 6	150-151	3.894222	-76.309597	Barrio 1	112-113	3.896431	-76.311408	Barrio 1
353-354	3.886779	-76.294624	Barrio 6	328-330	3.889968	-76.295853	Barrio 6	150-139	3.894341	-76.309933	Barrio 1	113-127	3.895944	-76.311188	Barrio 1
297-353	3.886755	-76.29535	Barrio 6	309-329	3.890908	-76.297648	Barrio 6	148-150	3.894487	-76.309563	Barrio 1	133-126	3.896414	-76.310418	Barrio 1
351-353	3.886941	-76.294696	Barrio 6	326-327	3.890416	-76.294737	Barrio 6	147-148	3.89478	-76.309514	Barrio 1	120-134	3.896137	-76.310452	Barrio 1
298-351	3.886998	-76.295452	Barrio 6	325-326	3.890499	-76.294207	Barrio 6	138-148	3.894712	-76.309886	Barrio 1	127-131	3.895882	-76.310659	Barrio 1
351-352	3.88721	-76.293086	Barrio 6	324-325	3.889979	-76.292577	Barrio 6	148-162	3.89465	-76.309344	Barrio 1	131-135	3.895844	-76.310303	Barrio 1
345-352	3.887593	-76.291422	Barrio 6	333-334	3.889352	-76.292137	Barrio 6	161-162	3.89472	-76.30912	Barrio 1	136-137	3.895286	-76.310179	Barrio 1
344-345	3.887818	-76.291607	Barrio 6	334-335	3.888778	-76.291249	Barrio 6	147-161	3.89485	-76.30929	Barrio 1	130-137	3.895135	-76.310379	Barrio 1
344-346	3.887774	-76.292315	Barrio 6	332-333	3.889702	-76.293523	Barrio 6	160-161	3.894969	-76.309052	Barrio 1	129-130	3.895173	-76.310731	Barrio 1
346-347	3.887705	-76.293362	Barrio 6	335-336	3.88898	-76.29226	Barrio 6	146-160	3.895144	-76.309242	Barrio 1	128-129	3.89541	-76.310887	Barrio 1
346-348	3.887898	-76.293374	Barrio 6	336-337	3.889366	-76.293215	Barrio 6	159-160	3.895259	-76.308974	Barrio 1	127-128	3.89577	-76.310853	Barrio 1
348-349	3.888039	-76.294112	Barrio 6	337-361	3.888513	-76.293762	Barrio 6	144-159	3.895441	-76.309182	Barrio 1	124-128	3.89562	-76.311074	Barrio 1
301-349	3.887895	-76.295433	Barrio 6	361-345	3.887743	-76.292862	Barrio 6	158-159	3.895534	-76.308892	Barrio 1	114-124	3.895649	-76.311267	Barrio 1
349-362	3.887827	-76.294296	Barrio 6	343-344	3.888198	-76.292681	Barrio 6	143-158	3.895701	-76.309131	Barrio 1	114-115	3.8956	-76.311269	Barrio 1
300-362	3.88754	-76.295338	Barrio 6	343-348	3.888322	-76.29374	Barrio 6	156-158	3.896039	-76.308704	Barrio 1	115-123	3.895507	-76.311216	Barrio 1
299-350	3.887253	-76.29555	Barrio 6	323-324	3.88947	-76.290909	Barrio 6	140-156	3.896488	-76.30894	Barrio 1	123-124	3.895557	-76.311214	Barrio 1
350-351	3.887184	-76.294795	Barrio 6	322-323	3.890561	-76.292927	Barrio 6	132-140	3.896609	-76.30968	Barrio 1	116-129	3.895213	-76.31112	Barrio 1
303-349	3.8882	-76.295599	Barrio 6	321-322	3.891705	-76.295016	Barrio 6	140-141	3.896434	-76.309335	Barrio 1	116-122	3.895104	-76.31127	Barrio 1
348-342	3.888352	-76.293927	Barrio 6	320-321	3.89117	-76.29333	Barrio 6	133-141	3.896339	-76.309703	Barrio 1	122-117	3.895012	-76.311479	Barrio 1
341-342	3.888654	-76.294409	Barrio 6	319-320	3.890591	-76.291632	Barrio 6	141-154	3.896276	-76.309152	Barrio 1	117-118	3.894964	-76.311756	Barrio 1
340-341	3.88875	-76.295463	Barrio 6	318-319	3.891504	-76.293253	Barrio 6	154-155	3.896114	-76.308967	Barrio 1	118-121	3.8948	-76.31136	Barrio 1
305-340	3.888922	-76.296618	Barrio 6	317-318	3.89243	-76.294906	Barrio 6	134-142	3.896056	-76.309739	Barrio 1	121-138	3.894737	-76.310597	Barrio 1
306-330	3.889533	-76.296724	Barrio 6	316-317	3.89289	-76.296735	Barrio 6	142-143	3.895876	-76.309394	Barrio 1	119-121	3.894611	-76.310966	Barrio 1
330-339	3.889324	-76.295544	Barrio 6	312-316	3.892958	-76.29863	Barrio 6	135-143	3.895781	-76.309768	Barrio 1	119-139	3.89444	-76.310624	Barrio 1
338-339	3.889228	-76.294554	Barrio 6	263-365	3.89058	-76.301547	Barrio 6	136-144	3.895473	-76.309793	Barrio 1	126-120	3.8963145	-76.31079	Barrio 1

Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio	Calle	Latitud	Longitud	Barrio
155-142	3.895988	-76.309174	Barrio 1	214-215	3.891042	-76.304729	Barrio 5	245-246	3.888254	-76.302739	Barrio 5	286-358	3.885034	-76.298298	Barrio 7
163-164	3.893234	-76.306172	Barrio 3	215-216	3.890637	-76.304613	Barrio 5	240-246	3.888175	-76.303562	Barrio 5	285-286	3.885164	-76.298812	Barrio 7
164-165	3.892077	-76.306443	Barrio 3	216-217	3.89101	-76.304376	Barrio 5	241-247	3.887833	-76.303622	Barrio 5	167-168	3.891430	-76.307824	Barrio 4
165-166	3.891806	-76.306493	Barrio 3	217-220	3.891365	-76.304116	Barrio 5	247-248	3.887489	-76.302856	Barrio 5	168-169	3.891494	-76.309017	Barrio 4
166-167	3.891449	-76.306583	Barrio 3	220-221	3.890934	-76.303977	Barrio 5	242-248	3.887414	-76.303687	Barrio 5	169-170	3.891119	-76.307909	Barrio 4
202-203	3.894179	-76.304829	Barrio 3	221-222	3.890522	-76.30387	Barrio 5	244-245	3.888707	-76.302671	Barrio 5	170-171	3.890718	-76.30678	Barrio 4
203-204	3.893794	-76.304587	Barrio 3	222-223	3.890902	-76.303647	Barrio 5	231-243	3.889681	-76.302562	Barrio 5	171-172	3.890792	-76.30797	Barrio 4
204-219	3.893404	-76.304361	Barrio 3	223-224	3.891274	-76.30342	Barrio 5	244-252	3.888929	-76.302304	Barrio 5	172-173	3.890875	-76.309163	Barrio 4
218-219	3.892509	-76.304175	Barrio 3	224-225	3.89121	-76.302843	Barrio 5	245-251	3.888377	-76.302197	Barrio 5	173-174	3.890474	-76.308042	Barrio 4
206-218	3.891802	-76.304649	Barrio 3	225-226	3.890935	-76.302413	Barrio 5	250-251	3.888101	-76.301591	Barrio 5	174-175	3.89007	-76.30691	Barrio 4
206-207	3.89179	-76.305045	Barrio 3	226-227	3.890766	-76.302849	Barrio 5	246-250	3.887977	-76.302133	Barrio 5	175-176	3.890137	-76.308091	Barrio 4
199-207	3.891685	-76.305385	Barrio 3	227-228	3.890642	-76.303334	Barrio 5	247-249	3.887584	-76.302333	Barrio 5	176-177	3.890206	-76.309293	Barrio 4
165-199	3.891814	-76.306093	Barrio 3	228-229	3.890365	-76.302836	Barrio 5	248-249	3.887388	-76.302364	Barrio 5	177-178	3.889796	-76.308174	Barrio 4
164-200	3.892135	-76.306016	Barrio 3	229-230	3.890087	-76.302312	Barrio 5	249-254	3.887633	-76.301562	Barrio 5	178-179	3.889393	-76.307054	Barrio 4
200-206	3.891977	-76.305327	Barrio 3	230-231	3.889898	-76.302418	Barrio 5	254-255	3.887644	-76.301197	Barrio 5	179-180	3.889485	-76.308243	Barrio 4
205-206	3.892477	-76.304886	Barrio 3	231-232	3.890146	-76.303899	Barrio 5	255-258	3.887841	-76.300265	Barrio 5	180-181	3.889567	-76.309418	Barrio 4
201-205	3.893134	-76.305061	Barrio 3	232-262	3.890114	-76.305262	Barrio 5	258-259	3.888373	-76.299603	Barrio 5	181-182	3.889161	-76.308291	Barrio 4
200-201	3.892633	-76.305502	Barrio 3	262-243	3.88965	-76.303924	Barrio 5	257-259	3.888247	-76.300378	Barrio 5	182-183	3.888762	-76.307175	Barrio 4
201-202	3.893739	-76.305252	Barrio 3	243-244	3.889215	-76.302618	Barrio 5	256-257	3.88868	-76.301336	Barrio 5	183-184	3.888847	-76.308371	Barrio 4
163-202	3.894246	-76.305537	Barrio 3	233-234	3.889154	-76.305441	Barrio 5	253-256	3.889351	-76.301854	Barrio 5	184-185	3.888932	-76.309561	Barrio 4
166-198	3.891645	-76.306344	Barrio 5	233-244	3.889143	-76.303645	Barrio 5	253-254	3.888525	-76.301644	Barrio 5	185-186	3.888519	-76.308416	Barrio 4
197-198	3.891252	-76.306269	Barrio 5	238-234	3.888895	-76.305122	Barrio 5	225-365	3.891177	-76.302437	Barrio 5	186-187	3.888106	-76.30728	Barrio 4
197-209	3.890862	-76.306132	Barrio 5	236-238	3.88863	-76.304719	Barrio 5	98-153	3.893096	-76.309785	Barrio 2	187-188	3.888207	-76.308499	Barrio 4
208-209	3.891203	-76.305838	Barrio 5	235-236	3.888564	-76.305442	Barrio 5	98-151	3.894073	-76.309673	Barrio 2	186-189	3.888186	-76.306855	Barrio 4
208-211	3.891569	-76.305622	Barrio 5	235-237	3.888498	-76.306225	Barrio 5	152-153	3.893198	-76.309449	Barrio 2	189-190	3.88827	-76.306449	Barrio 4
210-211	3.891173	-76.305531	Barrio 5	237-241	3.88814	-76.305366	Barrio 5	284-285	3.885657	-76.298811	Barrio 7	183-190	3.888518	-76.30682	Barrio 4
210-212	3.89076	-76.305395	Barrio 5	240-241	3.888133	-76.304383	Barrio 5	285-294	3.885352	-76.298311	Barrio 7	182-191	3.888842	-76.306743	Barrio 4
212-213	3.891105	-76.305147	Barrio 5	239-240	3.888477	-76.304326	Barrio 5	294-358	3.885222	-76.297796	Barrio 7	191-192	3.888922	-76.306323	Barrio 4
213-214	3.891453	-76.304876	Barrio 5	239-245	3.888556	-76.303504	Barrio 5	358-359	3.885093	-76.297487	Barrio 7	179-192	3.88916	-76.306697	Barrio 4

Calle	Latitud	Longitud	Barrio
178-193	3.88948	-76.306614	Barrio 4
193-194	3.889583	-76.306192	Barrio 4
175-194	3.88983	-76.30655	Barrio 4
174-195	3.89014	-76.306505	Barrio 4
195-196	3.89024	-76.306079	Barrio 4
171-196	3.89049	-76.306424	Barrio 4
167-170	3.891052	-76.306704	Barrio 4

Anexo C. Algoritmo KMeans

S = Conjunto de calles
 x_j = Calle j
 k = Número de cluster
 μ_i = Centroide de cada cluster i

Para la asignación de calles a cada cluster:

$$\min_S E(\mu_i) = \min_S \sum_{i=1}^k \sum_{x_j \in S_i} \|x_j - \mu_i\|^2$$

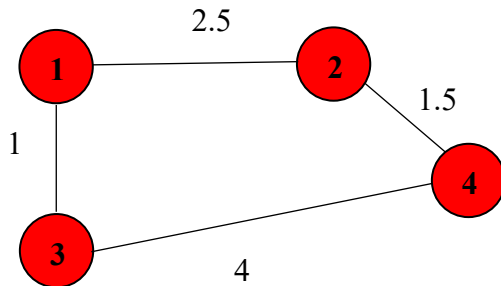
Para actualizar los centroides de cada cluster:

t = Número de iteraciones

$$\mu_i^{(t+1)} = \frac{1}{|S_i^{(t)}|} \sum_{x_j \in S_i^{(t)}} x_j$$

Anexo D. Algoritmo de Dijkstra

Parámetros:



d_{ij} = Distancia entre el nodo i y nodo j
 $n = 4$, Número de nodos
 $n = 1$, nodo de inicio
 $n = 4$, nodo de fin
 $n = 2,3$, nodos de transbordo

Variables:

x_{ij} = variable binaria, si el nodo i se conecta con el nodo j

Función objetivo:

$$\text{Min } Z: \sum_{ij}^4 d_{ij} * x_{ij} = 2.5 * x_{12} + 1 * x_{13} + 1.5 * x_{24} + 4 * x_{34}$$

Restricciones:

$$\text{Nodo inicial: } x_{12} + x_{13} = 1$$

$$\text{Nodo final: } x_{34} + x_{24} = 1$$

$$\text{Transbordo 2: } x_{12} = x_{24}$$

$$\text{Transbordo 2: } x_{13} = x_{34}$$

$$x_{ij} \geq 0$$

Anexo E. Algoritmo de Fleury

G = Grafo euleriano

V = Vértices

v_1 = Vértice inicial

E = Arcos

P = Ciclo euleriano en G

Pasos:

1. Definir $P_0 = (v_1), G_0 = G$
2. Si el paseo $P_k = (v_1, e_1, v_2, \dots, e_{k-1}, v_k)$ ha sido construido, elegir $e_k \in E(G_{k-1})$ tal que,
(a) $= e_k$ incide con v_k
(b) $= a$ menos que no haya otra elección, e_k no es puente de G_{k-1}
3. Definir $P_k = (P_{k-1}, e_k, v_{k+1})$, donde v_{k+1} es el otro extremo de e_k
4. Definir $G_k = G_{k-1} - e_k$
5. Definir $P = P_k$

Anexo F. Visualización del código desarrollado en Google colabatory

Instrucciones:

En el siguiente enlace podrá encontrar el código del algoritmo KMeans- al igual que el código para las rutas de entrada y salida a cada cluster- los datos generales del problema- los recorridos actuales y los recorridos propuestos. Tenga en cuenta que para ingresar- deberá hacerlo con el correo institucional de Univalle- y seleccionar la opción “Abrir con Google Colaboratory”:

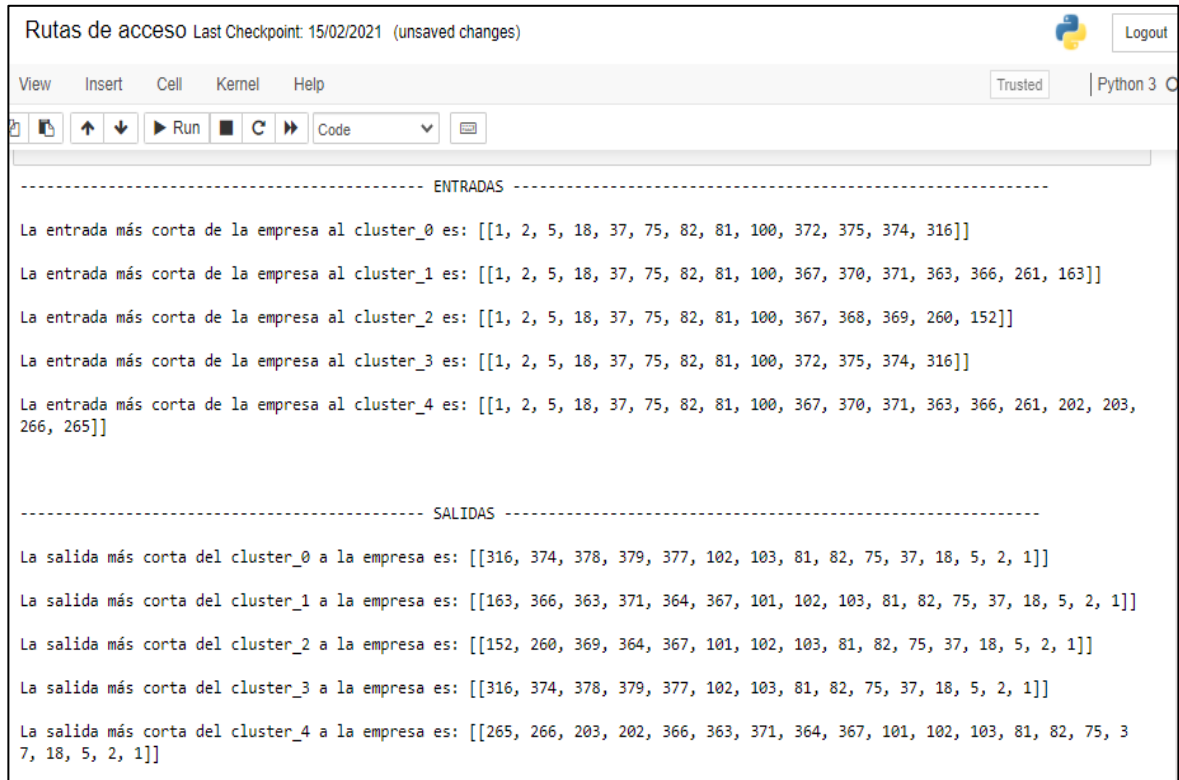
Link:

<https://colab.research.google.com/drive/1ssDzbQCSliVTazFj5Aey29OuTSxgEaRG?usp=ssharing>

Nota: Debido a la dimensión del código para la solución del problema del cartero chino- se anexa link del código base utilizado.

Link: <https://github.com/supermitch/Chinese-Postman/find/master>

Anexo G. Resultados en Python de las rutas de entrada y salida a cada cluster



```
Rutas de acceso Last Checkpoint: 15/02/2021 (unsaved changes)
View Insert Cell Kernel Help Trusted Python 3

----- ENTRADAS -----
La entrada más corta de la empresa al cluster_0 es: [[1, 2, 5, 18, 37, 75, 82, 81, 100, 372, 375, 374, 316]]
La entrada más corta de la empresa al cluster_1 es: [[1, 2, 5, 18, 37, 75, 82, 81, 100, 367, 370, 371, 363, 366, 261, 163]]
La entrada más corta de la empresa al cluster_2 es: [[1, 2, 5, 18, 37, 75, 82, 81, 100, 367, 368, 369, 260, 152]]
La entrada más corta de la empresa al cluster_3 es: [[1, 2, 5, 18, 37, 75, 82, 81, 100, 372, 375, 374, 316]]
La entrada más corta de la empresa al cluster_4 es: [[1, 2, 5, 18, 37, 75, 82, 81, 100, 367, 370, 371, 363, 366, 261, 202, 203, 266, 265]]

----- SALIDAS -----
La salida más corta del cluster_0 a la empresa es: [[316, 374, 378, 379, 377, 102, 103, 81, 82, 75, 37, 18, 5, 2, 1]]
La salida más corta del cluster_1 a la empresa es: [[163, 366, 363, 371, 364, 367, 101, 102, 103, 81, 82, 75, 37, 18, 5, 2, 1]]
La salida más corta del cluster_2 a la empresa es: [[152, 260, 369, 364, 367, 101, 102, 103, 81, 82, 75, 37, 18, 5, 2, 1]]
La salida más corta del cluster_3 a la empresa es: [[316, 374, 378, 379, 377, 102, 103, 81, 82, 75, 37, 18, 5, 2, 1]]
La salida más corta del cluster_4 a la empresa es: [[265, 266, 203, 202, 366, 363, 371, 364, 367, 101, 102, 103, 81, 82, 75, 37, 18, 5, 2, 1]]
```

Anexo H. Resultados en Python de los recorridos de cada cluster

```
Anaconda Prompt (anaconda3)
Loading graph: cluster_0_5
<67> edges
Converting to Eulerian path...
  Doubling dead ends
  Building possible odd node pairs
    (66 pairs)
  Finding pair solutions
    (132 solutions)
  Building path sets
  Finding cheapest route
  Adding new edges
Conversion complete
  Added 15 edges
  Total cost is 7947.79
Attempting to solve Eulerian Circuit...
  Solved in <1> attempts
Solution: (<82> edges)
[1, 309, 308, 307, 274, 275, 307, 306, 357, 305, 304, 305, 340, 305, 357, 276, 277, 304, 303, 278, 279, 302, 303, 349, 362, 349, 301, 300, 362, 300, 299, 298, 2
91, 290, 289, 288, 287, 283, 284, 285, 294, 358, 359, 358, 286, 285, 284, 295, 296, 360, 296, 297, 296, 293, 292, 291, 292, 297, 298, 351, 353, 297, 353, 354, 353, 351,
350, 299, 281, 280, 301, 302, 279, 282, 258, 259, 275, 306, 330, 328, 329, 309, 1]
(TG) C:\Users\jeank\Google Drive (sanchez.jean@correounivalle.edu.co)\INGENIERIA INDUSTRIAL\X\TRABAJO DE GRADO II\MODELO\PYTHON\CARTERO\Chinese-Postman-master>
```

```
Anaconda Prompt (anaconda3)
Loading graph: cluster_1_5
<80> edges
Converting to Eulerian path...
  Doubling dead ends
  Building possible odd node pairs
    (231 pairs)
  Finding pair solutions
    (462 solutions)
  Building path sets
  Finding cheapest route
  Adding new edges
Conversion complete
  Added 20 edges
  Total cost is 9315.93
Attempting to solve Eulerian Circuit...
  Solved in <1> attempts
Solution: (<100> edges)
[1, 164, 200, 164, 165, 166, 167, 168, 169, 170, 167, 170, 171, 172, 173, 174, 175, 176, 177, 178, 179, 192, 191, 182, 181, 180, 179, 178, 193, 194, 175, 174, 1
95, 196, 171, 196, 232, 262, 233, 234, 238, 236, 235, 237, 189, 186, 187, 188, 187, 186, 185, 184, 183, 182, 183, 190, 189, 237, 241, 237, 235, 236, 238, 234, 233, 262,
232, 222, 221, 220, 217, 216, 215, 214, 213, 207, 199, 165, 166, 198, 197, 209, 208, 211, 210, 212, 213, 207, 206, 218, 219, 204, 203, 202, 201, 205, 206, 200, 201, 20
2, 1]
(TG) C:\Users\jeank\Google Drive (sanchez.jean@correounivalle.edu.co)\INGENIERIA INDUSTRIAL\X\TRABAJO DE GRADO II\MODELO\PYTHON\CARTERO\Chinese-Postman-master>
```

```
Anaconda Prompt (anaconda3)
Loading graph: cluster_2_5
<61> edges
Converting to Eulerian path...
  Doubling dead ends
  Building possible odd node pairs
    (91 pairs)
  Finding pair solutions
    (182 solutions)
  Building path sets
  Finding cheapest route
  Adding new edges
Conversion complete
  Added 10 edges
  Total cost is 3798.27
Attempting to solve Eulerian Circuit...
  Solved in <1> attempts
Solution: (<71> edges)
[1, 153, 98, 151, 150, 139, 119, 121, 118, 117, 122, 116, 129, 121, 138, 148, 147, 161, 160, 159, 144, 136, 137, 130, 129, 128, 124, 123, 115, 114, 124, 128, 12
7, 131, 135, 143, 158, 159, 158, 156, 140, 141, 140, 132, 125, 112, 113, 127, 131, 135, 143, 142, 155, 154, 141, 133, 126, 120, 134, 142, 143, 144, 146, 147, 146, 160,
161, 162, 148, 150, 151, 1]
(TG) C:\Users\jeank\Google Drive (sanchez.jean@correounivalle.edu.co)\INGENIERIA INDUSTRIAL\X\TRABAJO DE GRADO II\MODELO\PYTHON\CARTERO\Chinese-Postman-master>
```

```

Anaconda Prompt (anaconda3)
Loading graph: cluster_4_5
<66> edges
Converting to Eulerian path...
  Doubling dead ends
  Building possible odd node pairs
    (153 pairs)
  Finding pair solutions
    (306 solutions)
  Building path sets
  Finding cheapest route
  Adding new edges
Conversion complete
  Added 16 edges
  Total cost is 8486.029999999997
Attempting to solve Eulerian Circuit...
  Solved in <1> attempts
Solution: (<82> edges)
[1, 265, 268, 267, 314, 315, 314, 313, 312, 311, 270, 271, 310, 309, 308, 273, 263, 259, 257, 259, 258, 255, 254, 257, 256, 253, 254, 249, 248, 249, 247, 248, 2
42, 241, 240, 241, 247, 246, 245, 251, 250, 246, 240, 239, 245, 244, 252, 244, 233, 244, 243, 262, 232, 231, 243, 231, 230, 229, 228, 227, 226, 225, 224, 223, 222, 223,
224, 225, 365, 263, 273, 272, 309, 310, 311, 312, 313, 269, 268, 269, 264, 265, 1]

(TG) C:\Users\jeank\Google Drive (sanchez.jean@correounivalle.edu.co)\INGENIERIA INDUSTRIAL\X\TRABAJO DE GRADO II\MODELO\PYTHON\CARTERO\Chinese-Postman-master>

```

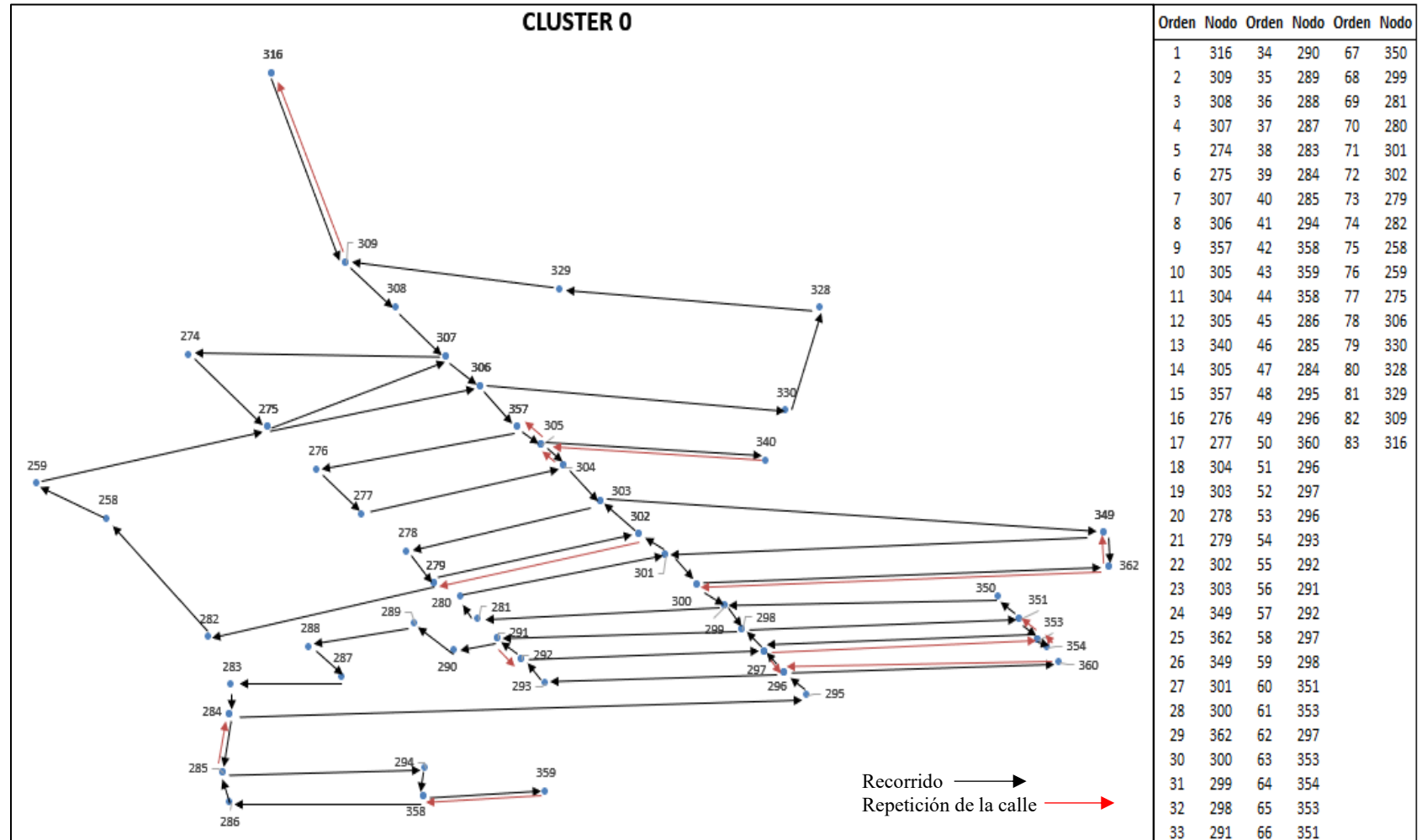
```

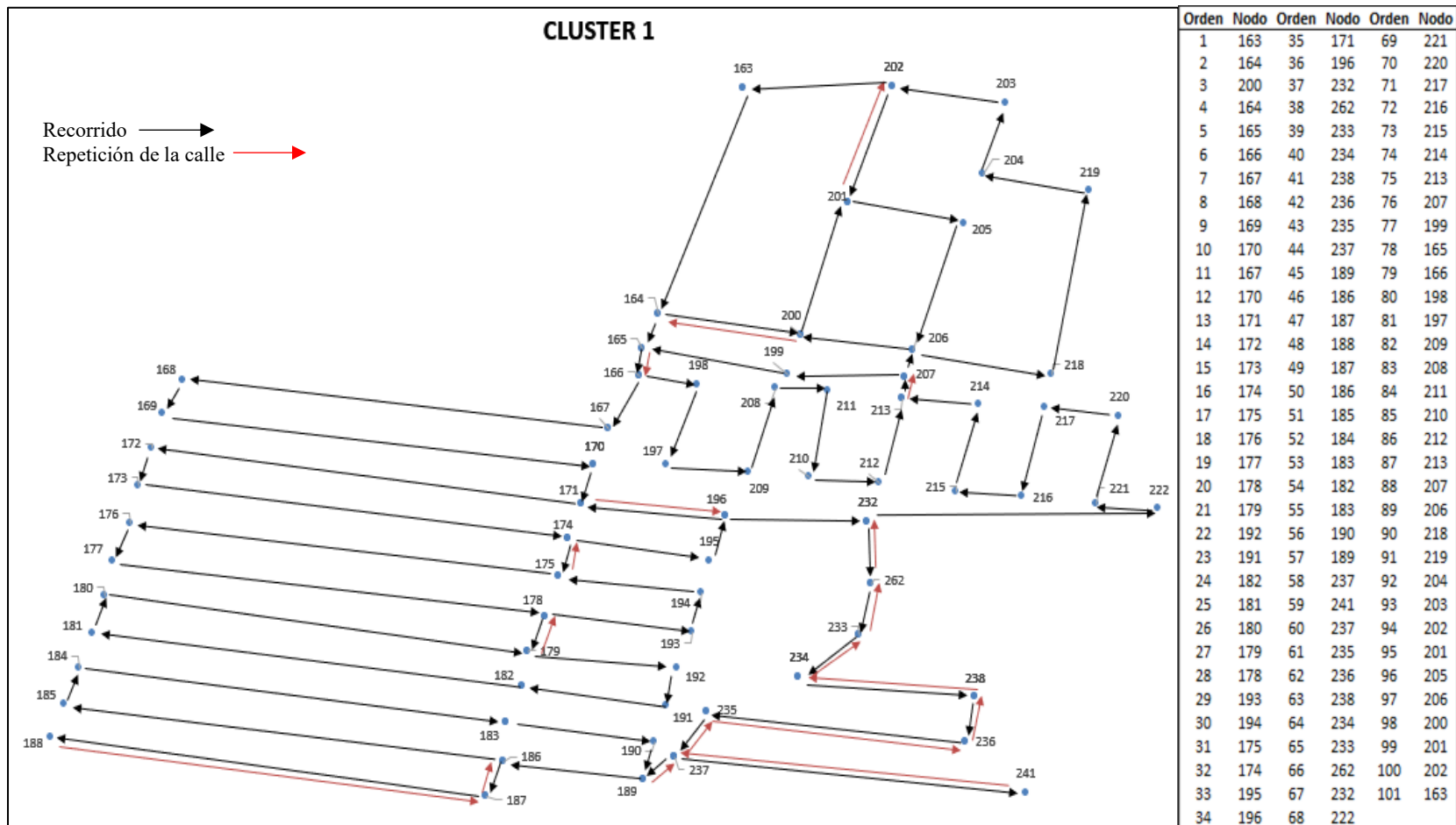
Anaconda Prompt (anaconda3)
Loading graph: cluster_3_5
<51> edges
Converting to Eulerian path...
  Doubling dead ends
  Building possible odd node pairs
    (66 pairs)
  Finding pair solutions
    (132 solutions)
  Building path sets
  Finding cheapest route
  Adding new edges
Conversion complete
  Added 12 edges
  Total cost is 7784.54
Attempting to solve Eulerian Circuit...
  Solved in <1> attempts
Solution: (<63> edges)
[1, 317, 318, 321, 322, 323, 324, 325, 326, 327, 328, 327, 331, 332, 338, 339, 330, 340, 341, 342, 348, 346, 348, 343, 344, 345, 352, 351, 353, 354, 355, 356, 3
80, 295, 380, 360, 354, 353, 351, 350, 351, 362, 349, 348, 347, 346, 344, 345, 361, 337, 336, 335, 334, 333, 332, 331, 327, 322, 321, 320, 319, 318, 317, 1]

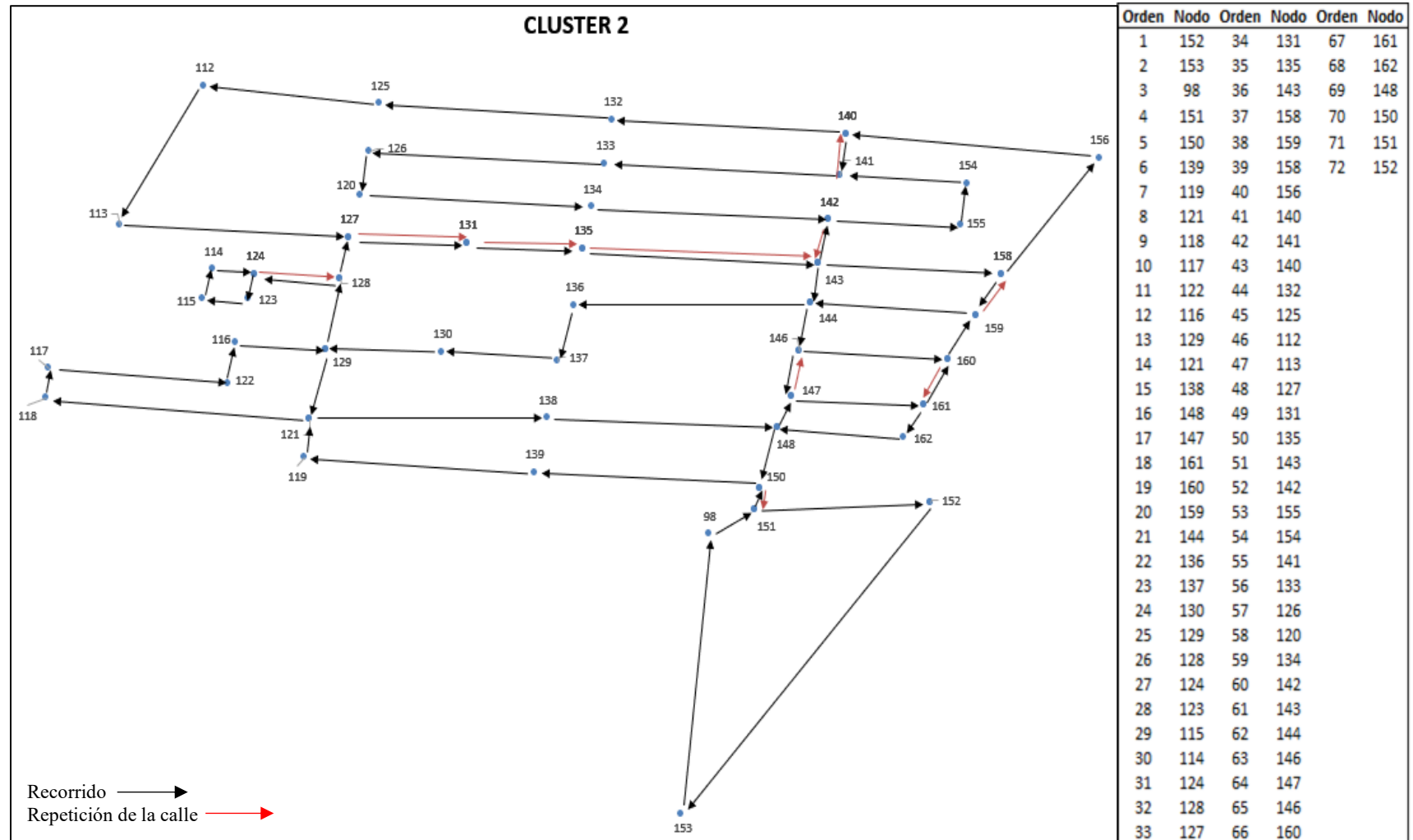
(TG) C:\Users\jeank\Google Drive (sanchez.jean@correounivalle.edu.co)\INGENIERIA INDUSTRIAL\X\TRABAJO DE GRADO II\MODELO\PYTHON\CARTERO\Chinese-Postman-master>

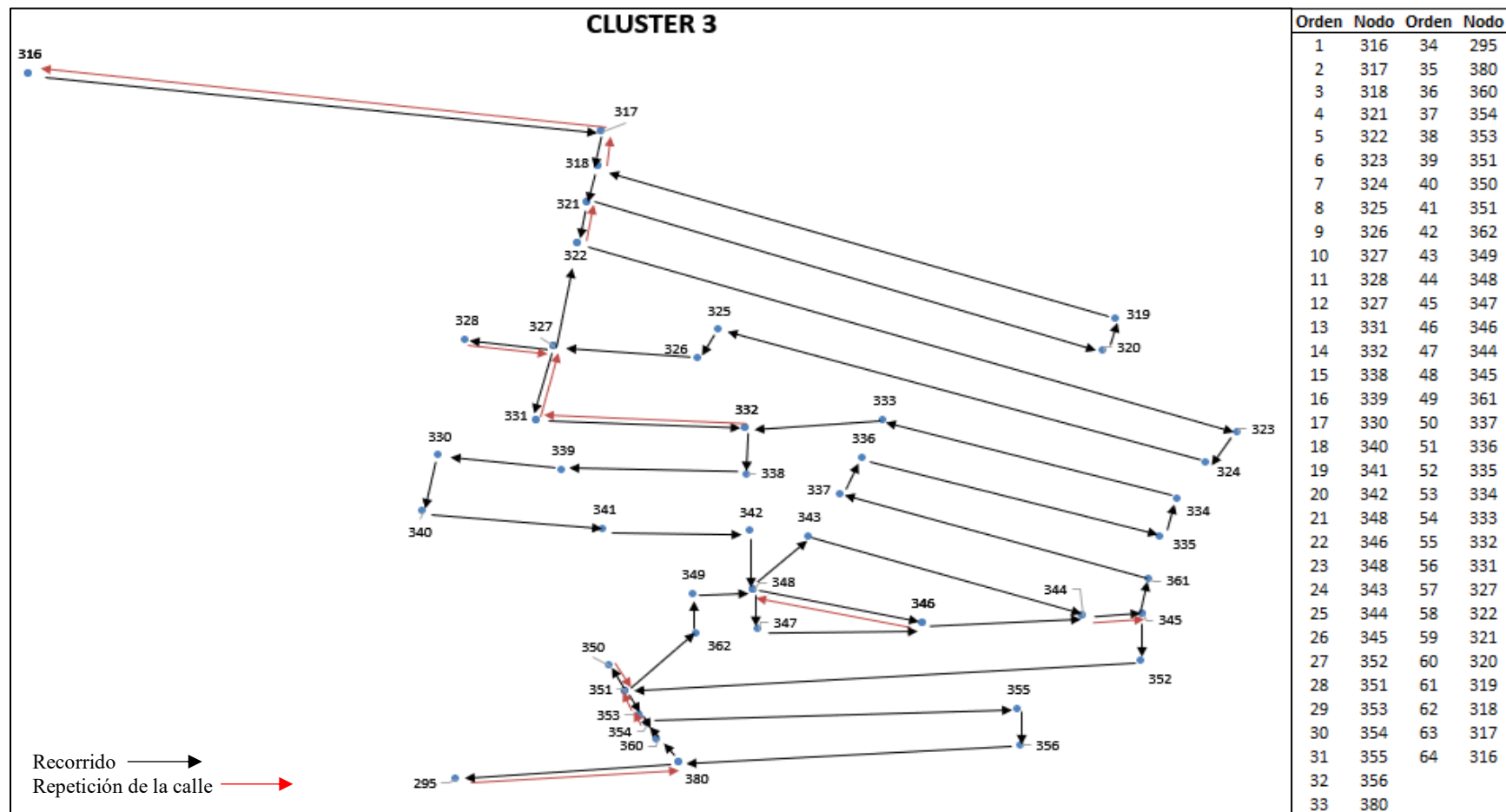
```

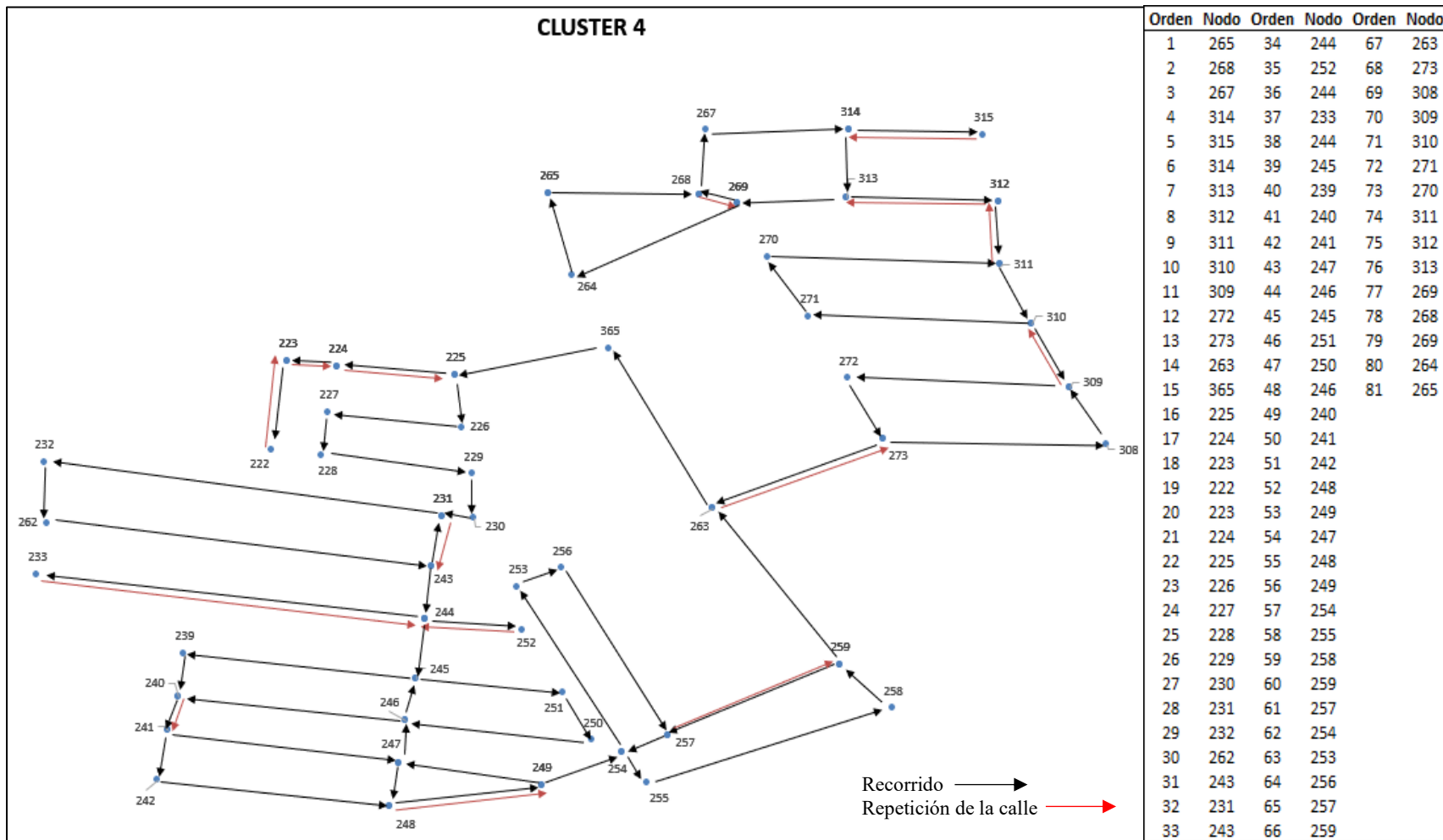
Anexo I. Recorridos de la solución propuesta











Anexo J. Código en AMPL para el CARP

```
#Modelo básico del CARP (Golden 1981)

param n:=49;                                #Número de nodos
set NODOS ordered:= {1..n};                 #Conjunto de nodos ordenados
set ARCOS:={i in NODOS, j in NODOS:i<>j};    #Conjunto de arcos
set K;                                       #Conjunto de vehículos

param C {(i,j) in ARCOS};
param W {k in K};
param q{(i,j) in ARCOS:i<>j}>=0;

var x{(i,j) in ARCOS, k in K:i<>j}>=0 binary;    #Si el arco ij es atravesado por el vehículo p
var fl{(i,j) in ARCOS, k in K:i<>j}>=0;          #Variable de flujo
var l{(i,j) in ARCOS, k in K:i<>j}>=0 binary;    #Si el vehículo p atiende la calle ij

#F.O
minimize total_cost:
sum{(i,j) in ARCOS, k in K:i<>j}C[i,j]*x[i,j,k];
```

```
#CONSTRAINS

subject to restricción_2{i in NODOS,k in K}:
sum{f in NODOS:i<>f}x[f,i,k]-sum{f in NODOS:i<>f}x[i,f,k]=0;

subject to restricción_3{(i,j) in ARCOS:i<>j}:
sum{k in K}(l[i,j,k]+l[j,i,k])=ceil(q[i,j]/sum{k in K}W[k]);

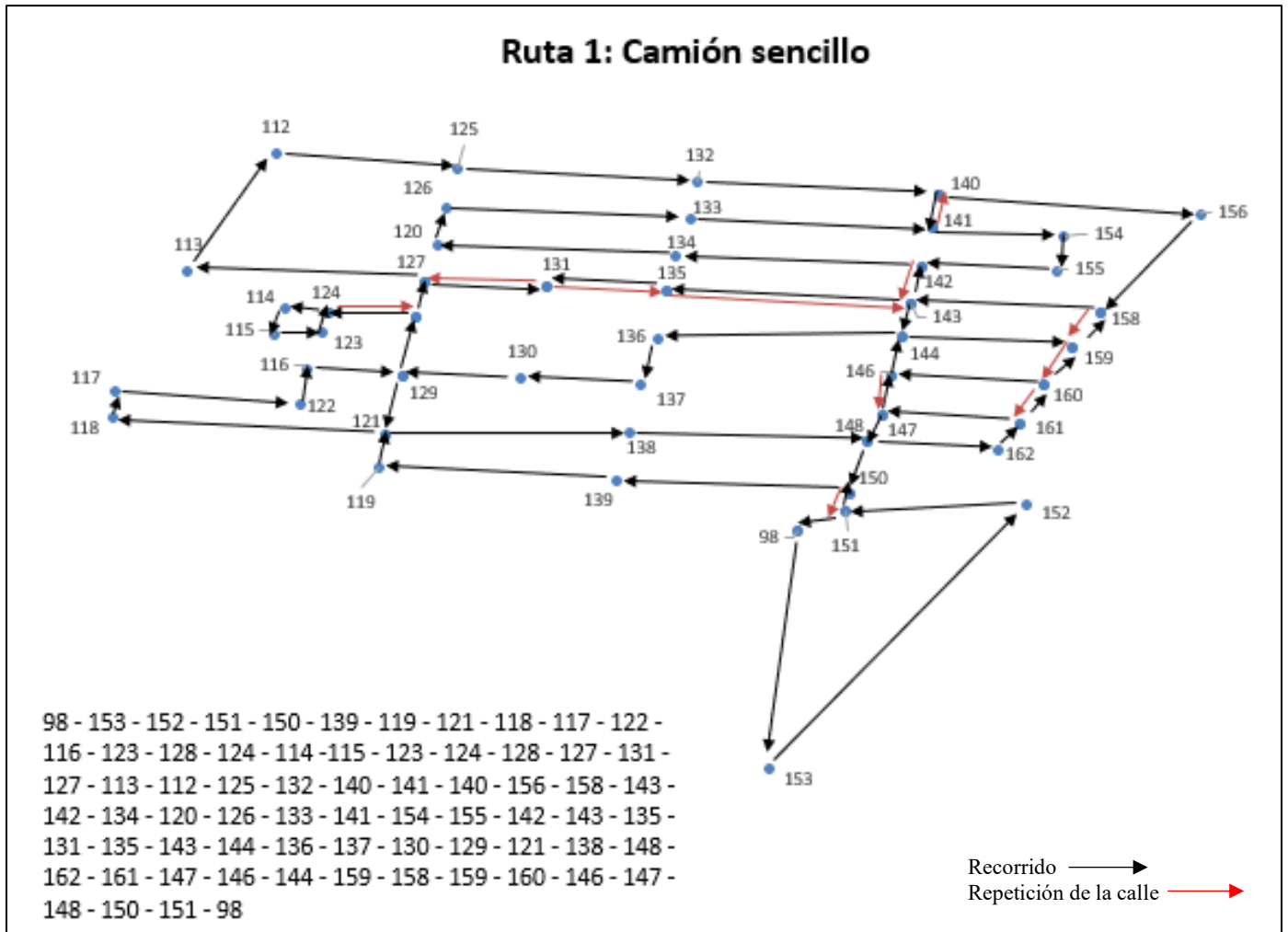
subject to restricción_4{(i,j) in ARCOS, k in K:i<>j}:
x[i,j,k]>=l[i,j,k];

subject to restricción_5{k in K}:
sum{(i,j) in ARCOS}l[i,j,k]*q[i,j]<=W[k];

subject to restricción_6prima{i in NODOS, k in K:i>1}:
sum{f in NODOS:i<>f}fl[i,f,k]-sum{f in NODOS:i<>f}fl[f,i,k]-sum{j in NODOS:i<>j}l[i,j,k]*q[i,j]=0;

subject to restricción_7{(i,j) in ARCOS,k in K:i<>j}:
fl[i,j,k]<=900*x[i,j,k];
```

Anexo K. Recorrido arrojado por el modelo CARP y la heurística propuesta



RECORRIDO SOLUCIÓN HEURISTICA

