



Universidad
Zaragoza

Trabajo Fin de Grado

Optimización de rutas de recogida de residuos
en Fuentes de Ebro

Waste Collection Route Optimization in Fuentes
de Ebro

Autor

Darío Fraile Tolón

Director

Luis Mariano Esteban Escaño

Escuela Universitaria Politécnica La Almunia

2022

Página intencionadamente en blanco.



**Escuela Universitaria
Politécnica** - La Almunia
Centro adscrito
Universidad Zaragoza

**ESCUELA UNIVERSITARIA POLITÉCNICA
DE LA ALMUNIA DE DOÑA GODINA (ZARAGOZA)**

MEMORIA

Optimización de rutas de recogida de residuos
en Fuentes de Ebro

Waste Collection Route Optimization in Fuentes
de Ebro

30134

Autor: Darío Fraile Tolón

Director: Luis Mariano Esteban Escaño

Fecha: Septiembre 2022

Página intencionadamente en blanco.

INDICE DE CONTENIDO BREVE

1. RESUMEN	1
2. ABSTRACT	2
3. INTRODUCCIÓN	3
4. DESARROLLO	6
5. RESULTADOS	41
6. CONCLUSIONES	46
7. OBJETIVOS DE DESARROLLO SOSTENIBLE	48
8. BIBLIOGRAFÍA	50
9. ACCESIBILIDAD PARA DISCAPACITADOS	¡ERROR! MARCADOR NO DEFINIDO.
10. EJEMPLOS DE USO DE QR	¡ERROR! MARCADOR NO DEFINIDO.

INDICE DE CONTENIDO

1. RESUMEN	1
1.1. PALABRAS CLAVE	1
2. ABSTRACT	2
2.1. KEY WORDS	2
3. INTRODUCCIÓN	3
3.1. OBJETIVOS DEL TFG	3
3.2. DEFINICIÓN DEL PROBLEMA	4
3.3. METODOLOGÍA PARA LOGRAR LOS OBJETIVOS	4
3.4. PARTES EN LAS QUE SE ESTRUCTURA EL TFG	5
4. DESARROLLO	6
4.1. GESTIÓN Y RECOGIDA DE RESIDUOS SÓLIDOS URBANOS	6
4.1.1. <i>Residuos Sólidos Urbanos (RSU)</i>	6
4.1.2. <i>Gestión de RSU</i>	6
4.1.3. <i>Recogida de RSU</i>	6
4.1.4. <i>Necesidad de la gestión de RSU</i>	7
4.1.5. <i>Gestión de residuos en Fuentes de Ebro</i>	8
4.2. PROBLEMA DEL VIAJANTE Y ALGORITMO TSP	9
4.2.1. <i>Definición teórica</i>	9
4.2.2. <i>Tipos de TSPs</i>	10
4.2.3. <i>Algoritmos más utilizados</i>	10
4.2.4. <i>Aplicaciones</i>	11
4.3. OBTENCIÓN DE LA SOLUCIÓN.	12
4.3.1. <i>Aplicación Shiny/R</i>	12
4.3.2. <i>Library TSP</i>	13
4.3.3. <i>Library rconnect</i>	13
4.3.4. <i>Archivo server.R</i>	14
4.3.5. <i>Archivo ui.R</i>	23
4.4. PÁGINA WEB	31
5. RESULTADOS	41
5.1. RESULTADOS CONTENEDOR VERDE	41
5.2. RESULTADOS CONTENEDOR AMARILLO	42
5.3. RESULTADOS CONTENEDOR AZUL	44
6. CONCLUSIONES	46
7. OBJETIVOS DE DESARROLLO SOSTENIBLE	48
8. BIBLIOGRAFÍA	50
9. ACCESIBILIDAD PARA DISCAPACITADOS	¡ERROR! MARCADOR NO DEFINIDO.
10. EJEMPLOS DE USO DE QR	¡ERROR! MARCADOR NO DEFINIDO.

INDICE DE ILUSTRACIONES

Ilustración 1. Librería Shiny server.R	12
Ilustración 2. Librería Shiny ui.R.....	13
Ilustración 3. Librería TSP server.R.....	13
Ilustración 4. Librería rconnect ui.R.....	14
Ilustración 5. Función.....	14
Ilustración 6. Marco de Datos Verde.....	14
Ilustración 7. Condiciones de Entrada Verde	15
Ilustración 8. Vectores Verde	15
Ilustración 9. Desarrollo Vectores Verde	16
Ilustración 10. Resultado Verde.....	16
Ilustración 11. Marco de Datos Amarillo y Condiciones de Entrada	16
Ilustración 12. Vectores Amarillo	17
Ilustración 13. Resultado Amarillo.....	17
Ilustración 14. Marco de Datos Azul y Condiciones de Entrada.....	17
Ilustración 15. Vectores Azul.....	18
Ilustración 16. Resultado Azul	18
Ilustración 17. Marco de Datos 2 Verde	19
Ilustración 18. Marco de Datos 2 Amarillo.....	19
Ilustración 19. Marco de Datos 2 Azul	19
Ilustración 20. Solución Ruta Verde	19
Ilustración 21. Solución Ruta Amarillo	20
Ilustración 22. Solución Ruta Azul.....	20
Ilustración 23. Salida de Tabla Verde	21
Ilustración 24. Salida de Tabla Amarillo	21
Ilustración 25. Salida de Tabla Azul	22
Ilustración 26. Salida de Texto	22
Ilustración 27. Imprimibles Salida de Texto de los 3 Objetos.....	22
Ilustración 28. Diseño Página Fluida.....	23
Ilustración 29. Título de la Aplicación	23
Ilustración 30. Creación Panel de Barra Lateral.....	23
Ilustración 31. Creación Conjunto de Pestañas	24
Ilustración 32. Términos Pestañas Verde	24
Ilustración 33. Título del Elemento Verde.....	24
Ilustración 34. Número de Muestras Verde.....	24
Ilustración 35. Número Semilla de Aleatoriedad Verde	25
Ilustración 36. Variable Puntos de Recogida Verde	25
Ilustración 37. Puntos de Recogida Verde	26
Ilustración 38. Selección Puntos de Recogida Verde	26
Ilustración 39. Términos Pestañas Amarillo	27
Ilustración 40. Puntos de Recogida Amarillo	27
Ilustración 41. Selección Puntos de Recogida Amarillo	28
Ilustración 42. Términos Pestañas Azul	28
Ilustración 43. Puntos de Recogida Azul	29
Ilustración 44. Selección Puntos de Recogida Azul	29
Ilustración 45. Opciones de Elección del Algoritmo	30
Ilustración 46. Elementos de Salida del Panel Principal	31
Ilustración 47. Apariencia General de la App	32
Ilustración 48. Elección Tipo de Contenedor	33
Ilustración 49. Elección del Número de Muestras Predeterminada	33
Ilustración 50. Ejemplo 1 Número de Muestras.....	33
Ilustración 51. Ejemplo 2 Número de Muestras.....	33
Ilustración 52. Elección Semilla de Aleatoriedad Predeterminada	34

Ilustración 53. Ejemplo 1 Semilla de Aleatoriedad.....	34
Ilustración 54. Puntos Seleccionados Verde	35
Ilustración 55. Puntos Seleccionados Amarillo	35
Ilustración 56. Puntos Seleccionados Azul.....	36
Ilustración 57. Elección del Algoritmo.....	36
Ilustración 58. Zona de Resultados de la App.....	38
Ilustración 59. Ejemplo 1 de Resultado.....	39
Ilustración 60. Ejemplo 2 de Resultado.....	40
Ilustración 61. Ejemplo 3 de Resultado.....	40
Ilustración 62. ODS 4.....	48
Ilustración 63. ODS 8.....	48
Ilustración 64. ODS 11	48
Ilustración 65. ODS 13.....	49

1. RESUMEN

La optimización de rutas es un proceso muy importante que debe realizarse siempre en cualquier empresa u organización que cuente con un servicio de reparto o de recogida, ya que con una buena optimización de dichas rutas se consigue una reducción de costes notable además de una gran reducción del tiempo dedicado a esta tarea de reparto o de recogida.

En este trabajo, la optimización de rutas se ha realizado sobre las tres diferentes rutas de recogida existentes de los tres diferentes residuos sólidos urbanos (RSU) en la localidad de Fuentes de Ebro (Zaragoza), por lo que se va a realizar la solución de un problema real.

Estos RSU serán la materia orgánica (contenedor verde), el papel y cartón (contenedor azul) y el plástico (contenedor amarillo).

Para realizar dicha optimización se ha realizado el estudio de las rutas de recogida de la localidad gracias a la información proporcionada por el propio Ayuntamiento de Fuentes de Ebro, conociendo así las ubicaciones de cada uno de los diferentes puntos de recogida, el calendario de recogida de los diferentes residuos y las distancias y tiempos empleados en dicha recogida.

Posteriormente, se ha detallado el algoritmo elegido para la resolución del problema en cuestión, el TSP (Travelling Salesman Problem), así como los diferentes programas y paquetes necesarios para esta optimización, con la intención de realizar una aplicación que sea capaz de ofrecer este recorrido optimizado, así como la función de incorporar futuros nuevos puntos de recogida para lograr la nueva ruta óptima.

Finalmente, se ha realizado la explicación detallada por pasos de cómo se ha obtenido la solución al problema, mostrando los resultados obtenidos y comparándolos con los reales que teníamos de partida, obteniendo así nuestras propias conclusiones.

1.1. PALABRAS CLAVE

Optimización, rutas, TSP, recogida, residuos.

2. ABSTRACT

Route optimisation is a very important process that should always be carried out in any company or organisation that has a delivery or collection service, since a good optimisation of these routes leads to a significant reduction in costs as well as a great reduction in the time dedicated to this task of delivery or collection.

In this work, route optimisation has been carried out on the three different existing collection routes for the three different types of municipal solid waste (MSW) in the town of Fuentes de Ebro (Zaragoza), so the solution to a real problem is going to be carried out.

These MSW will be organic matter (green container), paper and cardboard (blue container) and plastic (yellow container).

In order to carry out this optimisation, a study of the collection routes of the town has been carried out thanks to the information provided by the Fuentes de Ebro Town Council itself, thus knowing the locations of each of the different collection points, the collection schedule of the different wastes and the distances and times used in this collection.

Subsequently, the algorithm chosen for the resolution of the problem in question, the TSP (Travelling Salesman Problem), has been detailed, as well as the different programs and packages necessary for this optimisation, with the intention of creating an application that is capable of offering this optimised route, as well as the function of incorporating future new collection points to achieve the new optimal route.

Finally, a detailed step-by-step explanation of how the solution to the problem has been obtained has been made, showing the results obtained and comparing them with the real ones we had as a starting point, thus obtaining our own conclusions.

2.1. KEY WORDS

Optimization, Routes, TSP, Collection, Waste

3. INTRODUCCIÓN

La elección de este Trabajo Fin de Grado (TFG) tiene su origen en el interés personal que me ha surgido a lo largo del Grado en los temas de optimización, y, sobre todo, en la optimización de rutas.

En el Grado de Ingeniería de Organización Industrial se le otorga una gran importancia a la mejora económica de las empresas con su adecuada gestión, y, en ello, la optimización, tanto de rutas como del resto de ámbitos empresariales, juega un papel importantísimo.

La optimización de rutas es algo vital para reducir costes en la empresa, y eso se ha enseñado en diferentes asignaturas a lo largo del grado como Investigación Operativa, Ampliación de Investigación Operativa o Logística, pero, sobre todo, tras realizar la asignatura de Investigación Operativa es cuando se despierta mi interés en temas de optimización.

Además, otra de las razones y motivaciones por las que realizo este trabajo es por el hecho de que Fuentes de Ebro es la localidad en la que resido, y por ello, planear una optimización de rutas en su recogida de residuos es algo que puede beneficiar a mi localidad y evitar pérdidas tanto temporales como económicas.

La gestión de estas tareas en Fuentes de Ebro es realizada por la empresa Seula, reconocida como una empresa referente regional en el sector de la gestión de residuos.

Por todo esto, el TFG tiene como fundamento lograr una optimización real de las rutas de recogida de residuos de Fuentes de Ebro para comparar estos resultados con los existentes hoy en día y poder ofrecer una solución al problema.

3.1. OBJETIVOS DEL TFG

Como objetivo principal de este trabajo se encuentra la optimización de las tres diferentes rutas de recogida de los tres diferentes residuos en Fuentes de Ebro, con la intención de minimizar la distancia que recorren los camiones en cada recogida, lo que conlleva una reducción de costes y de tiempo empleado en esta tarea.

Esto genera otros dos objetivos a cumplir:

- Una ruta óptima que pase por todos los diferentes puntos de recogida de residuos generada en una aplicación de la que dispondrá el conductor de cada camión.

- Una nueva funcionalidad en la que se incorporen en la aplicación los nuevos puntos de recogida que puedan crearse en un futuro y ofrecer así las nuevas rutas óptimas.

3.2. DEFINICIÓN DEL PROBLEMA

El problema a resolver del TFG es la consecución de las rutas óptimas de la recogida de residuos de Fuentes de Ebro, por lo que se trata de un problema y una situación real.

Para ello, se comenzará obteniendo todas las coordenadas GPS específicas de cada uno de los puntos de recogida de los diferentes tipos de residuos de la localidad, para, posteriormente, convertir estas coordenadas en una matriz de distancias entre cada uno de los puntos de recogida en las que relacionemos todas ellas directamente con sus distancias exactas.

Una vez obtenida esta matriz, se procederá a crear el código necesario mediante la librería TSP perteneciente al software R, con el que se obtendrá la solución a este denominado problema del viajante, que consiste en encontrar el recorrido más corto posible pasando una vez por cada uno de los puntos existentes y regresando al punto de partida.

Este software se ha empleado a lo largo del grado universitario en diversas asignaturas como Estadística, Investigación Operativa y Ampliación de Investigación Operativa.

3.3. METODOLOGÍA PARA LOGRAR LOS OBJETIVOS

Se van a seguir 6 diferentes pasos para lograr los objetivos propuestos:

1. Estudio de cómo es la gestión de recogida de RSU en Fuentes de Ebro.
2. Obtención de las coordenadas GPS de los 82 puntos diferentes de recogida de materia orgánica, de los 61 puntos diferentes de recogida de plástico y de los 56 puntos diferentes de recogida de papel y cartón.
3. Creación de la matriz de distancias entre puntos de recogida eligiendo cuál es el punto de partida y cuál es el punto final de cada una de las tres rutas.
4. Creación del código y ejecución del algoritmo en el software R que realice el cálculo de la distancia más corta en cada una de las tres rutas
5. Implementación de la aplicación web con los datos obtenidos de las rutas óptimas.

6. Conclusiones y propuestas de mejora (si procede)

3.4. PARTES EN LAS QUE SE ESTRUCTURA EL TFG

El trabajo constará de los siguientes apartados:

- Gestión y recogida de residuos sólidos urbanos
- Problema del viajante y algoritmo TSP
- Herramientas usadas para el cálculo de la ruta óptima.
- Obtención y análisis de la solución
- Obtención de conclusiones y propuestas de mejora.

4. DESARROLLO

4.1. GESTIÓN Y RECOGIDA DE RESIDUOS SÓLIDOS URBANOS

4.1.1. Residuos Sólidos Urbanos (RSU)

Los residuos sólidos urbanos (RSU) se definen en la Ley de Residuos como los generados en los domicilios particulares, comercios, oficinas y servicios, así como todos aquellos que no tengan la calificación de peligrosos y que por su naturaleza o composición puedan asimilarse a los producidos en los anteriores lugares o actividades.

Tienen también la consideración de residuos urbanos según la citada ley, los siguientes:

- Residuos procedentes de la limpieza de vías públicas, zonas verdes, áreas recreativas y playas.
- Animales domésticos muertos, así como muebles, enseres y vehículos abandonados.
- Residuos y escombros procedentes de obras menores de construcción y reparación domiciliaria.

4.1.2. Gestión de RSU

Se considera como gestión de los residuos sólidos urbanos al conjunto de operaciones que se realizan con ellos desde que se generan en los hogares y servicios hasta la última fase en su tratamiento. Abarca pues tres etapas:

1. Depósito y recogida.
2. Transporte.
3. Tratamiento.

(UNED Biblioteca, 2003. *Gestión de los residuos sólidos urbanos.*)

4.1.3. Recogida de RSU

La recogida de los residuos urbanos consiste en su recolección para efectuar su traslado a las plantas de tratamiento.

Básicamente existen dos tipos fundamentales de recogida:

- Recogida no selectiva.
- Recogida selectiva.

En la primera, los residuos se depositan mezclados en los contenedores, sin ningún tipo de separación.

La recogida selectiva se hace separando los residuos según su clase y depositándolos en los contenedores correspondientes. Así, existen normalmente contenedores para el papel, vidrio, envases y la materia orgánica.

(UNED Biblioteca, 2003. *Gestión de los residuos sólidos urbanos.*)

Hasta hace unos años, la recogida habitual era la recogida no selectiva, pero esto ha cambiado y ha generado que la recogida de nuestro país sea selectiva.

A su vez, el método de recogida empleado mayoritariamente es mediante vehículos, es decir, diferentes vehículos preparados especialmente para cada uno de los diferentes residuos; sin embargo, también existe otro método de recogida, la recogida neumática.

El sistema de recogida neumática de residuos consiste en una serie de buzones de vertido conectados mediante tuberías subterráneas al punto de captura desde donde se realiza una aspiración del circuito. (Gobierno de España, 2020. *Sistema de Recogida.*)

4.1.4. Necesidad de la gestión de RSU

La gestión de RSU tiene una gran importancia hoy en día y es un proceso vital tratando desde el punto de vista ambiental. Una buena y adecuada gestión de estos residuos logra reducir los impactos negativos que existen sobre el medioambiente, así como los impactos negativos existentes sobre la salud.

Además, el buen manejo de estos residuos ayuda a mitigar la presión que se ejerce sobre los recursos naturales y los ecosistemas.

Por todo ello, es imprescindible incluir el reciclaje en esta gestión de residuos, ya que favorece enormemente a los buenos resultados mencionados anteriormente; además de la máxima reutilización posible de los materiales de los distintos RSU.

Finalmente, cabe destacar la importancia que tiene a nivel económico, ya que conseguimos reducir muchos gastos en las empresas a la hora de compras de nuevos materiales o de la producción de estos, debido a que la reutilización de ellos es un proceso más barato. También, permite reducir muchos gastos en la fase de recogida y transporte de los residuos, en calidad de combustibles y tiempos empleados.

4.1.5. Gestión de residuos en Fuentes de Ebro

En este apartado se va a analizar y detallar la gestión de los RSU en Fuentes de Ebro, localización en la cual se va a desarrollar el objetivo de este trabajo.

Según la Memoria aportada por la empresa Seula, encargada de esta función mencionada, al Ayuntamiento de Fuentes de Ebro la recogida de RSU constará de materia orgánica, papel-cartón y plásticos.

Seula se encargará de destinar un camión diferente para cada una de estas recogidas y cada camión tendrá su propio recorrido y su propio personal.

En los tres tipos de camiones se encontrarán 2 trabajadores, el conductor encargado de conducir el camión y de las operaciones en cabina y un peón encargado de colocar el contenedor en la posición adecuada para su recogida.

Seula establece un calendario para cada uno de los 3 tipos de residuos que se van a recoger.

En el caso de la materia orgánica (contenedor verde) existe una recogida diaria de lunes a sábado, es decir, 6 días a la semana. El horario de recogida se establecerá desde las 23:00 horas hasta las 07:00 horas.

En el caso del papel-cartón (contenedor azul) existe una recogida semanal siendo este día a concretar según la semana. El horario de recogida se establecerá desde las 23:00 horas hasta las 07:00 horas.

En el caso de los plásticos (contenedor amarillo) al igual que en el caso anterior, existe una recogida semanal siendo este día a concretar según la semana. El horario de recogida se establecerá desde las 23:00 horas hasta las 07:00 horas.

Además de todo esto, también se incluye el número de puntos de recogida de cada tipo de residuo, siendo de 82 puntos diferentes para el contenedor verde, 61 puntos diferentes para el contenedor amarillo y 56 para el contenedor azul.

Por último, se indica aproximadamente el tiempo empleado en la recogida de cada residuo, así como la distancia recorrida de cada ruta.

Para el contenedor verde se estima un recorrido de 49.4 km y un tiempo aproximado de 4 horas.

Para el contenedor amarillo se estima un recorrido de 40.8 km y un tiempo aproximado de 2 horas y 55 minutos.

Para el contenedor azul se estima un recorrido de 22.1 km y un tiempo aproximado de 2 horas y 30 minutos.

4.2. PROBLEMA DEL VIAJANTE Y ALGORITMO TSP

El problema del viajante (TSP) es un problema basado en encontrar el camino más corto entre los diferentes puntos a recorrer pasando una vez por cada uno y regresando al punto de origen. Es uno de los problemas más famosos de la optimización combinatoria computacional.

En el caso de este trabajo, el objetivo es encontrar la ruta más corta entre todos los diferentes puntos de recogida de residuos de Fuentes de Ebro pasando una vez por cada punto y regresando al lugar de partida.

4.2.1. Definición teórica

Se va a exponer una de las formulaciones clásica del TSP, y para ello debemos tener en cuenta los siguientes conceptos:

- Un grafo $G = (N, A)$ consistente en un conjunto N de elementos llamados vértices o nodos y un conjunto A cuyos elementos llamaremos arcos o aristas. Cada arco o arista representa una conexión directa entre dos nodos o elementos de N .
- Un grafo orientado o dirigido es aquel en el que los arcos son pares ordenados; el arco (i, j) empieza en el nodo i y termina en el nodo j .
- Un grafo no orientado o no dirigido, es un grafo en el que (i, j) y (j, i) representan el mismo arco.
- Un grafo completo es un grafo (orientado o no) en el que todas las aristas posibles están presentes.
- Un camino es una secuencia de aristas en las que todos los vértices son distintos.
- Un circuito es una secuencia de aristas en la que el primer vértice y el último son el mismo y además no hay más vértices coincidentes que estos dos.
- Un camino hamiltoniano es un camino que recorre todos los nodos del grafo una y sólo una vez.
- Un circuito hamiltoniano es un circuito que recorre todos los nodos del grafo una y sólo una vez.

Conociendo estas definiciones, el problema del TSP puede ser descrito según la teoría de grafos de la siguiente manera:

Sea $G = (N, A)$ un grafo completo, donde $N = 1, \dots, n$ es el conjunto de nodos o vértices y A es el conjunto de arcos. Los nodos $i = 2, \dots, n$ se

corresponden con los clientes a visitar mientras que el nodo 1 es considerado la ciudad de origen y destino. A cada arco (i, j) se le asocia un valor no negativo d_{ij} , que representa la distancia del vértice i al j . El uso de los arcos (i, i) no está permitido, por lo que se impone que $d_{ii}=1$ $\forall i \in N^2$.

(Universidad de Valladolid, 2015. *Resolución del Problema del Viajante de Comercio (TSP) y su variante con Ventanas de Tiempo (TSPTW) usando métodos heurísticos de búsqueda local*)

Este problema es del tipo NP-Hard, es decir, un problema muy difícil de resolver en el que la solución óptima y exacta no es lo que se busca, sino la mayor aproximación posible a ella.

4.2.2. Tipos de TSPs

Se pueden diferenciar cuatro tipos de TSPs:

- Simétrico: en este tipo las distancias entre un par de puntos son siempre la misma independientemente del sentido a tomar. En el TSP simétrico la matriz de distancias es simétrica y es el TSP que se ha utilizado para realizar este TFG.
- Asimétrico: en este tipo las distancias pueden no ser las mismas entre dos puntos dependiendo del sentido que se tome, es por ello que la matriz de distancias en este caso no es simétrica.
- Euclidiano: en este tipo las distancias entre dos puntos son las distancias euclidianas.

La distancia euclidiana es un número positivo que indica la separación que tienen dos puntos en un espacio donde se cumplen los axiomas y teoremas de la geometría de Euclides. La distancia entre dos puntos A y B de un espacio euclidiano es la longitud del vector AB perteneciente a la única recta que pasa por dichos puntos. (Cajal, A., 2019, diciembre 3. *Distancia euclidiana: Concepto, fórmula, cálculo, ejemplo. Lifeder*).

- Con desigualdad triangular: en este tipo se expone que la distancia desde el punto A hasta el punto B no sea mayor que la distancia de A hasta B pasando por el punto C: $d_{AC} + d_{CB} \geq d_{AB}$

4.2.3. Algoritmos más utilizados

En este apartado se van a diferenciar entre dos algoritmos principales para el problema del viajante:

- Algoritmo del vecino más cercano: comenzando desde cualquier punto, se elige el punto más cercano entre los puntos en los que todavía no haya estado, hasta que se recorran todos los puntos existentes y se regrese al punto de partida.
- Algoritmos de inserción: estos algoritmos consisten en construir ciclos que primeramente visiten unos determinados puntos, para después extender estos ciclos insertando los puntos que quedaban fuera del ciclo. En cada paso se va incorporando un nuevo punto en el ciclo hasta lograr obtener el ciclo óptimo.

Existen cuatro criterios destacables dentro de estos algoritmos:

- Inserción más cercana: Seleccionar el vértice j más cercano al ciclo. $d_{min}(j) = \min\{d_{min}(v)/v \in W\}$
- Inserción más lejana: Seleccionar el vértice j más lejano al ciclo. $d_{min}(j) = \max\{d_{min}(v)/v \in W\}$
- Inserción más barata: Seleccionar el vértice j con el menor incremento del coste
- Inserción aleatoria: Seleccionar el vértice j al azar
(Cádiz, U. de., 2015. *TSP: Algoritmos de resolución*).

4.2.4. Aplicaciones

El problema del viajante tiene muchas diversas aplicaciones en el mundo en el que vivimos.

Las principales aplicaciones son para la planificación de rutas de logística, debido a que este problema genera una solución de ruta óptima que es en lo que se basa esta primera aplicación mencionada.

Además, también se aplica en otras áreas muy distintas como en la fabricación de placas de circuitos; en este caso se enfoca a conseguir el orden óptimo en el que taladrar las placas y los caminos óptimos de comunicación mediante cableado entre los chips.

Por último, encontramos una aplicación en los servidores de Red, ya que al fin y al cabo Internet es una plataforma de distribución y se quiere conseguir el uso óptimo de esta.

Estas son las aplicaciones más importantes y destacables de este problema, pero existen otras tantas como en la secuenciación del ADN o para el análisis estadístico de datos.

4.3. OBTENCIÓN DE LA SOLUCIÓN.

En este apartado se va a desarrollar y detallar el trabajo realizado con sus respectivos pasos para obtener la solución al problema descrito y tratado anteriormente para este proyecto.

4.3.1. Aplicación Shiny/R

Para comenzar, se describe la aplicación Shiny del Software R que va a ser con la que se van a desarrollar las rutas óptimas de cada recogida de residuos tratada.

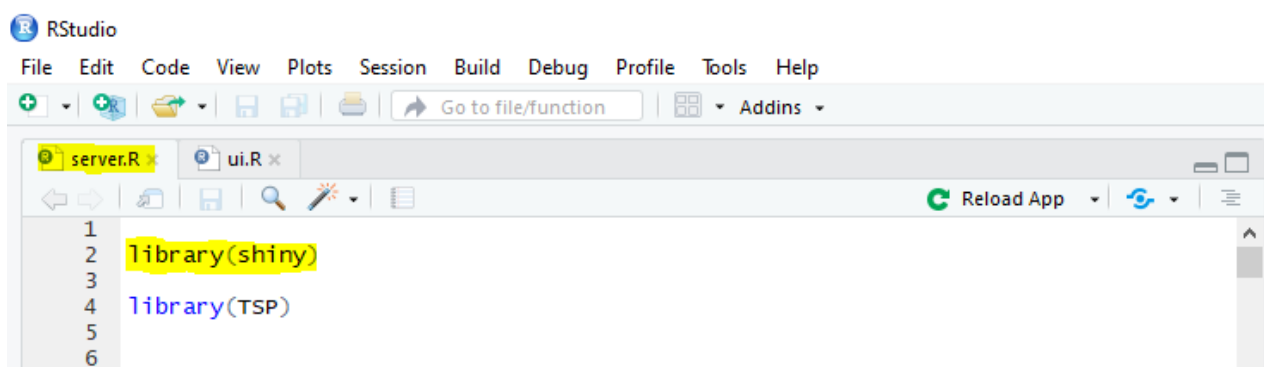
Mediante el software R y el paquete Shiny se genera todo el código necesario para que se cree una Aplicación Web en la que se muestre la solución al problema definido.

Esta aplicación Shiny está formada por dos archivos diferentes:

- Server.R en el cual se define la lógica del servidor y donde es generado el contenido que depende de las interacciones que hace el usuario en la pantalla.
- Ui.R es donde están especificados los elementos de la interfaz y la disposición de estos en la propia pantalla, es decir, donde se encuentra el código necesario para mostrar la información en la pantalla y los parámetros que la hacen interactiva.

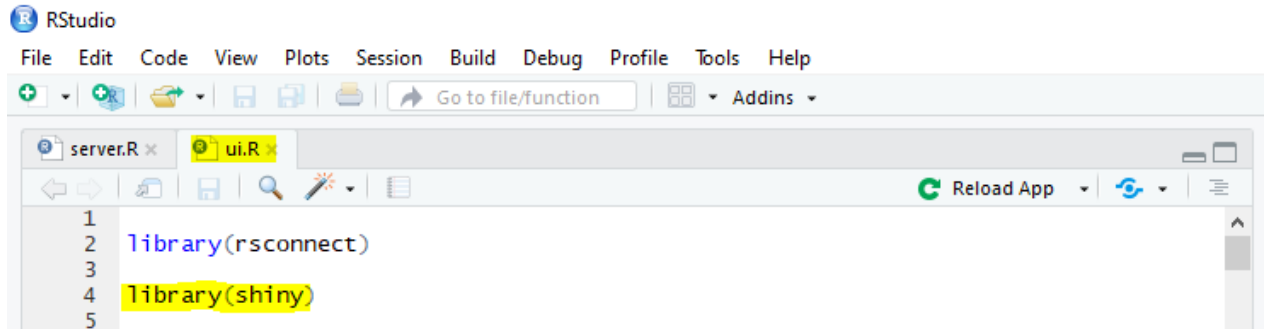
Con estos dos archivos será suficiente para la creación de la app en la que se mostrará la solución del problema.

A continuación, se muestran dos imágenes en las que podemos ver el código con la librería shiny.



```
1  
2 library(shiny)  
3  
4 library(TSP)  
5  
6
```

Ilustración 1. Librería Shiny server.R



```

1
2 library(rsconnect)
3
4 library(shiny)
5

```

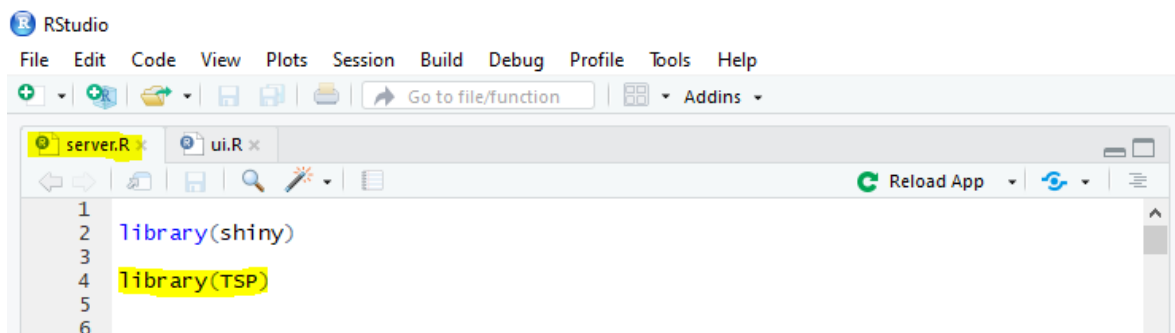
Ilustración 2. Librería Shiny ui.R

4.3.2. Library TSP

Mediante esta librería se va a aportar la infraestructura y los algoritmos para el problema del viajante, es decir, esta librería va a ser la encargada de obtener la solución de las rutas óptimas de nuestro problema.

Esta librería va a ser utilizada en el archivo server.R de la aplicación Shiny.

A continuación, se muestra imagen de la librería mencionada.



```

1
2 library(shiny)
3
4 library(TSP)
5
6

```

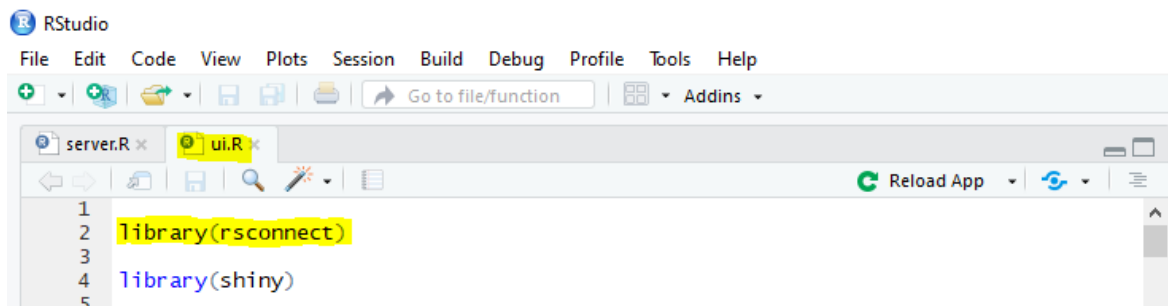
Ilustración 3. Librería TSP server.R

4.3.3. Library rsconnect

Instalando esta librería se obtiene un paquete que sirve para poder desplegar las aplicaciones del Shiny.

Esta librería va a ser utilizada en el archivo ui.R de la aplicación Shiny.

A continuación, se muestra imagen de la librería mencionada.



```

1
2 library(rsconnect)
3
4 library(shiny)
5

```

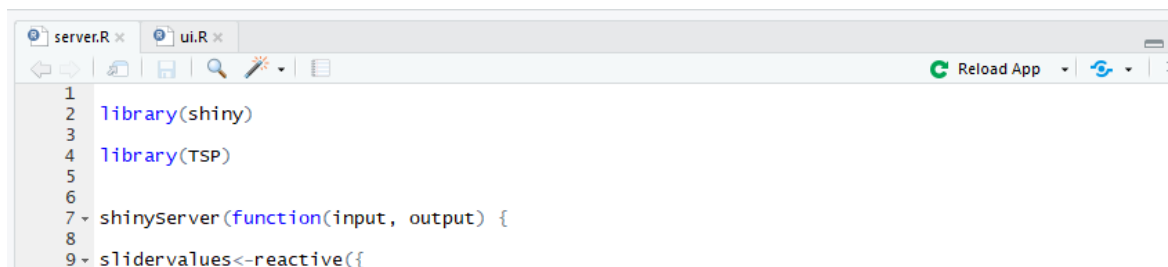
Ilustración 4. Librería rsconnect ui.R

4.3.4. Archivo server.R

En este apartado se va a explicar cómo se ha construido y desarrollado el archivo server de la aplicación.

Primero, se debe definir la función de nuestro código con sus respectivas entradas (input) y salidas (output) y mediante el término "function".

Posteriormente, con "slidervalues<-reactive" se crea el marco de datos que contiene todos los valores de entrada.



```

1
2 library(shiny)
3
4 library(TSP)
5
6
7 shinyServer(function(input, output) {
8
9 slidervalues<-reactive({

```

Ilustración 5. Función

Una vez introducidos estos términos, se empieza definiendo la primera matriz de distancias; en este caso, la de los contenedores verdes (materia orgánica). Se llamará "MatrizDistanciasVerde" y estará previamente cargada en el software.

El término "as.numeric" se usa para establecer un factor como numérico.

```

9 slidervalues<-reactive({
10
11   Matriz<-MatrizDistanciasVerde[as.numeric(c(input$variable)),as.numeric(c(input$variable))]
12
13

```

Ilustración 6. Marco de Datos Verde

Una vez definida la matriz con la primera entrada denominada "variable", añadimos la primera condición con la primera entrada en la que siempre se cumplirá el "if" que determina la longitud de esta.

Con esto cumplido, se inserta la dummy, que ejerce de corte, para hacer el tour hamiltoniano tal y como se indica en el comentario.

Con el "method" se recoge el valor del input relativo al algoritmo empleado.

```
17  if (length(input$variable)<2) stop("")
18  else {
19    PuntosRepartoTSP<-TSP(Matriz, labels = NULL)
20
21    #Hay que insertar la dummy para hacer el tour hamiltoniano
22    CRTsp <- insert_dummy(PuntosRepartoTSP, label = "cut")
23    method1<-input$Method
24
```

Ilustración 7. Condiciones de Entrada Verde

Con el siguiente paso se empieza a codificar el cálculo de la solución. Para obtener una solución óptima utilizando el método aleatorio ϵ HO, lo que hay que hacer es crear tres vectores diferentes para obtener tres colecciones de datos iguales; en este caso, con la entrada "Nmuestras" que fija el número de muestras de cada vector.

Estos vectores son utilizados para la función siguiente a desarrollar, el "for".

```
28  x<-vector("list", input$Nmuestras)
29  w<-vector("list", input$Nmuestras)
30  v<-vector("list", input$Nmuestras)
```

Ilustración 8. Vectores Verde

Con los vectores creados, se define el "for" para ejecutar el bucle ya que se generan aleatoriamente un número de rutas para su posterior optimización.

Dentro de esta función aparecen los vectores definidos anteriormente.

Además, se establece la semilla de aleatoriedad, con la que se consigue que el código sea reproducible ya que se genera una semilla inicial; en este caso, empezando la ruta siempre en el punto 1.

```

31 for (i in 1:input$Nmuestras){
32
33
34   set.seed(i+input$Semilla)
35   x[[i]]<-solve_TSP((CRTsp),method1,control=list(start=1))
36
37
38   w[[i]]<-as.integer(x[[i]])
39
40
41   v[[i]]<-tour_length(CRTsp, w[[i]])
42
43 }

```

Ilustración 9. Desarrollo Vectores Verde

La última parte del “for” incluye el resultado nombrado como “results1” en el que se guarda la longitud mínima de la ruta.

La segunda parte, el “labels”, hace referencia a la etiqueta de la ruta; ofrece los nombres de este resultado como aparece en el comentario.

```

45 results1<-x[[which.min(unlist(v))]]
46 #función labels ofrece los nombres
47 labels(results1)[labels(results1)!="cut"]
48
49 }
50 })

```

Ilustración 10. Resultado Verde

Con esto se termina el marco de datos de la matriz de distancias del primer contenedor y se procede a realizar lo mismo con el segundo, que es el contenedor amarillo; “slidervaluesamarillo”.

La matriz se denomina “MatrizDistanciasAmarillo” y la entrada será “variableamarilla”. También estará previamente cargada en el software.

El resto de código usado tiene la misma estructura que la parte anterior.

```

51 slidervaluesamarillo<-reactive({
52
53   Matriz<-MatrizDistanciasAmarillo[as.numeric(c(input$variableamarilla)),as.numeric(c(input$variableamarilla))
54   if (length(input$variableamarilla)<2) stop("")
55   else {
56     PuntosRepartoTSP<-TSP(Matriz, labels = NULL)
57
58     #Hay que insertar la dummy city para hacer el tour hamiltoniano
59     CRTsp <- insert_dummy(PuntosRepartoTSP, label = "cut")
60     method1<-input$Method

```

Ilustración 11. Marco de Datos Amarillo y Condiciones de Entrada

En el caso de los vectores, para que el software no cargue los mismos datos se le añade “Am” y en la entrada de las Nmuestras se le añade también la palabra “amarillo” para diferenciar.

También, en la semilla de aleatoriedad se debe cambiar el nombre y será "semillaamarillo".

El resto de estructura se realiza de igual modo que para el contenedor verde.

```

65     xAm<-vector("list", input$Nmuestrasamarillo)
66     wAm<-vector("list",input$Nmuestrasamarillo)
67     vAm<-vector("list",input$Nmuestrasamarillo)
68     for (i in 1:input$Nmuestrasamarillo){
69
70
71         set.seed(i+input$Semillaamarillo)
72         xAm[[i]]<-solve_TSP((CRTsp),method1,control=list(start=1))
73
74
75         wAm[[i]]<-as.integer(xAm[[i]])
76         |
77
78         vAm[[i]]<-tour_length(CRTsp, wAm[[i]])
79

```

Ilustración 12. Vectores Amarillo

Por último, en cuanto al resultado se le nombra como "results1Amarillo" pero también funciona del mismo modo.

```

82     results1Amarillo<-xAm[[which.min(unlist(vAm))]]
83     #función labels ofrece los nombres
84     labels(results1Amarillo)[labels(results1Amarillo)!="cut"]
85
86 }
87 })

```

Ilustración 13. Resultado Amarillo

El último slidervalues a desarrollar es para la última de las rutas a tratar que es para el contenedor azul; "slidervaluesazul".

Para esta matriz, se le cambia el nombre a "MatrizDistanciasAzul" y al igual que las dos anteriores debe estar cargada previamente en R.

La variable es "variableazul" y el resto se mantiene igual que los dos casos previamente descritos.

```

88     slidervaluesazul<-reactive({
89
90     Matriz<-MatrizDistanciasAzul[as.numeric(c(input$variableazul)),as.numeric(c(input$variableazul))]
91     if (length(input$variableazul)<2) stop("")
92     else {
93         CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)
94
95         #Hay que insertar la dummy city para hacer el tour hamiltoniano
96         CRTsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")
97         method1<-input$Method
98

```

Ilustración 14. Marco de Datos Azul y Condiciones de Entrada

En cuanto a los vectores, se nombran como "xAz", "wAz" y "vAz" para diferenciar de los amarillos y los verdes y las Nmuestras para esta ruta es "Nmuestrasazul"

La semilla también se renombra y se llama "semillaazul". El resto de código también se mantiene idéntico y con la misma funcionalidad que lo explicado anteriormente.

```

102 xAz<-vector("list", input$Nmuestrasazul)
103 wAz<-vector("list", input$Nmuestrasazul)
104 vAz<-vector("list", input$Nmuestrasazul)
105 for (i in 1:input$Nmuestrasazul){
106
107
108     set.seed(i+input$Semillaazul)
109     xAz[[i]]<-solve_TSP((CRTsp),method1,control=list(start=1))
110
111
112     wAz[[i]]<-as.integer(xAz[[i]])
113
114
115     vAz[[i]]<-tour_length(CRTsp, wAz[[i]])
116
117 }

```

Ilustración 15. Vectores Azul

Por último, al igual que los dos casos vistos, se describe el resultado con la longitud mínima de esta ruta.

```

119 results1<-xAz[[which.min(unlist(vAz))]]
120 #función labels ofrece los nombres
121 labels(results1)[labels(results1)!="cut"]
122
123 }
124 })

```

Ilustración 16. Resultado Azul

Una vez se termina toda esta parte del código con la cual se logra obtener los resultados de las longitudes mínimas de la ruta, se pasa a la segunda parte en la que se logra obtener las rutas mínimas.

Para ello, se duplica todo el código anterior desde la función "slidervalues" para cada uno de los tres tipos de contenedores cambiando el nombre de este por "slidervalues1"

El nombre de las variables se mantiene igual que la primera parte del código.

```
126 - slidervalues1<-reactive({
127
128     Matriz<-MatrizDistanciasVerde[as.numeric(c(input$variable)),as.numeric(c(input$variable))]
129 }
```

Ilustración 17. Marco de Datos 2 Verde

```
169 - slidervalues1Amarillo<-reactive({
170
171     Matriz<-MatrizDistanciasAmarillo[as.numeric(c(input$variableamarilla)),as.numeric(c(input$variableamarilla))]
```

Ilustración 18. Marco de Datos 2 Amarillo

```
213 - slidervalues1Azul<-reactive({
214
215     Matriz<-MatrizDistanciasAzul[as.numeric(c(input$variableazul)),as.numeric(c(input$variableazul))]
```

Ilustración 19. Marco de Datos 2 Azul

La única diferencia en este nuevo código es la parte del cálculo de la solución, donde "results1<-xAz[[which.min(unlist(vAz))]]" y "labels(results1)[labels(results1)!="cut"]" es sustituido por una única frase de código: "v[[which.min(unlist(v))]]"

Con esto, como se ha descrito anteriormente, se obtiene como resultado la ruta mínima en lugar de la longitud mínima.

En la siguiente imagen se observa el caso de la ruta mínima para el contenedor verde.

```
148 #Cálculo de la solución
149
150 x<-vector("list", input$Nmuestras)
151 w<-vector("list",input$Nmuestras)
152 v<-vector("list",input$Nmuestras)
153 - for (i in 1:input$Nmuestras){
154
155
156     set.seed(i+input$Semilla)
157     x[[i]]<-solve_TSP((CRTsp),method1,control=list(start=1))
158
159
160     w[[i]]<-as.integer(x[[i]])
161
162
163
164     v[[i]]<-tour_length(CRTsp, w[[i]])
165 }
166 v[[which.min(unlist(v))]]
167 }
168 }
```

Ilustración 20. Solución Ruta Verde

En la siguiente imagen se observa el caso de la ruta mínima para el contenedor amarillo.

```
191 #Cálculo de la solución
192
193 xAm<-vector("list", input$Nmuestrasamarillo)
194 wAm<-vector("list", input$Nmuestrasamarillo)
195 vAm<-vector("list", input$Nmuestrasamarillo)
196 for (i in 1:input$Nmuestrasamarillo){
197
198
199     set.seed(i+input$Semillaamarillo)
200     xAm[[i]]<-solve_TSP((CRTsp),method1,control=list(start=1))
201
202
203     wAm[[i]]<-as.integer(xAm[[i]])
204
205
206
207     vAm[[i]]<-tour_length(CRTsp, wAm[[i]])
208 }
209 vAm[[which.min(unlist(vAm))]]
210 }
211 }
```

Ilustración 21. Solución Ruta Amarillo

Por último, en la siguiente imagen se observa el caso de la ruta mínima para el contenedor azul.

```
235 #Cálculo de la solución
236
237 xAz<-vector("list", input$Nmuestrasazul)
238 wAz<-vector("list", input$Nmuestrasazul)
239 vAz<-vector("list", input$Nmuestrasazul)
240 for (i in 1:input$Nmuestrasazul){
241
242
243     set.seed(i+input$Semillaazul)
244     xAz[[i]]<-solve_TSP((CRTsp),method1,control=list(start=1))
245
246
247     wAz[[i]]<-as.integer(xAz[[i]])
248
249
250
251     vAz[[i]]<-tour_length(CRTsp, wAz[[i]])
252 }
253 vAz[[which.min(unlist(vAz))]]
254 }
255 }
```

Ilustración 22. Solución Ruta Azul

Una vez terminada esta segunda parte del código, se empiezan a establecer los outputs, las salidas.

El primer output es establecido para obtener como salida una tabla, en específico, una tabla con nombre "Ruta de recogida" para el primer caso que es el del contenedor verde.

Para ello, añadimos el "values" perteneciente al contenedor verde y se usa "renderTable" con el "slidervalues" también perteneciente al contenedor verde para que se dé esta salida que se quiere obtener.

```
257 ▾ output$values<- renderTable({
258     Tabla<-data.frame(slidervalues())
259     names(Tabla)<-"Ruta de recogida"
260     Tabla
261 })
```

Ilustración 23. Salida de Tabla Verde

Repetimos el mismo proceso para la salida de la tabla correspondiente al contenedor amarillo.

Para este segundo caso introducimos el "valuesAmarillo" y dentro del "renderTable" el "slidervaluesamarillo" perteneciente al contenedor de este color.

También se le nombra como "Ruta de recogida" a esta segunda tabla.

```
263 ▾ output$valuesAmarillo<- renderTable({
264     Tabla<-data.frame(slidervaluesamarillo())
265     names(Tabla)<-"Ruta de recogida"
266     Tabla
267 })
```

Ilustración 24. Salida de Tabla Amarillo

Para terminar con las salidas de tablas, se realiza el mismo procedimiento para el último contenedor que es el azul.

La diferencia se encuentra al igual que en el caso anterior en el "valuesAzul" y el "slidervaluesazul" correspondientes a este último caso.

El nombre de esta última tabla tampoco varía y sigue siendo "Ruta de recogida".

```
269 ▾ output$valuesAzul<- renderTable({  
270     Tabla<-data.frame(slidervaluesazul())  
271     names(Tabla)<-"Ruta de recogida"  
272     Tabla  
273 })
```

Ilustración 25. Salida de Tabla Azul

El siguiente output, la siguiente salida a establecer es una salida de texto, particularmente para la distancia de las rutas óptimas que se calculan.

Para ello, se usa "renderText" y se establece como nombre a esta distancia "Longitud de la ruta (m)".

```
276 ▾ output$Distancia<- renderText({  
277     "Longitud de la ruta(m)"  
278 })
```

Ilustración 26. Salida de Texto

Por último, se desarrollan otros tres outputs para terminar con todo el código del archivo server.R.

En este caso, la función de estas salidas es "renderPrint" y lo que hace es imprimir la salida de texto, como número esta vez, de los objetos creados en el código de la ruta mínima que son los "slidervalues1".

```
282 ▾ output$values<- renderPrint({  
283     slidervalues1()  
284 })  
285 ▾ output$valuesAmarillo<- renderPrint({  
286     slidervalues1Amarillo()  
287 })  
288 ▾ output$valuesAzul<- renderPrint({  
289     slidervalues1Azul()  
290 })  
291 })
```

Ilustración 27. Imprimibles Salida de Texto de los 3 Objetos

Con esto está desarrollado el primer archivo de la aplicación Shiny y se desarrolla el segundo archivo, tratado a continuación.

4.3.5. Archivo *ui.R*

En este apartado se va a explicar cómo se ha construido y desarrollado el archivo *ui* de la aplicación.

Lo primero que se debe establecer es el "shinyUI" con la función "fluidPage", la cual sirve para crear una página fluida. Este diseño de página fluida consiste en filas que contienen a su vez columnas.

```
1  
2 library(rsconnect)  
3  
4 library(shiny)  
5  
6 shinyUI(fluidPage(  
7
```

Ilustración 28. Diseño Página Fluida

Al mismo tiempo, lo primero que establecemos en esta función, en esta página fluida es el título de la aplicación, mediante el término "titlePanel".

El título de esta es "Red de Recogida".

```
6 shinyUI(fluidPage(  
7  
8 # título de la aplicación  
9 titlePanel("Red de Recogida"),  
10
```

Ilustración 29. Título de la Aplicación

A continuación, se crea la función "sidebarPanel" en la que están incluidos todos los inputs del archivo *ui* y sirve para crear un panel de barra lateral en la aplicación a desarrollar.

```
9 titlePanel("Red de Recogida"),  
10  
11 # sidebarPanel con los inputs  
12  
13 sidebarPanel(  
14
```

Ilustración 30. Creación Panel de Barra Lateral

Una vez este término esté creado, se desarrolla dentro de este el "tabsetPanel". Este sirve para crear un conjunto de pestañas que corresponden a la optimización de cada tipo de contenedor.

```

13     sidebarPanel(
14
15
16     tabsetPanel(

```

Ilustración 31. Creación Conjunto de Pestañas

Estas pestañas las forman diferentes elementos, "tabPanel". En este código hay tres "tabPanel", uno por cada ruta a tratar.

Para desarrollar cada "tabPanel" se deben incluir cuatro términos en cada uno.

Estos términos son: "title", dos "numericInput" y "checkboxGroupInput".

```

16     tabsetPanel(
17         tabPanel(title="Contenedor Verde",
18                 numericInput("Nmuestras", "Basado en Muestras", min=100, max=1000, step=100, value=100),
19                 numericInput("Semillas", "Basado en Semillas", min=100, max=1000, step=100, value=100),
20                 checkboxGroupInput("Semillas", "Basado en Semillas", min=100, max=1000, step=100, value=100),
21

```

Ilustración 32. Términos Pestañas Verde

Dentro del primer término, se debe definir el título del elemento. En este caso será "Contenedor Verde".

```

16     tabsetPanel(
17         tabPanel(title="Contenedor Verde",

```

Ilustración 33. Título del Elemento Verde

El primer "numericInput" se usa para definir el número de muestras, es decir, el número de veces a repetir el cálculo para el cual obtendremos el óptimo; en este caso el mínimo de veces serán 100 y el máximo 1000 ("Nmuestras"); y el margen de cambio entre estas muestras; en este caso empieza por defecto en 100 veces y cambia de 100 veces en 100 veces hasta esas 1000 veces ("Basado en Muestras").

```

17         tabPanel(title="Contenedor Verde",
18                 numericInput("Nmuestras", "Basado en Muestras", min=100, max=1000, step=100, value=100),

```

Ilustración 34. Número de Muestras Verde

El segundo "numericInput" se usa para la misma función que el caso anterior, pero con la semilla de aleatoriedad.

La semilla será como mínimo 1 y como máximo 1000 ("Semilla") y se establecerá por defecto en 1, cambiando de 1 en 1 hasta el máximo fijado ("Semilla de aleatoriedad").

```
17 tabPanel(title="Contenedor Verde",
18         numericInput("Nmuestras", "Basado en Muestras", min=100, max=1000, step=100, value=100)
19         numericInput("Semilla", "Semilla de aleatoriedad", min=1, max=1000, step=1, value=1),
```

Ilustración 35. Número Semilla de Aleatoriedad Verde

El último término es "checkboxGroupInput" con el que se genera una entrada múltiple en la que eligen los puntos de recogida ("variable") que son 82 y cuales están seleccionados por defecto en la aplicación ("Seleccionar los puntos de recogida de contenedores verdes que se desean incluir en la ruta") que son todos ellos.

```
17 inel(title="Contenedor Verde",
18       numericInput("Nmuestras", "Basado en Muestras", min=100, max=1000, step=100, value=100),
19       numericInput("Semilla", "Semilla de aleatoriedad", min=1, max=1000, step=1, value=1),
20       checkboxGroupInput("variable", "Seleccionar los puntos de recogida de contenedores verdes que se desean incluir en la ruta",
```

Ilustración 36. Variable Puntos de Recogida Verde

Dentro de este último término, los puntos de recogida se establecen con la función "choices".

```

choices=c("1"="1", "2"="2", "3"="3",
          "4"="4",
          "5"="5",
          "6"="6", "7"="7",
          "8"="8", "9"="9", "10"="10",
          "11"="11", "12"="12",
          "13"="13", "14"="14",
          "15"="15", "16"="16",
          "17"="17",
          "18"="18", "19"="19", "20"="20",
          "21"="21", "22"="22", "23"="23",
          "24"="24", "25"="25",
          "26"="26", "27"="27",
          "28"="28", "29"="29",
          "30"="30", "31"="31",
          "32"="32", "33"="33", "34"="34", "35"="35", "36"="36", "37"="37",
          "38"="38", "39"="39", "40"="40", "41"="41", "42"="42", "43"="43",
          "44"="44", "45"="45", "46"="46", "47"="47", "48"="48", "49"="49",
          "50"="50", "51"="51", "52"="52", "53"="53", "54"="54", "55"="55",
          "56"="56", "57"="57", "58"="58", "59"="59", "60"="60", "61"="61",
          "62"="62", "63"="63", "64"="64", "65"="65", "66"="66", "67"="67",
          "68"="68", "69"="69", "70"="70", "71"="71", "72"="72", "73"="73",
          "74"="74", "75"="75", "76"="76", "77"="77", "78"="78", "79"="79",
          "80"="80", "81"="81", "82"="82"),

```

Ilustración 37. Puntos de Recogida Verde

Y los puntos de recogida seleccionados se establecen con la función "selected".

```

selected=c("1"="1", "2"="2", "3"="3",
          "4"="4",
          "5"="5",
          "6"="6", "7"="7",
          "8"="8", "9"="9", "10"="10",
          "11"="11", "12"="12",
          "13"="13", "14"="14",
          "15"="15", "16"="16",
          "17"="17",
          "18"="18", "19"="19", "20"="20",
          "21"="21", "22"="22", "23"="23",
          "24"="24", "25"="25",
          "26"="26", "27"="27",
          "28"="28", "29"="29",
          "30"="30", "31"="31",
          "32"="32", "33"="33", "34"="34", "35"="35", "36"="36", "37"="37",
          "38"="38", "39"="39", "40"="40", "41"="41", "42"="42", "43"="43",
          "44"="44", "45"="45", "46"="46", "47"="47", "48"="48", "49"="49",
          "50"="50", "51"="51", "52"="52", "53"="53", "54"="54", "55"="55",
          "56"="56", "57"="57", "58"="58", "59"="59", "60"="60", "61"="61",
          "62"="62", "63"="63", "64"="64", "65"="65", "66"="66", "67"="67",
          "68"="68", "69"="69", "70"="70", "71"="71", "72"="72", "73"="73",
          "74"="74", "75"="75", "76"="76", "77"="77", "78"="78", "79"="79",
          "80"="80", "81"="81", "82"="82"),
inline=TRUE)) ,

```

Ilustración 38. Selección Puntos de Recogida Verde

Con esto se termina el desarrollo del primero de los tres "tabPanel" y se comienza a desarrollar el segundo, el referente al contenedor amarillo.

Para ello, se realiza la misma estructura de código que para el de color verde, pero se deben cambiar los nombres.

Se pone como título "Contenedor Amarillo", las muestras serán "Nmuestrasamarillo", la semilla será "Semillaamarillo" y la variable será "variableamarilla".

Los datos de los "numericInput" se mantienen establecidos del mismo modo, pero los datos del "checkboxGroupInput" varían, debido a que el número de puntos de recogida no es el mismo en los tres tipos de residuos.

Para el contenedor amarillo son 61 puntos diferentes los cuales también estarán seleccionados todos al principio.

```
71 tabPanel(title="Contenedor Amarillo",
72   numericInput("Nmuestrasamarillo", "Basado en Muestras", min=100, max=1000, step=100, value=100),
73   numericInput("Semillaamarillo", "Semilla de aleatoriedad", min=1, max=1000, step=1, value=1),
74   checkboxGroupInput("variableamarilla", "Seleccionar los puntos de recogida de contenedores amarillos que se desean incluir en la ruta",
```

Ilustración 39. Términos Pestañas Amarillo

```
choices=c("1"="1", "2"="2", "3"="3",
          "4"="4",
          "5"="5",
          "6"="6", "7"="7",
          "8"="8", "9"="9", "10"="10",
          "11"="11", "12"="12",
          "13"="13", "14"="14",
          "15"="15", "16"="16",
          "17"="17",
          "18"="18", "19"="19", "20"="20",
          "21"="21", "22"="22", "23"="23",
          "24"="24", "25"="25",
          "26"="26", "27"="27",
          "28"="28", "29"="29",
          "30"="30", "31"="31",
          "32"="32", "33"="33", "34"="34", "35"="35", "36"="36", "37"="37",
          "38"="38", "39"="39", "40"="40", "41"="41", "42"="42", "43"="43",
          "44"="44", "45"="45", "46"="46", "47"="47", "48"="48", "49"="49",
          "50"="50", "51"="51", "52"="52", "53"="53", "54"="54", "55"="55",
          "56"="56", "57"="57", "58"="58", "59"="59", "60"="60", "61"="61"),
```

Ilustración 40. Puntos de Recogida Amarillo

```
selected=c("1"="1", "2"="2", "3"="3",
          "4"="4",
          "5"="5",
          "6"="6", "7"="7",
          "8"="8", "9"="9", "10"="10",
          "11"="11", "12"="12",
          "13"="13", "14"="14",
          "15"="15", "16"="16",
          "17"="17",
          "18"="18", "19"="19", "20"="20",
          "21"="21", "22"="22", "23"="23",
          "24"="24", "25"="25",
          "26"="26", "27"="27",
          "28"="28", "29"="29",
          "30"="30", "31"="31",
          "32"="32", "33"="33", "34"="34", "35"="35", "36"="36", "37"="37",
          "38"="38", "39"="39", "40"="40", "41"="41", "42"="42", "43"="43",
          "44"="44", "45"="45", "46"="46", "47"="47", "48"="48", "49"="49",
          "50"="50", "51"="51", "52"="52", "53"="53", "54"="54", "55"="55",
          "56"="56", "57"="57", "58"="58", "59"="59", "60"="60", "61"="61"),
inline=TRUE)) ,
```

Ilustración 41. Selección Puntos de Recogida Amarillo

Así termina el segundo "tabPanel" y se da paso al tercero y último, dedicado al contenedor de color azul.

Al igual que los dos últimos, la estructura del código se mantiene idéntica y solamente cambian los nombres definidos.

Se pone como título "Contenedor azul", las muestras serán "Nmuestrasazul", la semilla será "Semillaazul" y la variable será "variableazul".

Los datos de los "numericInput" se mantienen establecidos del mismo modo, pero los datos del "checkboxGroupInput" varían, debido a que el número de puntos de recogida no es el mismo en los tres tipos de residuos.

Para el contenedor azul son 56 puntos diferentes los cuales también estarán seleccionados todos al principio.

```
tabPanel(title="Contenedor azul",
         numericInput("Nmuestrasazul", "Basado en Muestras", min=100, max=1000, step=100, value=100),
         numericInput("Semillaazul", "Semilla de aleatoriedad", min=1, max=1000, step=1, value=1),
         checkboxGroupInput("variableazul", "Seleccionar los puntos de recogida de contenedores azules que se desean incluir en la ruta",
```

Ilustración 42. Términos Pestañas Azul

```

122     choices=c("1"="1", "2"="2", "3"="3",
123             "4"="4",
124             "5"="5",
125             "6"="6", "7"="7",
126             "8"="8", "9"="9", "10"="10",
127             "11"="11", "12"="12",
128             "13"="13", "14"="14",
129             "15"="15", "16"="16",
130             "17"="17",
131             "18"="18", "19"="19", "20"="20",
132             "21"="21", "22"="22", "23"="23",
133             "24"="24", "25"="25",
134             "26"="26", "27"="27",
135             "28"="28", "29"="29",
136             "30"="30", "31"="31",
137             "32"="32", "33"="33", "34"="34", "35"="35", "36"="36", "37"="37",
138             "38"="38", "39"="39", "40"="40", "41"="41", "42"="42", "43"="43",
139             "44"="44", "45"="45", "46"="46", "47"="47", "48"="48", "49"="49",
140             "50"="50", "51"="51", "52"="52", "53"="53", "54"="54", "55"="55",
141             "56"="56"),

```

Ilustración 43. Puntos de Recogida Azul

```

142     selected=c("1"="1", "2"="2", "3"="3",
143              "4"="4",
144              "5"="5",
145              "6"="6", "7"="7",
146              "8"="8", "9"="9", "10"="10",
147              "11"="11", "12"="12",
148              "13"="13", "14"="14",
149              "15"="15", "16"="16",
150              "17"="17",
151              "18"="18", "19"="19", "20"="20",
152              "21"="21", "22"="22", "23"="23",
153              "24"="24", "25"="25",
154              "26"="26", "27"="27",
155              "28"="28", "29"="29",
156              "30"="30", "31"="31",
157              "32"="32", "33"="33", "34"="34", "35"="35", "36"="36", "37"="37",
158              "38"="38", "39"="39", "40"="40", "41"="41", "42"="42", "43"="43",
159              "44"="44", "45"="45", "46"="46", "47"="47", "48"="48", "49"="49",
160              "50"="50", "51"="51", "52"="52", "53"="53", "54"="54", "55"="55",
161              "56"="56"),
162     inline=TRUE))) ,

```

Ilustración 44. Selección Puntos de Recogida Azul

Una vez desarrollado este último "tabPanel", se finaliza la parte del código del archivo ui perteneciente a la función "tabsetPanel".

A continuación, se define la segunda y última parte del código perteneciente a los inputs del archivo, perteneciente al "siderbarPanel".

Se crea una función para seleccionar que opciones va a tener la aplicación para realizar el cálculo de la solución mediante su algoritmo.

Las opciones se incluyen en el término "choices" y son la inserción más cercana ("nearest_insertion"), la inserción más rápida ("farthest_insertion"), la inserción más barata ("cheapest_insertion") y por último, una inserción arbitraria ("arbitrary_insertion").

La seleccionada para el cálculo es la inserción más cercana y se realiza mediante el término "selected".

```
164         selectInput("Method",label="Algoritmo",  
165                     choices=c("nearest_insertion", "farthest_insertion",  
166                               "cheapest_insertion", "arbitrary_insertion"  
167                               ),selected="nearest_insertion")  
168     ),
```

Ilustración 45. Opciones de Elección del Algoritmo

Con toda la parte de los inputs terminada, se desarrolla la última parte del código del archivo; la parte de las salidas, los outputs.

Para ello, se realiza mediante la función "mainPanel", la cual ofrece estos elementos de salida que se incluyen en el panel principal.

Dentro de este "mainPanel" se encuentra otro "tabsetPanel" con sus tres respectivos "tabPanel" por cada tipo de residuo.

Cada "tabpanel" tiene 3 términos en su estructura, que son:

El "textOutput" con el cual se crea una salida de texto y los nombres de estas salidas son "Distancia verde" para el contenedor de materia orgánica, "Distancia amarillo" para el contenedor de plásticos y "Distancia azul" para el contenedor de papel y cartón.

El "verbatimTextOutput" que es otra salida que se le dice a la página web en la que nos referimos al "valuess" establecido en el archivo server y que se refiere a la longitud. Estos son "valuess" para el contenedor verde, "valuessAmarillo" para el contenedor del propio color nombrado y "valuessAzul" para, al igual que el último, el contenedor de este color.

Por último, se desarrolla el "tableOutput" que también es una salida que se le dice a la web y, en este caso, referida a la tabla de la ruta y a su vez también establecida en el archivo server con el nombre de "values". Son "values", "valuesAmarillo" y "valuesAzul" referidos de la misma forma que los nombrados en el apartado anterior.

```

169     #mainpanel con los outputs
170     mainPanel(
171         tabsetPanel(
172             tabPanel("contenedor verde",
173                 textOutput("Distancia verde")
174             ),
175             verbatimTextOutput("valuess"),
176             tableOutput("values")
177         ),
178     ),
179 ),
180
181     tabPanel("contenedor amarillo",
182             textOutput("Distancia amarillo"),
183             verbatimTextOutput("valuessAmarillo"),
184             tableOutput("valuesAmarillo")
185         ),
186     tabPanel("contenedor azul",
187             textOutput("Distancia azul"),
188             verbatimTextOutput("valuessAzul"),
189             tableOutput("valuesAzul")
190         )
191     )

```

Ilustración 46. Elementos de Salida del Panel Principal

Con esta última parte del código, se termina el desarrollo del archivo ui.R y, a su vez, todo el código ejecutado para la obtención de la solución a resolver para este problema del viajante expuesto.

4.4. PÁGINA WEB

En este apartado se va a mostrar la solución del desarrollo del código explicado y detallado en el punto 4.3., la cual es una página web.

En este proyecto, como se ha mencionado a lo largo de él, la solución está proyectada en una página web interactiva en la que se pueden observar los diferentes resultados obtenidos.

Esta solución y estos resultados se ven en forma de la longitud mínima a recorrer por cada una de las tres diferentes rutas existentes por cada tipo de contenedor y residuo y, a su vez, la propia ruta a seguir entre los puntos de recogida que ofrece cada una de esas tres longitudes mínimas.

A continuación, se muestra la apariencia de la web creada con el código descrito y se desarrollan y explican las funcionalidades que esta tiene.

A rasgos generales, se ve el título de esta página tal y como se comenta párrafos antes, el cual es "Red de Recogida" y, a partir de ahí, se diferencian dos zonas.

La primera zona es la situada en el lado izquierdo de la página, la cual es una barra lateral debido a la función definida en el código, en la que se pueden elegir las principales opciones desarrolladas de la aplicación.

La segunda zona es la que muestra los principales resultados del problema a tratar.

La apariencia general de esta página es como se observa en la siguiente imagen:

Red de Recogida

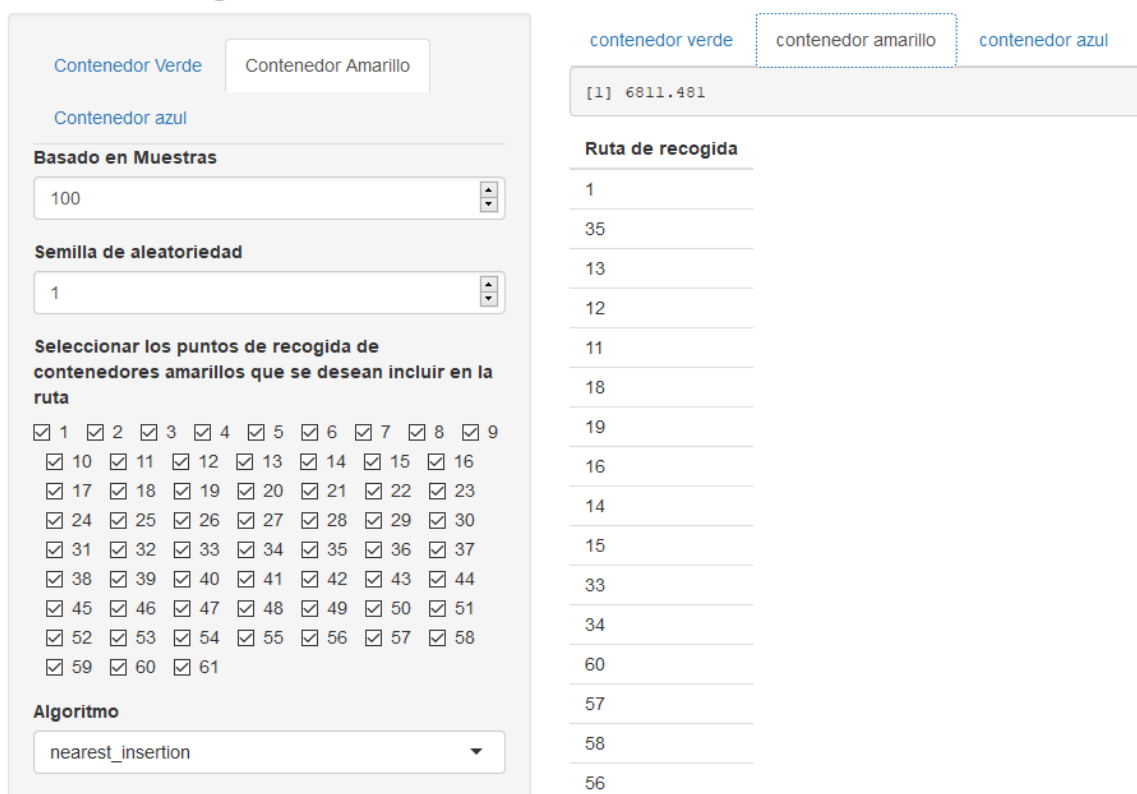


Ilustración 47. Apariencia General de la App

Una vez vista la apariencia general, se va a detallar cada una de las partes que componen esta página.

La primera parte a describir es de la zona de las principales opciones y es la elección del tipo de contenedor; verde, amarillo o azul.

Esta elección hace que se carguen los diferentes puntos de recogida que tiene cada tipo de contenedor; en el caso de elegir la pestaña "Contenedor Verde" se cargan los 82 puntos disponibles; en el caso de elegir la pestaña "Contenedor Amarillo" se cargan los 61 puntos

disponibles y en el caso de elegir la pestaña "Contenedor Azul" se cargan los 56 puntos disponibles.

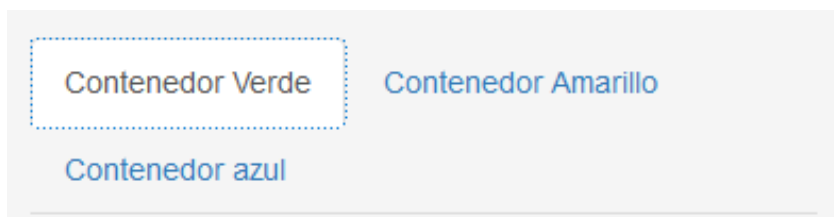


Ilustración 48. Elección Tipo de Contenedor

La siguiente parte de esta zona es la opción del número de veces a realizar el cálculo del problema que, como se menciona en el desarrollo del código, empieza por defecto en 100.

Además, como también se menciona, esta opción cambia de 100 en 100 hasta un máximo de 1000 veces.

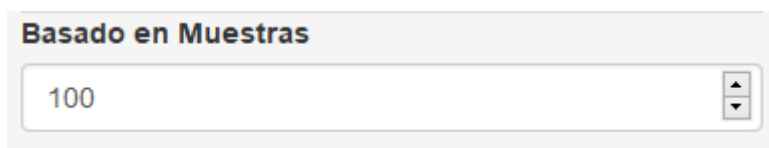


Ilustración 49. Elección del Número de Muestras Predeterminada

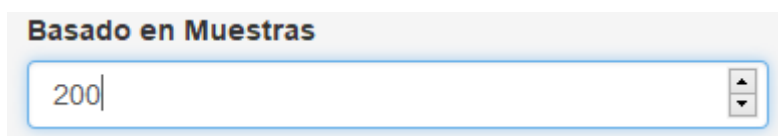


Ilustración 50. Ejemplo 1 Número de Muestras

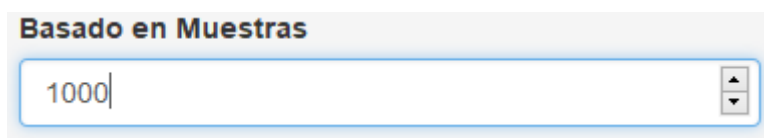
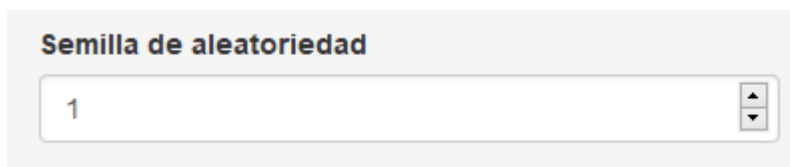


Ilustración 51. Ejemplo 2 Número de Muestras

La tercera parte de esta primera zona es la opción de la semilla de aleatoriedad que se usa para este cálculo que, como se menciona en el desarrollo del código, empieza por defecto en 1.

Además, como también se menciona, esta opción cambia de 1 en 1 hasta un máximo de 1000.



Semilla de aleatoriedad

1

Ilustración 52. Elección Semilla de Aleatoriedad Predeterminada



Semilla de aleatoriedad

2

Ilustración 53. Ejemplo 1 Semilla de Aleatoriedad

La cuarta y penúltima parte de esta zona es la opción de seleccionar los puntos de recogida de cada color de contenedor que se quieran incluir en la ruta.

En este caso, salen por defecto todos debido a que, en el caso real del problema a resolver, el camión de recogida pasa por todos los puntos diferentes de la localidad; pero siempre se pueden seleccionar los puntos que se deseen.

Además, según la primera opción de contenedor que escojas, aparecen los diferentes puntos disponibles. Se observa en las siguientes imágenes:

**Seleccionar los puntos de recogida de
contenedores verdes que se desean incluir en la
ruta**

- 1 2 3 4 5 6 7 8 9
 10 11 12 13 14 15 16
 17 18 19 20 21 22 23
 24 25 26 27 28 29 30
 31 32 33 34 35 36 37
 38 39 40 41 42 43 44
 45 46 47 48 49 50 51
 52 53 54 55 56 57 58
 59 60 61 62 63 64 65
 66 67 68 69 70 71 72
 73 74 75 76 77 78 79
 80 81 82

Ilustración 54. Puntos Seleccionados Verde

**Seleccionar los puntos de recogida de
contenedores amarillos que se desean incluir en la
ruta**

- 1 2 3 4 5 6 7 8 9
 10 11 12 13 14 15 16
 17 18 19 20 21 22 23
 24 25 26 27 28 29 30
 31 32 33 34 35 36 37
 38 39 40 41 42 43 44
 45 46 47 48 49 50 51
 52 53 54 55 56 57 58
 59 60 61

Ilustración 55. Puntos Seleccionados Amarillo

**Seleccionar los puntos de recogida de
contenedores azules que se desean incluir en la
ruta**

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

Ilustración 56. Puntos Seleccionados Azul

La última parte a describir de esta primera zona es la opción del algoritmo a utilizar.

Por defecto se elige el de la inserción más cercana ya que es la solución principal que se quiere obtener, pero existen las cuatro opciones desarrolladas en el código.

Dependiendo del algoritmo seleccionado, el resultado del problema varía.

Algoritmo

nearest_insertion

nearest_insertion

farthest_insertion

cheapest_insertion

arbitrary_insertion

Ilustración 57. Elección del Algoritmo

Una vez esta zona explicada, se pasa a describir la otra zona de la página; la zona de los resultados.

La primera parte que se observa es el tipo de contenedor a seleccionar, de modo que, dependiendo del color de contenedor elegido, te muestre la longitud mínima en metros para este recorrido elegido y su ruta de punto de recogida a punto de recogida.

Posteriormente, debajo de estas tres pestañas comentadas, se observa un recuadro donde aparece un número. Este número es la mencionada longitud mínima del recorrido de recogida del color de contenedor seleccionado.

La tercera y última parte es, como el nombre indica, la ruta a seguir para conseguir esa longitud mínima en metros. Esta ruta aparece definida en una tabla vertical de una sola columna en la que cada fila es un número el cual expresa un punto diferente de recogida, que a su vez está asociado a unas coordenadas exactas.

A continuación, se muestra una imagen de la apariencia de esta zona de la página:

contenedor verde
contenedor amarillo
contenedor azul

[1] 7956.57

Ruta de recogida

1

44

49

50

51

52

53

57

56

55

54

65

66

75

74

76

Ilustración 58. Zona de Resultados de la App

Cabe destacar que se observa un solo ejemplo, y que no es la solución final para el caso seleccionado; sino que dependiendo de las opciones marcadas en todas estas partes descritas el resultado será diferente.

Se muestran algunos ejemplos:

En este caso vemos como para un número de muestras 100, con semilla de aleatoriedad 1 y el algoritmo de inserción más cercana nos da esos dos resultados como longitud y ruta.

Red de Recogida

Contenedor Verde Contenedor Amarillo

Contenedor azul

Basado en Muestras

100

Semilla de aleatoriedad

1

Seleccionar los puntos de recogida de contenedores amarillos que se desean incluir en la ruta

1 2 3 4 5 6 7 8 9

10 11 12 13 14 15 16

17 18 19 20 21 22 23

24 25 26 27 28 29 30

31 32 33 34 35 36 37

38 39 40 41 42 43 44

45 46 47 48 49 50 51

52 53 54 55 56 57 58

59 60 61

Algoritmo

nearest_insertion

contenedor verde contenedor amarillo contenedor azul

[1] 6811.481

Ruta de recogida

1

35

13

12

11

18

19

16

14

15

33

34

60

57

58

56

Ilustración 59. Ejemplo 1 de Resultado

Sin embargo, únicamente cambiando una de las opciones, las soluciones son completamente distintas.

A continuación, se observa como únicamente cambiando el número de muestras en el primer caso y el algoritmo en el segundo, los resultados varían completamente.

Red de Recogida

Contenedor Verde Contenedor Amarillo

Contenedor azul

Basado en Muestras

300

Semilla de aleatoriedad

1

Seleccionar los puntos de recogida de contenedores amarillos que se desean incluir en la ruta

1 2 3 4 5 6 7 8 9

10 11 12 13 14 15 16

17 18 19 20 21 22 23

24 25 26 27 28 29 30

31 32 33 34 35 36 37

38 39 40 41 42 43 44

45 46 47 48 49 50 51

52 53 54 55 56 57 58

59 60 61

Algoritmo

nearest_insertion

contenedor verde contenedor amarillo contenedor azul

[1] 6795.905

Ruta de recogida

1

35

39

40

41

42

49

50

37

38

26

27

36

59

48

47

Ilustración 60. Ejemplo 2 de Resultado

Red de Recogida

Contenedor Verde Contenedor Amarillo

Contenedor azul

Basado en Muestras

100

Semilla de aleatoriedad

1

Seleccionar los puntos de recogida de contenedores amarillos que se desean incluir en la ruta

1 2 3 4 5 6 7 8 9

10 11 12 13 14 15 16

17 18 19 20 21 22 23

24 25 26 27 28 29 30

31 32 33 34 35 36 37

38 39 40 41 42 43 44

45 46 47 48 49 50 51

52 53 54 55 56 57 58

59 60 61

Algoritmo

farthest_insertion

contenedor verde contenedor amarillo contenedor azul

[1] 7080.751

Ruta de recogida

1

4

3

10

9

5

6

8

7

21

18

19

16

17

20

22

Ilustración 61. Ejemplo 3 de Resultado

5. RESULTADOS

En este apartado número 5 del proyecto se van a desarrollar e indicar los diferentes resultados obtenidos para el problema planteado a resolver.

En primer lugar, como se ha explicado a lo largo de esta memoria, los resultados están incorporados en una página interactiva de la cual dispondrán los diferentes conductores de recogida de residuos de Fuentes de Ebro para poder realizar la ruta mínima a la hora de realizar su trabajo.

Los resultados que se van a comentar a continuación son obtenidos para la actual situación de la red de recogida de la localidad; con sus 82 puntos para el contenedor verde, 61 puntos para el contenedor amarillo y 56 puntos para el contenedor azul con las respectivas coordenadas de cada punto existente.

Sin embargo; en caso de que los puntos de recogida cambien tanto de número como de posición, esta página siempre será actualizada para obtener la nueva ruta optimizada.

5.1. RESULTADOS CONTENEDOR VERDE

Los primeros resultados a detallar son los correspondientes a los contenedores de color verde, los contenedores de materia orgánica.

Para conseguir estos resultados, los cuales van a ser la ruta óptima con su respectiva longitud mínima, se deben elegir las opciones adecuadas de las pestañas comentadas en el apartado anterior 4.4.

Para comenzar, se seleccionan las pestañas de "Contenedor Verde" tanto de la barra lateral izquierda como de la zona de la derecha.

Una vez seleccionadas ambas pestañas, se establece como número de muestras el mayor valor posible, en esta ocasión 1000, ya que cuantas más veces se repita el cálculo, mayor probabilidad existe de que se encuentre una ruta más óptima; y así ocurre en este proyecto.

A continuación, como se ha objetado anteriormente, la semilla de aleatoriedad se mantiene en 1 para establecer siempre el mismo punto de inicio y en cuanto a los puntos seleccionados, estos son todos los existentes, los 81, debido a que, en la situación actual de la red de recogida, el camión pasa por todos ellos a la hora de realizar su ruta de recogida.

La última pestaña a marcar es la del algoritmo elegido, y tal y como se nombra anteriormente, se marca el algoritmo de "nearest_insertion",

ya que con este se obtiene la distancia más corta entre cada punto, y, a su vez, la longitud mínima para la ruta a optimizar.

De este modo, ya se tienen todas las opciones seleccionadas para obtener el mejor resultado posible.

El resultado de la longitud de la ruta óptima para el contenedor de color verde es de una longitud de 7796,428 metros, es decir, aproximadamente 7,8 kilómetros.

En cuanto a la ruta a seguir, esta es 1-44-49-50-51-52-53-57-56-55-54-65-66-67-73-75-77-74-76-79-80-81-72-82-71-70-69-68-58-60-59-62-63-61-78-64-45-46-47-48-30-31-32-33-35-34-25-27-26-29-28-23-8-9-7-6-10-20-13-18-21-24-22-36-37-19-16-17-38-39-40-41-42-43-15-14-12-11-4-3-5-2.

Con este resultado y comparando con los datos reales, se puede observar como la distancia recorrida es muchísimo menor, en particular, pasa de una distancia de 49,4 kilómetros a 7,8 kilómetros.

Se debe destacar que en esa distancia real recorrida está incluida la recogida del punto limpio, la cual en este proyecto no se ha incluido debido a que no se encuentra dentro de los límites de la localidad.

Aun así, esa distancia sería de 45,4 kilómetros, una distancia 5,8 veces mayor.

Además, figura el tiempo estimado de este trayecto real, que es en torno a 4 horas. Con esta nueva ruta optimizada, el tiempo estimado de recogida sería de 1,5 horas, casi 3 veces menos.

Para terminar, se ha realizado un cálculo aproximado del gasto económico en combustible que se realiza con cada una de las dos rutas. En el caso de la ruta real, se ha estimado un gasto de unos 13 litros que con el precio aproximado del combustible hoy día sería un coste de unos 25 euros; mientras que, con la ruta óptima calculada, el gasto sería de unos 2,5 litros que sería un coste de unos 4,5 euros.

Esto implica un ahorro de aproximadamente 20 euros por día de recogida.

5.2. RESULTADOS CONTENEDOR AMARILLO

Los siguientes resultados a detallar son los correspondientes a los contenedores de color amarillo, los contenedores de plásticos.

Al igual que el caso del contenedor verde, para conseguir estos resultados, los cuales van a ser la ruta óptima con su respectiva longitud mínima, se deben elegir las opciones adecuadas de las pestañas comentadas en el apartado anterior 4.4.

Para comenzar, se seleccionan las pestañas de "Contenedor Amarillo" tanto de la barra lateral izquierda como de la zona de la derecha.

Una vez seleccionadas ambas pestañas, se establece como número de muestras el mayor valor posible, en esta ocasión también será 1000, por el mismo motivo descrito antes.

A continuación, de la misma forma que para el primer contenedor, la semilla de aleatoriedad se mantiene en 1 para establecer siempre el mismo punto de inicio y en cuanto a los puntos seleccionados, estos son todos los existentes, los 61, debido a que, en esta situación actual de la red de recogida, el camión también pasa por todos ellos a la hora de realizar su ruta de recogida.

La última pestaña a marcar es la del algoritmo elegido, y por la misma razón que se explica en el primer caso, se marca el algoritmo de "nearest_insertion".

De este modo, ya se tienen todas las opciones seleccionadas para obtener el mejor resultado posible de esta segunda ruta.

El resultado de la longitud de la ruta óptima para el contenedor de color verde es de una longitud de 6795.905 metros, es decir, aproximadamente 6,8 kilómetros. Esto es 1 kilómetro menos que la ruta óptima del contenedor verde, ya que esta segunda ruta tiene menos puntos de recogida.

En cuanto a la ruta a seguir, esta es 1-35-39-40-41-42-49-50-37-38-26-27-36-59-48-47-53-52-46-45-43-44-51-56-58-57-60-55-61-54-32-31-30-28-29-20-17-14-15-33-34-13-12-11-18-19-16-22-23-24-25-21-7-8-6-5-9-10-3-24.

Se observa como el primer punto también es el punto número 1 debido a que así está establecido en el código. El resto de puntos a seguir no son en ningún caso similares.

Con este resultado y comparando con los datos reales, se puede observar como la distancia recorrida es muchísimo menor, en particular, pasa de una distancia de 40,8 kilómetros a 6,8 kilómetros.

Se debe destacar que en esa distancia real recorrida está incluida la recogida del punto limpio como en el primer caso, la cual en este proyecto no se ha incluido debido a que tampoco no se encuentra dentro de los límites de la localidad.

Aun así, esa distancia sería de 36,8 kilómetros, una distancia 5,4 veces mayor.

Además, figura el tiempo estimado de este trayecto real, que es en torno a 3 horas. Con esta nueva ruta optimizada, el tiempo estimado de recogida sería de 1 hora, es decir, 3 veces menos.

Para terminar, se ha realizado también un cálculo aproximado del gasto económico en combustible que se realiza con cada ruta. En el caso de la ruta real, se ha estimado un gasto de unos 11 litros que con el precio aproximado del combustible hoy día sería un coste de unos 21 euros; mientras que, con la ruta óptima calculada, el gasto sería de unos 2 litros que sería un coste de uno 4 euros.

Esto implica un ahorro de aproximadamente 17 euros por día de recogida.

5.3. RESULTADOS CONTENEDOR AZUL

Los siguientes y últimos resultados a detallar son los correspondientes a los contenedores de color azul, los contenedores de papel y cartón.

Al igual que los casos del contenedor verde y del contenedor amarillo, para conseguir estos resultados, los cuales van a ser la ruta óptima con su respectiva longitud mínima, se deben elegir las opciones adecuadas de las pestañas comentadas en el apartado anterior 4.4.

Para comenzar, se seleccionan las pestañas de "Contenedor Azul" tanto de la barra lateral izquierda como de la zona de la derecha.

Una vez seleccionadas ambas pestañas, se establece como número de muestras el mayor valor posible, en esta tercera ocasión también será 1000.

A continuación, de la misma forma que para los dos primeros contenedores, la semilla de aleatoriedad se mantiene en 1 para establecer siempre el mismo punto de inicio y en cuanto a los puntos seleccionados, estos son todos los existentes, los 56, debido a que, en esta situación actual de la red de recogida, el camión de recogida de papel y cartón también pasa por todos ellos para realizar esta recogida.

La última pestaña a marcar es la del algoritmo elegido, y del mismo modo que los otros dos casos, se marca el algoritmo de "nearest_insertion".

De este modo, ya se tienen todas las opciones seleccionadas para obtener el mejor resultado posible de esta segunda ruta.

El resultado de la longitud de la ruta óptima para el contenedor de color verde es de una longitud de 6232.915 metros, es decir, aproximadamente 6,2 kilómetros. Esta distancia es la mínima de las tres obtenidas ya que es la ruta con menos puntos a recoger.

En cuanto a la ruta a seguir, esta es 1-3-2-8-10-9-15-7-4-5-6-17-21-20-19-18-16-14-13-11-12-29-31-30-55-54-51-52-50-49-56-28-26-

27-25-24-23-22-34-33-32-53-43-45-44-42-48-47-46-40-41-39-38-37-36-35.

Se observa también como el primer punto sigue siendo el punto número 1 y que el resto de puntos a seguir no son en ningún caso similares a ninguno de los otros dos casos.

Con este resultado y comparando con los datos reales, se puede observar como la distancia recorrida también es muchísimo menor, en particular, pasa de una distancia de 22,1 kilómetros a 6,2 kilómetros.

Este tercer caso es en el que menos diferencia existe, ya que existen menos puntos y es más fácil realizar caminos más cortos.

Se debe destacar que, en esa distancia real recorrida, al igual que ambos contenedores previos, está incluida la recogida del punto limpio, la cual en este proyecto no se ha incluido debido a que no se encuentra dentro de los límites de la localidad.

Aun así, esa distancia sería de 18,1 kilómetros, una distancia casi 3 veces mayor.

Además, figura el tiempo estimado de este trayecto real, que es en torno a 2,5 horas. Con esta nueva ruta optimizada, el tiempo estimado de recogida sería también de 1 hora, casi 3 veces menos.

Para terminar, se ha realizado al igual que los dos primeros contenedores un cálculo aproximado del gasto económico en combustible que se realiza con estas rutas. En el caso de la ruta real, se ha estimado un gasto de unos 5,5 litros que con el precio aproximado del combustible hoy día sería un coste de unos 10,3 euros; mientras que, con la ruta óptima calculada, el gasto sería de unos 2 litros que sería un coste de unos 4 euros, de igual manera que para el amarillo.

Esto implica un ahorro de aproximadamente 6 euros por día de recogida.

6. CONCLUSIONES

La principal conclusión que se puede sacar de la realización de este proyecto es la importancia de la optimización, en este caso, de la optimización de rutas. Es un muy buen ejemplo ya que es una optimización aplicable a la vida real y con datos tangibles y que son fáciles de comparar a primera vista; todo el mundo sabe comparar y ve claramente la diferencia entre distancias, tiempos y, sobre todo, aunque en este estudio no es objeto principal a desarrollar, entre costes económicos.

Se debe destacar como una buena planificación con sus correspondientes herramientas de optimización puede ser algo vital para el desarrollo de todo tipo de gestiones y empresas, ya que todo objetivo de cualquiera que emprende una acción, es que se haga de manera eficaz y eficiente, es decir, que tenga un buen resultado gastando los menores recursos posibles.

Esto es algo que se debe globalizar y empezar a ser habitual en todos los ámbitos posibles, ya que mejorando uno mismo, mejora el resto que lo rodea.

Incrementar los costes en el estudio de la optimización de cada proyecto a realizar no es una pérdida económica es una inversión a corto plazo, ya que, con la correcta planificación y desarrollo de esta, se obtienen unos beneficios notables desde muy temprano.

En este proyecto se puede observar como una empresa tan importante en su sector no tiene en cuenta esto recién planteado, y es algo que debería cuanto menos estudiarse, sin entrar en la dificultad que conlleve aplicarlo o no.

Destacando ahora más en concreto la optimización de rutas, en lugar de la optimización en general, se puede observar cómo, además de reducir los costes y tiempos, se reduce el impacto ambiental debido a que, como todo el mundo sabe, el tráfico es uno de los principales factores de contaminación y reduciendo el tiempo que estos vehículos están circulando, se contribuye a disminuir la nombrada contaminación y, con ello, a mejorar el medio ambiente y el entorno, algo muy importante a tener en cuenta hoy en día para mantener vivo el lugar en el que habitamos.

En este tema, ya no es solamente importante mejorar para que mejore lo de alrededor, es necesario mejorar para no empeorar uno mismo ni el resto ya que con el incremento existente en la actualidad de contaminación y la progresión que esta lleva, lo único que se genera son aspectos perjudiciales.

Por ello, la idea de implantar una buena optimización de rutas a todo tipo de gestión de recogida/reparto/transporte es vital para los intereses de todos.

Otra conclusión que se puede sacar de este proyecto es la importancia que se le da a los conocimientos adquiridos sobre algún tema específico cuando estos son aplicables a la vida real, al mundo tangible.

Cuando se aprende acerca de la optimización, solamente se ven números y resultados ficticios, algo a lo que no se le otorga valor real; sin embargo, con un estudio de un caso real en el que se deben aplicar todas las nociones que se han aprendido acerca del tema y se observa al final de él los resultados obtenidos y el buen trabajo realizado, es cuando se valora lo importante que es realizar el "trabajo que no se ve".

En el tema específico de este problema descrito, se valora de verdad el conocimiento adquirido en optimización de rutas y en sus sistemas de cálculo trabajados de manera ficticia anteriormente, cuando observas que obtienes un resultado de una distancia mucho menor a la que se realiza en la vida real, y que la localidad en la que se reside puede aplicar este estudio y estos cálculos para mejorar en su red de recogida de residuos, algo tan importante para todo lugar habitable.

Por último, la última conclusión a nombrar y el último aspecto a destacar es la importancia de establecer unos objetivos específicos y medibles a lograr y conseguir.

En el caso de este proyecto, se fijó el objetivo de conseguir tres rutas óptimas para la recogida de residuos de Fuentes de Ebro, de manera que todo el trabajo realizado va enfocado en conseguir ese objetivo de la mejor forma posible.

Sin un objetivo específico y medible a conseguir, lo más seguro a ocurrir es la desviación en el tema y no conseguir unos resultados claros y concisos para el problema.

De esta manera se pretendía obtener una distancia física y una sucesión de puntos para cada una de las tres rutas existentes, y el resultado ha sido una distancia y una sucesión de puntos para cada una de ellas; exactamente el objetivo a conseguir antes de empezar dicho proyecto.

7. OBJETIVOS DE DESARROLLO SOSTENIBLE

Los objetivos de este Trabajo Fin de Grado están alineados con los siguientes Objetivos de Desarrollo Sostenible (ODS) y metas, de la Agenda 2030:

- **Objetivo 4** - Garantizar una educación inclusiva y equitativa de calidad y promover oportunidades de aprendizaje permanente para todos



Ilustración 62. ODS 4

- **Meta 4.4.** De aquí a 2030, aumentar considerablemente el número de jóvenes y adultos que tienen las competencias necesarias, en particular técnicas y profesionales, para acceder al empleo, el trabajo decente y el emprendimiento.

- **Objetivo 8** - Promover el crecimiento económico sostenido, inclusivo y sostenible, el empleo pleno y productivo y el trabajo decente para todos.



Ilustración 63. ODS 8

- **Meta 8.4.** Mejorar progresivamente, de aquí a 2030, la producción y el consumo eficientes de los recursos mundiales y procurar desvincular el crecimiento económico de la degradación del medio ambiente, conforme al Marco Decenal de Programas sobre modalidades de Consumo y Producción Sostenibles, empezando por los países desarrollados.

- **Objetivo 11** - Lograr que las ciudades sean más inclusivas, seguras, resilientes y sostenibles.



Ilustración 64. ODS 11

- **Meta 11.6.** De aquí a 2030, reducir el impacto ambiental negativo per cápita de las ciudades, incluso prestando especial atención a la calidad del aire y la gestión de los desechos municipales y de otro tipo.

- **Objetivo 13.** Adoptar medidas urgentes para combatir el cambio climático y sus efectos.



Ilustración 65. ODS 13

- **Meta 13.2.** Incorporar medidas relativas al cambio climático en las políticas, estrategias y planes nacionales.

8. BIBLIOGRAFÍA

UNED Biblioteca. (2003). *Gestión de los residuos sólidos urbanos*. Recuperado 18 de abril de 2022, de <https://www2.uned.es/biblioteca/rsu/pagina3.htm>

Gobierno de España. (2020). *Sistema de Recogida*. <https://www.miteco.gob.es/es/calidad-y-evaluacion-ambiental/temas/prevencion-y-gestion-residuos/flujos/domesticos/gestion/sistema-recogida/Neumatica.aspx>

Universidad de Valladolid. (2015). *Resolución del Problema del Viajante de Comercio (TSP) y su variante con Ventanas de Tiempo (TSPTW) usando métodos heurísticos de búsqueda local*. <https://uvadoc.uva.es/bitstream/handle/10324/13378/TFG-I-252.pdf;jsessionid=9E2A4295C700514D659B9D066130EE9D?sequence=1>

Cádiz, U. de. (2015). *TSP: Formulación*. Recuperado 20 de abril de 2022, de <https://knuth.uca.es/moodle/mod/page/view.php?id=3416>

Cajal, A. (2019, diciembre 3). *Distancia euclidiana: Concepto, fórmula, cálculo, ejemplo*. Lifeder. <https://www.lifeder.com/distancia-euclidiana/>

Problema del viajante. (2022). En *Wikipedia, la enciclopedia libre*. https://es.wikipedia.org/w/index.php?title=Problema_del_viajante&oldid=142560496

Cádiz, U. de. (2015). *TSP: Algoritmos de resolución*. <https://knuth.uca.es/moodle/mod/page/view.php?id=3417>

Vision, S. (2017). *Aplicaciones Web en R con Shiny*. Synergy Vision. Recuperado 19 de septiembre de 2022, de <https://synergy.vision/corpus/shiny/2017-08-15-shiny.html>

Brillante: Uso de controles deslizantes. (2017). https://shiny-rstudio-com.translate.google.com/articles/sliders.html? x tr sl=en& x tr tl=es& x tr hl=es& x tr _pto=sc

Relación de documentos

(X) Memoria 61 páginas

(_) Anexos 31 páginas

La Almunia, a 19 de septiembre de 2022



Firmado: Darío Fraile Tolón



**Escuela Universitaria
Politécnica - La Almunia**
Centro adscrito
Universidad Zaragoza

**ESCUELA UNIVERSITARIA POLITÉCNICA
DE LA ALMUNIA DE DOÑA GODINA (ZARAGOZA)**

ANEXOS

Optimización de rutas de recogida de residuos
en Fuentes de Ebro

Waste Collection Route Optimization in Fuentes
de Ebro

30134

Autor: Darío Fraile Tolón

Director: Luis Mariano Esteban Escaño

Fecha: Septiembre 2022

Página intencionadamente en blanco.



INDICE DE CONTENIDO

1. ANEXO 1. RELACIÓN PUNTOS DE RECOGIDA - COORDENADAS	1
1.1. COORDENADAS CONTENEDOR VERDE	1
1.2. COORDENADAS CONTENEDOR AMARILLO	3
1.3. COORDENADAS CONTENEDOR AZUL	5
2. CÓDIGO DE R	7
2.1. ARCHIVO SERVER.R	7
2.2. ARCHIVO UI.R	14
3. ASPECTOS DE INTERÉS MEMORIA SEULA	18

1. ANEXO 1. RELACIÓN PUNTOS DE RECOGIDA - COORDENADAS

1.1. COORDENADAS CONTENEDOR VERDE

	CONTENEDOR VERDE
PUNTOS	COORDENADAS VERDE
1	41.507785, -0.622368
2	41.508766, -0.623055
3	41.509184, -0.624262
4	41.509700, -0.624223
5	41.509353, -0.623315
6	41.511401, -0.623067
7	41.512081, -0.623297
8	41.512219, -0.624241
9	41.512895, -0.622705
10	41.510765, -0.624936
11	41.509967, -0.625206
12	41.509794, -0.626192
13	41.510126, -0.626682
14	41.508674, -0.626520
15	41.508174, -0.627549
16	41.510037, -0.627976
17	41.510191, -0.628770
18	41.510958, -0.627396
19	41.511137, -0.628284
20	41.510649, -0.626273
21	41.511123, -0.626696
22	41.511846, -0.628340
23	41.512645, -0.625920
24	41.511903, -0.627372
25	41.512616, -0.627932
26	41.513585, -0.626972
27	41.513622, -0.627422
28	41.514005, -0.626378
29	41.514508, -0.626709
30	41.514566, -0.628615
31	41.513663, -0.629425
32	41.513423, -0.628971



Anexo 1. Relación Puntos de Recogida - Coordenadas

33	41.513233, -0.629412
34	41.512808, -0.628932
35	41.512874, -0.629817
36	41.511782, -0.629229
37	41.511142, -0.630034
38	41.509785, -0.629555
39	41.508886, -0.630114
40	41.508161, -0.629963
41	41.507528, -0.630024
42	41.506446, -0.629878
43	41.505658, -0.629332
44	41.507202, -0.623896
45	41.513771, -0.630276
46	41.514355, -0.630655
47	41.515086, -0.630706
48	41.515125, -0.629579
49	41.513835, -0.638974
50	41.514441, -0.638110
51	41.514633, -0.636357
52	41.513758, -0.635597
53	41.512252, -0.636625
54	41.510438, -0.635692
55	41.510903, -0.635756
56	41.511679, -0.635051
57	41.512423, -0.634576
58	41.512738, -0.633444
59	41.513126, -0.633958
60	41.512921, -0.633248
61	41.513565, -0.632543
62	41.514192, -0.633729
63	41.515096, -0.632926
64	41.513993, -0.631819
65	41.510397, -0.635191
66	41.510719, -0.634448
67	41.511395, -0.633655
68	41.512056, -0.633472
69	41.512151, -0.632832
70	41.511808, -0.632036
71	41.511452, -0.632210
72	41.511018, -0.631889
73	41.510764, -0.633458
74	41.509661, -0.633377
75	41.509854, -0.634170

Anexo 1. Relación Puntos de Recogida - Coordenadas

76	41.509158, -0.632768
77	41.508804, -0.634176
78	41.513611, -0.631778
79	41.509692, -0.631303
80	41.510347, -0.630858
81	41.510520, -0.631464
82	41.511227, -0.631596

1.2. COORDENADAS CONTENEDOR AMARILLO

CONTENEDOR AMARILLO	
PUNTOS	COORDENADAS AMARILLO
1	41.507785, -0.622368
2	41.509184, -0.624262
3	41.509700, -0.624223
4	41.509353, -0.623315
5	41.511401, -0.623067
6	41.512081, -0.623297
7	41.512219, -0.624241
8	41.512895, -0.622705
9	41.510765, -0.624936
10	41.509967, -0.625206
11	41.509794, -0.626192
12	41.508674, -0.626520
13	41.508174, -0.627549
14	41.510037, -0.627976
15	41.510191, -0.628770
16	41.510958, -0.627396
17	41.511137, -0.628284
18	41.510649, -0.626273
19	41.511123, -0.626696
20	41.511846, -0.628340
21	41.512645, -0.625920
22	41.511903, -0.627372
23	41.512616, -0.627932
24	41.513585, -0.626972
25	41.514005, -0.626378
26	41.514566, -0.628615
27	41.513663, -0.629425
28	41.513233, -0.629412

Anexo 1. Relación Puntos de Recogida - Coordenadas

29	41.512808, -0.628932
30	41.512874, -0.629817
31	41.512228, -0.630516
32	41.511142, -0.630034
33	41.509785, -0.629555
34	41.508886, -0.630114
35	41.507202, -0.623896
36	41.513771, -0.630276
37	41.515086, -0.630706
38	41.515125, -0.629579
39	41.513835, -0.638974
40	41.514441, -0.638110
41	41.514633, -0.636357
42	41.513758, -0.635597
43	41.512252, -0.636625
44	41.510799, -0.636115
45	41.511679, -0.635051
46	41.512423, -0.634576
47	41.513126, -0.633958
48	41.513565, -0.632543
49	41.514192, -0.633729
50	41.515096, -0.632926
51	41.510719, -0.634448
52	41.511395, -0.633655
53	41.512056, -0.633472
54	41.511808, -0.632036
55	41.511018, -0.631889
56	41.509661, -0.633377
57	41.509158, -0.632768
58	41.508804, -0.634176
59	41.513611, -0.631778
60	41.509456, -0.631887
61	41.511227, -0.631596

1.3. COORDENADAS CONTENEDOR AZUL

CONTENEDOR AZUL	
PUNTOS	COORDENADAS AZUL
1	41.507785, -0.622368
2	41.509700, -0.624223
3	41.509353, -0.623315
4	41.511401, -0.623067
5	41.512081, -0.623297
6	41.512219, -0.624241
7	41.510765, -0.624936
8	41.509967, -0.625206
9	41.509794, -0.626192
10	41.508674, -0.626520
11	41.510037, -0.627976
12	41.510191, -0.628770
13	41.510958, -0.627396
14	41.511137, -0.628284
15	41.510649, -0.626273
16	41.511846, -0.628340
17	41.512645, -0.625920
18	41.511903, -0.627372
19	41.512616, -0.627932
20	41.513585, -0.626972
21	41.514005, -0.626378
22	41.514566, -0.628615
23	41.513663, -0.629425
24	41.513233, -0.629412
25	41.512874, -0.629817
26	41.511941, -0.630859
27	41.512228, -0.630516
28	41.511142, -0.630034
29	41.509785, -0.629555
30	41.508886, -0.630114
31	41.508000, -0.629807
32	41.513771, -0.630276
33	41.515086, -0.630706
34	41.515125, -0.629579
35	41.513835, -0.638974
36	41.514441, -0.638110
37	41.514633, -0.636357
38	41.513758, -0.635597



Anexo 1. Relación Puntos de Recogida - Coordenadas

39	41.512252, -0.636625
40	41.511679, -0.635051
41	41.512423, -0.634576
42	41.513126, -0.633958
43	41.513565, -0.632543
44	41.514192, -0.633729
45	41.515096, -0.632926
46	41.510719, -0.634448
47	41.511395, -0.633655
48	41.512056, -0.633472
49	41.511808, -0.632036
50	41.509661, -0.633377
51	41.509158, -0.632768
52	41.508804, -0.634176
53	41.513611, -0.631778
54	41.509456, -0.631887
55	41.509692, -0.631303
56	41.511227, -0.631596

2. CÓDIGO DE R

2.1. ARCHIVO SERVER.R

```
library(shiny)

library(TSP)

shinyServer(function(input, output) {

  slidervalues<-reactive({

    Matriz<-
    MatrizDistanciasVerde[as.numeric(c(input$variable)),as.numeric(c(input
    $variable))]

    if (length(input$variable)<2) stop("")
    else {
      PuntosRepartoTSP<-TSP(Matriz, labels = NULL)

      CRtsp <- insert_dummy(PuntosRepartoTSP, label = "cut")
      method1<-input$Method

      solve_TSP((CRtsp),method1,control=list(start=initialtour))
      x<-vector("list", input$Nmuestras)
      w<-vector("list",input$Nmuestras)
      v<-vector("list",input$Nmuestras)
      for (i in 1:input$Nmuestras){

        set.seed(i+input$Semilla)
        x[[i]]<-solve_TSP((CRtsp),method1,control=list(start=1))

        w[[i]]<-as.integer(x[[i]])

      }

    }

  })

})
```



```
v[[i]]<-tour_length(CRtsp, w[[i]])
}
results1<-x[[which.min(unlist(v))]]
labels(results1)[labels(results1)!="cut"]
}
})
slidervaluesamarillo<-reactive({
  Matriz<-
MatrizDistanciasAmarillo[as.numeric(c(input$variableamarilla)),as.nume
ric(c(input$variableamarilla))]
  if (length(input$variableamarilla)<2) stop("")
  else {
    PuntosRepartoTSP<-TSP(Matriz, labels = NULL)

    CRtsp <- insert_dummy(PuntosRepartoTSP, label = "cut")
    method1<-input$Method

solve_TSP((CRtsp),method1,control=list(start=initialtour))
  xAm<-vector("list", input$Nmuestrasamarillo)
  wAm<-vector("list",input$Nmuestrasamarillo)
  vAm<-vector("list",input$Nmuestrasamarillo)
  for (i in 1:input$Nmuestrasamarillo){

    set.seed(i+input$Semillaamarillo)
    xAm[[i]]<-solve_TSP((CRtsp),method1,control=list(start=1))

    wAm[[i]]<-as.integer(xAm[[i]])

    vAm[[i]]<-tour_length(CRtsp, wAm[[i]])
  }

  results1Amarillo<-xAm[[which.min(unlist(vAm))]]
```

```

labels(results1Amarillo)[labels(results1Amarillo)!="cut"]

}
})
slidervaluesazul<-reactive({

  Matriz<-
  MatrizDistanciasAzul[as.numeric(c(input$variableazul)),as.numeric(c(in
put$variableazul))]
  if (length(input$variableazul)<2) stop("")
  else {
    CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)
    CRtsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")
    method1<-input$Method

  solve_TSP((CRtsp),method1,control=list(start=initialtour))
  xAz<-vector("list", input$Nmuestrasazul)
  wAz<-vector("list",input$Nmuestrasazul)
  vAz<-vector("list",input$Nmuestrasazul)
  for (i in 1:input$Nmuestrasazul){

    set.seed(i+input$Semillaazul)
    xAz[[i]]<-solve_TSP((CRtsp),method1,control=list(start=1))

    wAz[[i]]<-as.integer(xAz[[i]])

    vAz[[i]]<-tour_length(CRtsp, wAz[[i]])
  }

  results1<-xAz[[which.min(unlist(vAz))]]
  #función labels ofrece los nombres
  labels(results1)[labels(results1)!="cut"]
}

```

```
})
```

```
slidervalues1<-reactive({
```

```
  Matriz<-  
  MatrizDistanciasVerde[as.numeric(c(input$variable)),as.numeric(c(input  
  $variable))]
```

```
  if(length(input$variable)<2)
```

```
    stop("Elegir minimo 2 ciudades para obtener la ruta")
```

```
  else {
```

```
    CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)
```

```
    CRtsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")
```

```
    method1<-input$Method
```

```
    x<-vector("list", input$Nmuestras)
```

```
    w<-vector("list",input$Nmuestras)
```

```
    v<-vector("list",input$Nmuestras)
```

```
    for (i in 1:input$Nmuestras){
```

```
      set.seed(i+input$Semilla)
```

```
      x[[i]]<-solve_TSP((CRtsp),method1,control=list(start=1))
```

```
      w[[i]]<-as.integer(x[[i]])
```

```
      v[[i]]<-tour_length(CRtsp, w[[i]])
```

```
    }
```

```
    v[[which.min(unlist(v))]]
```

```
  }
```

```
})
```

```
slidervalues1Amarillo<-reactive({
```

```
Matriz<-  
MatrizDistanciasAmarillo[as.numeric(c(input$variableamarilla)),as.nume  
ric(c(input$variableamarilla))]  
  
if(length(input$variableamarilla)<2)  
  stop("Elegir minimo 2 ciudades para obtener la ruta")  
else {  
  
  CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)  
  
  CRtsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")  
  
  method1<-input$Method  
  
  xAm<-vector("list", input$Nmuestrasamarillo)  
  wAm<-vector("list",input$Nmuestrasamarillo)  
  vAm<-vector("list",input$Nmuestrasamarillo)  
  for (i in 1:input$Nmuestrasamarillo){  
  
    set.seed(i+input$Semillaamarillo)  
    xAm[[i]]<-solve_TSP((CRtsp),method1,control=list(start=1))  
  
    wAm[[i]]<-as.integer(xAm[[i]])  
  
    vAm[[i]]<-tour_length(CRtsp, wAm[[i]])  
  }  
  vAm[[which.min(unlist(vAm))]]  
}  
})  
  
slidervalues1Azul<-reactive({
```

```
Matriz<-  
MatrizDistanciasAzul[as.numeric(c(input$variableazul)),as.numeric(c(in  
put$variableazul))]  
  
if(length(input$variableazul)<2)  
  stop("Elegir minimo 2 ciudades para obtener la ruta")  
else {  
  
  CiudadesRepartoTSP<-TSP(Matriz, labels = NULL)  
  
  CRtsp <- insert_dummy(CiudadesRepartoTSP, label = "cut")  
  
  method1<-input$Method  
  
  xAz<-vector("list", input$Nmuestrasazul)  
  wAz<-vector("list",input$Nmuestrasazul)  
  vAz<-vector("list",input$Nmuestrasazul)  
  for (i in 1:input$Nmuestrasazul){  
  
    set.seed(i+input$Semillaazul)  
    xAz[[i]]<-solve_TSP((CRtsp),method1,control=list(start=1))  
  
    wAz[[i]]<-as.integer(xAz[[i]])  
  
    vAz[[i]]<-tour_length(CRtsp, wAz[[i]])  
  }  
  vAz[[which.min(unlist(vAz))]]  
}  
})  
  
output$values<- renderTable({
```

```
Tabla<-data.frame(slidervalues())
names(Tabla)<-"Ruta de recogida"
Tabla
})

output$valuesAmarillo<- renderTable({
  Tabla<-data.frame(slidervaluesamarillo())
  names(Tabla)<-"Ruta de recogida"
  Tabla
})

output$valuesAzul<- renderTable({
  Tabla<-data.frame(slidervaluesazul())
  names(Tabla)<-"Ruta de recogida"
  Tabla
})

output$Distancia<- renderText({
  "Longitud de la ruta(m)"
})
output$valuess<- renderPrint({
  slidervalues1()
})
output$valuessAmarillo<- renderPrint({
  slidervalues1Amarillo()
})
output$valuessAzul<- renderPrint({
  slidervalues1Azul()
})
})
```

2.2. ARCHIVO UI.R

```
library(rsconnect)

library(shiny)

shinyUI(fluidPage(

  titlePanel("Red de Recogida"),

  sidebarPanel(

    tabsetPanel(
      tabPanel(title="Contenedor Verde",
        numericInput("Nmuestras","Basado en
Muestras",min=100,max=1000,step=100,value=100),
        numericInput("Semilla","Semilla de
aleatoriedad",min=1,max=1000,step=1,value=1),
        checkboxGroupInput("variable", "Seleccionar
los puntos de recogida de contenedores verdes que se desean incluir en
la ruta",

choices=c("1"="1","2"="2","3"="3","4"="4","5"="5","6"="6","7"="7","8
"="8","9"="9","10"="10","11"="11","12"="12","13"="13","14"="14","1
5"="15","16"="16","17"="17","18"="18","19"="19","20"="20","21"="2
1","22"="22","23"="23","24"="24","25"="25","26"="26","27"="27","28
"="28","29"="29","30"="30","31"="31","32"="32","33"="33","34"="34
","35"="35","36"="36","37"="37","38"="38","39"="39","40"="40","41"
="41","42"="42","43"="43","44"="44","45"="45","46"="46","47"="47"
,"48"="48","49"="49","50"="50","51"="51","52"="52","53"="53","54"
="54","55"="55","56"="56","57"="57","58"="58","59"="59","60"="60"
,"61"="61","62"="62","63"="63","64"="64","65"="65","66"="66","67"
="67","68"="68","69"="69","70"="70","71"="71","72"="72","73"="73"
,"74"="74","75"="75","76"="76","77"="77","78"="78","79"="79","80"
="80","81"="81","82"="82"),
```

```
selected=c("1"="1","2"="2","3"="3","4"="4","5"="5","6"="6","7"="7",  
"8"="8","9"="9","10"="10","11"="11","12"="12","13"="13","14"="14",  
"15"="15","16"="16","17"="17","18"="18","19"="19","20"="20","21"="21",  
"22"="22","23"="23","24"="24","25"="25","26"="26","27"="27","28"="28",  
"29"="29","30"="30","31"="31","32"="32","33"="33","34"="34",  
"35"="35","36"="36","37"="37","38"="38","39"="39","40"="40",  
"41"="41","42"="42","43"="43","44"="44","45"="45","46"="46","47"="47",  
"48"="48","49"="49","50"="50","51"="51","52"="52","53"="53",  
"54"="54","55"="55","56"="56","57"="57","58"="58","59"="59","60"="60",  
"61"="61","62"="62","63"="63","64"="64","65"="65","66"="66",  
"67"="67","68"="68","69"="69","70"="70","71"="71","72"="72","73"="73",  
"74"="74","75"="75","76"="76","77"="77","78"="78","79"="79",  
"80"="80","81"="81","82"="82"),inline=TRUE)) ,
```

```
tabPanel(title="Contenedor Amarillo",
```

```
  numericInput("Nmuestrasamarillo","Basado en Muestras",min=100,max=1000,step=100,value=100),
```

```
  numericInput("Semillaamarillo","Semilla de aleatoriedad",min=1,max=1000,step=1,value=1),
```

```
  checkboxGroupInput("variableamarilla",  
"Seleccionar los puntos de recogida de contenedores amarillos que se desean incluir en la ruta",
```

```
choices=c("1"="1","2"="2","3"="3","4"="4","5"="5","6"="6","7"="7","8"="8",  
"9"="9","10"="10","11"="11","12"="12","13"="13","14"="14","15"="15",  
"16"="16","17"="17","18"="18","19"="19","20"="20","21"="21","22"="22",  
"23"="23","24"="24","25"="25","26"="26","27"="27","28"="28",  
"29"="29","30"="30","31"="31","32"="32","33"="33","34"="34",  
"35"="35","36"="36","37"="37","38"="38","39"="39","40"="40","41"="41",  
"42"="42","43"="43","44"="44","45"="45","46"="46","47"="47",  
"48"="48","49"="49","50"="50","51"="51","52"="52","53"="53","54"="54",  
"55"="55","56"="56","57"="57","58"="58","59"="59","60"="60",  
"61"="61"),
```

```
selected=c("1"="1","2"="2","3"="3","4"="4","5"="5","6"="6","7"="7",  
"8"="8","9"="9","10"="10","11"="11","12"="12","13"="13","14"="14",  
"15"="15","16"="16","17"="17","18"="18","19"="19","20"="20","21"="21",  
"22"="22","23"="23","24"="24","25"="25","26"="26","27"="27","28"="28",  
"29"="29","30"="30","31"="31","32"="32","33"="33","34"="34",  
"35"="35","36"="36","37"="37","38"="38","39"="39","40"="40",  
"41"="41","42"="42","43"="43","44"="44","45"="45","46"="46",  
"47"="47",  
"48"="48","49"="49","50"="50","51"="51","52"="52","53"="53",  
"54"="54"),
```



```
= "54", "55" = "55", "56" = "56", "57" = "57", "58" = "58", "59" = "59", "60" = "60",  
"61" = "61"), inline = TRUE)) ,
```

```
tabPanel(title = "Contenedor azul",  
numericInput("Nmuestrasazul", "Basado en  
Muestras", min = 100, max = 1000, step = 100, value = 100),  
numericInput("Semillaazul", "Semilla de  
aleatoriedad", min = 1, max = 1000, step = 1, value = 1),  
checkboxGroupInput("variableazul",  
"Seleccionar los puntos de recogida de contenedores azules que se  
desean incluir en la ruta",
```

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

```
selected = c("1" = "1", "2" = "2", "3" = "3", "4" = "4", "5" = "5", "6" = "6", "7" = "7", "8"  
= "8", "9" = "9", "10" = "10", "11" = "11", "12" = "12", "13" = "13", "14" = "14", "15"  
= "15", "16" = "16", "17" = "17", "18" = "18", "19" = "19", "20" = "20", "21" = "21",  
"22" = "22", "23" = "23", "24" = "24", "25" = "25", "26" = "26", "27" = "27", "28"  
= "28", "29" = "29", "30" = "30", "31" = "31", "32" = "32", "33" = "33", "34" = "34",  
"35" = "35", "36" = "36", "37" = "37", "38" = "38", "39" = "39", "40" = "40", "41"  
= "41", "42" = "42", "43" = "43", "44" = "44", "45" = "45", "46" = "46", "47" = "47",  
"48" = "48", "49" = "49", "50" = "50", "51" = "51", "52" = "52", "53" = "53", "54"  
= "54", "55" = "55", "56" = "56"), inline = TRUE))) ,
```

```
selectInput("Method", label = "Algoritmo",  
choices = c("nearest_insertion", "farthest_insertion",  
"cheapest_insertion", "arbitrary_insertion"  
), selected = "nearest_insertion")  
,  
mainPanel(  
tabsetPanel(  

```

```
        tabPanel("contenedor verde",
                textOutput("Distancia verde")
                ,
                verbatimTextOutput("values"),
                tableOutput("values")
        ),

        tabPanel("contenedor amarillo",
                textOutput("Distancia amarillo"),
                verbatimTextOutput("valuesAmarillo"),
                tableOutput("valuesAmarillo")
        ),
        tabPanel("contenedor azul",
                textOutput("Distancia azul"),
                verbatimTextOutput("valuesAzul"),
                tableOutput("valuesAzul")
        )
    )
)
))
```

3. ASPECTOS DE INTERÉS MEMORIA SEULA

SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS (FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE DE ENVASES LIGEROS Y PAPEL CARTÓN ASÍ COMO DE LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021

2 PROYECTO DE GESTIÓN

2.1 Recogida de Residuos Sólidos Urbanos (RSU)

La recogida de los residuos descritos en el apartado 1.3.1 Recogida de RSU, contará con las siguientes frecuencias:

A nivel orientativo, se presenta el siguiente calendario de recogidas para el año 2022.

enero	febrero	marzo
L M X J V S D 2 6 9 16 23 30	L M X J V S D 6 13 20 27	L M X J V S D 6 13 20 27

abril	mayo	junio
L M X J V S D 3 10 17 24	L M X J V S D 1 8 15 22 29	L M X J V S D 5 12 19 26

julio	agosto	septiembre
L M X J V S D 3 10 17 24 31	L M X J V S D 7 14 21 28	L M X J V S D 4 11 18 25

octubre	noviembre	diciembre
L M X J V S D 2 9 16 23 30	L M X J V S D 1 8 15 22 29	L M X J V S D 4 11 18 25

La recogida de RSU se realiza de modo que se busca la ruta más óptima, para que sea más sostenible medioambientalmente y la huella de carbono del servicio se vea reducida.


25

CONFIDENCIAL

Seula

SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS
(FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE
DE ENVASES LIGEROS Y PAPEL CARTÓN ASÍ COMO DE
LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021

En esta ruta se recogerán los contenedores de Fuentes de Ebro y de la pedanía de Rodén.



CONFIDENCIAL

Para la recogida de todos los contenedores se estima un recorrido de 49'4km que se realizan en un tiempo estimado de 4 horas.

El servicio se realiza con dos trabajadores. Un conductor, encargado de conducir el camión y de las operaciones necesarias a realizar en cabina. La otra persona será un peón que en cada punto de recogida se bajará del camión para colocar el contenedor en la posición para que éste pueda ser recogido y vaciado en el compactador.

El recorrido dentro del núcleo urbano de Fuentes de Ebro será el siguiente:

26



SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS (FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE DE ENVASES LIGEROS Y PAPEL CARTÓN ASÍ COMO DE LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021

2.2 Recogida de papel-cartón

A nivel orientativo, se presenta el siguiente calendario de recogidas para el año 2022.

L	M	X	J	V	S	D
						1 2
4	5	6	7	8	9	
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

L	M	X	J	V	S	D
		2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

L	M	X	J	V	S	D
		2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

L	M	X	J	V	S	D
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

L	M	X	J	V	S	D
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

L	M	X	J	V	S	D
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

L	M	X	J	V	S	D
						1 2 3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

L	M	X	J	V	S	D
			3	4	5	6 7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

L	M	X	J	V	S	D
						1 2 3 4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

L	M	X	J	V	S	D
						1 2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

L	M	X	J	V	S	D
		1	2	3	4	5 6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

L	M	X	J	V	S	D
						1 2 3 4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

CONFIDENCIAL


En esta ruta se recogerán los contenedores de Fuentes de Ebro y de la pedanía de Rodén.

La recogida de papel-cartón se realiza de modo que se busca la ruta más óptima, para que sea más sostenible medioambientalmente y la huella de carbono del servicio se vea reducida.

El emplazamiento de los contenedores del municipio es el que se muestra en el siguiente plano:

Seula

SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS
(FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE
DE ENVASES LIGEROS Y PAPEL CARTÓN ASÍ COMO DE
LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021



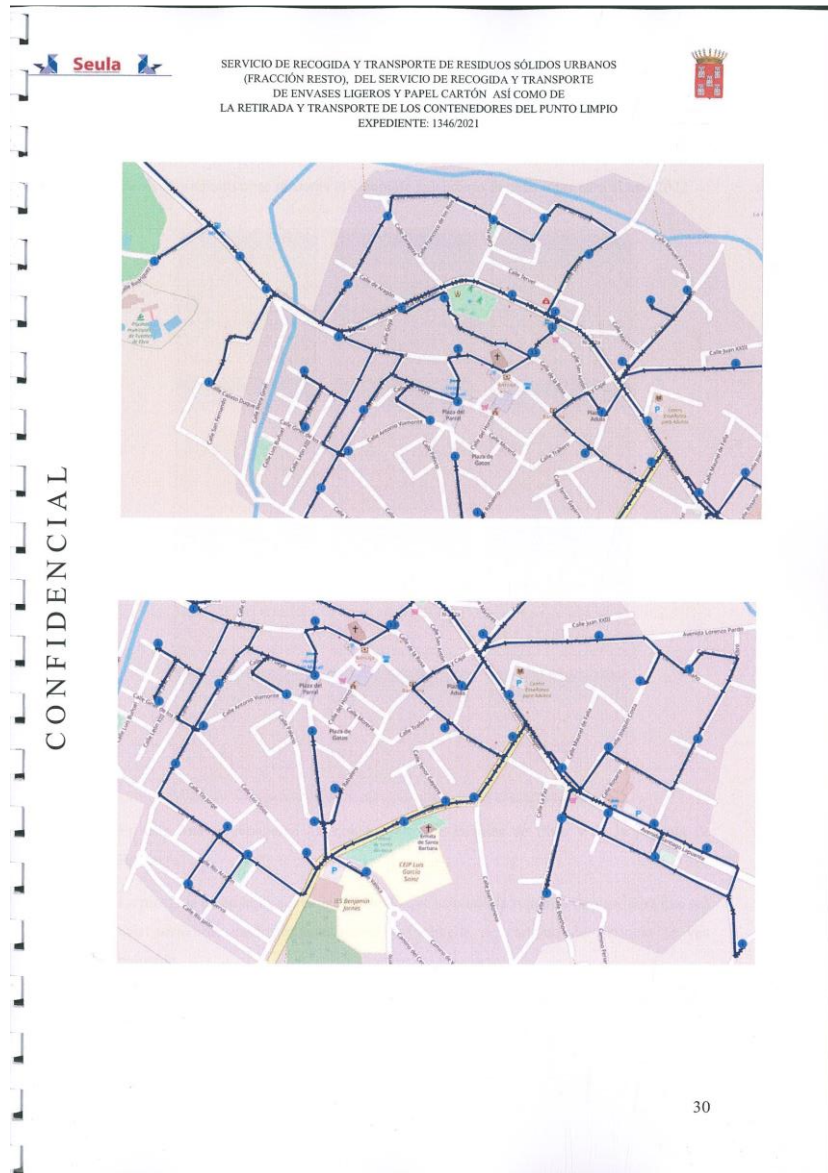
CONFIDENCIAL

Para la recogida de todos los contenedores se estima un recorrido de 22'1km que se realizan en un tiempo estimado de 2 horas y 30 minutos.

El servicio se realiza con dos trabajadores. Un conductor, encargado de conducir el camión y de las operaciones necesarias a realizar en cabina. La otra persona será un peón que en cada punto de recogida se bajará del camión para colocar el contenedor en la posición para que éste pueda ser recogido y vaciado en el compactador.

El recorrido dentro del núcleo urbano de Fuentes de Ebro será el siguiente:

29



SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS
(FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE
DE ENVASES LIGEROS Y PAPEL CARTÓN, ASÍ COMO DE
LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021

2.3 Recogida de envases

A nivel orientativo, se presenta el siguiente calendario de recogidas para el año 2022.

L	M	X	J	V	S	D
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

L	M	X	J	V	S	D
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28						

L	M	X	J	V	S	D
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

L	M	X	J	V	S	D
			1	2	3	
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

L	M	X	J	V	S	D
						1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31					

L	M	X	J	V	S	D
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28	29	30			

L	M	X	J	V	S	D
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	31

L	M	X	J	V	S	D
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31			

L	M	X	J	V	S	D
				1	2	3
4	5	6	7	8	9	10
11	12	13	14	15	16	17
18	19	20	21	22	23	24
25	26	27	28	29	30	

L	M	X	J	V	S	D
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31						

L	M	X	J	V	S	D
	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30				

L	M	X	J	V	S	D
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

CONFIDENCIAL

En esta ruta se recogerán los contenedores de Fuentes de Ebro y de la pedanía de Rodén.

La recogida de envases se realiza de modo que se busca la ruta más óptima, para que sea más sostenible medioambientalmente y la huella de carbono del servicio se vea reducida.

El emplazamiento de los contenedores del municipio es el que se muestra en el siguiente plano:

Seula

SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS
(FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE
DE ENVASES LIGEROS Y PAPEL CARTÓN ASÍ COMO DE
LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021

CONFIDENCIAL

Para la recogida de todos los contenedores se estima un recorrido de 40'8km que se realizan en un tiempo estimado de 2 horas y 55 minutos.

El servicio se realiza con dos trabajadores. Un conductor, encargado de conducir el camión y de las operaciones necesarias a realizar en cabina. La otra persona será un peón que en cada punto de recogida se bajará del camión para colocar el contenedor en la posición para que éste pueda ser recogido y vaciado en el compactador.

El recorrido dentro del núcleo urbano de Fuentes de Ebro será el siguiente:

32

Seula

SERVICIO DE RECOGIDA Y TRANSPORTE DE RESIDUOS SÓLIDOS URBANOS
(FRACCIÓN RESTO), DEL SERVICIO DE RECOGIDA Y TRANSPORTE
DE ENVASES LIGEROS Y PAPEL CARTÓN ASÍ COMO DE
LA RETIRADA Y TRANSPORTE DE LOS CONTENEDORES DEL PUNTO LIMPIO
EXPEDIENTE: 1346/2021

CONFIDENCIAL

33

Relación de documentos

Memoria 61 páginas
 Anexos 31 páginas

La Almunia, a 19 de septiembre de 2022



Firmado: Darío Fraile Tolón