

PROPUESTA DE RUTA DE RECOLECCIÓN DE RESIDUOS SÓLIDOS URBANOS EN UNA ZONA DE LA CIUDAD DE TRENQUE LAUQUEN MEDIANTE EL USO DE HERRAMIENTAS LIBRES Y GRATUITAS

Marcos, Carlos Eduardo; Martínez Micakoski, Fernanda; Enrietti, Adhemar;
Arqué, Emanuel

Facultad Regional Trenque Lauquen, Universidad Tecnológica Nacional.
marcoscarloseduardo@gmail.com; fer_micakoski@yahoo.com; adhemarenrietti@gmail.com;
ciberema@gmail.com

RESUMEN

El presente trabajo tiene como objetivo optimizar el recorrido que realiza el camión recolector de residuos sólidos urbanos en una zona de la ciudad de Trenque Lauquen, Provincia de Buenos Aires. Para obtener la solución se utilizan herramientas de código abierto, libre y gratuito como Open Street Map y R. El procedimiento se inicia descargando la zona de interés desde el repositorio geográfico, la cual se simplifica para luego ser transformada en un grafo representativo de las características viales de ese sector de la ciudad. Este grafo es modificado de manera que pueda obtenerse a través de él un ciclo euleriano a partir de la generación ficticia de arcos entre sus nodos. A continuación, basado en el algoritmo del Problema del Cartero Chino se obtiene una secuencia de arcos que reducen la distancia total al recorrer todos sus arcos. Finalmente se ordenan para representar un recorrido real. La solución obtenida se compara con la distancia media del trayecto actual de los camiones recolectores. A partir de los valores conseguidos se puede estimar la distancia que los operarios realizan a pie para reducir el tiempo de recolección. La zona fue seleccionada entre las cinco existentes por tratarse de un área cuyos residuos no requieren una descarga intermedia en el basural. Posibles mejoras incluyen la reducción en la cantidad de giros debido al desgaste que produce en los camiones.

Palabras Claves: Problema del cartero chino. Software libre. Grafos. Logística.

ABSTRACT

The present work aims to optimize the route made by the urban solid waste collection truck in an area of the city of Trenque Lauquen, Province of Buenos Aires. To obtain the solution, open source, free and free tools such as Open Street Map and R are used. The procedure begins by downloading the area of interest from the geographical repository, which is simplified and then transformed into a graph representative of the road characteristics of that sector of the city. This graph is modified so that a Eulerian cycle can be obtained through it from the fictitious generation of arcs between its nodes. Then, based on the algorithm of the Chinese Postman's Problem, a sequence of arcs is obtained that reduce the total distance by traveling all its arcs. Finally, they are ordered to represent a real tour. The solution obtained is compared with the average distance of the current journey of the collection trucks. From the values obtained, the distance that the operators make on foot can be estimated to reduce the collection time. The area was selected among the five existing ones because it is an area whose waste does not require an intermediate discharge into the landfill. Possible improvements include a reduction in the number of turns due to the wear and tear it produces on trucks.

Keywords: Chinese Postman Problem. Free software. Graph. Logistics.

1. INTRODUCCIÓN

La gestión de residuos sólidos urbanos es una tarea compleja que tiene implicaciones sociales, económicas, tecnológicas y ambientales para la sociedad y para las administraciones locales. La fase de recolección domiciliaria llega a representar entre el 70 y el 85 % de los costos totales de la gestión de los residuos sólidos, por lo que es un aspecto crítico dentro de la prestación del servicio (Betanzo-Quezada et al. 2016).

En esencia el problema consiste en asignar rutas a un conjunto de vehículos para recolectar los residuos domiciliarios bajo ciertas restricciones (Beliën et al., 2014).

El presente trabajo se enfoca en una de las cuatro zonas en que la Dirección de Higiene divide la ciudad de Trenque Lauquen para efectuar la recolección de residuos domiciliarios.

Trenque Lauquen es la ciudad cabecera del partido homónimo, de la provincia de Buenos Aires, Argentina. Se encuentra a 445 km de la Ciudad Autónoma de Buenos Aires por RN 5, y a 80 km de la Provincia de La Pampa. Se halla en la intersección de las rutas nacionales N°5 (que conecta a C.A.B.A y Santa Rosa) y N°33 (que une Rosario y Bahía Blanca). Cuenta con aproximadamente 50.000 habitantes (Instituto Nacional de Estadística y Censos INDEC, 2016). La ciudad se distingue por ser todas sus arterias avenidas o bulevares, lo que impactará directamente en la forma de tratar el problema del recorrido.

La Dirección de Higiene Urbana realiza la recolección en la zona urbana y quintas aledañas, con servicios diferenciados por tipo de residuos, según se observa en la Figura 1.



Figura 1 Servicios de Recolección Urbanos

La ruta del servicio a diseñar se basa en la recolección de residuos secos o húmedos de gestión privada correspondiente a la Zona 3 (Figura 2). A dicha zona se le asigna un camión para realizar la tarea mediante un recorrido único de domingo a viernes en horario nocturno.

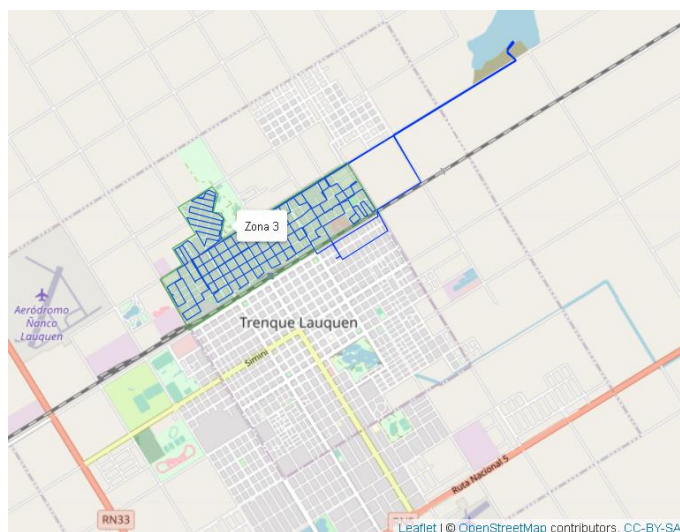


Figura 2 Servicios de Recolección Urbanos

El recorrido inicia y finaliza en el garaje de la Dirección de Higiene Urbana, previa descarga del camión en el Basural a Cielo Abierto o en la planta de procesamiento de residuo sólido urbano, de acuerdo con el cronograma de tipo de residuos que el vecino puede sacar en base al día de la semana. Cada vecino deposita la basura permitida en un canasto individual frente a su domicilio a partir de las 20 horas.

El alcance de este trabajo no incluye la posibilidad de re-zonificar las áreas de servicio que tiene asignadas cada camión.

En este artículo se utilizan técnicas de investigación operativa y recursos de sistemas de información geográfica (SIG) para modelar la zona bajo estudio y obtener un trayecto de recorrido mínimo que recorra todas las calles del área, lo que implica resolver variantes del Problema del Cartero Chino (Edmonds & Johnson, 1973).

1.1. Teoría de Grafos.

El problema de los siete puentes de Königsberg es un célebre problema matemático resuelto por Leonhard Euler en 1736 y cuya resolución dio origen a la teoría de grafos. El problema se formuló en el siglo XVIII y consistía en encontrar un recorrido para cruzar a pie toda la ciudad pasando solo una vez por cada uno de los puentes y regresando al mismo punto de inicio.

Euler realiza una abstracción del problema, enfocándose en las regiones terrestres y las conexiones entre ellas (Figura 3).



Figura 3 Mapa de época y su transformación en un grafo.

Así el problema se reduce a decidir si se puede recorrer el dibujo sin repetir las líneas y regresando al punto donde se inició el mismo. Euler demostró que esto no es posible si hay puntos en la figura en los cuáles el número de líneas que inciden es impar.

Este tipo de dibujos, donde a los puntos se los denomina *nodos* y a las líneas *aristas* es lo que constituye un *grafo*.

El número de aristas que inciden en un punto se denomina grado del nodo.

Cuando existe un trayecto que recorre todas las aristas del grafo una única vez regresando al nodo de inicio se lo denomina circuito euleriano. El teorema que manifiesta esta condición se enuncia de la siguiente manera: *Un grafo posee un circuito euleriano si y sólo si es conexo y todos sus nodos tienen grado par.*

Para el problema que debemos resolver, las calles representan las aristas del grafo y sus intersecciones los nodos. Por lo tanto, debemos verificar si el grafo asociado a la zona de recolección es euleriano, y en caso contrario, cuántas y cuáles calles se deberían repetir para transformarlo en uno donde sea posible hallar un circuito euleriano.

1.1.1 Definiciones y Nomenclatura.

Un *grafo* G consiste en un par ordenado $G = (V, E)$, donde V es un conjunto finito no vacío de nodos y E es un conjunto de pares no ordenados vw con $v, w \in V$ y $v \neq w$, denominados aristas.

Un *grafo dirigido* es un grafo $G = (V, A)$ donde $A \subseteq \{(a, b) \in V \times V: a \neq b\}$, es un conjunto de pares ordenados de elementos de V . Dado un arco $(a, b) \in A$, a es su nodo inicial y b su nodo final.

Un *grafo mixto* es una tupla $G = (V, E, A)$ donde V es un conjunto finito no vacío de nodos, E es un conjunto de pares no ordenados vw con $v, w \in V$, denominados *aristas* y A es un conjunto de pares ordenados (v, w) con $v, w \in V$, denominados *arcos*.

Dado un grafo mixto $G = (V, E, A)$, para todo $v \in V$ definimos como *grado* de v y lo notamos $\delta(v)$ al número de aristas de E incidentes en v . De manera similar, definimos el *grado de entrada* de v , denotado $\delta_-(v)$, como el número de arcos de A que tiene a v como su vértice final, y grado de salida $\delta_+(v)$, como el número de arcos de A que tiene a v como su nodo inicial.

Para todo $v \in V$ definimos $\Gamma_-(v)$ al conjunto de arcos tales que v es su extremo final, y de manera similar definimos $\Gamma_+(v)$ al conjunto de arcos tales que v es su extremo inicial.

Se denomina *camino* en un grafo a una secuencia de vértices tal que existe una arista entre cada vértice y el siguiente. Se dice que dos vértices están conectados si existe un camino que comience en uno y termine en otro. Dos vértices pueden estar conectados por varios caminos. Un *circuito* es un camino donde el vértice inicial y final son el mismo.

Un *circuito euleriano* es aquel que contiene todas las aristas de un grafo exactamente una vez. Un grafo que contiene un circuito euleriano se dice que el mismo es euleriano.

A partir de estas definiciones, el Teorema de Euler puede enunciarse formalmente de la siguiente manera:

Teorema: Sea un grafo $G = (V, E)$. Entonces, G es euleriano si y sólo si G es conexo y $\delta(v)$ es par, para todo $v \in V$.

Si se trata de un grafo dirigido:

Teorema: Sea un grafo dirigido $G = (V, A)$. Entonces, G es euleriano si y sólo si G es conexo y $\delta_+(v) = \delta_-(v)$ es par, para todo $v \in V$.

Cada arista o arco de un grafo puede tener una función costo asociada, la cual puede representar por ejemplo la distancia, tiempo u otro atributo.

2. METODOLOGÍA.

2.1 Obtención del plano de la zona

Accediendo al sitio web del SIG OpenStreetMap (OSM) se seleccionó el área de interés (Figura 4) y se descargó la información cartográfica asociada en un archivo de formato *osm*. Este tipo de archivos posee un formato Extensible Markup Language (XML) que básicamente consiste en una lista de elementos etiquetados. Para el caso de OSM esas etiquetas se vinculan con *node* y *way*. El elemento *node* consiste de un punto geolocalizado (latitud y longitud) y un identificador del mismo. El elemento *way* se refiere a una lista ordenada de *nodes* que conforman distintas figuras geométricas.

En concreto para nuestro trabajo se requirieron los *ways* que representan las calles y los *nodes* que identifican las intersecciones de estas.

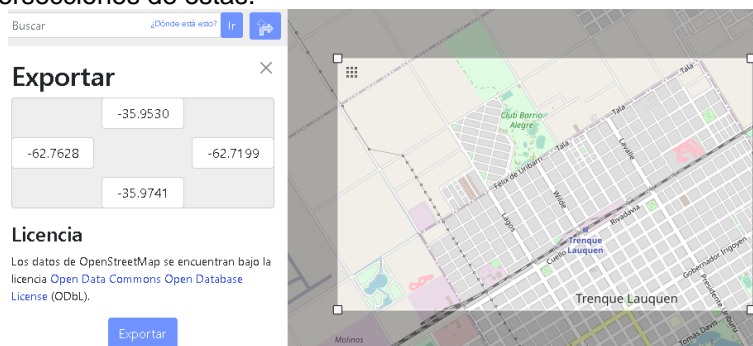


Figura 4 Exportación de datos cartográficos a través de OpenStreetMap.

Se utilizó el editor Java OpenStreetMap (JOSM) para eliminar la información que no era relevante y simplificar las calles con rambla al medio (boulevard) de manera que solo quede una vía de ambas manos conforme a la modalidad de trabajo de recolección que implica en una sola pasada recoger los residuos de ambas veredas (Figura 5).

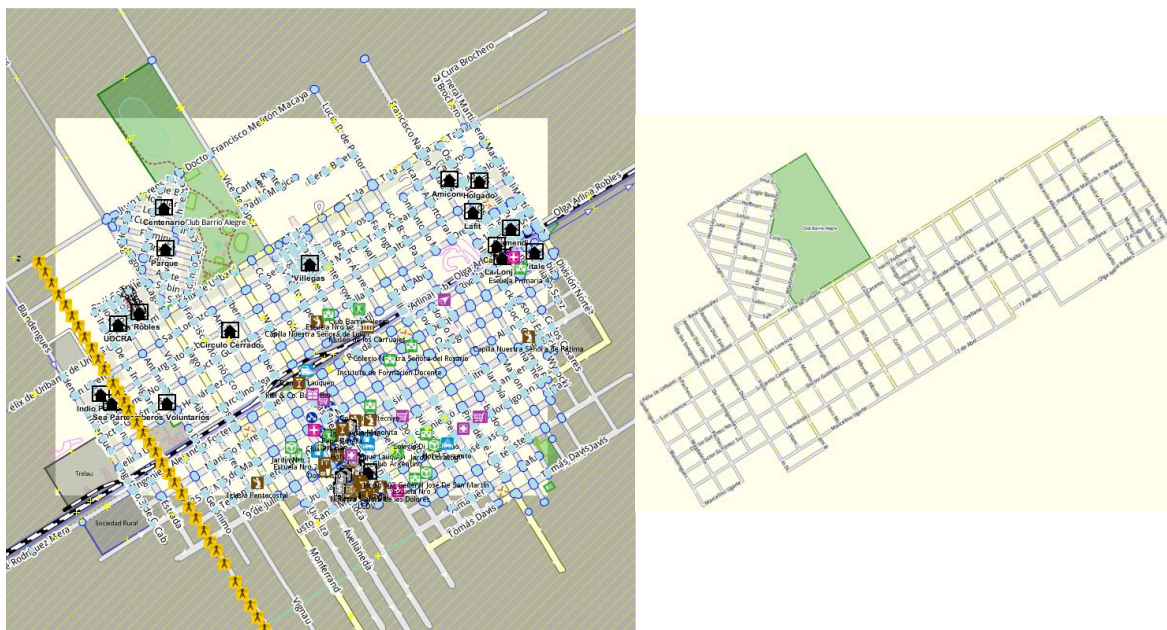


Figura 5 Limpieza de datos cartográficos mediante editor JOSM.

2.2 Construcción del grafo

La información del archivo osm se importó al software de análisis de datos R a través del uso del paquete *sf* (Pebesma, 2018) que permite obtener las figuras (en este caso la capa de calles) con sus atributos geográficos (latitud y longitud de los nodos que las constituyen).

De esta manera se obtuvo una tabla con 108 observaciones y 10 variables entre las que se destacan el identificador de osm para el objeto importado (*osm_id*), nombre de la calle (*name*), y nodos que la integran con su correspondiente latitud y longitud (*geometry*). El objeto *sf* se transformó en otro *sfnetworks* ya que el mismo permite con mayor sencillez interactuar con las aristas y nodos (Figura 6).

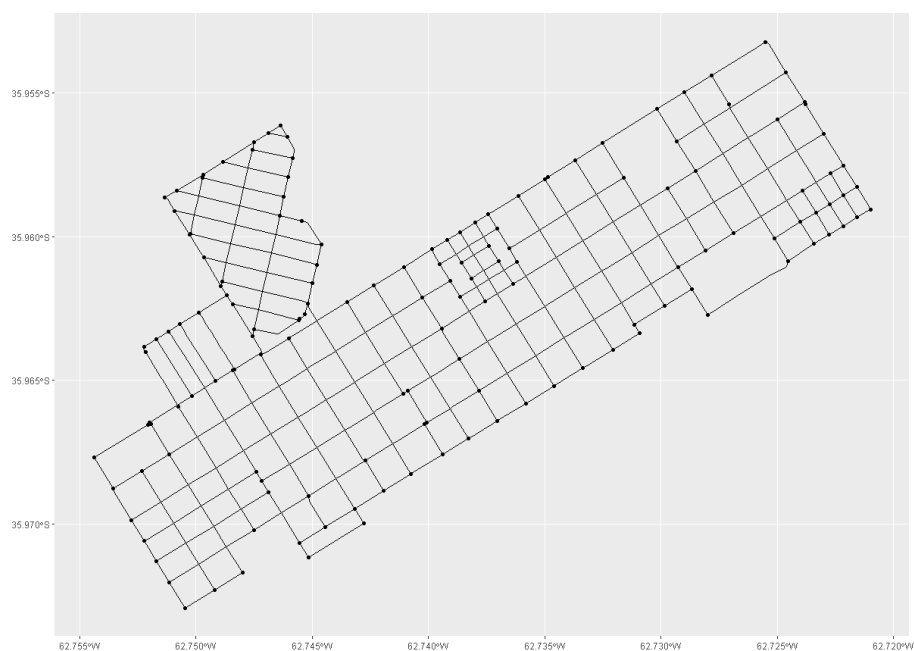


Figura 6 Visualización geográfica de los datos importados a R.

Mediante el uso del comando *to_spatial_subdivision* del paquete *sfnetwork* (van der Meer, Abad, Gilardi, & Lovelace, 2022) se transformó la red de calles en una nueva red segmentada en cada

intersección, ya que es necesario identificar cada tramo de manera particular para asegurar que sea recorrido por el camión.

De esta manera se llegó al arreglo de la Figura 7.

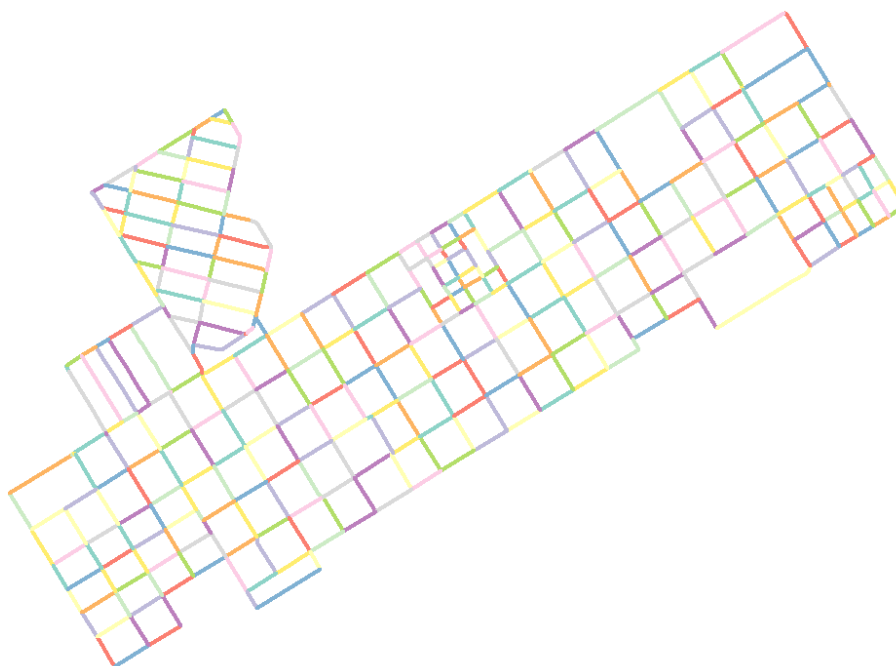


Figura 7 Plano segmentado en cada intersección.

Si bien en general se obtuvo un par de nodos por segmento, en algunos casos el número es mayor ya que deben representar curvas en el espacio. Para estos casos se desarrolló un script que permitía obtener únicamente los nodos iniciales y finales del segmento ya que solo estos son los que conforman los nodos del grafo a resolver.

A partir de la tabla con los segmentos (aristas) y nodos se creó el grafo asociado a la zona mediante el uso del paquete *igraph*. (Csardi & Nepusz, 2006).

Si bien la visualización del grafo obtenido es independiente de la geolocalización de sus nodos, se optó por mantener la misma ya que contribuía a interpretar los recorridos y nodos de grado impar en el grafo (Figura 8).

El grado de los nodos del grafo se observa en la Tabla 1. Como suele ocurrir, el grafo posee nodos impares y por lo tanto se lo debió transformar en euleriano para poder hallar un circuito que recorra todas las calles, pero repitiendo alguna de ellas.

Tabla 1 Grado de los nodos del grafo.

Grado	Cantidad
2	20
3	112
4	111

Al ser par la cantidad de nodos con grado 3, se generó un script en R que recorrió el grafo buscando para cada nodo impar su nodo impar vecino más próximo (utilizando la información de latitud, longitud y la fórmula de *Haversine*). A continuación se agregó una arista ficticia entre ellos transformándolos así en nodos pares manteniendo al mínimo la dimensión total del grafo. La Figura 9 representa estos arcos en color rojo y curvados.

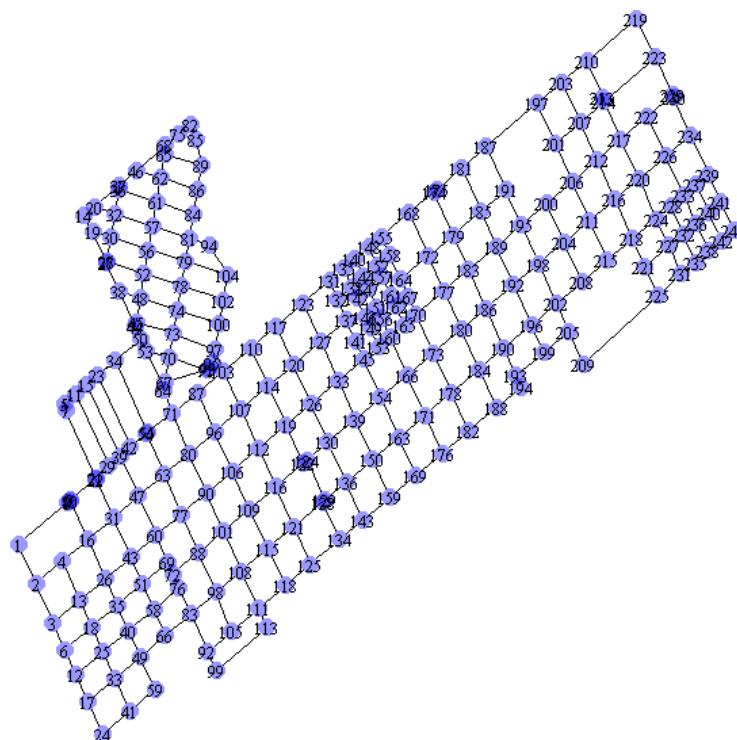


Figura 8 Representación gráfica del grafo con nodos geolocalizados

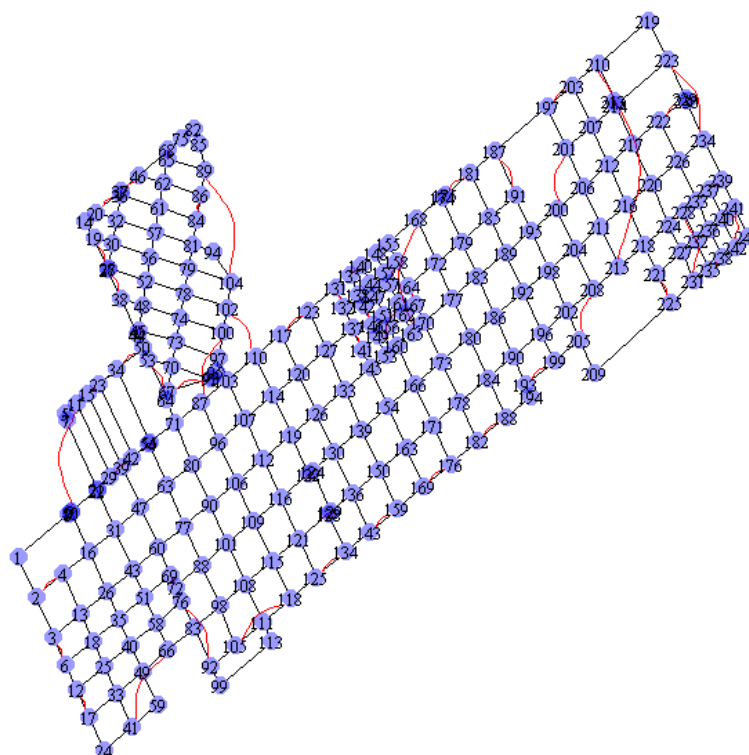


Figura 9 Grafo euleriano procesado completamente con R.

A través de la función *eulerian* del paquete *PairViz* (Hurley & Oldford, 2022), obtuvimos la propuesta de trayecto euleriano como un listado de nodos a recorrer, partiendo del nodo inicial 243.

Los primeros 10 nodos que encabezaron el listado fueron: 243-241-239-234-223-214-207-201-197-187.

Para comparar la idoneidad de la solución obtenida se generó un archivo con los nodos y arcos asociados al grafo (sin incluir los arcos ficticios propuestos por el script en R) y se procesaron los mismos mediante el software SCIP, el cual es uno de los solvers no comerciales más rápidos para la programación de enteros mixtos (MIP) y la programación no lineal de enteros mixtos (MINLP). Su uso es gratuito con fines de investigación como miembro de una institución académica o no comercial.

Como el grafo no es euleriano, se duplicaron las aristas intercambiando sus nodos extremos de manera que el *solver* pueda generar trayectos de ida y vuelta por la misma cuadra cuando sea necesario (o recorrerla más de una vez en el mismo sentido) siempre atendiendo a recorrer todas las aristas al menos una vez en cualquier sentido y minimizando el recorrido total. El costo de cada arista se asoció a la longitud de la misma.

El código procesado en SCIP fue el siguiente:

```
# adquisición de los nodos del grafo
set V := { read "nodos.txt" as "<1s>" comment "#" };
set P := { <i,j> in V*V };

# adquisición de los arcos del grafo: 1s = nodo1, 2s = nodo2, 3n = longitud del arco
param d[V*V] := read "arcos.txt" as "<1s,2s> 3n" default 0;
set E := { <i,j> in V*V with d[i,j] > 0 };

# x = cantidad de veces que se recorre el arco.
var x[<a,b> in E] integer >= 0;

# se minimiza la suma total del costo (distancia) multiplicada por la cantidad de veces que
# se recorrió el arco
minimize fobj: sum <i,j> in E: (d[i,j] * x[i,j]);

# la cantidad de veces que se ingresa a un nodo debe ser igual a la cantidad de veces que
# se sale del mismo para respetar el flujo dentro del nodo
subto flujo:
    forall <i> in V:
        (sum <j> in V with <j, i> in E: x[j,i]) ==
        (sum <j> in V with <i,j> in E: x[i,j]);

# con recorrer una arista en un solo sentido es suficiente
subto unavez:
    forall <i,v> in E:
        x[i,v] + x[v,i] >= 1;
```

La solución obtenida se importó nuevamente al software R para poder graficarla. Con color rojo se representan los arcos que se recorren una única vez, con color azul aquellos que se recorren dos veces, pero en distinto sentido, y con color verde los arcos que se recorren dos veces en el mismo sentido. La flecha en el arco indica el sentido propuesto en la solución.

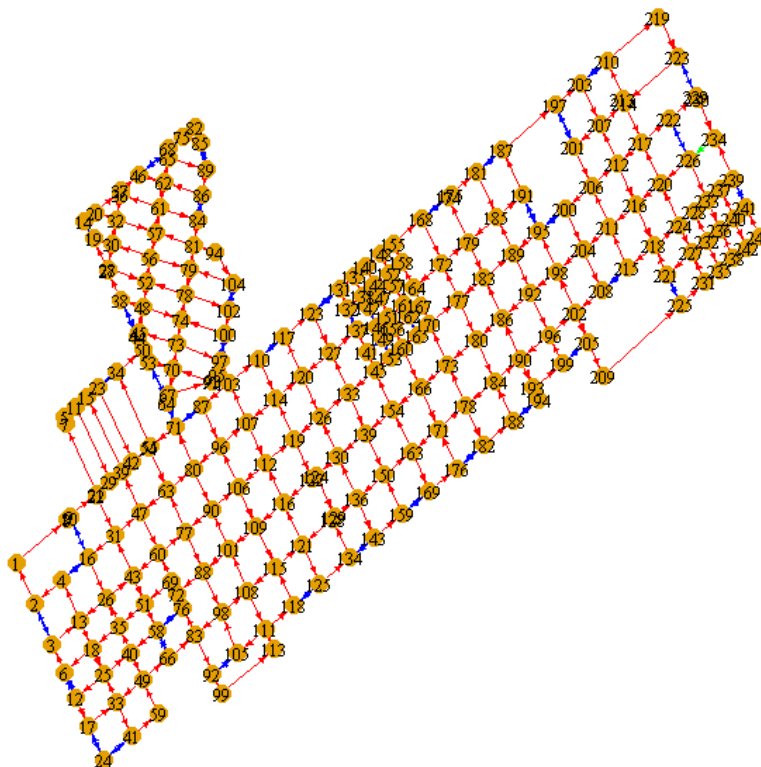


Figura 10 Grafo euleriano procesado con SCIP.

3. RESULTADOS.

Mediante el procedimiento para obtener el circuito euleriano enteramente con el software R se obtuvo un recorrido total aproximado de 52.120 metros, incluyendo el añadido de 6.140 metros de arcos ficticios. Se considera aproximado ya que la distancia de los arcos ficticios fue considerada en línea recta. El tiempo para el procesamiento y generación de gráficos es de 3 minutos. Puntualmente, el tiempo de generación del grafo euleriano (mediante aristas ficticias) y la generación del trayecto fue de 44 segundos.

El tiempo del *solver SCIP* para encontrar un circuito euleriano en el grafo es de 2 segundos, con una distancia total de 52.205 metros.

La distancia del circuito propuesto por ambos softwares es similar.

Comparando con el recorrido actual realizado por el camión de residuos se observa que las distancias obtenidas son superiores a las registradas por el sistema de seguimiento satelital del camión. El motivo es que actualmente el camión no recorre todas las calles. Para cubrir el área total lleva una dotación de tres recolectores (adicionales al chofer) que se turnan para ir reuniendo en las esquinas los residuos de las cuadras transversales de manera de reducir el tiempo del recorrido total. La Figura 11 visualiza esta situación. El cuadro inferior derecho resume el tiempo que demandó el recorrido y los kilómetros asociados.

El recorrido del camión es de 34 km, y posee 1 km para llegar y regresar de la zona al garaje donde se guardan los camiones. Por lo tanto, el recorrido realizado dentro de la zona es de 33 km. A partir de la sumatoria de la longitud de cada arista se obtiene que la zona posee una distancia total a recorrer de 46 km en total, lo que implica que los operarios recorren a pie 13 km (28% del total).

La duración actual es de 4 horas (237 minutos) para recorrer 34 km. Por lo que asumiendo una rapidez promedio similar, para recorrer los 52 km propuestos, el tiempo que requeriría el servicio sin que los operarios realicen una recolección paralela a pie, sería de aproximadamente 6 horas. La Tabla 2 resume estos valores.

Con esta información, el Municipio debería evaluar si, en base a las lesiones de trabajo y la ventana de tiempo disponible para prestar el servicio, es conveniente inclinarse por alguno de los circuitos eulerianos obtenidos en el presente trabajo. El beneficio que obtendría el Municipio es que al reemplazar la mayor parte de los recorridos a pie puede reducir la dotación del camión obteniendo

una diferencia positiva luego de descontar el mayor consumo de gasoil. Los trabajadores asignados no generan mayor costo ya que no se deben abonar horas extras debido a que la jornada municipal es de 6 horas. La reducción de accidentes laborales implica un beneficio adicional ya que trotar con bolsas de residuo es considerado una actividad riesgosa. El saldo positivo podría reasignarse a aumentar el número de trabajadores que realizan la separación de los residuos mejorando la eficiencia total del servicio.

Una desventaja que suelen exhibir estos circuitos es que presentan un número elevado de giros para minimizar el recorrido total evitando recorrer calles más de una vez.



Figura 11 Recorrido actual del camión recolector en la zona bajo estudio.

Tabla 2 Comparativo de parámetros de recorrido en la zona.

Variable	Actual	Propuesta
Distancia en cuadras de la zona	46 km	
Recorrido en camión	33 km	52 km
Recorrido a pie	13 km	0 km
Duración aproximada	4 h	6 h

El proyecto de investigación, que finaliza el 31/12, contempla enriquecer la propuesta del trayecto actual modelizando una penalidad en función del cambio de dirección al llegar a una intersección.

4. CONCLUSIONES.

Hoy existen herramientas libres y gratuitas que permiten encontrar soluciones para modelizar situaciones reales de logística como en el caso de recolección de residuos sólidos urbanos.

La solución obtenida consume mayor tiempo que la de los paquetes de software comercial, pero es aceptable para una ciudad del interior dividida en zonas como el caso estudiado.

Si bien el trayecto cumple con la exigencia de minimizar la distancia total recorrida, en la práctica debe incluir otros aspectos como por ejemplo la reducción en la cantidad de giros, tanto por el desgaste que se ocasiona en los vehículos como para facilitar a los conductores recordar la nueva ruta.

La disponibilidad de herramientas y repositorios cartográficos libres y gratuitos abre una nueva oportunidad para plantear soluciones a problemas con información cada vez más cercana a la realidad.

5. REFERENCIAS.

Beliën, J., De Boeck, L., & Van Ackere, J. (2014). Municipal Solid Waste Collection and Management Problems: A Literature Review.. *Transportation Science*, 48(1), 78-102.

Betanzo-Quezada, E., Torres-Gurrola, M. Á., Romero-Navarrete, J. A., & Obregón-Biosca, S. A. (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.

Csardi, G., & Nepusz, T. (2006). The igraph software package for complex network research. *InterJournal, Complex Systems*, 1695. Obtenido de <https://igraph.org>

Edmonds, J., & Johnson, E. L. (1973). Matching Euler tours and the Chinese postman. *Mathematical Programming*, 5(1), 88-124. <https://doi.org/10.1007/BF01580113>

Hurley C, Oldford R (2022). PairViz: Visualization using Graph Traversal. R package version 1.3.6, <https://cbhurley.github.io/PairViz/>.

Instituto Nacional de Estadísticas y Censos, Indec (2016). Población estimada al 1 de julio de cada año calendario por sexo, según partido. Provincia de Buenos Aires. Años 2010-2025.

Pebesma, E. (2018). Simple Features for R: Standardized Support for Spatial Vector Data. *The R Journal*, 10(1), 439--446. doi:10.32614/RJ-2018-009

Van der Meer, L., Abad, L., Gilardi, A., & Lovelace, R. (2022). sfnetworks: Tidy Geospatial Networks. Obtenido de <https://CRAN.R-project.org/package=sfnetworks>

Agradecimientos

Los autores de este trabajo desean agradecer a la Universidad Tecnológica Nacional y la Municipalidad de Trenque Lauquen por su apoyo para realizar el presente proyecto de investigación.