UNIVERSIDAD POLITÉCNICA SALESIANA
SEDE CUENCA

CARRERA DE TELECOMUNICACIONES

DESARROLLO DE UN PROGRAMA PARA LA ASIGNACIÓN DE ESTUDIANTES Y RUTAS APLICADO AL TRANSPORTE ESCOLAR.

Trabajo de titulación previo a la obtención del título de Ingeniero en Telecomunicaciones.

AUTOR: JOSÉ ANTONIO MARTÍNEZ CORONEL

TUTOR: ING. JUAN PAÚL INGA ORTEGA, MgT.

Cuenca – Ecuador

2024

CERTIFICADO DE RESPONSABILIDAD Y AUTORÍA DEL TRABAJO DE TITULACIÓN

Yo, José Antonio Martínez Coronel con documento de identificación N° 0107610081; manifiesto que:

Soy el autor y responsable del presente trabajo; y, autorizo a que sin fines de lucro la Universidad Politécnica Salesiana pueda usar, difundir, reproducir o publicar de manera total o parcial el presente trabajo de titulación.

Cuenca, 29 de julio de 2024

Atentamente,

José Antonio Martínez Coronel
0107610081

CERTIFICADO DE CESIÓN DE DERECHOS DE AUTOR DEL TRABAJO DE TITULACIÓN A LA UNIVERSIDAD POLITÉCNICA SALESIANA

Yo, José Antonio Martínez Coronel con documento de identificación N° 0107610081, expreso mi voluntad y por medio del presente documento cedo a la Universidad Politécnica Salesiana la titularidad sobre los derechos patrimoniales en virtud de que soy autor del Proyecto Técnico: "Desarrollo de un programa para la asignación de estudiantes y rutas aplicado al transporte escolar.", el cual ha sido desarrollado para optar por el título de: Ingeniero en Telecomunicaciones, en la Universidad Politécnica Salesiana, quedando la Universidad facultada para ejercer plenamente los derechos cedidos anteriormente.

En concordancia con lo manifestado, suscribo este documento en el momento que hago la entrega del trabajo final en formato digital a la Biblioteca de la Universidad Politécnica Salesiana.

Cuenca, 29 de julio de 2024

Atentamente,

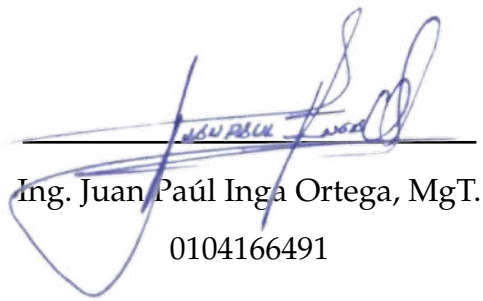
José Antonio Martínez Coronel
0107610081

CERTIFICADO DE DIRECCIÓN DEL TRABAJO DE TITULACIÓN

Yo, Juan Paúl Inga Ortega con documento de identificación N° 0104166491, docente de la Universidad Politécnica Salesiana, declaro que bajo mi tutoría fue desarrollado el trabajo de titulación: DESARROLLO DE UN PROGRAMA PARA LA ASIGNACIÓN DE ESTUDIANTES Y RUTAS APLICADO AL TRANSPORTE ESCOLAR., realizado por José Antonio Martínez Coronel con documento de identificación N° 0107610081, obteniendo como resultado final el trabajo de titulación bajo la opción proyecto técnico que cumple con todos los requisitos determinados por la Universidad Politécnica Salesiana.

Cuenca, 29 de julio de 2024

Atentamente,



Ing. Juan Paúl Inga Ortega, MgT.
0104166491

AGRADECIMIENTOS

Fiel a mis creencias religiosas, quiero empezar por agradecer a Dios, pues todo proyecto realizado en su nombre prosperará y este trabajo para mí es evidencia de aquello, de la gracia del señor bendiciendo y resguardando los anhelos de mi corazón.

Aprovecho para agradecer a mi papá, Antonio Martínez, por las enseñanzas inculcadas, por su ejemplo de trabajo, perseverancia y lucha, virtudes que asimile y que me han acompañado durante toda mi carrera Universitaria.

Agradezco a mi mamá, Katherine Coronel, por el sacrificio realizado para que pueda culminar mis estudios, pues en suma a su apoyo incondicional, conformaron la motivación que me permitieron superar los desafíos en las diferentes etapas de mi vida como estudiante Universitario.

No quiero terminar sin mencionar a aquellas personas que fueron esenciales para alcanzar esta meta, a mis abuelos Jaime Coronel, María González, Soledad González, quienes fueron un apoyo incondicional, demostrándome una vez más con acciones concretas el cariño que tienen hacía mí, ellos son seres a quienes tengo un cariño inmensurable.

Para finalizar a mi tío Jaime Coronel y Alexandra Coronel quienes con su respaldo me motivaron a continuar con ímpetu con mis estudios. Gracias a todos ellos, mi cariño y gratitud siempre estará con cada uno de ellos.

José Antonio Martínez

DEDICATORIA

El presente trabajo de titulación es dedicado a aquel que puso especial interés en cada paso de mi carrera Universitaria, para quien exaltó cada uno de mis logros conseguidos y los ostento con orgullo, este trabajo va dedicado a Jaime Coronel, también quiero incluir a María González y Soledad González, quienes no solo me respaldaron en la obtención de este título, también atestiguaron el arduo sacrificio que significó el culminar mi carrera Universitaria y para quienes este logro conseguido representa un gran orgullo.

Dedico el trabajo a mis padres porque estos logros obtenidos son el fruto de sus enseñanzas y de su devoción en mi formación como persona, me han demostrado un apoyo incondicional, pues el conjunto de todos estos aspectos ha sido crucial en la obtención de este logro.

Por último, dedico este trabajo a mis hermanos, para que les sirva de motivación a trabajar con perseverancia y disciplina por alcanzar los logros que se han propuesto en sus vidas, pues les sobran cualidades, pero no basta con tener talento sino con explotarlo.

Índice general

Índice de figuras

Índice de tablas

Resumen

Diversas instituciones educativas presentan problemas en la asignación de estudiantes a sus unidades de transporte, ya sea por falta de gestión, proceso tedioso o uso adecuado del tiempo. Además, el problema se ve complicado cuando la cantidad de unidades y estudiantes aumenta ya que la asignación manual de estudiantes a unidades de transporte no siempre optimiza la asignación y tampoco las rutas. Esto provoca que rutas extensas con largos tiempos de espera o que el personal administrativo a cargo del proceso deje de realizar otras actividades para concentrarse en el proceso de planificación de rutas. Además, si no hay un coordinador del servicio de transporte, los representantes deben gestionar el servicio de manera directa con los choferes, lo que puede ser un proceso complicado y desorganizado.

Para abordar estos problemas, se propone el desarrollo de un programa especializado que optimice la asignación de estudiantes y el cálculo de rutas de transporte escolar haciendo uso de heurísticas que resuelven problemas generales de ingeniería a través de optimización. Así, el presente trabajo propone un aplicativo en software libre que combina algoritmos de optimización y agrupación para mejorar la asignación de recursos y la planificación de trayectos en la ciudad de Cuenca - Ecuador. El programa lee datos desde una base de datos con información básica que puede ser proporcionada a través de un sencillo archivo de excel que incluirán información sobre estudiantes, conductores y unidades de transporte. La solución propuesta integra funciones de geolocalización para mapear ubicaciones y definir rutas óptimas basadas en distancia y tiempo. Además, cuenta con una interfaz gráfica intuitiva para visualizar rutas en un mapa georeferenciado y un archivo de salida que indica la buseta que le toca usar a cada estudiante inscrito en el sistema de transporte escolar. Establece una ruta de transporte para cada buseta para el conjunto

de estudiantes sugiriendo el orden en que el conductor debe recoger a cada pasajero, para que sea la ruta más corta desde cada estudiante hacia el siguiente hasta llegar a la unidad educativa.

El programa es flexible, adaptándose a cambios en la base de datos y recalculando de forma automática las rutas hacia el punto de destino.

Palabras clave: Asignación de recursos; Agrupamiento de Conjuntos; Cálculo de rutas; GMM; Optimización; Planificación de rutas; Problemas de enrutamiento de Buses Escolares.

Abstract

Several educational institutions have problems assigning students to their transportation units, either due to a lack of management, a tedious process, or an inadequate use of time. In addition, the problem becomes more complicated when the number of units and students increases since manually assigning students to transportation units does not always optimize the assignment and routes. This results in long routes with long waiting times, or the administrative staff in charge of the process stops performing other activities to concentrate on the route planning process. In addition, if there is no transportation service coordinator, the representatives must manage the service directly with the drivers, which can be a complicated and disorganized process.

Our proposed solution, a specialized program that optimizes student assignment and transportation route planning, offers significant benefits to educational institutions. By leveraging heuristics to solve general engineering problems through optimization, it promises to streamline operations and improve resource allocation. Thus, this work proposes a free software application that combines optimization and clustering algorithms to improve resource allocation and route planning in Cuenca, Ecuador. The program reads data from a database with basic information that can be provided through a simple Excel file, which will include information about students, drivers, and transportation units. The proposed solution integrates geolocation functions to map locations and define optimal routes based on distance and time. It also has an intuitive graphical interface to visualize routes on a geo-referenced map and an output file that indicates which bus to use for each student enrolled in the school transportation system. It establishes a transportation route for each bus for the assigned set of students by calculating the shortest route from each

student to the next one until reaching the educational unit. This process is designed to minimize travel time and ensure that students arrive at their destination in the most efficient manner possible.

Moreover, the program is designed to be flexible, seamlessly adapting to changes in the database. This means that it can automatically recalculate routes to the destination point, ensuring that the transportation system remains efficient and up-to-date.

Keywords: Clustering; GMM; Optimization; Resource Allocation; Routing; School bus routing problems.

Antecedentes

El problema de rutas de autobuses escolares, conocido por sus siglas SBRP (Del inglés, *School Bus Routing Problem*) se constituye en uno de los problemas más complejos de resolver dentro del campo del transporte y ha sido objeto de estudio por muchos años, ya que engloba una serie de subproblemas operativos, como lo es la selección de la parada de bus, la selección de los usuarios, la asignación de usuarios, encontrar la ruta de menor tiempo hacia la unidad educativa, etc. [1], [2].

El problema de SBRP es algo con lo que tienen que lidiar las autoridades escolares como también los proveedores de servicio de transporte escolar, ya que son los encargados de ofrecer el servicio y se encargan de que este sea confiable, seguro y puntual. Para lograr este propósito va mucho más allá de calcular la ruta del transporte escolar, que dicho sea de paso tiene que ser el menor trayecto, sino abarca también la cantidad de usuarios que serán asignados a las unidades de transporte [3].

Por otro lado, dentro del problema de asignación de estudiantes, este proceso se realiza de forma manual y son los mismos representantes quienes deben acercarse al coordinador del servicio de transporte, en caso de que exista alguno, e indicar la información básica del estudiante (nombre y dirección en donde vive). Con estos datos el coordinador buscará alguna unidad de transporte que tenga una ruta cercana a la dirección de la casa del estudiante y, además, que la unidad tenga disponibilidad de pasajeros. Se debe tener presente que un exceso de pasajeros es penado por la ley de Tránsito de nuestro país (artículo 382 del Código Orgánico Integral Penal, COIP). En caso de no haber un coordinador de las unidades de transporte el representante o padre de familia deberá gestionar con la institución o, en el peor de los casos, conversar con los choferes de las unidades uno por uno hasta localizar quién le puede ayudar con el servicio de transporte [4].

Justificación

A mediados del siglo XX se empezaron a utilizar algoritmos informáticos capaces de ejecutar una serie de instrucciones con el fin de resolver problemas mediante una serie de cálculos, los algoritmos son esenciales en la informática y en la creación de diversos sistemas. En la actualidad, se utilizan diversos algoritmos para enrutamiento, búsqueda, análisis probabilísticos, entre otros; la amplia variedad de éstos permite que puedan ser utilizados en diversos campos, sobre todo cuando hablamos de automatización y optimización de sistemas. [5].

El presente proyecto pretende la creación de un programa especializado en la asignación de estudiantes a busetas escolares, contemplando la cantidad de unidades de transporte escolar disponibles por cada institución educativa, el programa cuenta con la capacidad de considerar la capacidad máxima de pasajeros por unidad de transporte escolar, para no exceder la cantidad de pasajeros permitida para cada unidad, ofreciendo mayor seguridad a los pasajeros y al conductor, además el programa cuenta con la cualidad de leer la información y datos de los estudiantes de una hoja de Excel, facilitando el proceso de cargar información de estudiantes inclusive para aquellos con poco o nulo conocimiento en programación. Por último, el programa sugiere cual es el orden que debe seguir el conductor para recoger a cada pasajero, para que la ruta que tome el conductor sea la de menor trayecto entre usuarios del servicio de transporte y además sea la ruta de menor trayecto hacia la unidad educativa.

Gracias a este programa se podrá realizar una distribución de los estudiantes a las unidades de transporte escolar de una forma eficiente sin exceder la capacidad de pasajeros de las busetas, agrupando a los usuarios cercanos e indicando la ruta de menor trayecto entre usuarios hasta llegar al punto de destino.

Objetivos

Objetivo General

- Desarrollar un programa para la asignación de estudiantes y rutas aplicado al transporte escolar.

Objetivos específicos:

- Realizar un estado del arte referente a los algoritmos y heurísticas utilizados en la asignación de recursos y cálculos de rutas.
- Desarrollar el programa para la asignación de estudiantes y cálculo de rutas basado en algoritmos de optimización.
- Realizar pruebas de funcionamiento y comprobar el correcto funcionamiento del programa.

Introducción

El ministerio de educación según cifras otorgadas, anuncia que para el año lectivo 2022-2023 cerca de 10.500 alumnos se matricularon en Primaria, más de 7.300 en Secundaria, más de 5.000 en Infantil y alrededor de 2.400 en Bachillerato, esto repercute que el uso de transporte escolar sea cada vez más común dentro de la ciudad de Cuenca-Ecuador, constituyéndose en una manera práctica de movilizar a niños y niñas a sus instituciones educativas, a su vez que los representantes pueden despreocuparse por la movilización de sus representados y enfocarse en sus actividades diarias con la certeza de que sus representados arribaran a la institución educativa [6].

Desde el punto de vista institucional el panorama no es tan favorable, los procesos implicados en asignar estudiantes al transporte escolar son rudimentarios considerando que es época de la cuarta revolución industrial, hay una evidente falta de modernización en el proceso, además de no considerar lo tedioso que implica para los representantes estudiantiles personarse para coordinar como será brindado el servicio [7].

Es evidente la notable carencia de optimización del servicio y la poca o nula intención de que este servicio se automatice. Por último la responsabilidad recae sobre la empresa transportista quienes se preocupan de sus pasajeros y además de la ruta que van a tomar para llegar a tiempo a la institución y de cumplir con las exigencias de ley para prestar el servicio y salvaguardar la vida de sus pasajeros.

La presente tesis tiene como objetivo principal realizar un programa de asignación de estudiantes al transporte escolar, que permita asignar de forma automática estudiantes a las diferentes busetas escolares sin exceder el número de pasajeros de cada unidad, considerando la ubicación de los estudiantes y el punto de

partida de las busetas y que además calcule la ruta más corta y la de menor tiempo hacia la institución educativa.

Exponga también las limitaciones del trabajo de titulación.

Capítulo 1

Marco Teórico

El siguiente apartado se enfoca en contextualizar los temas relevantes y de interés sobre los cuales se fundamenta este trabajo y que constituyen la base para la elaboración del presente proyecto de grado.

En el campo de la ingeniería los grafos son esenciales para resolver problemas matemáticos relacionados con mapas, contadores, dibujo computacional, entre otras, sin embargo han encontrado un nicho importante en el cálculo de rutas o encontrar caminos más cortos o de menos peso computacional, posicionando la teoría de grafos en parte esencial para resolver problemas computacionales complejos como los problemas de enrutamiento de vehículos y problemas computacionales derivados del mismo [8].

Así se contextualiza en primera instancia un breve estado del arte en el que se busca definir cuáles son los algoritmos (heurísticas y metaheurísticas) que pudiesen ser usadas para resolver el problema definido en este proyecto. Se expone un breve fundamento teórico de los principales algoritmos usados para agrupación, asignación de recursos y cálculo de rutas.

Luego, se expone las principales librerías que son de interés para resolver problemas con grafos.

1.1. Estado del Arte

El buscar optimizar rutas se han constituido en un tema de estudio bastante amplio, lo que ha repercutido en la creación de estrategias para encontrar rutas óptimas, muchos plantean como parte de la solución la asignación de recursos, ya que una buena asignación repercute en un correcto manejo y orden de la información, este campo de estudio se ha aplicado a servicios de paquetería, problema del autobús escolar, etc. Esto impulsó el desarrollo de algoritmos que permitan encontrar rutas óptimas para la entrega de paquetes y realizar una asignación de recursos que favorezca el encontrar rutas óptimas, el éxito de estos algoritmos ha permitido expandir este campo de estudio a servicios de entrega de comida, servicios de entrega de paquetes y servicios de buses escolares para transportar estudiantes a las unidades educativas.

El enfoque propuesto por Sciortino [9], busca utilizar la menor cantidad de buses escolares para recoger estudiantes, asignar a los estudiantes a las paradas de bus más cercanas a la ubicación del estudiante, disminuir el tiempo del viaje a la institución y que las distancias recorridas por todos los buses sea la menor posible, el artículo propone un algoritmo heurístico con restricciones de tiempo y formulado para resolver problemas de enrutamiento, el algoritmo elige de entre todas las paradas aquellas que considera óptimas para reducir el tiempo y en base a la ubicación de estas paradas asignar a los estudiantes, se utilizan modelos muy similares a los de programación entera mixta, obteniendo como resultado formular soluciones en cortos períodos de tiempo.

Además, se han propuesto algoritmos de programación lineal entera y el principio del problema del agente viajero (TSP, del Inglés *Travelling Salesman Problem*) para ofrecer una solución a la asignación de estudiantes y calcular la menor ruta a la institución educativa. Esto se lleva a cabo con el uso del algoritmo genético o modelos matemáticos como el MTZ (Miller, Tucker y Zemlin) que es de forma cuadrática, que buscan ofrecer soluciones eficientes mediante soluciones aproximadas. Por ejemplo, el artículo de Florez [10] se centra de manera esencial en el cálculo de rutas, presenta una solución basada en teoría de grafos, mediante un modelo basado en una serie de sumatorias busca minimizar los costos de recorrido de cada ruta y así disminuir

los tiempos de arribo para recoger a cada estudiante, la asignación de estudiantes se realiza asignando a un grupo de estudiantes en paradas de buses cercanas a sus casas, contemplando la distancia en pasos de cada estudiante a la parada de bus y que esta sea la menor posible, usando el método MTZ, busca que cada estudiante sea recogido una vez y que sea en el menor tiempo posible, sin embargo su eficiencia disminuye a medida que aumentan los datos ya que tarda más en ofrecer una solución.

El enfoque propuesto por Díaz [11] propone la utilización del algoritmo genético, mediante un parámetro de selección analiza y elige dos paradas de bus como posible solución. Se utiliza un operador de crossover para seleccionar la posible mejor solución de las dos opciones, con el fin de encontrar una ruta de menor tiempo, con el algoritmo genético se evalúa la solución tomada para aprobar o descartar esa solución, en caso de ser descartadas se toman dos puntos nuevos, con el algoritmo genético se exploran una gran cantidad de soluciones para elegir la más factible. Para la selección de puntos el programa considera una distancia prudente entre la casa del estudiante, calculando la menor cantidad de pasos del estudiante a la parada y al cumplirse está condición el estudiante se asigna a esa parada de bus, se crean paradas de buses cercanas a esta ruta para asignar a los estudiantes.

La heurística planteada en López [12] propone dividir el problema de asignación y rutas en subproblemas, definen el problema como combinatorio y hacen uso del método de generación de columnas, además plantea una asignación de estudiantes considerando la capacidad de la buseta, para la asignación, propusieron agrupar a aquellos estudiantes considerando su lugar de residencia y sectorizando la ciudad, agrupan a aquellos que vivan dentro del sector, para lo cual emplean el algoritmo de Shin & Han, que se encarga de crear y ajustar clústeres con las ubicaciones de los estudiantes, proponen además la generación de rutas para vehículos con ventanas de tiempo (VRPTW, del inglés *Vehicle Routing Problem with Time Windows*), lo cual considera la hora de entrada y salida de la unidad educativa para la selección de rutas y programar las rutas de los buses.

Por otro lado, los mismos principios para la resolución del problema de asignación y cálculo de rutas han sido planteados para entrega de correo, como por ejemplo las metaheurísticas tratadas por Gussmag [13] que caracteriza el problema

de entrega de correo de tres formas independientes: la agrupación de clientes en distritos, la elección de un modo para cada distrito y la ruta del cartero por su distrito. El enfoque de la solución se implementó en dos fases, utilizando una heurística de pétalos de rosa para desplegar una serie de rutas como posible solución y después aplicar partición de conjuntos para definir los distritos, se utilizan metaheurísticas como K-Means para el agrupamiento de los clientes en cada distrito.

De igual manera basándose en agrupamiento la propuesta tratada por Wang [14] propone el uso de algoritmos de agrupamiento para resolver el problema de agrupamiento de rutas de bus, definiendo las rutas de embarque, considerando los destinos y el número de rutas de autobús, se busca también disminuir el tiempo de espera de los usuarios focalizando las unidades de autobús por paradas.

El enfoque plateado por Park [15] propone un algoritmo que considera el cálculo de carga de trabajo y plantea un modelo que incluye el modelo de carga de trabajo para realizar la entrega postal, considera el tiempo de clasificación de los paquetes, el tiempo de entrega y los expresa como una sumatoria, con el uso de algoritmos como Modelos de Mezcla Gaussiana(GMM, del inglés *Gaussian mixture models*) sectorizan los diferentes puntos de entrega para obtener el menor tiempo posible, para así encontrar el que genere la menor carga en la entrega, el algoritmo también contempla los tiempos de movilización dentro de la zona urbana y rural.

Por otro lado, el enfoque de Yang [16] plantea realizar la optimización de entrega de paquetes haciendo uso del Problema del árbol de Steiner, de todas las oficinas de correo se seleccionan a aquellas oficinas de correos que se encuentren en ubicaciones consideradas óptimas esto se determina con el uso del árbol de Steiner, se calcula la interconexión más corta entre distintos puntos, a su vez esto determina las oficinas de correo óptimas que permiten encontrar la ruta más eficaz.

Por otra parte, el enfoque propuesto por Rahman [17] va mucho más allá que el postulado anterior, ya que propone realizar la entrega de paquetes, pero con la optimización de agentes de entrega, realizando un agrupamiento de los puntos de entrega con el algoritmo K-Means y asignando un agente a cada punto para después trazar la ruta de entrega con algoritmos genéticos.

La propuesta planteada por Sgarro en [18] guarda ciertas similitudes con

propuestas anteriores, está propuesta busca resolver los problemas de entrega de paquetes en China, ya que para optimizar las entregas se hace uso de algoritmos de agrupamiento y para delimitar los sectores de entrega, para estos procesos se plantea una mejora del algoritmo K-Means para tratar de mejorar la asignación de aquellos usuarios lejanos, para el cálculo de rutas se utiliza el algoritmo de colonia de hormigas, este algoritmo ha sido poco utilizado para este tipo de propuestas, como lo es en la optimización de entrega de paquetes, por lo tanto sus autores contrastan la solución obtenida con el uso del algoritmo de colonia de hormigas contra el algoritmo genético, tratando de optimizar las rutas de entrega, la cantidad de vehículos para la entrega y optimizar el número de personal, reduciéndolo al mínimo necesario.

Por su parte, Lee en [19] analiza una propuesta para brindar servicio de taxi a personas con capacidades diferentes, por lo tanto calculan la ruta de menor tiempo desde la casa del usuario hacia su destino y trata de reducir el tiempo de espera de los usuarios, se considera además el número de usuarios, el número de taxis disponibles, para la asignación se utiliza el algoritmo Modelo de Mezcla Gaussiana, además se empleó ventanas de tiempo para el cálculo de rutas.

Se han propuesto diversos métodos, algoritmos y metaheurísticas para el agrupamiento y asignación de usuarios y el cálculo de una ruta óptimas para la solución del problema de enrutamiento de autobús escolar, se han implementado algoritmos muy similares para solucionar problemas de entrega de correo y paquetes. Sin embargo, las propuestas son tan amplias y se emplean algoritmos muy distintos que requiere pruebas exhaustivas en diversos escenarios para comprobar el funcionamiento de cada programa y determinar cuál de ellos se adapta mejor a los diversos escenarios y ofrece los mejores resultados de asignación y cálculo de rutas.

La tabla 1.1, resume la comparación realizada entre trabajos similares y la propuesta desarrollada en este trabajo. Las principales diferencias son el hecho de leer la información de Excel como la ubicación de los estudiantes en sistemas de coordenadas, además la asignación de estudiantes considera la capacidad de la buseta y ofrece servicio puerta a puerta, proceso para el cual es necesario hacer agrupamiento de individuos, haciendo uso del Algoritmo de Mezcla Gaussiana, que ofrece resultados menos variables que K-Means y por último el tiempo de ejecución

del programa es bastante rápido.

Tabla 1.1: Comparativa entre el presente trabajo y Casos Similares

Fuente: Autor

Referencias	Heurísticas										Métricas			Tipo de Aplicación					
	Teoría de Grafos	Algoritmo Genético	Pétalos de Rosa	Shin & Han	Carga de Trabajo	Árbol de Steiner	K-Means	Colonia de Hormigas	Gaussian Mixture Model	Algoritmos de Agrupamiento	Asignación	Agrupamiento	Tiempo	Sectorización de Estudiantes	Entrega de Paquetes	Asignación conductor más cercano	Agrupamiento de individuos	Cálculo de Rutas	Explorar la mejor Parada de Bus
Sciortino[9]	X										X								X
Florez[10]	X	X											X					X	
Díaz[11]		X											X			X		X	X
López[12]		X		X									X	X			X	X	
Gussmag[13]			X				X			X	X	X			X		X		
Wang[14]										X	X	X				X	X		X
Park[15]					X				X				X		X				
Yang[16]	X					X							X		X			X	
Rahman[17]		X					X				X	X			X	X	X		
Sgarro[18]							X	X		X	X	X			X		X		
Lee[19]								X		X	X	X				X		X	
Este Trabajo					**X**				**X**	**X**	**X**	**X**	**X**	**X**			**X**	**X**	

1.2. Algoritmo de Greedy

El algoritmo de Greedy también conocido como esquema voraz, es un algoritmo muy utilizado en problemas de optimización. El algoritmo de Greedy tiene la particularidad de ofrecer soluciones construidas por pasos sin la necesidad de volver a considerar las decisiones ya tomadas, lo cual disminuye el tiempo de espera para poder obtener una respuesta frente a un determinado problema [20].

Greedy es utilizado para resolver problemas como hallar dentro de un grupo de n posibles candidatos una posible solución, que se capaz de maximizar o minimizar una función objetivo y que a su vez está resulte ser una solución óptima,

está peculiaridad ha hecho que Greedy sea utilizado para resolver problemas de planificación de datos, problemas de grafos, problemas de cálculo de rutas, optimizar rutas, etc [20].

1.3. Algoritmos de Agrupamiento

Los algoritmos de agrupamiento (en inglés *Clustering*) se encargan de automatizar tareas en las que, para un gran número de datos ingresados se encontrarán las mejores agrupaciones posibles, en general basados en condiciones similares, por ejemplo por distancia, color o cualquier condición que se pueda especificar; estos grupos generados se denominan "Agrupaciones". Estas actividades de agrupamiento son muy usadas en procesos de aprendizaje de máquina [21].

Los grupos o también llamados clústeres de datos que conforman una agrupación son similares entre sí o guardan cierta cercanía entre datos y facilitan encontrar cierta información que guarde similitudes o patrones. El conocimiento que tiene el algoritmo de cada grupo de datos permite dar una descripción de los diversos conjuntos de datos, por este motivo es muy utilizado en la minería de datos [21].

La función principal de estos algoritmos es catalogar aquellos datos que no hayan sido categorizados, es decir aquellos grupos de datos que no tengan grupo ni categoría, su funcionamiento radica en la búsqueda de datos en todos los grupos formados y todos los datos que se ingresen serán asignados a cada grupo, dependiendo los parámetros que tenga cada grupo, por ejemplo, estos parámetros pueden ser distancia o similitud [22].

En general, los datos suelen tener una etiqueta que contiene información para identificar al dato y ubicarlo, sin embargo, con estos algoritmos de agrupamiento no es necesario. Los algoritmos de agrupamiento de forma automática ubican datos, información y descubren agrupaciones sin la necesidad de supervisar el proceso y lo hacen en función de su índice o posición dentro del listado de datos [21].

Los distintos tipos de algoritmos de agrupación, se subdividen de acuerdo a su capacidad para manejar información y datos sin etiquetas. Sin embargo en este trabajo tan solo se manejarán los algoritmos utilizados para cumplir con los objetivos

propuestos, estos son algoritmo K-Means, BIRCH y Guassiano [22].

1.3.1. Algoritmo K-Means

Es un método de agrupamiento cuya simpleza y velocidad lo han consolidado como uno de los algoritmos más reconocidos, consiste en un algoritmo basado en centroides (punto equidistante del grupo y más representativo de cada agrupación o clúster). En K-Means, cada clúster está representado por un punto llamado centroide, que es el promedio de todos los puntos en ese clúster. El objetivo de este algoritmo es minimizar la suma de las distancias cuadradas entre cada punto y el centroide de su clúster y lo hace buscando la mínima distancia entre un objeto y su centroide dentro de un determinado grupo [23]. Con respecto a la sensibilidad de datos atípicos, los centroides son sensibles a dichos valores, ya que el promedio se ve afectado por valores extremos lo que puede modificar la agrupación resultante.

Se puede decir que este algoritmo se ejecuta en cuatro pasos [24]:

1. Especificar la cantidad deseada de grupos (K) y se establecerán un K número de centroides aleatorios dentro de todo el espacio de datos, estos se denominan centroides iniciales.

2. Se asignarán los objetos al centroide de menor distancia, es decir al más cercano, calcula la menor distancia entre el centroide y el objeto.

3. Recabar los centroides, se actualiza el centroide de cada grupo formado, el nuevo centroide elegido resulta de la media de todos los demás puntos que conforman el clúster y así se consolida el nuevo grupo formado.

4. Se debe repetir el segundo y tercer paso de forma iterativa hasta que los clústeres se mantengan constantes.

El algoritmo K-Means presenta ciertas desventajas como el hecho de indicar el número de clústeres de forma previa y la necesidad de ir ajustando los centroides con respecto a los valores iniciales [24]. Esto implica que K-Means, al usar un estado estadístico de los datos, no siempre dará la misma respuesta de agrupación frente al mismo conjunto de datos.

1.3.2. K-Medoids

Es otro algoritmo de agrupamiento particional muy usado al igual que K-Means, en procesos de aprendizaje no supervisado. También busca dividir un conjunto de datos en k grupos (o clústeres) de manera que los elementos dentro de cada grupo sean más similares entre sí que con los elementos de otros grupos.

Sin embargo, presentan diferencias fundamentales en la forma de seleccionar los centros de cada clúster y en su robustez ante valores atípicos conocidos como *outliers* [25]-[27].

En K-medoids, cada clúster está representado por un punto de datos real del conjunto, llamado "*medoide*". El medoide es el punto que minimiza la suma de las distancias a todos los otros puntos del clúster. El objetivo de K-medoids sigue siendo minimizar la suma de las disimilitudes (en general son las distancias) entre cada punto y el centro de masa de su clúster que ahora no sera un centroide sino el medoide. En cuanto a la robustez frente a datos anómalos, los medoides son más robustos a los outliers, ya que al ser puntos de datos reales, no se ven tan afectados por valores extremos. No obstante, esto implica que K-medoids sea más lento [26].

Al igual que K-Means, K-medoids, también tiene el problema de que no siempre dará la misma respuesta de agrupación frente al mismo conjunto de datos [26].

1.3.3. Algoritmo BIRCH

El algoritmo de Reducción Iteractiva Balaceada y agrupación usando Jerarquías (BIRCH, del inglés, *Balance Iterative Reducing and Clustering using Hierarchies*) es un algoritmo de Equilibrio Iterativo de Reducción y Agrupación mediante Jerarquías, caracterizado por su eficiencia al trabajar con grandes volúmenes de datos, como su nombre indica crea los grupos de forma jerárquica e incremental, realizando un solo escaneo de datos. Se trata de un algoritmo de agrupamiento en clústeres que se destaca por su eficiencia en el manejo de grandes conjuntos de datos [28].

En lugar de procesar cada punto de datos de forma individual, BIRCH crea una estructura de datos jerárquica llamada Función de Agrupación en Árbol (CFT,

del inglés *Clustering Feature Tree*) que resume la información de los datos. Es decir, el algoritmo realiza una selección de datos o puntos agrupando en pequeños grupos conocidos como "resúmenes". Estos resúmenes contienen toda la información sobre cómo se encuentran distribuidos los datos. Este algoritmo se vale de otros algoritmos como el K-Means para realizar para generar mejores resúmenes [29].

Este algoritmo realiza agrupamientos basado en jerarquías buscando minimizar las distancias entre los registros de datos y sus agrupaciones, este cálculo de distancias se puede realizar con distancias euclidianas. La función de agrupación CF que funciona para un conjunto de n datos y se define como una serie de tres valores, el primero corresponde al número de datos, suma de todos los datos y la suma datos al cuadrado [30]. Esta función de agrupación en árbol (CFT, del inglés *clustering feature tree*) es una representación compacta de los datos en cada hoja y representa sub clústeres. Cada nodo no hoja (en inglés *no leaf*) tiene b entradas, cada entrada es un puntero a los nodos hoja (en inglés *leaf*) y cada CF del no leaf es la sumatoria de cada leaf. Cada nodo leaf tiene L entradas, cada entrada tiene que ser menor a un parámetro T (puede ser el radio o diámetro del clúster).

El algoritmo BIRCH realiza los grupos siguiendo los siguientes pasos [28]:

1. **Creación del árbol:** Una vez que se ha construido el árbol CF, se pueden aplicar diferentes técnicas de agrupamiento a los nodos hoja. Por ejemplo, se puede utilizar un algoritmo de agrupamiento jerárquico para agrupar los nodos hoja y obtener los clústeres finales.Los datos se insertan de manera secuencial en el árbol CF. Cada nodo del árbol contiene información sobre los puntos de datos que representa, como el número de puntos, la suma de los puntos y la suma de los cuadrados de los puntos. Si un nodo se llena, se divide en dos nodos hijos.

2. **Agrupamiento:** Una vez que se ha construido el árbol CF, se pueden aplicar diferentes técnicas de agrupamiento a los nodos hoja. Por ejemplo, se puede utilizar un algoritmo de agrupamiento jerárquico para agrupar los nodos hoja y obtener los clústeres finales. Para obtener un resultado óptimo el algoritmo BIRCH necesita de otro algoritmo de agrupamiento como K-Means [29].

Se utiliza una técnica de agrupamiento para hacer clústeres de las entradas de

los nodos hoja y la última fase refina los grupos, ya que los datos originales solo se escanean una vez, en esta fase se corrigen las inexactitudes ocasionadas por agrupar sobre un resumen de los datos [30].

- VENTAJAS: Una ventaja es que optimiza el tiempo en la creación de grupos ya que solo escanea la base de datos una vez, rápido para bases de datos grandes consiguiendo una respuesta óptima de forma rápida, ofrece alta estabilidad y escalabilidad.

- DESVENTAJAS: Este algoritmo presenta algunas desventajas como [28], [29] :

  - Sensibilidad a los parámetros: La calidad de los resultados puede depender en gran medida de la elección de los parámetros, como el tamaño máximo de un nodo y el umbral de radio.
  - Limitaciones en la forma de los clústeres: BIRCH puede tener dificultades para encontrar clústeres de formas arbitrarias o con densidades muy variables.

1.3.4. Algoritmo Gaussiano (GMM)

El Modelo Gaussiano Mixto (GMM, del inglés *Gaussian Mixture Model*), es un algoritmo de clustering probabilístico que asume que los datos se generan a partir de una mezcla de distribuciones gaussianas (o normales). En términos más simples, GMM intenta encontrar grupos de datos donde cada grupo sigue una distribución normal multidimensional. GMM presenta algunas ventajas en cuanto a flexibilidad ya que a diferencia de K-means, que asume clústeres esféricos o elípticos y modelar clústeres de diferentes formas y tamaños debido a la flexibilidad de las distribuciones gaussianas [31]. También, GMM proporciona una probabilidad de pertenencia a cada clúster, lo que permite un análisis más profundo de los datos. En lo que respecta a la detección de datos anómalos que pueden tener elementos muy distantes del centroide, los puntos tienen una baja probabilidad de pertenecer a cualquier clúster pueden considerarse como *outliers*.

Los pasos básicos del algortimo son [32]:

1. **Asumir una mezcla gaussiana:** Selecciona el número de clústeres y ejecuta los parámetros de la distribución gaussiana con los cuales se identificarán si un valor es igual, mayor o inferior a cierto dato, en este caso los obtenidos en la desviación estándar y la media. Este proceso se realiza para cada uno de los grupos. Se asume que los datos provienen de una mezcla de varias distribuciones gaussianas. Cada mezcla gaussiana representa un clúster potencial.

2. E**stimación de parámetros:** A partir de las distribuciones Gaussianas un dato formará parte de cierto grupo dependiendo su cercanía con el centro de la distribución gaussiana. Es decir, para la creación de los grupos se consigue valiéndose de la media y la desviación estándar de los datos; la desviación estándar se utiliza para cuantificar la variación de los datos o la dispersión de los datos, la media indica el promedio de los datos del grupo considerando todos los puntos del grupo para ubicar las tendencias centrales de los datos [33]. Entonces, el algoritmo estima los parámetros de media y covarianza gaussiana y las proporciones relativas de cada componente en la mezcla.

3. **Asignación de puntos a clústeres**: Para aumentar la probabilidad de que un dato pertenezca a un grupo se realiza una suma ponderada de las posiciones de los datos y donde la ponderación indica la probabilidad de que un dato pertenezca a cierto clúster o grupo. Por lo tanto, cada punto de datos se asigna al clúster más probable, basado en la probabilidad de que pertenezca a cada gaussiana.

4. **Iteración:** Los pasos 2 y 3 se repiten de forma iterativa hasta que se alcanza un criterio de convergencia, como la maximización de la verosimilitud de los datos.

Para lograr resultados óptimos el proceso dos y tres deben repetirse de forma iterativa hasta que la respuesta no varie.

El proceso de optimización del algoritmo GMM busca maximizar la verosimilitud de los datos dados los parámetros del modelo. En otras palabras, busca encontrar los parámetros de los componentes gaussianos que mejor expliquen los datos observados.

El algoritmo Gaussiano presenta una mejoría sobre K-Means, ya que en este último los datos se agrupan considerando una forma circular del clúster siendo el

dato más lejano el límite del círculo y siempre va a buscar elaborar agrupaciones en forma circular, por tanto, los datos no circulares no se agrupan de forma adecuada, a su vez K-Means presenta problemas cuando trata de agrupar.

1.4. Algoritmo de Asignación de Recursos

El algoritmo de asignación de recursos es un logaritmo que permite calcular la distancia entre nodos de un grafo, considerando como referencia los nodos vecinos de un nodo en particular, además permite determinar cual será el comportamiento de los enlaces del grafo.

Esta clase de algoritmos se los considera capaces de detectar bloqueo mutuo; este bloqueo ocurre cuando dos procesos solicitan recursos y ninguno de los dos puede procesar esos recursos y ambos esperan que el otro proceso suelte el recurso para poder continuar, por lo tanto hay un bloqueo mutuo. Los algoritmos de asignación representan de forma gráfica los recursos asignados a cada uno de los procesos indicando que procesos no necesitan recursos para que sean ejecutados [34].

Este algoritmo además es capaz de realizar un recorrido a través del grafo donde revisara las conexiones entre nodos, lo cual facilita obtener información precisa acerca de los recursos y procesos involucrados además de detectar si el recorrido del grafo se encuentra en espera circular, lo que indicaría que existe un bloqueo [35].

1.4.1. Definición de Grafos

En el ámbito de las matemáticas y la informática, un grafo es una estructura formada por un conjunto de elementos llamados vértices o nodos, que están conectados por vínculos llamados aristas o arcos. Los gráficos se utilizan para representar conexiones binarias entre los elementos de un conjunto y se examinan dentro del campo de la teoría de grafos [36].

De manera general, un grafo se representa de forma visual trazando puntos e interconectándolos con trazos en forma de línea o arco. Desde un punto de vista práctico, los grafos ayudan a analizar las relaciones entre varias unidades. Por ejemplo, una red informática se puede representar y evaluar mediante un grafo, donde

los vértices simbolizan los dispositivos y los bordes las conexiones, ya sean cableadas o inalámbricas [36].

Los grafos se pueden emplear para representar una amplia gama de problemas y su estudio abarca tanto las ciencias precisas como las sociales [37].

En resumen, un grafo se representa como un diagrama con puntos o círculos para los vértices, conectados por trazos en forma de líneas o curvas para las aristas. Los grafos juegan un papel crucial en el estudio de las matemáticas discretas [37].

1.4.2. Tipos de Grafos

- **Grafo Simple:** Este es un tipo de grafo que con una sola arista conecta dos vértices específicos. En otras palabras, cada par de vértices está unido por una única arista. Esta es la definición clásica de un grafo [38].
- **Multigrafo:** Son grafos que permiten la existencia de múltiples aristas entre dos vértices. Estas aristas adicionales se conocen como aristas múltiples o lazos. Los grafos simples son una subcategoría dentro de esta clase de grafos. Además, a estos grafos se les denomina también grafos no dirigidos [39].
- **Grafo Dirigido:** Son aquellos grafos en los que se ha incorporado una dirección a las aristas, la cual se muestra de forma gráfica mediante flechas [38].
- **Grafo No Dirigido:** Es un grafo sin un sentido determinado ya que las líneas o arcos que unen los nodos están dispuestas en ambas direcciones [39].

1.5. Algoritmo de Enrutamiento

El problema general de enrutamiento es uno de los principales problemas usados en ingeniería [40]. Este problema consiste en encontrar la mejor ruta o camino entre un nodo origen y un nodo destino en una red de nodos que están interconectados entre si. La cantidad de enlaces posibles entre los diferentes nodos puede diferir entre cada problema de aplicación y las restricciones que cada escenario puede tener [40].

Entre los diversos campos de aplicación puede ser el de una red de computadoras, una red de transporte o cualquier otra red que pueda representarse

como un grafo. La "mejor ruta"puede definirse de diversas maneras, como la ruta más corta en términos de distancia, tiempo, costo o una combinación de estos factores. Este costo se determina mediante el conjunto de operaciones que asignan un valor a una determinada ruta o distancia, definidas como métricas [41].

En el problema de enrutamiento general se define la necesidad de que un flujo de información pueda ir desde un nodo de origen s a un nodo de destino t. Para esto, es necesario que exista un grupo de nodos o vértices y posibles conexiones o enlaces entre estos siendo e_i el i-ésimo enlace. La mejor ruta que se elija será aquella que tenga un costo menor, el mejor será aquel camino que permita llegar a uno o n puntos para llegar a su destino [42] donde n puede no ser la totalidad de nodos en el escenario [41].

Para el cálculo de las métricas de red, puede también conserarse el número de saltos que se debe dar desde un nodo para ir a otro, los resultados que arroja este proceso no siempre serán los óptimos, pero se consideran buenos resultados lo que hace de este algoritmo muy difundido en aplicaciones. Otras de las métricas es el cálculo de retardo entre nodos, con la particularidad que la distancia entre nodos expresada en tiempo no es constante y es de naturaleza variante, ya que depende del tráfico existente en cada ruta [43].

1.5.1. Características de los Algoritmos de Enrutamiento

Los algoritmos de enrutamiento deben considerar una serie de características, a partir de las cuales se puede elegir un algoritmo para obtener una solución [41], [43].

- **Óptimo:** Se refiere a aquella capacidad que presenta el algoritmo para calcular una métrica que le permita obtener la mejor ruta. Cada software de enrutamiento cuenta con sus propias métricas y algoritmos destinados al cálculo de rutas.
- **Sencillez:** Los algoritmos de enrutamiento deben ser de fácil manejo es decir deben tratar de ser lo más sencillos que sea posible sin descuidar la eficiencia y funcionalidad del software, es de suma importancia considerar que el tiempo para procesar información debe ser el menor posible.
- **Robusto:** El algoritmo debe tener la capacidad de solucionar problemas e imprevistos cuando cae un enlace o hay cambios en las topologías, además

de estar en la capacidad para operar de forma apropiada y estable frente a sobrecargas de red.

- **Rápida Convergencia:** Se entiende como convergencia de un algoritmo a aquella capacidad para establecer rutas estables con rapidez, además de identificar posibles problemas o cambios en la red y tener una rápida reacción ante los mismos. Se dice que al poseer una convergencia lenta se producen caídas en la red o loops.

- **Flexible:** Los problemas que pueden presentarse al ejecutar el algoritmo lo obligan a adaptarse con rapidez frente a posibles eventos, como lo son el Ancho de Banda disponible, los retardos en la red y por lo tanto el tamaño de cola, esto determina la eficiencia del algoritmo.

Bajo estas características cada nodo debe tener cierta inteligencia y realizar en cada nodo procesos independientes de manera que no exista un nodo central, también se trata de un sistema asíncrono, de manera que la información se procesa cuando se transmite y llega información [43].

1.5.2. Propiedades de un Algoritmo de Enrutamiento

Aquellos algoritmos de enrutamiento que funcionan en forma correcta y son bastante estables, cumplen con las siguientes propiedades [44]:

- **Corrección:** El algoritmo debe encontrar la manera de que el paquete llegue al nodo de destino.

- **Simplicidad:** El algoritmo necesita dar soluciones sencillas, lo cual facilita mucho cuando se trabaja con grandes cantidades de datos y aquellos que cuentan con protocolos simples son los más utilizados.

- **Robustez:** Es aquella capacidad del algoritmo de comportarse en forma adecuada ante posibles problemas.

- **Estabilidad:** El algoritmo debe funcionar sin problemas durante su ejecución, en caso de existir alguna caída de nodo, un usuario ya no forma parte de la red, etc. El algoritmo debe recalcular las rutas y el algoritmo continúa funcionando.

- **Gestionabilidad:** Cuenta con información de los procesos que realiza la red para que cuando ocurran eventos pueda solucionar los mismos.
- **Escalabilidad:** El funcionamiento óptimo debe mantenerse a pesar de que la red crezca o cambie su topología .

1.5.3. Algoritmo de Dijkstra

El algoritmo de Dijkstra se ha ganado un lugar como una herramienta crucial para hallar la ruta más corta entre los nodos de un grafo, motivo por el cual este algoritmo se lo conoce como "Algoritmo de caminos mínimos", las soluciones de este algoritmo se generan a manera de un árbol que contiene los caminos de menor trayecto [45].

Para ejecutar el algoritmo de Dijkstra es necesario especificar el nodo de origen como punto de partida y el algoritmo se encarga de crear una ruta, proceso para el cual explora todos los posibles caminos y llevando un registro de las distancias de menos trayecto hacia cada nodo, una vez que el algoritmo selecciona el camino mínimo hacia el nodo lo marca como "visitado" y es agregado al camino solución [45].

1.6. Librerías en Python

Python se establece dentro de los Sistema de Información Geográfico (SIG) por ser de fácil uso y amigables para el usuario por tal motivo se ha posesionado como un lenguaje de uso común. Python utiliza librerías como un componente esencial del lenguaje de programación porque permiten acceder a múltiples funcionalidades, se las cataloga también como una colección de funciones y métodos encargadas de la manipulación de información, acceder a bases de datos, una serie de operaciones matemáticas, entre otras [46].

Como por ejemplo la librería Folium que permite que Python manipule los datos y Leaflet permite visualizar los datos, las dos se constituyen como bibliotecas de código abierto (open source), cuyo propósito es crear mapas interactivos que puedan ser visualizados, mediante el uso de las bibliotecas Javascript y Leaflet permite que

estos mapas sean escalables o integrables a sitios web.

1.6.1. Librerías Folium y Leaflet

Folium es una herramienta de visualización y permite la creación de mapas interactivos mediante una serie de Datasets, los cuales contienen un conjunto de información y datos ordenados (en este caso información cartográfica), en forma de tabulación, crea una conexión con Python, como lenguaje de programación, el mismo que permite editar y manipular la información obtenida y con el uso de la biblioteca Leaflet se crea y visualiza la cartografía y hace los mapas interactivos [47].

La información que contienen los datasets está organizada de forma estructurada en forma de tabulación dentro de una base de datos, que organiza los datos con los que se desea trabajar o se va a realizar cierta tarea. Poseen filas y columnas, las filas le corresponden a las variables y las columnas a los datos. Las características de los datasets y sus prestaciones lo establecen como una base de datos de origen [48].

Por otro lado, Leaflet es una librería JavaScript que cuenta con la capacidad publicar mapas en la web sin mayor dificultad. Forma parte de las librerías de Python para visualización geoespacial de datos. Gracias a su fácil manejo y por no ser una librería pesada se ha posesionado como una herramienta ideal para realizar procesos de cartografía en la web permitiendo obtener datos geo espaciales [49].

Permiten localizar en un mapa las diferentes locaciones, además permite desplegar y visualizar un gran número de objetos, como por ejemplo desplegar un mapa interactivo de la tasa de criminalidad, elaborado a partir de un dataset que contiene datos como la tasa de criminalidad y delitos que ocurren en cierta área urbana, país, etc [47].

1.6.2. Usos de Folium

Permite ubicar en un mapa una serie de diferentes locaciones, además permite desplegar y visualizar un gran número de objetos, como por ejemplo desplegar un mapa interactivo de la tasa de criminalidad, elaborado a partir de un dataset que

contiene datos como la tasa de criminalidad y delitos que ocurren en cierta área urbana, país, etc [50].

1.6.3. Mapas Interactivos

Un Mapa interactivo se define como el desarrollo y representación cartográfica realizada en medios computacionales, permiten al usuario interactuar con el mapa, es decir permite al usuario desplazarse, hacer zoom, activar o desactivar capas y demás posibilidades de interacción [50].

Así, a través de trazos digitales se establecen vínculos entre datos y técnicas de representación gráfica, gracias a estos procesos es posible elaborar mapas con múltiples funciones y prestaciones; buscando ubicar o geo referenciar puntos por medio del internet [51].

1.6.4. Características de los Mapas Interactivos

Los mapas interactivos permiten ajustar la visión según la necesidad del usuario, es decir permite hacer zoom y así poder manipular las escalas del mapa para mejorar la visualización de los puntos de interés y así observar con mayor detalle nombres de calles, avenidas y demás sitios. Dispone al usuario la capacidad de crear mapas en múltiples capas y así almacenar diferente información en cada capa, lo que faculta el hecho de visualizar distintos relieves, tiempo, humedad y pronósticos del clima para los días venideros. Además, facilitan la colocación de marcadores en forma de ícono, puntos líneos o delimitar áreas, dado la posibilidad de ubicar lugares, usuarios, entre otros; toda esta información puede presentarse de forma gráfica, textual, multimedia, hiperenlace, etc [51].

1.6.5. Librería GeoPy

La librería de Python GeoPy permite convertir direcciones o ubicaciones determinadas en coordenadas geográficas, considerando una ubicación real en la superficie terrestre es decir hacer procesos de geocodificación, para poder plasmarlas en representaciones cartográficas. Además de facilitar la rápida búsqueda e

identificación de puntos de interés, como por ejemplo determinados negocios, parques nacionales, entre otros, para lograr este cometido GeoPy trabaja con sistemas de coordenadas geográficas como son la latitud y la longitud o puede hacer uso de diversos sistemas de referencia como el sistema de referencia de cuadrícula militar [46].

La librería GeoPy proporciona clases de geo codificación para los siguientes proveedores de servicios geográficos: OpenStreetMap, Nominatim, ESRI ArcGIS, Google Geocoding API (V3), entre otros [52].

1.6.6. Librería Scikit-Learn

Es una librería de Python de código abierto y aprendizaje automático que permite acceder a una gran variedad de algoritmos comunes, como algoritmos para la clasificación, algoritmos de regresión, algoritmos de clusterización y algoritmos de dimensionalidad.

La librería scikit-learn cuenta la característica de ser compatible con varias librerías en Python como Numpy, matplotlib, SciPy, lo cual facilita el trabajo en diversos módulos y algoritmos, facilita además la extracción de datos o información de bases de datos, constituyendo como una librería que facilita el análisis de datos y de patrones [53].

Capítulo 2

Dessarrollo de la Metaheurística

En este capítulo se aborda la selección de las heurísticas usadas en la metaheurística resultante de este trabajo. Se exponen los diagramas de flujo de cada etapa elaborada para conseguir una asignación de usuarios a busetas considerando la capacidad máxima de esta.

Entonces, a partir de aquí, se desarrolla la metaheurística de asignación, basada en la heurística de agrupamiento GMM, como punto de partida para reducir el problema de optimización y así mediante una serie de ajustes poder mejorar la solución al problema de asignación. Como paso final se busca calcular la ruta de menor distancia de cada grupo desde el usuario más lejano hasta el más cercano a la unidad educativa.

Cabe destacar que, para simplificar el proceso de asignación se asume que la demanda no excede a la oferta de espacios disponibles en total. Para lograrlo se plantea que en caso de que el número de usuarios supere la capacidad máxima se descartarán los usuarios más lejanos hasta igualar la demanda de usuarios con la capacidad.

2.1. Implementación de Algoritmos para la Creación de Mapas

Para el despliegue del mapa interactivo que indique la ubicación geográfica de los estudiantes, se utilizará la librería de visualización geográfica Folium y la librería

Pandas que permite la manipulación y análisis de datos, se debe considerar que la base de datos en la cual se maneja la información de cada estudiante estará en una hoja de Excel, organizada de tal manera que indique el nombre completo del estudiante, dirección del domicilio y sobre todo su ubicación geográfica, cabe recalcar que las coordenadas son de suma importancia para poder ubicar en el mapa a cada estudiante.

Con el uso de las ya mencionadas librerías el proceso se vuelve bastante sencillo, como primer paso se debe llamar y cargar el archivo de Excel y leer los datos de la hoja de cálculo de Excel, la misma que contiene la información detallada de los estudiantes y los carga en una *DataFrame* (Estructura Tabular que organiza datos en filas y columnas).

Una vez cargada la información se procede a crear el mapa con el comando: folium.Map, se indican las coordenadas de la ubicación en la que se desea crear el mapa, en este caso las coordenadas de la ciudad de Cuenca Ecuador y se indica el nivel de zoom en el que se visualizará el mapa. Para poder usar los datos de un *DataFrame* se utiliza una estructura encargada de recorrer cada fila de un DataFrame, identificando el índice de cada fila y los datos contenidos en cada fila, con el fin de poder realizar operaciones específicas con cada fila del conjunto de datos, se crea la estructura datos capaz de contener cualquier tipo de datos, en este caso aquí se van a almacenar los datos con coordenadas y nombres de los estudiantes. Una vez recorridas las filas se identifican los valores almacenados en cada columna del *DataFrame* e ir organizando los datos tanto de las filas como de las columnas y permite que los nuevos datos vayan al final de la lista, para de esta manera organizar los datos del *DataFrame* y que no se sobrescriban.

Para finalizar se indica la columna de Excel donde se ubica las coordenadas, separar el valor de latitud y longitud, se agrega el nombre del estudiante a esa coordenada y se crea un marcador en cierta ubicación, se añade el nombre del estudiante y se añade al mapa, por último, se guarda el mapa.

2.2. Implementación del Algoritmo Gaussiano

Usar un algoritmo de agrupamiento significa que le dará al algoritmo una gran cantidad de datos de entrada sin etiquetas y le permitirá encontrar cualquier agrupamiento en los datos que pueda. En este procedimiento la agrupación se realiza en base a una serie criterios. Para este trabajo los criterios son distancia y similitud. Entonces, para la creación de clústeres se utilizará una serie de algoritmos comunes, no solo por su fácil manejo sino por su alto rendimiento y eficacia al momento de agrupar, la agrupación en clústeres es una tarea de aprendizaje automático no supervisada.

De acuerdo con la literatura revisada en el capítulo anterior, el algoritmo GMM fue elegido para realizar la primera fase, la agrupación. Esto se debe a que GMM presenta una mejor agrupación frente a elementos dispersos y que no sea necesario mantener una relación circular con respecto al centroide de cada agrupación definida, dando una mejor solución frente a elementos dispersos sobre otros algoritmos como K-means o K-medoids que dejan de lado a los elementos dispersos o sesgados. Otro motivo para descartar a estos dos últimos algoritmos es debido a que, para un mismo escenario, al ejecutar varias veces el algoritmo de asignación, no siempre se consigue una respuesta única. El algoritmo BIRCH fue descartado por el hecho de que no se puede elegir el número de grupos, para este programa el número de grupos debe ser igual al número de busetas. Por tal motivo el algoritmo de agrupamiento electo para este trabajo es GMM. El proceso de agrupamiento permite que los estudiantes que vivan cerca unos de otros conformen un grupo y este grupo será asignado a una de las busetas escolares, respetando su capacidad, una vez conformados los clústeres se calculará la ruta de inicio y llegada a la institución.

El resultado obtenido lo constituye una serie grupos conformados por los datos y coordenadas de los estudiantes, cabe recalcar que se debe especificar el número de clústeres que se desea, para en base a este dato hacer las agrupaciones, para este trabajo el número de busetas disponibles de la institución es igual al número de clústeres.

Para aplicar el algoritmo GMM se utilizó las librerías “sklearn.mixture”. Una vez cargada la librería aplicamos el algoritmo de mezcla Gaussiana a todas las coordenadas de ubicación de los estudiantes, extraídas con anterioridad en una

hoja de Excel, por último, se utilizan comandos que permite ajustar los clústeres, obteniendo datos como la media, la covarianza y obtener el índice del clúster, además, se calcula el centroide de cada clúster, el cual es esencial para los ajustes que se harán a futuro.

2.3. Aplicación de la Metaheurística para la Asignación

La figura 2.1 muestra el diagrama de flujo sobre el funcionamiento general del programa de asignación.

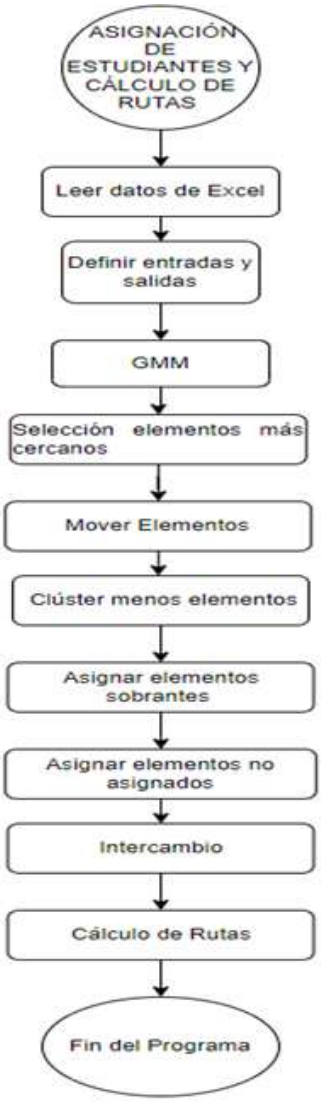


Figura 2.1: Diagrama de flujo de todo el Programa.
Fuente: Autor

Una vez aplicado el algoritmo de agrupamiento surge el problema de que

no se respeta la capacidad de las busetas, debido a que la distribución de datos no es equitativa, por lo tanto es necesario realizar una metaheurística que consiste en adicionar una serie de pasos que harán óptima la asignación y sobre todo permite respetar la capacidad de la buseta escolar.

2.3.1. Ordenamiento de Datos y Asignación de Busetas a los Clústeres

La figura 2.2 muestra el diagrama de flujo sobre el primer proceso de asignación, en el cual se asigna una Buseta a cada clúster.

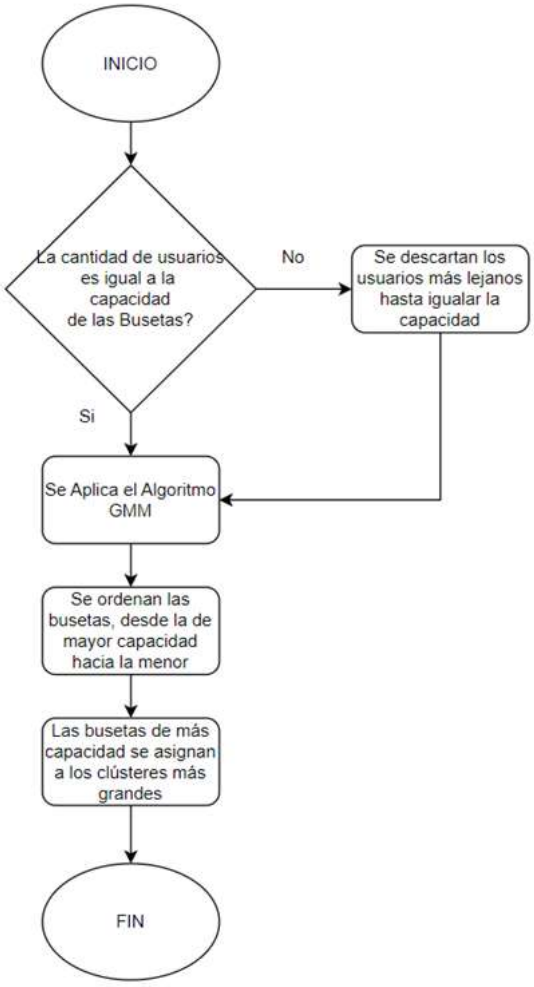


Figura 2.2: Diagrama de flujo de la asignación de busetas a los clústeres.
Fuente: Autor

Como primer ajuste a los clústeres, se crea un *DataFrame* con las coordenadas de

los usuarios, el nombre del usuario y el índice asignado en la hoja de Excel al usuario, con el fin de poder manejar la información con mayor facilidad y además se crea un respaldo de la información independiente de cualquier modificación. Para empezar, se identifica al clúster de mayor cantidad de elementos y se le asigna a la buseta con la mayor capacidad de espacio disponible.

2.3.2. Asignación de Busetas a los Clústeres

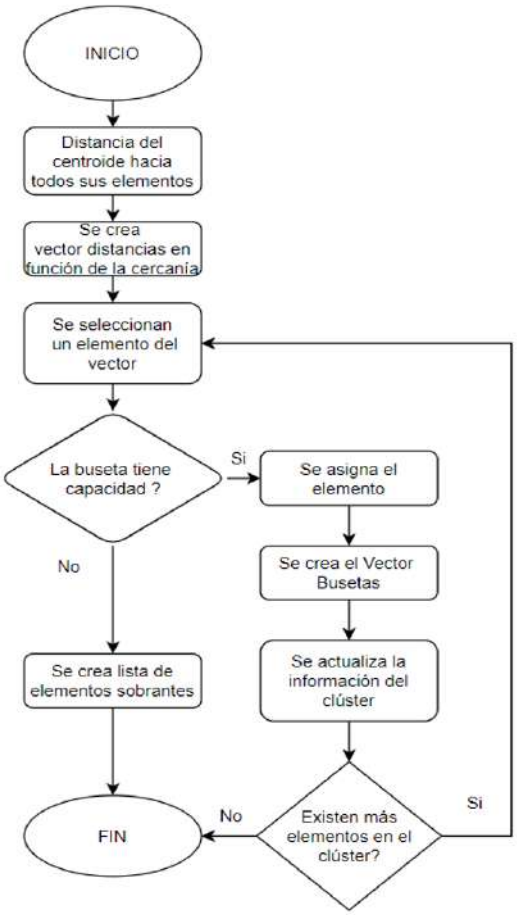


Figura 2.3: Diagrama de flujo de la asignación de busetas a los clústeres.
Fuente: Autor

Una vez asignada la buseta se procede a calcular una matriz de distancias desde el centroide hacia todos los elementos de su clúster, este paso se itera para todos los clústeres.

En base a las distancias obtenidas en cada clúster, se asignarán los elementos más cercanos al centroide de cada clúster, para este paso se considera la capacidad de

la buseta y se asignarán los elementos más cercanos siempre y cuando la cantidad de usuarios asignados sea menor o igual a la capacidad máxima de la buseta, este proceso se itera para todos los demás clústeres.

Como resultado se obtiene el Vector Busetas, el cual contiene un arreglo que muestra todos los usuarios asignados a cada clúster, además indica el índice del usuario y la numeración del clúster al que pertenece, este vector contiene la asignación de estudiantes a cada buseta. Estos mismos datos se muestran para todos los demás clústeres.

Tomar en cuenta que aquellos usuarios que no han sido asignados debido a que la buseta ya no cuenta con espacio para atender a estos usuarios, se almacenan en el vector denominado como “elementos sobrantes” y los cuales serán asignados en los próximos ajustes.

2.3.3. Mover Usuarios en Caso de Encontrar un Clúster más Cercano

En este proceso lo que se busca es mover un elemento que ya fue asignado con el propósito de encontrar un clúster más cercano que le pueda dar servicio, es decir que tenga capacidad y además tratar de hacer espacio para asignar los elementos sobrantes, que corresponden a la lista de estudiantes no asignados por falta de capacidad. La figura 2.4 muestra el diagrama de flujo sobre la primera modificación en el proceso de asignación, en el cual se asigna un usuario asignado busca un clúster que le resulte más cerca.

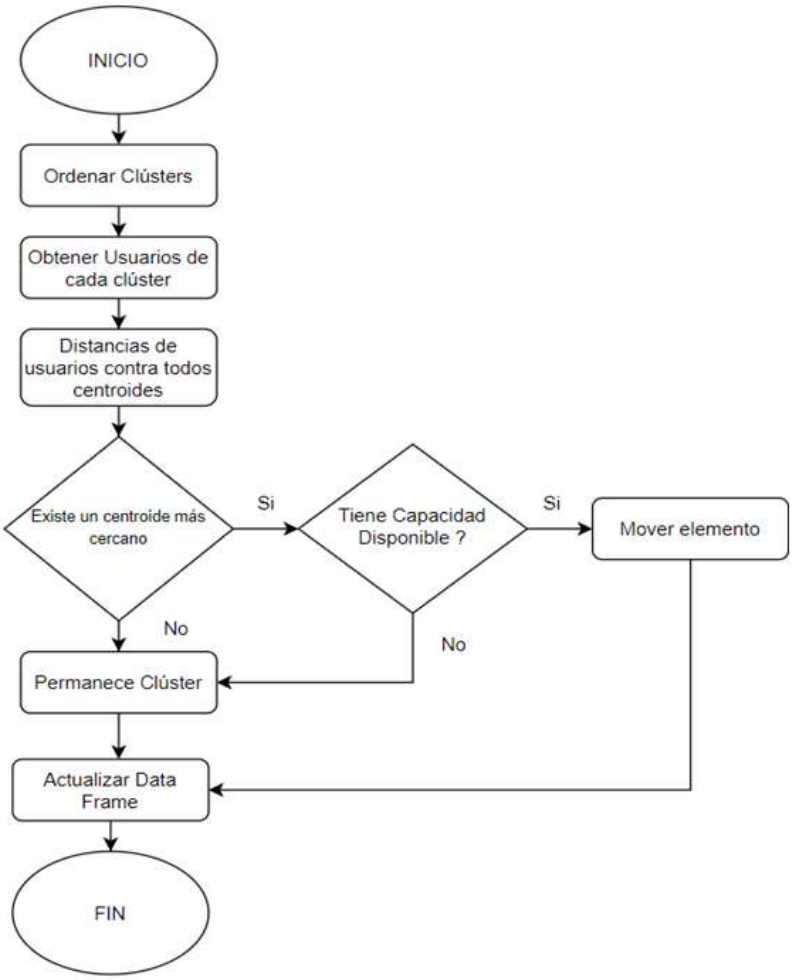


Figura 2.4: Diagrama de flujo obre el movimiento de usuarios.
Fuente: Autor

Para realizar este movimiento se define una función que contiene el vector Busetas, la lista de centroides de cada clúster y el *DataFrame* que contiene información de los estudiantes como sus índices, el primer clúster en ser analizado es aquel que contenga mayor número de elementos, el cálculo empieza ubicando al usuario más lejano del clúster hasta el más cercano, para estas mediciones se utiliza la función "geodesic" la cual calcula las distancias considerando la curvatura de la tierra, está información se almacena en una lista llamada "elementos a mover", a continuación se calcula la distancia de cada uno de los elementos dentro de la lista contra todos los demás centroides.

Para que el usuario cambie de clúster la distancia de ese usuario hacia el centroide de destino debe ser menor que la distancia hacia el centroide actual y además el clúster de destino debe tener capacidad para recibir al usuario, si cumple con estas

dos condiciones el elemento cambia de clúster, la capacidad de la buseta y la distancia del usuario hacia el centroide de destino son determinantes para el movimiento del usuario.

Para tener control sobre la cantidad de usuarios que hay asignados a cada buseta se crea un DataFrame, el cual indica la cantidad de usuarios asignados a cada buseta, el número de clúster, el número de Buseta e indica cuantos usuarios faltan para completar la capacidad de la buseta.

2.3.4. Equilibrar Clúster con Menor Cantidad de Elementos

Para este paso lo que se busca es llenar con mayor cantidad de elementos a aquel clúster que tiene la menor cantidad de elementos y así poder contar con mayor capacidad para asignar a aquellos elementos sobrantes. Para cumplir con este propósito se fija una distancia de restricción que comienza desde el centroide, la distancia después de probar con varias distancias se fijó en uno punto cinco kilómetros, ya que se ajustaba mejor al tamaño del escenario al que se aplicó, en este caso la ciudad de Cuenca-Ecuador. La figura 2.5 muestra el diagrama de flujo sobre el cual el clúster con menos elementos toma usuarios que esten dentro de su área de restricción.

Primero se debe ubicar la buseta con menor cantidad de elementos, una vez encontrado el clúster de menos elementos se creará un vector donde se almacenarán los datos llamado “buseta menos elementos”, se procede a crear el diccionario de distancias hacia los centroides, que será el encargado de almacenar la distancia de cada elemento hacia el centroide del clúster de menos elementos, se le asignarán todos los elementos que estén ubicados en una distancia menor o igual a la distancia de restricción.

Se realizó un ajuste adicional, en caso de que los usuarios caigan dentro de varias distancias de restricción, estos usuarios siempre serán asignados al clúster con menor cantidad de elementos, para verificar que se cumpla con la capacidad de la buseta se verifica que la cantidad de elementos sea menor o igual a la capacidad de la buseta.

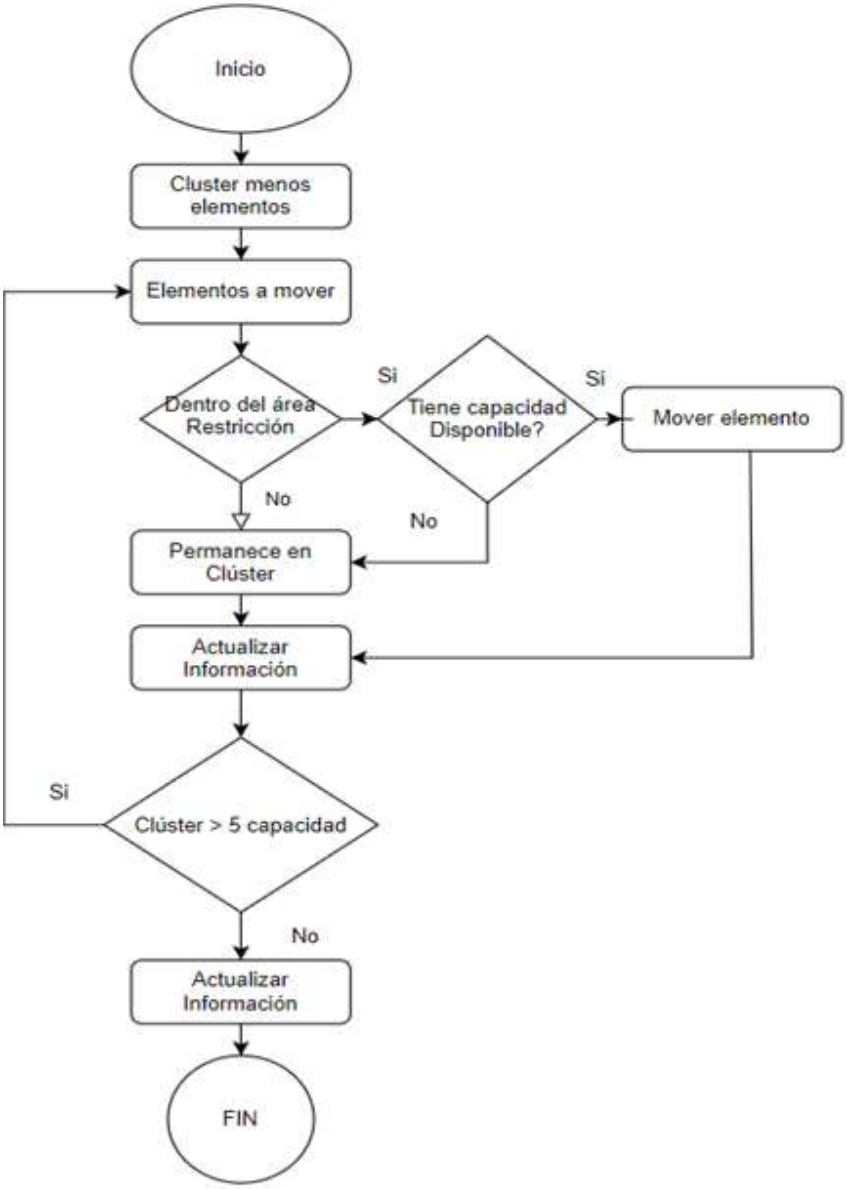


Figura 2.5: Diagrama de flujo sobre la asignación de usuarios al clúster de menos usuarios.

Fuente: Autor

Antes de finalizar se trata de localizar un clúster al que le falten más de cinco elementos, en caso de existir ese clúster, el proceso se repite, asignando a este clúster aquellos elementos que se encuentren dentro de su distancia de restricción y que no exceda su capacidad máxima.

Por último, se actualiza el *DataFrame* que contiene la información de la buseta indicando que elementos fueron removidos y reasignados.

2.3.5. Asignación de Elementos Sobrantes

La figura 2.6 muestra el diagrama de flujo sobre la asignación usuarios que no han sido asignados con anterioridad a clústeres que estén dentro de su área de restricción y tengan capacidad.

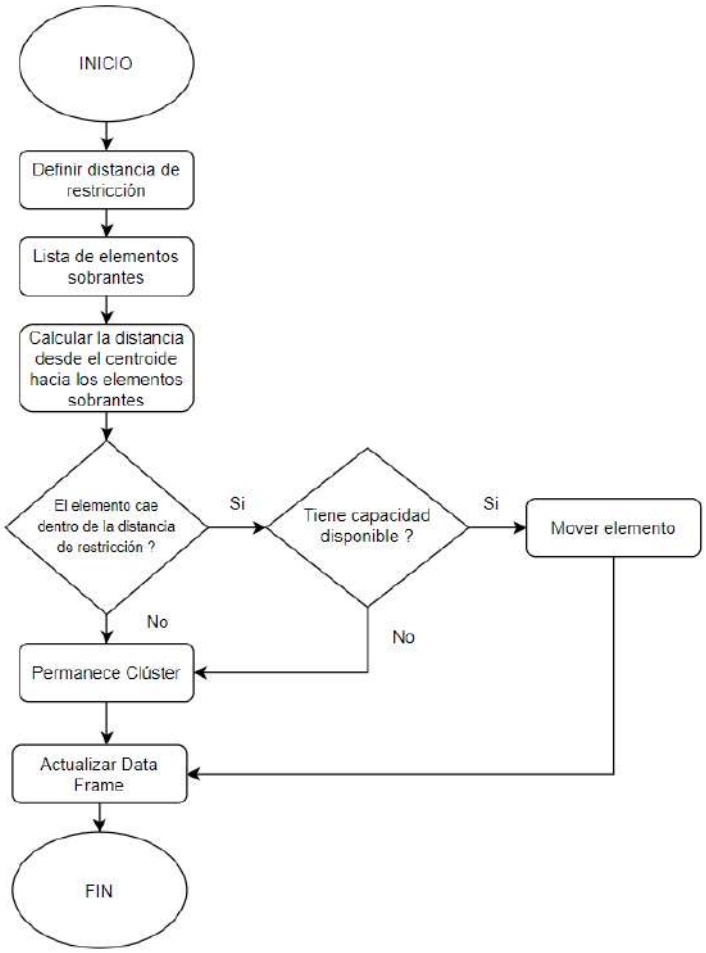


Figura 2.6: Diagrama de flujo sobre la asignación de usuarios sobrantes según la distancia de restricción.

Fuente: Autor

Cabe recalcar que en este proyecto se denominan elementos sobrantes a aquellos usuarios que no fueron asignados debido a que su clúster más cercano tiene la capacidad llena, por lo tanto, los pasos anteriores tenían la intención de generar el espacio necesario para poder asignar la mayor cantidad de elementos sobrantes o todos, en caso de ser posible.

Para ejecutar está asignación de nuevo se trabajará con una distancia de restricción de uno punto cinco kilómetros, se calcula la distancia de cada elemento

sobrante contra cada centroide y se almacena en un diccionario, se procede a identificar a aquellos elementos cuya distancia sea menor o igual a la distancia de restricción y además que cuya capacidad no exceda la capacidad máxima de la buseta, si cumple con esos requisitos el usuario es asignado y es eliminado de la lista de elementos sobrantes, caso contrario permanece en la lista de elementos sobrantes para ser asignado en pasos posteriores.

Para finalizar se procede a actualizar el DataFrame, el mismo que sirve para verificar cuanta capacidad sobra en las busetas, cuantos elementos fueron asignados e indica cuanto espacio hay disponible.

2.3.6. Asignación desde los Elementos Sobrantes hacia los Clústeres

En este paso se procede con la asignación de usuarios sobrantes, pero desde la perspectiva del usuario sobrante o también denominados elementos sobrantes en este trabajo, es decir es ahora el elemento sobrante el que elegirá el clúster al que ir, siempre y cuando se cumplan los siguientes requisitos:

- Que el clúster tenga capacidad para recibir al usuario.
- Que el clúster de destino este comprendido dentro de la distancia de restricción.

Para empezar, se crea un diccionario para almacenar la información de los elementos a mover, con el uso de la función geodesic se calcula la distancia de los elementos hacia todos los centroides, se hace un ordenamiento de la información de forma ascendente, es decir primero la menor distancia hacia la mayor, de esta manera se puede identificar los clústeres más cercanos.

A continuación, se sigue un proceso muy similar al de los pasos anteriores, de la distancia calculada se debe verificar que el usuario se encuentre en una distancia menor o igual a la distancia de restricción y que el clúster tenga capacidad para que pueda ser asignado a ese clúster, en caso de que haya sido asignado se removerá ese usuario de la lista de elementos sobrantes y se actualizará el *DataFrame* con la información de las busetas. La figura 2.7 muestra el diagrama de flujo sobre la asignación usuarios que todavía no han sido asignados a clústeres, aquí los usuarios sobrantes buscan el clúster más cercano con capacidad para recibir al usuario.

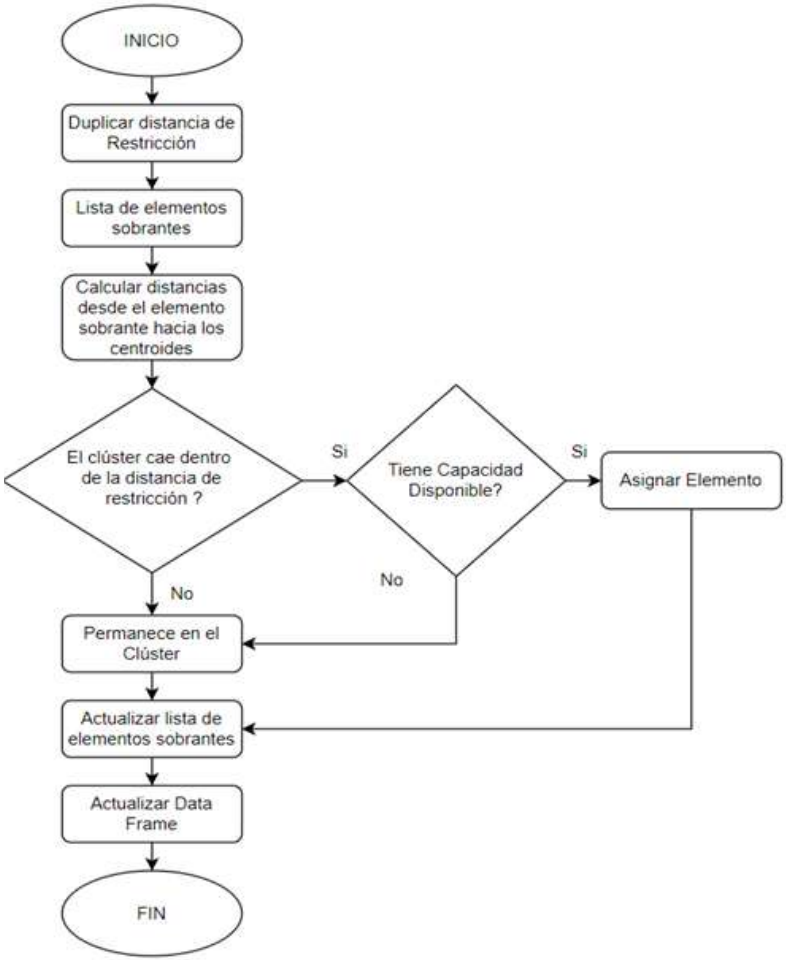


Figura 2.7: Diagrama de flujo sobre la búsqueda de los usuarios sobrantes por un clúster con capacidad

Fuente: Autor

2.3.7. Intercambio de Elementos

Este es el proceso más complejo de la asignación, ya que en este paso se asignarán los elementos sobrantes que no fueron asignados en pasos anteriores, en este proceso se realiza un intercambio de elementos.

Los usuarios sobrantes serán asignados al clúster más cercano y este clúster cederá elementos al siguiente clúster más cercano y este a su vez trata de intercambiar elementos con el siguiente clúster más cercano, hasta llegar al clúster con mayor capacidad o menor número de usuarios, siempre el intercambio está orientado hacia el clúster de menos usuarios, para poder intercambiar los usuarios con aquellos clústeres que tengan mayor capacidad y así terminar de llenar todos los clústeres.

La figura 2.8 muestra el diagrama de flujo sobre el intercambio de elementos.

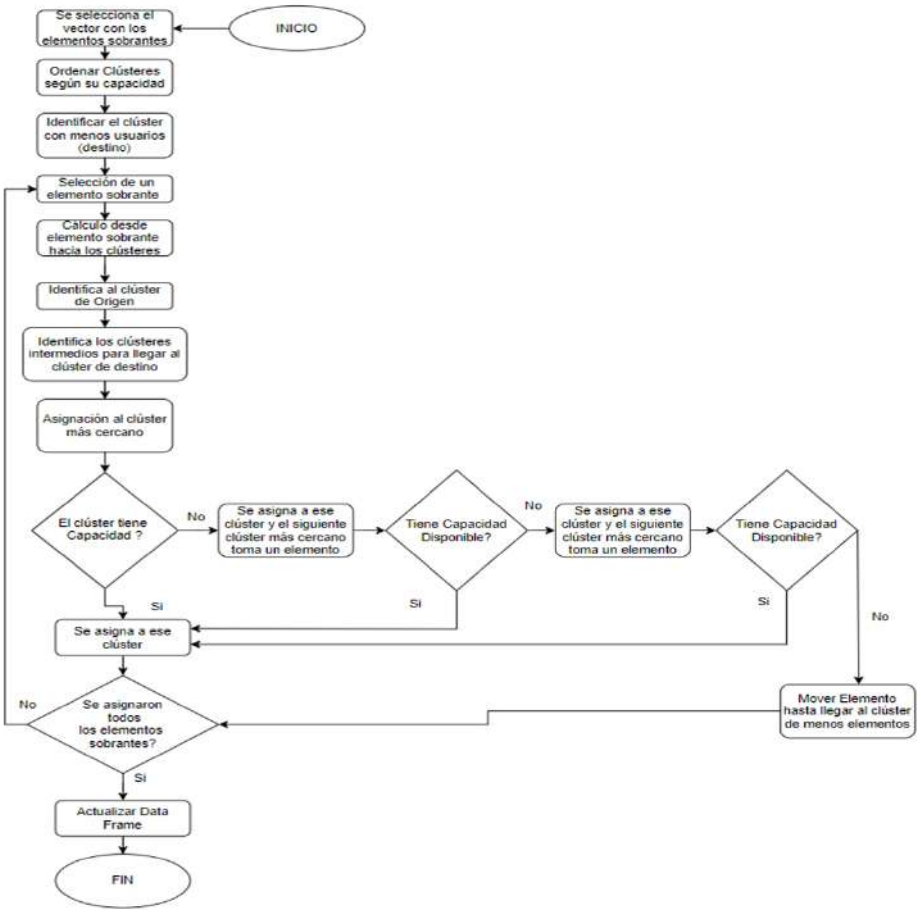


Figura 2.8: Diagrama de flujo sobre el intercambio de elementos.
Fuente: Autor

Primero se empieza creando la función para los usuarios sobrantes, la cual contiene la información del *DataFrame* con los índices y número de clúster de los elementos sobrantes, se procede a calcular la distancia de los elementos sobrantes hacia los demás centroides, se calcula utilizando la función geodesic y se ordenan las distancias desde la menor hacia la mayor.

Las busetas están ordenadas por capacidad de mayor a menor, se identifica a la buseta de mayor capacidad como clúster de destino y se identifican a las busetas que se encuentran en el camino desde el clúster actual hasta el clúster de menos usuarios o también llamado clúster de destino. Este proceso es necesario para poder intercambiar usuarios desde el clúster con capacidad llena hacia el clúster con menor cantidad de usuarios asignados, pasando por las busetas que queden entre estos dos clústeres.

Una vez identificado el clúster de destino, selecciona un elemento sobrante y se asigna al clúster que le sea más cercano, se verifica con el *DataFrame* de información de

busetas si esté clúster tiene capacidad para recibir al usuario, en caso de que no cuente con capacidad inicia un movimiento secuencial orientado a la siguiente buseta más cercana con capacidad para recibir usuarios, para realizar este movimiento, se calcula la secuencia de busetas a seguir para poder llegar al clúster de destino y asignar los usuarios.

Una vez el usuario es asignado al clúster más cercano, llamado clúster inicial, se verifica la capacidad del clúster inicial, en caso de que este clúster no disponga de capacidad, el siguiente clúster más cercano al clúster inicial selecciona el usuario que le resulte más cercano y toma ese usuario, se procede a verificar si este nuevo clúster tiene capacidad para recibir al usuario caso contrario el siguiente clúster más cercano a este selecciona al usuario que le sea más cercano y será asignado a este nuevo clúster, este proceso se repite hasta llegar al clúster de destino.

Se actualizan las estructuras de datos que contiene la información de busetas y la lista de elementos sobrantes, se retorna True si se logra asignar el elemento, o False si no es posible.

Por último, se utiliza un bucle while para verificar la lista de elementos sobrantes, en caso de que queden usuarios sobrantes, estos volverán al proceso anterior hasta ser asignados y se hará este proceso de forma iterativa hasta que ya no queden elementos por asignar.

2.4. Implementación del Algoritmo de Enrutamiento

Para este paso se planteó una resolución basada en teoría de grafos, el algoritmo de Dijkstra ya que permite encontrar la ruta de menor trayecto entre nodos, el punto de partida es un nodo, el cual buscará la ruta de menor distancia hacia otro nodo, una vez hallado la ruta se marca como visitado, este proceso continúa hasta haber visitado todos los nodos del grafo y de está forma se crea un camino que recorre todos los nodos procurando que sea la menor distancia entre los nodos.

La figura 2.9 muestra el diagrama de flujo sobre el funcionamiento de la heurística de enrutamiento.

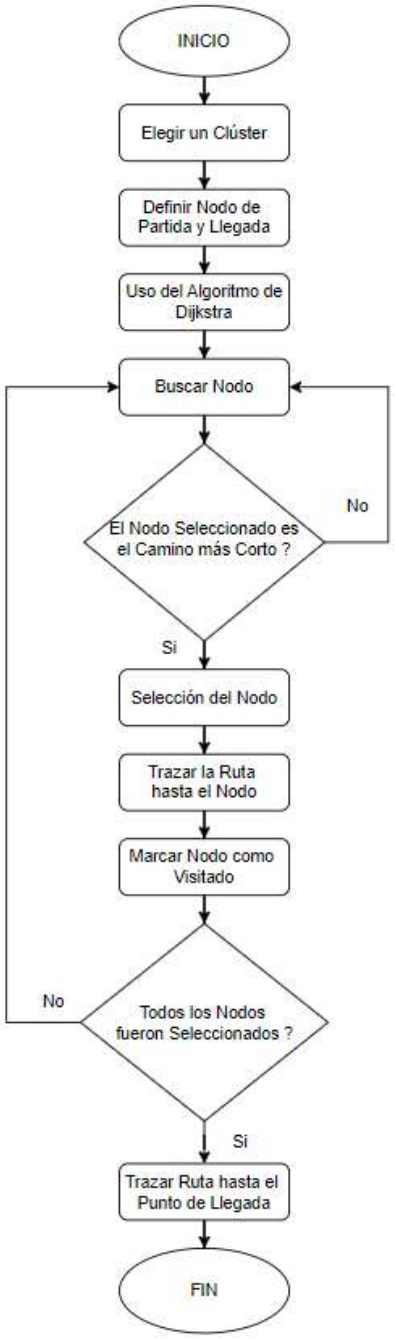


Figura 2.9: Diagrama de flujo sobre la heurística de enrutamiento.
Fuente: Autor

En este proceso se trabajará con cada clúster por separado, se define una función con dos coordenadas, la primera coordenada corresponde al punto de partida y la segunda corresponde al punto de llegada, las distancias serán calculadas haciendo uso de Haversine. El nodo de partida para este trabajo será el nodo más lejano de la unidad educativa.

El proceso continúa calculando las distancias entre cada par de usuarios dentro

del clúster y se crean las listas de ruta que tiene como punto de partida el nodo inicial, además se crea una lista para agregar aquellos nodos que hayan sido visitados.

El proceso continúa mientras la longitud de la ruta sea menor que el número total de usuarios, de esa manera se garantiza que todos los usuarios hayan sido visitados, se genera una lista con aquellos usuarios que aún no han sido visitados, cuando un usuario es visitado se marca como visitado y continúa localizando la ruta más corta hasta llegar al usuario denominado como "último nodo".

Una vez recorridos todos los nodos, se traza la ruta a la unidad educativa y así se conforma la ruta completa, la cual representa la ruta cercana óptima, según el vecino más cercano.

Capítulo 3

Análisis de Resultados

El presente apartado presenta el análisis de resultados en las diferentes fases de ejecución del programa. Considerando las soluciones de cada escenario planteado para el desarrollo del programa.

3.1. Análisis del Algoritmo de Agrupamiento

Para este trabajo en particular el agrupamiento Gaussiano presenta poca volatibilidad en los resultados de la asignación y por lo tanto ofrece una respuesta repetitiva frente a escenarios aleatorios planteados para este trabajo, lo cual hace que la respuesta del programa siempre sea la misma frente a un determinado escenario, sin embargo uno de los problemas presentados es el mismo que presentan otros algoritmos de agrupación y es que no permite tener un control sobre cuantos elementos serán asignados a cada grupo, por lo tanto en este trabajo se evidencio que existen grupos que tienen mayor cantidad de elementos que otros, lo cual para este trabajo representa un problema ya que sobre carga un grupo con demasiados datos y otros grupos tienen muy pocos elementos, lo que obliga a realizar una serie de procesos posteriores para ajustar el agrupamiento.

En las siguientes gráficas se observa los resultados de la mezcla gaussiana aplicado para cada uno de los escenarios, cabe recalcar que se trata de un escenario aleatorio.

Los Resultados de ejecutar el Algoritmo Gaussiano para un escenario de 100

estudiantes se ven la figura, donde todos los elementos fueron asignados, inclusive los elementos dispersos, sin embargo el clúster de color amarillo es el que contiene la mayor cantidad de elementos y el clúster celeste es el que tiene la menor cantidad de elementos, por lo tanto se demuestra que los grupos no son equitativos. Esto se ve en 3.1.

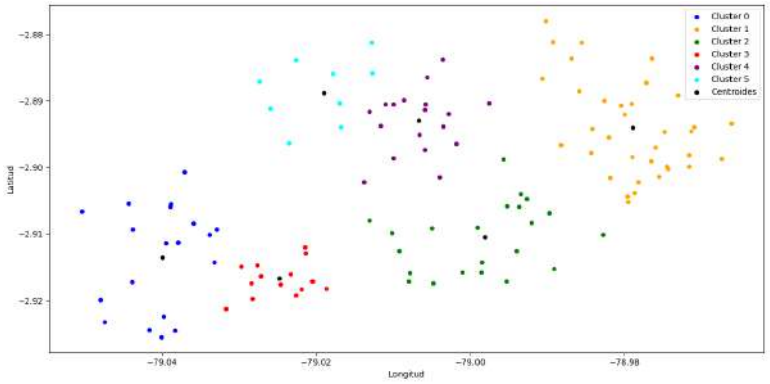


Figura 3.1: Resultados GMM para un escenario de 100 estudiantes.
Fuente: Autor

Se muestra los resultados de aplicar el Algoritmo Gaussiano para un escenario de 160 estudiantes aleatorio, este escenario presenta una mayor dispersión sobre 3.1 y se puede evidenciar como todos los elementos son agrupados con aquellos elementos que para estos sean los más cercanos, Con claridad se puede ver como en clúster de elementos en marrón, llamado clúster siete, apenas tiene elementos, esto debido a que el algoritmo formo un grupo con los elementos más dispersos y mientras que hay otros grupos con varios elementos, esto demuestra que la distribución tampoco fue equitativa, los resultados se ven en la fig3.2.

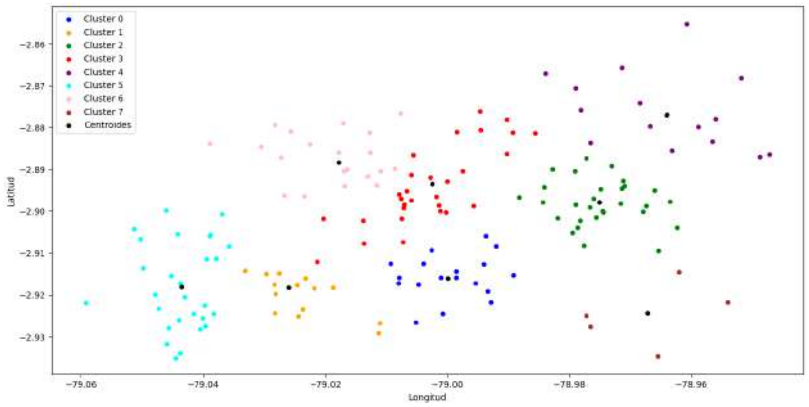


Figura 3.2: Resultados GMM para un escenario de 160 estudiantes.
Fuente: Autor

A continuación se muestran los resultados de aplicar el Algoritmo Gaussiano para un nuevo escenario aleatorio con 160 estudiantes, se demuestra como el algoritmo tiende a incluir los elementos dispersos, como el caso del cluster azul, llamado cluster cero, que incluye un elemento lejano a su grupo, de nuevo existe una distribución poco equitativa, considerando que el clúster seis de color rosado tiene una cantidad mucho mayor de elementos sobre el clúster siete de color marrón en la cual varía la distribución de los usuarios, todo este se ve en 3.3.

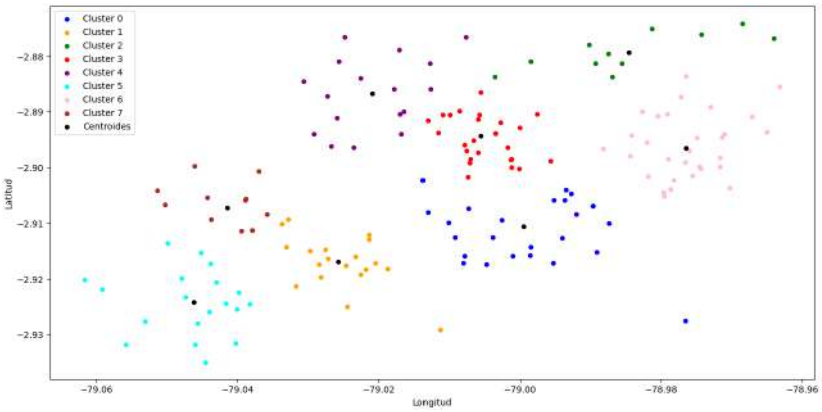


Figura 3.3: Resultados GMM para un escenario de 160 estudiantes.
Fuente: Autor

3.2. Análisis de la Heurística de Asignación

El análisis de resultados en este punto se divide en seis pasos.

3.2.1. Análisis Selección Usuarios más Cercanos

En este proceso se puede evidenciar como la mayor cantidad de elementos sobrantes se ubican cerca del clúster al cual el algoritmo de GMM agrupo con más elementos. En este paso el algoritmo selecciona los usuarios más cercanos del clúster con respecto al centroide del clúster y respetando la capacidad.

Se puede observar que las agrupaciones hechas con GMM no siempre son equitativas ya que hay clústeres que no se acercan a su capacidad máxima.

En la gráfica se evidencia aquellos usuarios que no fueron asignados porque la capacidad de la buseta se llenó, por lo tanto, los elementos de color verde fluorescente corresponden a lo que se denominan elementos sobrantes o usuarios sobrantes.

A continuación se muestra como los clústeres toman los elementos más cercanos a su centroide en función de la capacidad de la buseta que les fue asignada, dejando de color verde fluorescente a los usuarios sobrantes, se presenta los resultados para un escenario de 100 estudiantes, esto ve en 3.4.

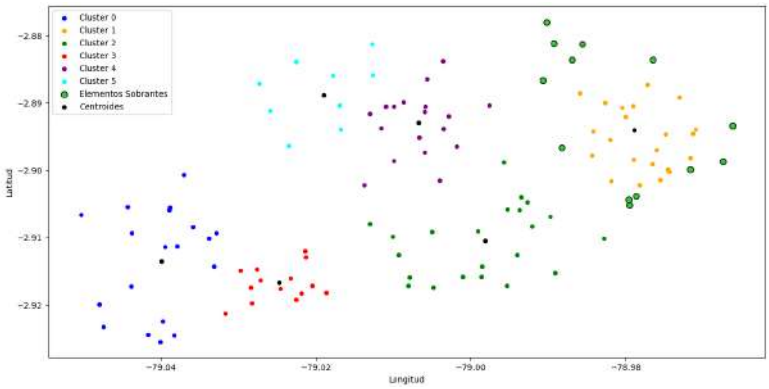


Figura 3.4: Resultados para un escenario de 100 estudiantes.
Fuente: Autor

Para el primer escenario de 160 estudiantes,los clústeres toman los elementos más cercanos a su centroide en función de la capacidad de la buseta que les fue asignado, dejando de color verde fluorescente a los usuarios sobrantes que no fueron asignado por exceder la capacidad de la buseta, estos se ven en la figura 3.5.

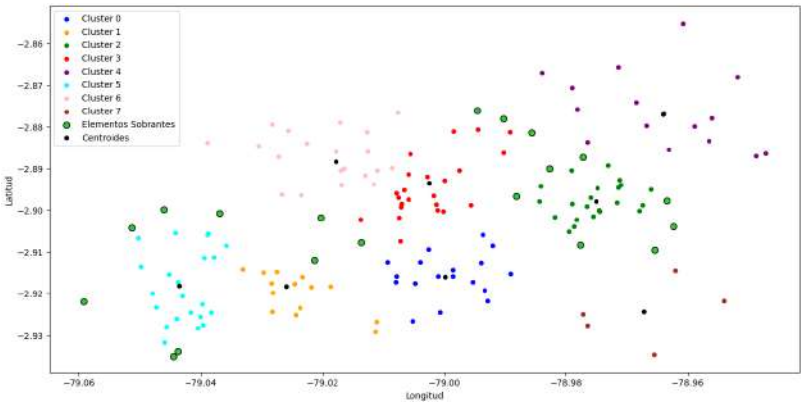


Figura 3.5: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

A continuación se presentan los resultados obtenidos para el segundo escenario de 160 usuarios, de igual manera cada centroide selecciona los elementos más cercanos y aquellos que no pudieron ser asignados por respetar la capacidad de la buseta se pintan de verde fluorescente y conforman la lista de usuarios sobrantes, se ven en 3.6.

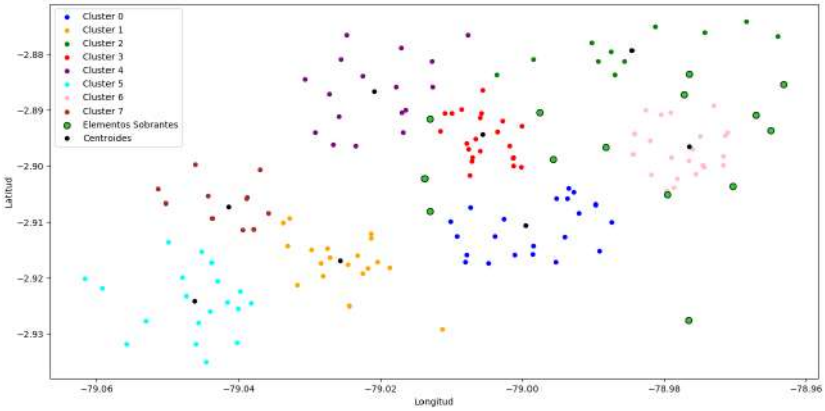


Figura 3.6: Resultados obtenidos para un escenario de 160 estudiantes.
Fuente: Autor

3.2.2. Movimiento de Elementos al Centroide más Cercano

En este paso el movimiento de elementos no es tan evidente ya que se mueven unos pocos elementos, respetando la capacidad de la buseta e identificando el clúster más cercano como nuevo clúster, se evidencia como se equilibran las asignaciones, donde en este paso hay más clústeres que ya completan su capacidad máxima.

A continuación se demuestra como los elementos de cada clúster buscan un clúster más cercano y son reasignados en caso de que el clúster de destino tenga capacidad. En la figura 3.7 el elemento perteneciente al clúster tres, de color rojo toma un elemento del clúster dos de color verde.

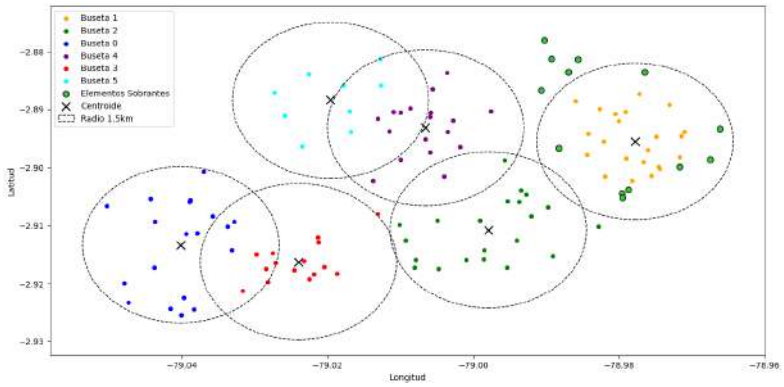


Figura 3.7: Resultados para un escenario de 100 estudiantes.
Fuente: Autor

Para el segundo escenario de 160 usuarios, la buseta tres de color rojo selecciona un elemento de la buseta seis, de color rosado y a su vez la buseta cero, de color azul,

toma un elemento de la buseta tres, de color rojo, cabe recalcar que estos procesos dependen de la capacidad de cada buseta, se ve en la figura 3.8.

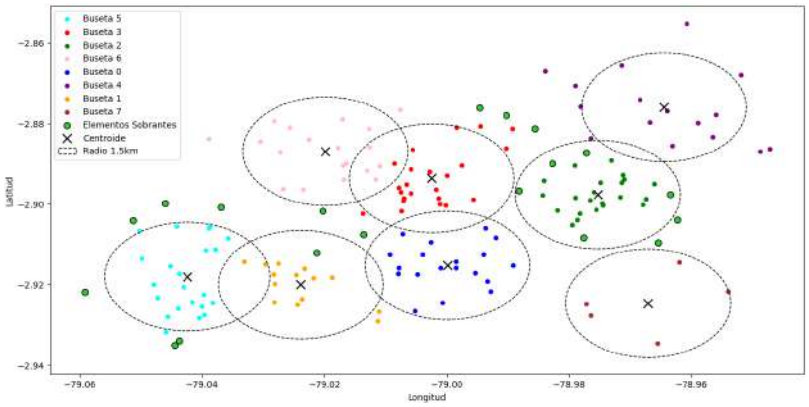


Figura 3.8: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

Para el escenario uno de 160 estudiantes, en la figura 3.9 la buseta cuatro, de color morado, toma un elemento perteneciente al clúster original a la buseta dos, de color verde. Considerar que este movimiento se hace en función de la capacidad máxima de cada buseta.

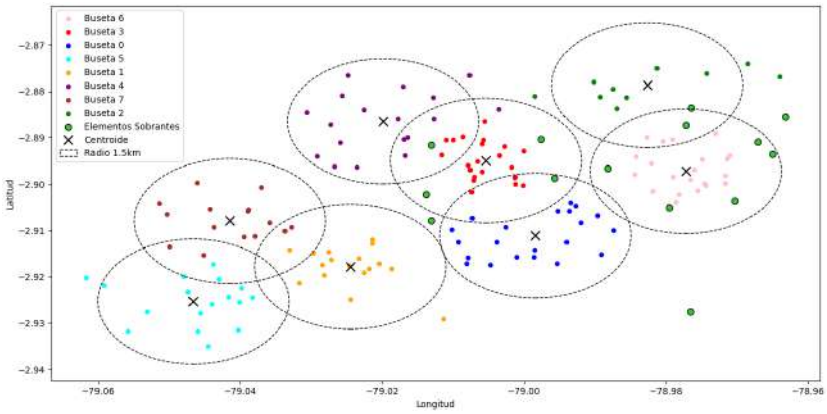


Figura 3.9: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

3.2.3. Asignación según Distancia de Restricción al Clúster de Menos Elementos

En este paso los resultados del movimiento se aprecian en los clústeres de menos elementos, ya que son los únicos que pueden realizar movimiento de elementos en este paso, el resto de clústeres solo ceden elementos al clúster identificado como

clúster de menos elementos, aquí el clúster de menos elementos toma elementos de sus vecinos si es que caen dentro del círculo, en cada paso se puede evidenciar como se llena cada clúster en función de la capacidad máxima de usuarios que pueden recibir.

En este paso la figura 3.10 evidencia como el clúster número cinco, de color celeste, agrega elementos que estén dentro de la distancia de restricción, siempre que el clúster pueda recibir estos elementos, solo el clúster con menos elementos ejecuta este paso y sin considerar elementos sobrantes.

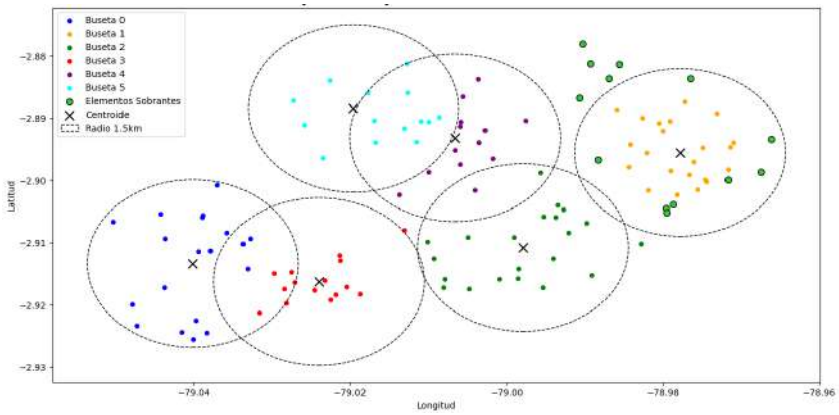


Figura 3.10: Resultados para un escenario de 100 estudiantes.
Fuente: Autor

En la figura 3.11 debido a que la buseta de menos elementos, para este caso la buseta siete, de color marrón, no tiene ningún elemento sobrante dentro de la distancia de restricción, por lo tanto no puede hacer este paso y el gráfico se mantiene.

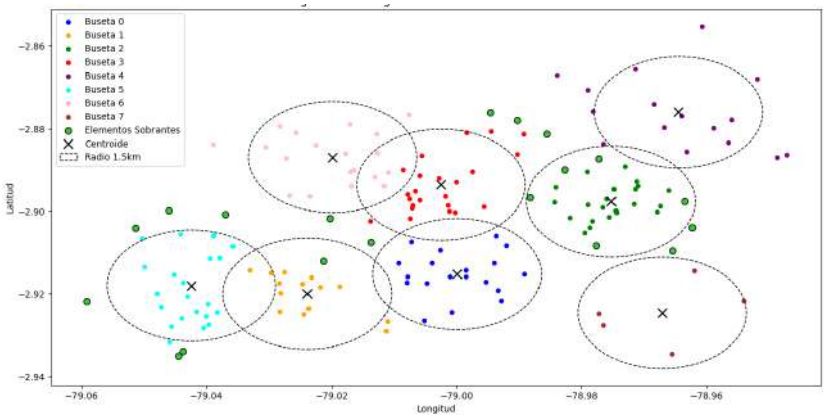


Figura 3.11: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

En la figura 3.12 se demuestra como la buseta dos, de color verde, selecciona a aquellos elementos que estén dentro de la distancia de restricción, se demuestra en la

gráfica que los elementos sobrantes no son considerados para este paso, por lo tanto estos elementos no son asignados.

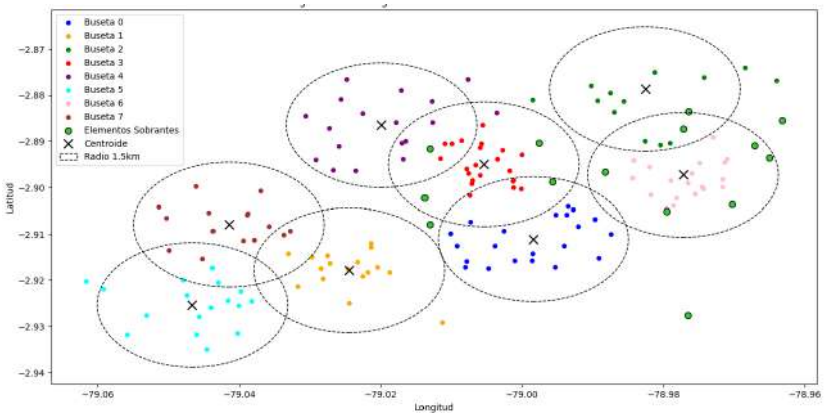


Figura 3.12: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

3.2.4. Asignación de Elementos Sobrantes

En este paso se asignan los elementos sobrantes que están dentro de la distancia de restricción, se puede evidenciar como la cantidad de elementos sobrantes disminuye, sin embargo, se observan todavía unos cuantos elementos sobrantes que no fueron asignados, eso se debe a que esos elementos no están dentro de la distancia de restricción o el clúster ya llenó su capacidad.

En la figura 3.13 se observa como los elementos sobrantes que estén dentro de la distancia de restricción son asignados, para este caso los elementos sobrantes que no fueron asignados se debe a que se lleno la capacidad de la buseta, por ese motivo estos elementos siguen sin ser asignados, se mantienen en la lista de elementos sobrantes y mantienen su color.

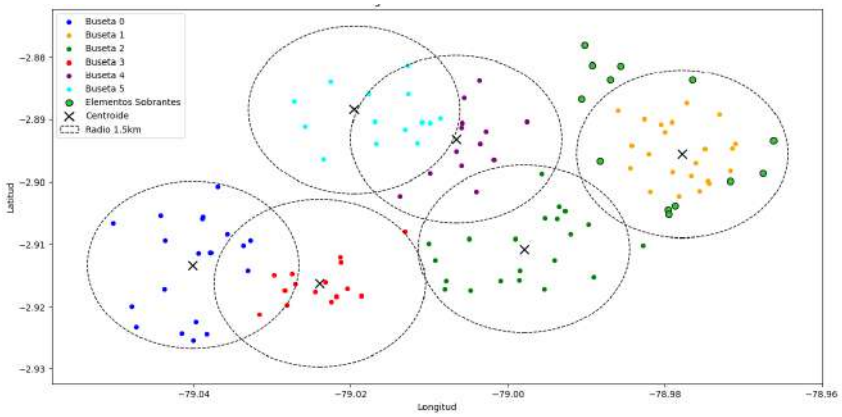


Figura 3.13: Resultados para un escenario de 100 estudiantes.
Fuente: Autor

En la figura 3.14 se asignan los elementos sobrantes que estén dentro de la distancia de restricción y se evidencia como aquellos elementos fuera de la distancia de restricción no son asignados.

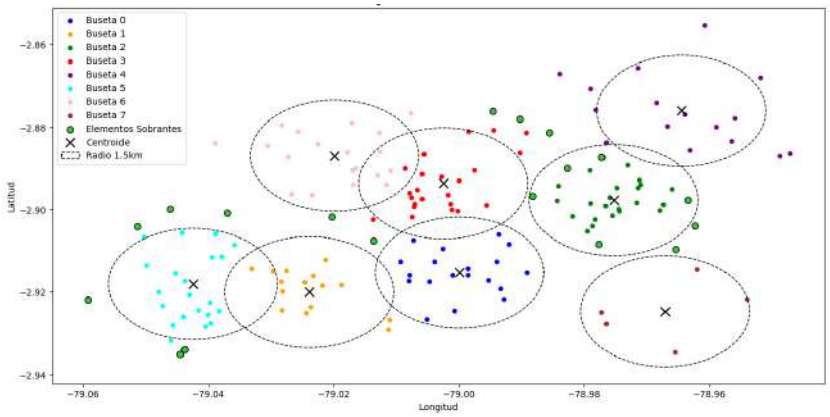


Figura 3.14: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

La figura 3.15 se puede apreciar que a pesar de que los elementos sobrantes caigan dentro de la distancia de restricción, si el clúster tiene la capacidad llena no se asignan más elementos, como es el caso del clúster rojo.

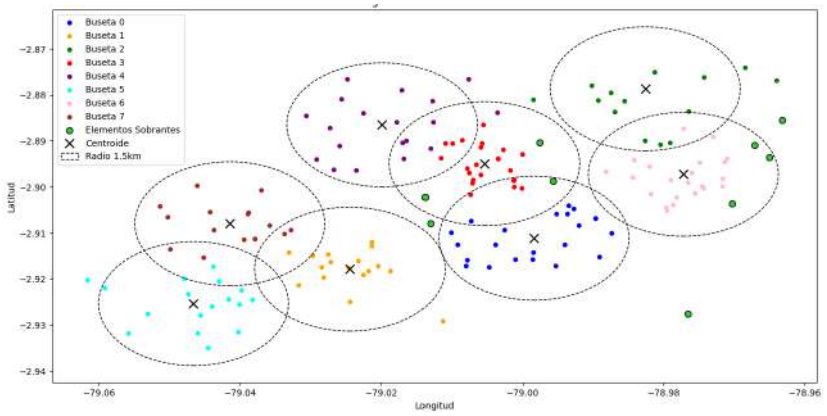


Figura 3.15: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

3.2.5. Asignación desde la Perspectiva del Usuario

En este paso el elemento sobrante busca el clúster al cual ser asignado, pero con la condición de que el clúster tenga capacidad. Como parte esencial de su búsqueda los elementos sobrantes tienen una distancia de restricción mayor a la usada en los pasos anteriores, es por ese motivo que se observa que no son asignados al clúster más cercano sino aquel que tenga capacidad y esté dentro de la distancia de restricción, por este motivo el elemento puede mantener cierta lejanía del clúster al que fue asignado. Por último aquellos elementos sobrantes que no son asignados es porque no hubo ningún clúster que dentro de la distancia de restricción pudo recibir este elemento, esto demuestra como en cada parte del proceso se respeta la capacidad máxima de la buseta.

En la figura 3.16 los usuarios sobrantes son asignados al clúster que estén dentro de su distancia de restricción y en función de la capacidad que tenga el clúster para recibir estos elementos, por tal motivo, los elementos sobrantes fueron asignados al clúster de color morado que pese a que no sea el clúster de menor distancia es el clúster que tiene capacidad para recibir estos elementos.

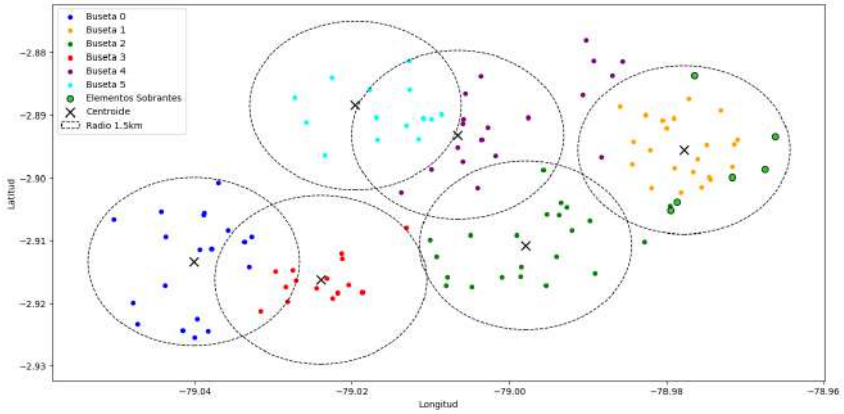


Figura 3.16: Resultados para un escenario de 100 estudiantes.
Fuente: Autor

En la figura 3.17 los elementos sobrantes que quedan no son asignados, porque los clústeres que están dentro de la distancia de restricción de los elementos sobrantes no tienen capacidad para recibir estos usuarios.

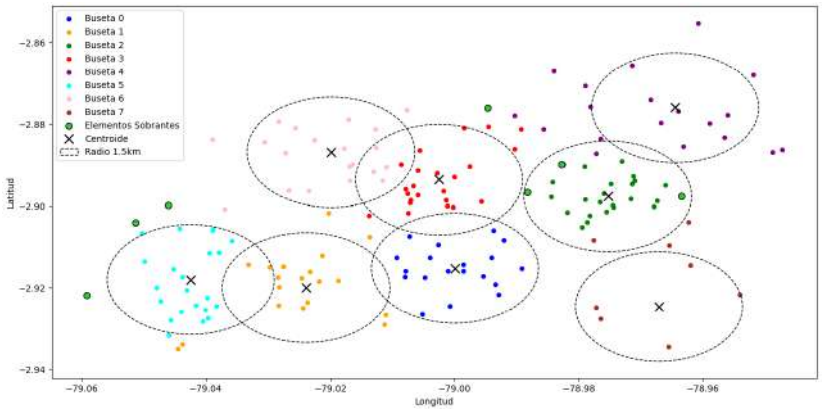


Figura 3.17: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

En la figura 3.18 se observa que no es necesario que los elementos sobrantes sean asignados al clúster que le quede más cerca, sino van a aquellos que tengan capacidad, como el caso del clúster rojo, que a pesar de que tiene elementos sobrantes cerca, no fueron asignados a su clúster porque no tiene capacidad para recibir elementos entonces buscan otro clúster que los pueda recibir.

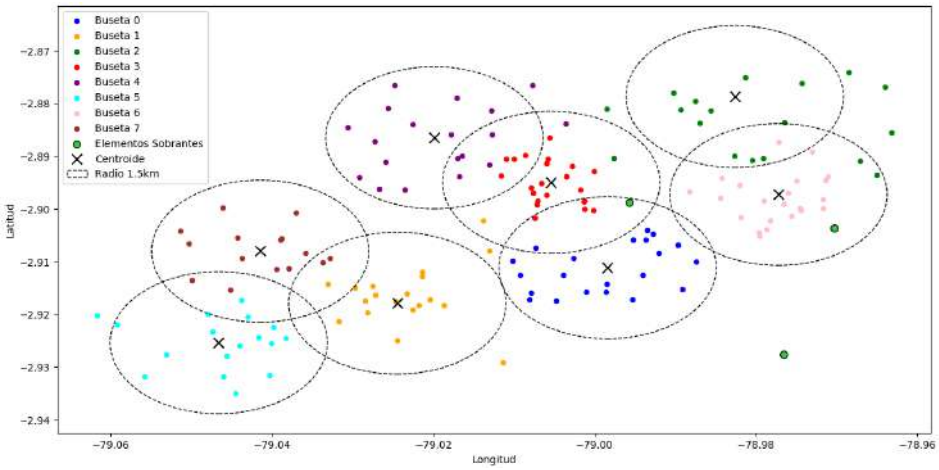


Figura 3.18: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

3.2.6. Intercambio de Elementos

En este proceso son asignados los elementos sobrantes que no fueron asignados en procesos anteriores, por tal motivo no hay más elementos sobrantes después de este proceso, en este proceso se observa como el clúster que en pasos anteriores tenía la menor cantidad de elementos en este paso completa o casi completa su capacidad máxima, este movimiento se hace siempre orientado a llenar el clúster de menos elementos, es por eso que en la gráfica se puede distinguir como los clústeres que intercambian elementos son aquellos que están cerca del clúster de menos elementos, para garantizar que siempre intercambien elementos con este clúster.

En la figura 3.19 el intercambio se hace en función del clúster con menos elementos, es por tal motivo que el clúster de color rojo es al que le fueron asignados más elementos.

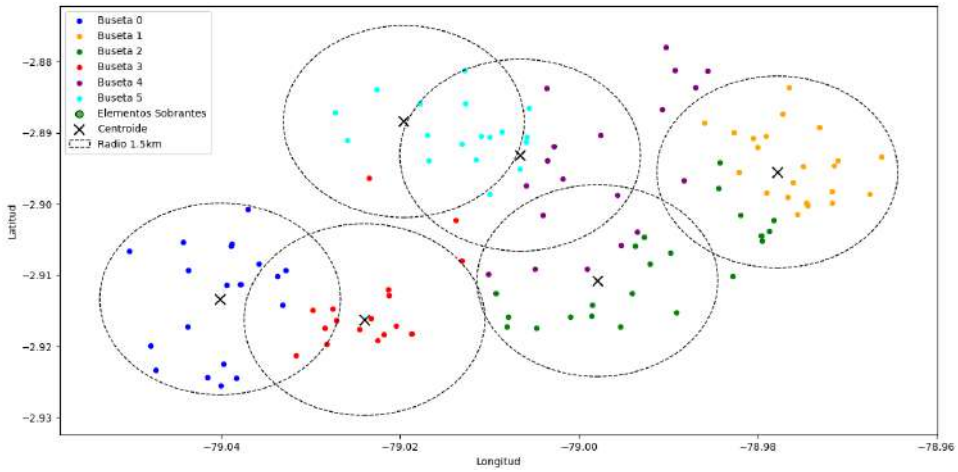


Figura 3.19: Resultados para un escenario de 100 estudiantes.
Fuente: Autor

En la figura 3.20 el clúster amarillo, es el clúster que intercambiará elementos, porque fue el clúster que tenía mayor capacidad disponible.

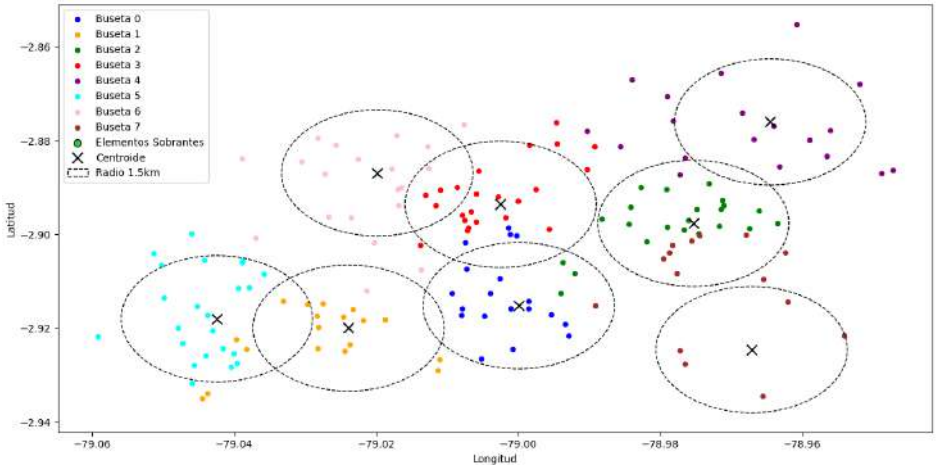


Figura 3.20: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

En la figura 3.21 se ve como es el clúster de color café el que intercambia elementos por ser el clúster con menor número de elementos, entonces todos los movimientos están orientados hacia este clúster.

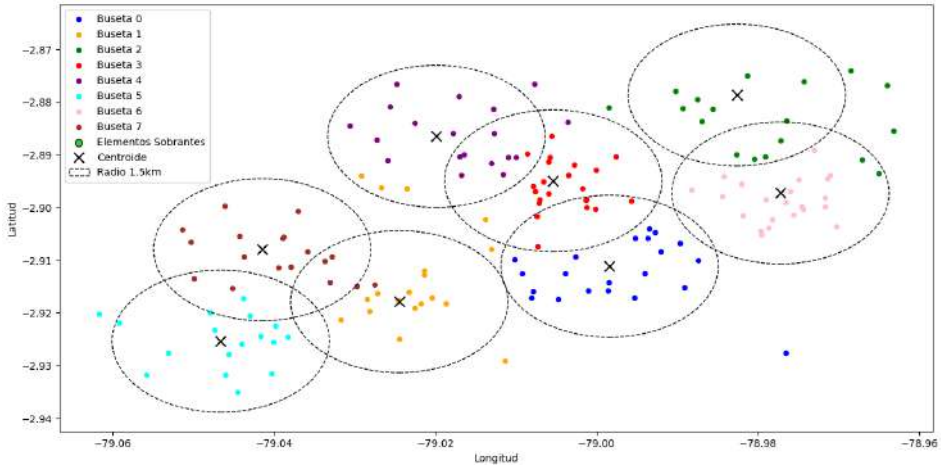


Figura 3.21: Resultados para un escenario de 160 estudiantes.
Fuente: Autor

3.3. Algoritmo de Enrutamiento

Basado en el algoritmo de Dijkstra, en este proceso se puede constatar que el punto de partida siempre será el elemento del clúster más lejano a la institución educativa. Una vez definido el punto de inicio, para determinar cual es el siguiente nodo, se procede a calcular cuál es el siguiente más cercano que no se haya visitado aún y que no sea el punto de destino final. La figura fig:8-1 muestra un ejemplo de ruta establecida donde es posible observar que cada nodo es visitado una sola vez, que para cada cluster, el punto de partida es el sitio más lejano a la institución educativa, también que el punto de llegada corresponde al nodo más cercano al punto de llegada. Como caso de estudio se propuso la Unidad Educativa Técnico Salesiano en la ciudad de Cuenca-Ecuador.

En la figura 3.22 se evidencia como el elemento más lejano al punto de destino es el punto de partida y aquel punto más cercano al punto de destino es el último punto en ser visitado. La figura 3.23 muestra que a pesar de que tenga dos elementos lejanos al punto de destino, el punto de partida es aquel que más lejos este del punto de destino. Como tercer ejemplo la figura 3.24 muestra la ruta de otro cluster donde también el punto de partida es el nodo más lejano a la institución y el punto antes de ir a destino es el punto más cercano a la institución. En la figura 3.25 a pesar de que exista dos punto cercanos al punto de destino, aquel que según Dijkstra considere la

ruta de menor trayecto será seleccionado como el último punto al cual calcular la ruta previo a ir al punto de destino.

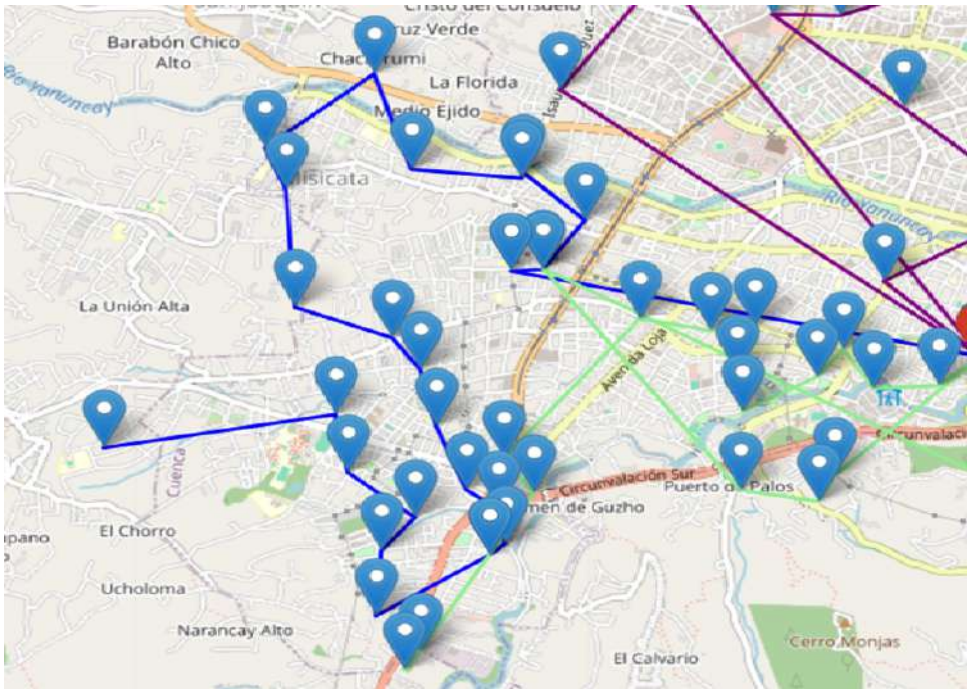


Figura 3.22: Resultados después de aplicar Dijkstra a uno de los clústeres.
Fuente: Autor

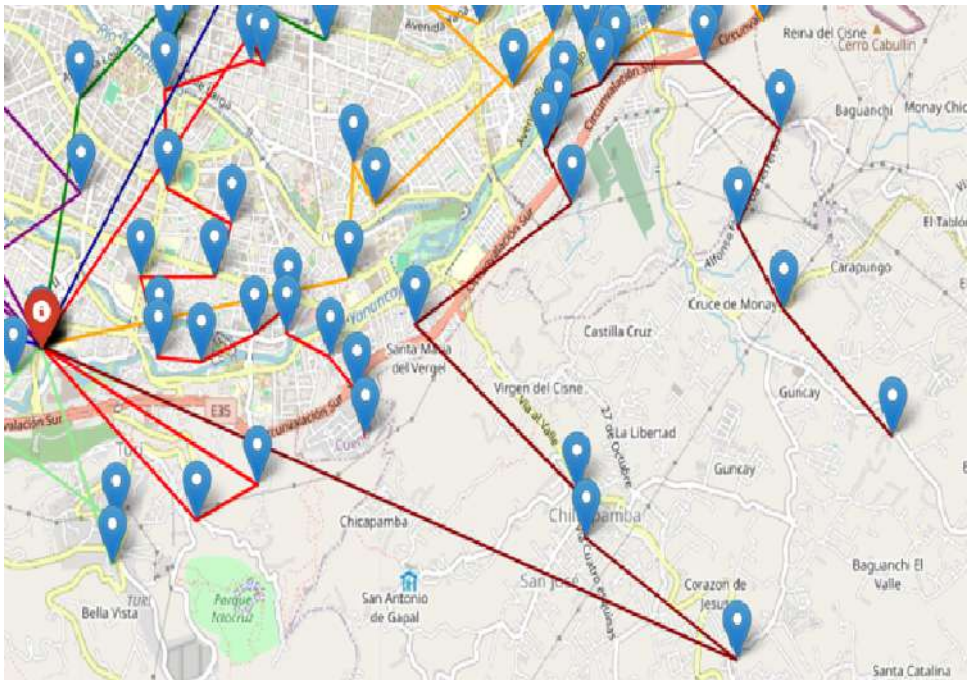


Figura 3.23: Resultados después de aplicar Dijkstra a uno de los clústeres.
Fuente: Autor

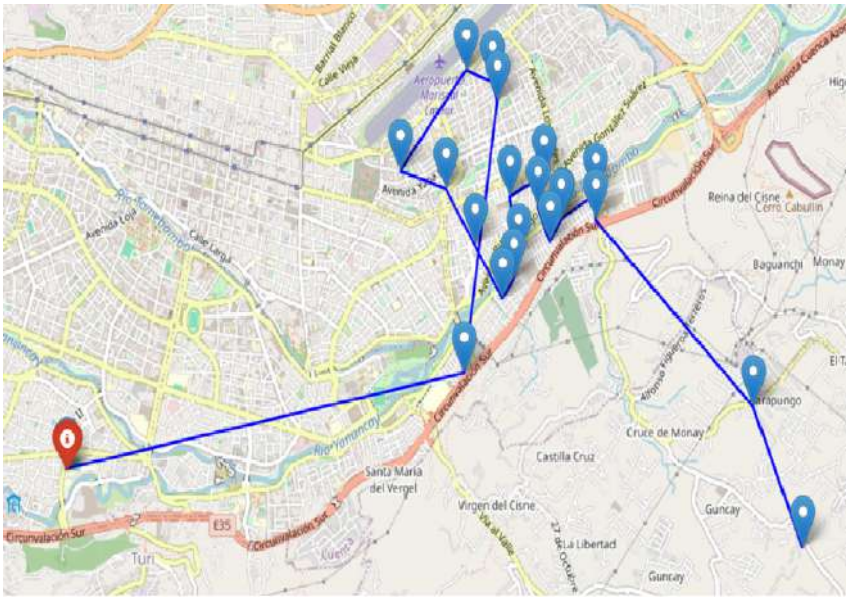


Figura 3.24: Resultados después de aplicar Dijkstra a uno de los clústeres.
Fuente: Autor

Figura 3.25: Resultados después de aplicar Dijkstra a uno de los clústeres.
Fuente: Autor

3.4. Pruebas de Tiempo

El presente apartado está dirigido al análisis de resultado obtenido tras las mediciones de tiempo. Se planteó dos tipos de mediciones cambiando las distancias de restricción del programa, primero se realizó la medición en saltos de 500km hasta los 3500km y para el segundo caso saltos de 1000km hasta llegar a los 4000km, se ejecutó un total de 500 veces para cada distancia.

En la figura 3.26 se observa que a medida que aumenta la distancia el tiempo disminuye, ya que al aumentar la distancia de restricción los elementos pueden encontrar de forma más rápida un clúster al cuál pueden ser asignados, esto ocasiona que todos los demás pasos de asignación se ejecuten con mayor rapidez, porque con una mayor distancia de restricción implica que hay menos elementos sin ser asignados y los procesos se ejecutan más rápido.

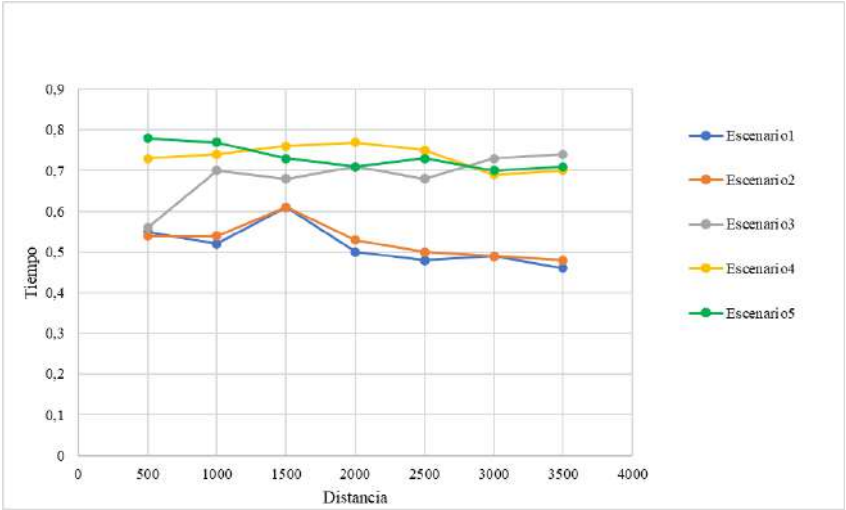


Figura 3.26: Resultados de medición de tiempo en saltos de distancia en 500km.
Fuente: Autor

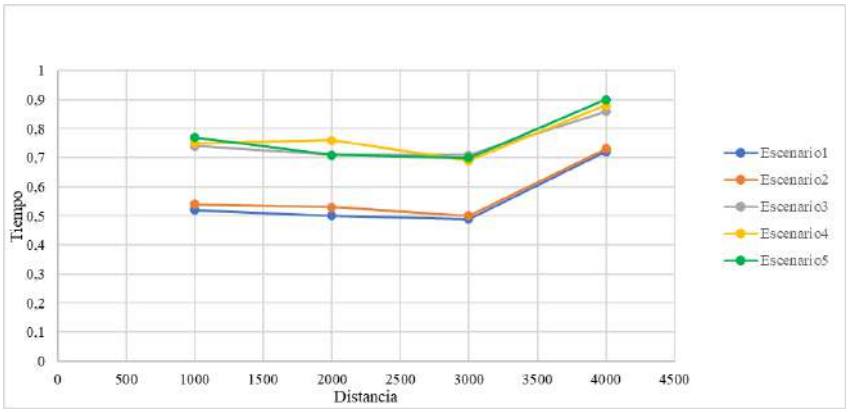


Figura 3.27: Resultados de medición de tiempo en saltos de distancia en 1000km.
Fuente: Autor

3.5. Creación del ejecutable

Antes de crear la versión ejecutable, se aplicó una pequeña interfaz basada en ventanas y botones para el usuario pueda elegir la base de datos que se desea procesar para realizar la asignación de busetas a cada estudiante (o asignación de estudiantes a cada buseta). Para esto, se uso la librería *tkinter* que permite la instanciación de ventanas y otro objetos gráficos en forma sencilla. Se agregaron tres botones y una caja de texto para mensajes de tal forma que, hay un botón que permite seleccionar el archivo de excel que contiene la base de datos (figura 3.28) y carga el nombre del archivo en una variable previo a que se ejecute el proceso de asignación. La caja de dialogo muestra un mensaje con el directorio del archivo seleccionado (figura 3.29).

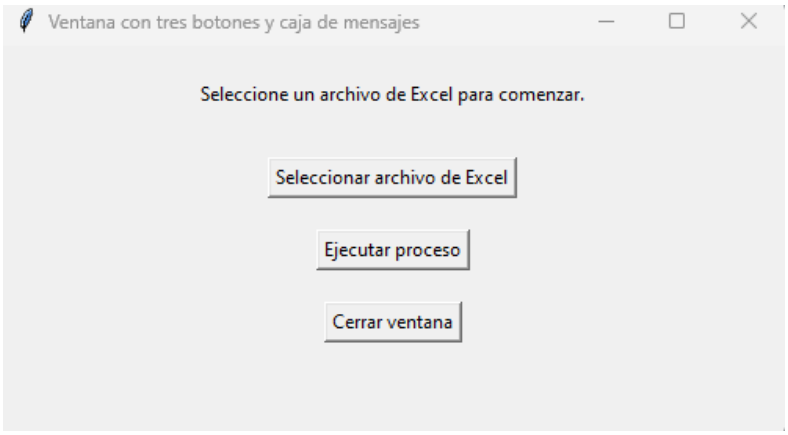


Figura 3.28: Ventana del aplicativo cuando es ejecutado
Fuente: Autor

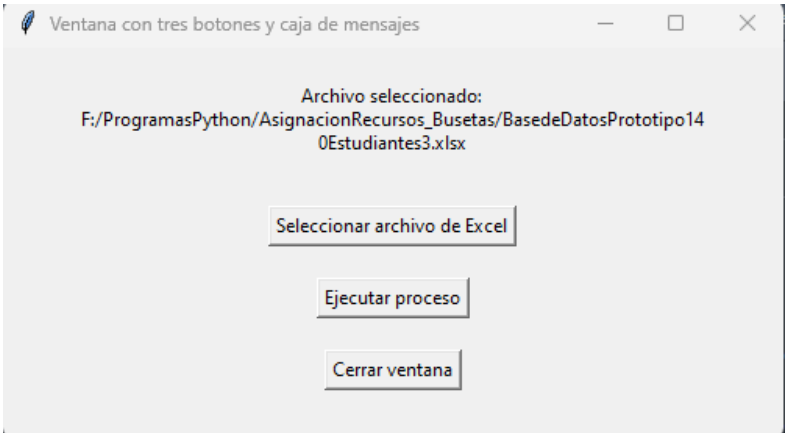


Figura 3.29: Aplicativo con archivo seleccionado.
Fuente: Autor

Existe un segundo botón para ejecutar el proceso de asignación Si no se ha elegido ningún archivo, se muestra el mensaje "Aún no ha seleccionado la base de datos" (Ver figura 3.30 y figura 3.31 respectivamente)

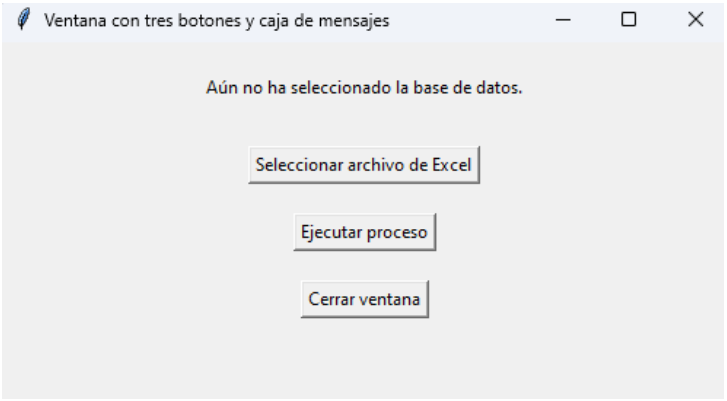


Figura 3.30: Intento de ejecución del aplicativo sin archivo seleccionado
Fuente: Autor

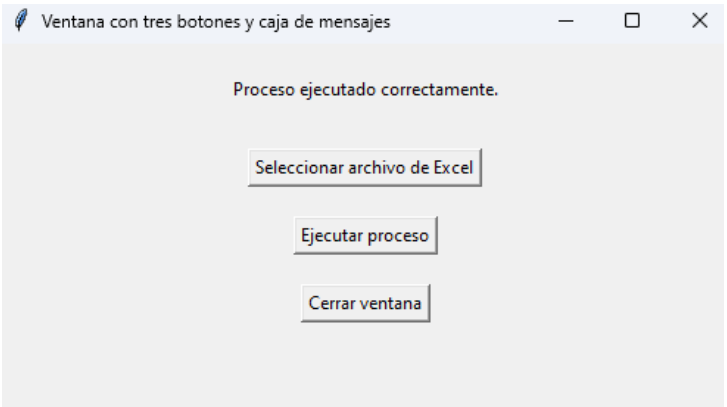


Figura 3.31: Estado del Aplicativo luego de ejecutarse correctamente.
Fuente: Autor

Como muestran las ventanas anteriores, el tercer botón permite que se cierre la ventana en cualquier momento deseado.

En lo que respecta al archivo de salida, se usó la librería *os* que permite crear un directorio dentro de la carpeta donde está el archivo ejecutable en el caso de que no exista. El nombre del archivo de salida dependerá de la fecha de ejecución, por lo tanto, se crea una versión diferente si el día de la ejecución cambia para lo cual es necesario usar la librería *datetime*.

Para la realización del ejecutable se utilizó el comando pyinstaller el cual se define como un módulo encargado de crear archivos ejecutables para diferentes plataformas como Windows, Linux, etc.

Es imprescindible tener el módulo pyinstaller, una vez culminada la instalación se procede a buscar en el directorio de archivos el nombre del documento del cual se desea realizar el ejecutable y se ejecuta el comando, el cual crea un archivo binario, el cual permite que los computadores lo interpreten como un programa.

Una vez que termina de crearse el ejecutable se crea una carpeta llamada “dist” y que contiene el archivo ejecutable y en el cual se debe agregar todas las bases de datos adicionales que sean necesarias para que se pueda ejecutar el programa; es imprescindible agregar el nombre del archivo. En el caso de este programa se deben cargar las bases de datos con las coordenadas de los estudiantes y su información personal (Ver Figura 3.32).

Por último cuando el ejecutable culmina da como resultado una hoja de Excel con el resultado de la asignación, está hoja indica nombre y apellido des estudiante y el número de buseta a la que fue asignado (Ver Figura 3.33).

```
a = Analysis(
    ['TesisFinalizada_Excel.py'],
    pathex=[],
    binaries=[],
    datas=[],
    hiddenimports=[],
    hookspath=[],
    hooksconfig={},
    runtime_hooks=[],
    excludes=[],
    noarchive=False,
    optimize=0,
)
pyz = PYZ(a.pure)

exe = EXE(
    pyz,
    a.scripts,
    a.binaries,
    a.datas,
    [],
    name='TesisFinalizada_Excel',
    debug=False,
    bootloader_ignore_signals=False,
    strip=False,
    upx=True,
    upx_exclude=[],
    runtime_tmpdir=None,
    console=False,
    disable_windowed_traceback=False,
    argv_emulation=False,
    target_arch=None,
    codesign_identity=None,
    entitlements_file=None,
)
```

Figura 3.32: Opciones para la creación del ejecutable.
Fuente: Autor

	Nombre	Buseta
2	Fausto Cisneros	6
3	Jurado Samanta	6
4	Leguizamon Veronica	6
5	Samanta Serrano	6
6	Angelica Díaz	6
7	Franco Listorti	6
8	García Fernando	6
9	Lorca Vinicio	6
10	Matías Carpio	6
11	José Vélez	6
12	Miguel Matute	6
13	Lorenzo Soliz	6
14	Castro Armando	6
15	Gómez Vicente	6
16	Fernandez Horacio	6
17	Cortázar Paul	6
18	Durán Alejandro	6
19	Romina Sarmiento	6
20	Edison Miranda	6
21	Juan Valdivieso	6
22	Hurtado Daniel	6
23	Llosa David	6
24	Lorca Federico	6
25	Orwell Diana	4
26	Icaza Jorge	4
27	Neftali Adrián	4

Figura 3.33: Resultados después correr el ejecutable.
Fuente: Autor

Capítulo 4

Conclusiones y Trabajos Futuros

A continuación se exponene las conclusiones de este trabajo.

4.1. Conclusiones y Recomendaciones

Este trabajo presenta una propuesta de asignación estudiantes (usuarios) a un conjunto de unidades de transporte escolar respetando su capacidad máxima pudiendo ser usada por cualquier institución educativa al ser desarrollada en código abierto como Python. Por esto, se decidió el desarrollo de una metaheurística partiendo de heurísticas ya establecidas como lo son los algoritmos de agrupamiento, en concreto GMM y el algoritmo goloso para establecer una solución sencilla con poco coste computacional. Por tanto, este trabajo ofrece resultados que pueden ser reproducibles frente a múltiples escenarios aleatorios y con nulas variaciones en la respuesta de asignación frente a diversas ejecuciones.

La complejidad computacional de los algoritmos de asignación crece de forma significativa con el aumento del tamaño de la instancia del problema, lo que puede llevar a tiempos de resolución prohibitivos. Por esta razón, es común recurrir a técnicas de relajación para obtener soluciones aproximadas pero eficientes. Entonces, en este trabajo, proponemos un enfoque que busca balancear la calidad de la solución con la eficiencia computacional. Para ello, se imponen restricciones a la solución, como limitar la capacidad de cada vehículo y descartar usuarios que se encuentren a una distancia excesiva de los centroides de demanda. Estas restricciones permiten reducir

el espacio de búsqueda y acelerar el proceso de resolución. Al relajar el problema y utilizar clustering para subdividir el problema, se busca obtener soluciones de alta calidad en tiempos de ejecución razonables, incluso para instancias de gran tamaño y complejidad.

En este trabajo de titulación se sugiere una ruta basada en el algoritmo de Dijkstra, el cual trabaja encontrando la mínima distancia entre nodos, para identificar en que orden se deben recoger a los pasajeros, sin embargo, sería importante aplicarlo siguiendo la trayectoria de las calles y con ciertas restricciones para ajustar mejor este modelo.

Se realizó un archivo ejecutable para iniciar el programa, para el cual es necesario indicar la base de datos que se desea ejecutar y el programa da como resultado una hoja de excel indicando el nombre y apellido del estudiante y la buseta a la que cada estudiante fue asignado. Para mejorar los tiempos de ejecución del programa se quitaron las imágenes de guía y únicamente se centra en la hoja de excel con el resultado final.

Al evaluar algoritmos, es fundamental establecer objetivos claros para identificar las restricciones específicas del problema. Estas restricciones delimitan el espacio de búsqueda y guían la elección de las variables de decisión.

Una evaluación exhaustiva de un algoritmo implica considerar no solo las metas y restricciones del problema, sino también las características del algoritmo en términos de eficiencia, robustez y interpretabilidad.

4.2. Trabajos Futuros

Para trabajos futuros se propone el uso de la API de Google para usar datos del tráfico en los horarios en los que las unidades de transporte escolar van a realizar los recorridos de forma que se pueda obtener rutas más reales al considerar el tráfico y/o posibles cierres de vías.

Glosario

BIRCH Reducción Iteractiva Balaceada y agrupación usando Jerarquías – Balance Iterative Reducing and Clustering using Hierarchies.

CF Característica del Agrupamiento – Clustering Feature (CFT), método que evalúa las características del agrupamiento..

CFT Característica de Agrupamiento en Árbol – Clustering Ferature TREE, que es una Estructura que indica como se conformaron los grupos de datos después de aplicar BIRCH.

Clúster Técnica informática que sirve para agrupar datos o elementos.

Data Frame Estructura que puede contener cualquier tipo de datos en dos dimensiones, es un dataset organizado en columnas..

Dataset Conjunto de datos organizados en una matriz..

GMM Modelo de Mezcla Gaussiana – Gaussian Mixture Model.

SBRP Problema de Enrutamiento de Bus Escolar – School Bus Routing Problem..

SIG Sistema de Información Geográfico – Geographic Information System.

Time Windows Ventanas de tiempo – Time Windows, método que indica el tiempo que transcurre entre el incio y el fin de un ruta..

TSP Problema del Agente Viajero – Travelling Salesman Problem.

VRP Problema de enrutamiento de vehículos – Vehicle Routing Problem..

Referencias

[1] M. Fisher, «Vehicle routing,» *Handbooks in operations research and management science*, vol. 8, págs. 1-33, 1995. DOI: https://doi.org/10.1016/S0927-0507(05)80105-7.

[2] W. A. Ellegood, S. Solomon, J. North y J. F. Campbell, «School bus routing problem: Contemporary trends and research directions,» *Omega*, vol. 95, pág. 102 056, 2020. DOI: https://doi.org/10.1016/j.omega.2019.03.014.

[3] J. Park y B.-I. Kim, «The school bus routing problem: A review,» *European Journal of operational research*, vol. 202, n.º 2, págs. 311-319, 2010. DOI: https://doi.org/10.1016/j.ejor.2009.05.017.

[4] J. F. Sarubbi, C. M. Mesquita, E. F. Wanner, V. F. Santos y C. M. Silva, «A strategy for clustering students minimizing the number of bus stops for solving the school bus routing problem,» en *NOMS 2016-2016 IEEE/IFIP network operations and management symposium*, IEEE, 2016, págs. 1175-1180. DOI: https://doi.org/10.1109/NOMS.2016.7502983.

[5] D. Du, K.-I. Ko, X. Hu et al., *Design and analysis of approximation algorithms*. Springer, 2012, vol. 62. DOI: https://doi.org/10.1007/978-1-4614-1701-9.

[6] M. d. C. P. Palomino y Á. F. R. Torres, «Las competencias digitales en estudiantes de las carreras de Educación en Ecuador,» *Campus Virtuales*, vol. 12, n.º 2, págs. 113-126, 2023. DOI: http://dx.doi.org/10.54988/cv.2023.2.1215.

[7] J. A. Cruz Mejillones, «Mejoramiento de atención a padres de familia mediante automatización de procesos de la Escuela de Educación Básica Zenón Macías vera, Guayas, 2021,» Universidad César Vallejo, 2022. dirección: https://hdl.handle.net/20.500.12692/78345.

[8] L. M. F. Vizcaya y O. d. J. R. Velásquez, «Enseñanza de la teoría de grafos en la escuela secundaria basada en la modelación geométrica y la resolución de problemas,» *CONTRIBUCIONES A LAS CIENCIAS SOCIALES*, vol. 17, n.º 1, págs. 6378-6399, 2024. DOI: https://doi.org/10.55905/revconv.17n.1-383.

[9] M. Sciortino, R. Lewis y J. Thompson, «A heuristic algorithm for school bus routing with bus stop selection,» en *Evolutionary Computation in Combinatorial Optimization: 21st European Conference, EvoCOP 2021, Held as Part of EvoStar 2021, Virtual Event, April 7–9, 2021, Proceedings 21*, Springer, 2021, págs. 202-218. DOI: https://doi.org/10.1007/978-3-030-72904-2_13.

[10] J. A. V. Flórez, C. J. Z. Grisales y G. Gatica, «Una aplicación del método MTZ a la solución del problema del agente viajero,» *Scientia et technica*, vol. 22, n.º 4, págs. 341-344, 2017. DOI: https://doi.org/10.22517/23447214.12751.

[11] O. Díaz-Parra, J. A. Ruiz-Vanoye, M. de los Ángeles Buenabad-Arias y A. C. Saenz, «Vertical transfer algorithm for the school bus routing problem,» *Transactions on Computational Science XXI: Special Issue on Innovations in Nature-Inspired Computing and Applications*, págs. 211-229, 2013. DOI: https://doi.org/10.1007/978-3-642-45318-2_9.

[12] E. R. López Santana y J. d. J. Romero Carvajal, «A hybrid column generation and clustering approach to the school bus routing problem with time windows,» *Ingeniería*, vol. 20, n.º 1, págs. 101-117, 2015. DOI: https://doi.org/10.14483/udistrital.jour.reving.2015.1.a07.

[13] E. Gussmagg-Pfliegl, F. Tricoire, K. F. Doerner, R. F. Hartl y S. Irnich, «Heuristics for a real-world mail delivery problem,» en *Applications of Evolutionary Computation: EvoApplications 2011: EvoCOMNET, EvoFIN, EvoHOT, EvoMUSART, EvoSTIM, and EvoTRANSLOG, Torino, Italy, April 27-29, 2011, Proceedings, Part II*, Springer, 2011, págs. 481-490. DOI: https://doi.org/10.1007/978-3-642-20520-0_49.

[14] S. Wang, W. Zhang, Y. Bie, K. Wang y A. Diabat, «Mixed-integer second-order cone programming model for bus route clustering problem,» *Transportation Research Part C: Emerging Technologies*, vol. 102, págs. 351-369, 2019. DOI: https://doi.org/10.1016/j.trc.2019.03.019.

[15] J.-H. Park y J.-H. Park, «Work Load Calculation Algorithm for Postal Delivery Operation,» en *Future Information Technology, Application, and Service: FutureTech 2012 Volume 2*, Springer, 2012, págs. 259-267. DOI: https://doi.org/10.1007/978-94-007-5064-7_36.

[16] S.-D. Yang, W.-G. Lyu y S.-J. Lee, «Optimal Location of Mail Distribution Center using Steiner Tree,» *Journal of the Korean Institute of Illuminating and Electrical Installation Engineers*, vol. 22, n.º 9, págs. 82-87, 2008. DOI: https://doi.org/10.1016/S0304-3975(00)00182-1.

[17] A. Rahman, H. Tan, W. Liew y N. Shahruddin, «Routing Mail Delivery from a Single Depot with Multiple Delivery Agents,» *Malaysian Journal of Mathematical Sciences*, vol. 14, págs. 15-29, 2020.

[18] G. A. Sgarro y L. Grilli, «Ant colony optimization for Chinese postman problem,» *Neural Computing and Applications*, vol. 36, n.º 6, págs. 2901-2920, 2024. DOI: https://doi.org/10.1007/s00521-023-09195-4.

[19] E. H. Lee y J. Jeong, «Accessible taxi routing strategy based on travel behavior of people with disabilities incorporating vehicle routing problem and Gaussian mixture model,» *Travel behaviour and society*, vol. 34, pág. 100 687, 2024. DOI: https://doi.org/10.1016/j.tbs.2023.100687.

[20] S. Wang, W. Rao e Y. Hong, «A distance matrix based algorithm for solving the traveling salesman problem,» *Operational Research*, vol. 20, págs. 1505-1542, 2020. DOI: https://doi.org/10.1007/s12351-018-0386-1.

[21] G. Fung, «Mastering Machine Learning with scikit-learn,» en Citeseer, 2001, cap. 15, ISBN: 9781788299879.

[22] D. Pascual, F. Pla y S. Sánchez, «Algoritmos de agrupamiento,» *Método Informáticos Avanzados*, págs. 164-174, 2007.

[23] S. Na, L. Xumin y G. Yong, «Research on k-means clustering algorithm: An improved k-means clustering algorithm,» en *2010 Third International Symposium on intelligent information technology and security informatics*, Ieee, 2010, págs. 63-67. DOI: https://doi.org/10.1109/IITSI.2010.74.

[24] Y. Li y H. Wu, «A clustering method based on K-means algorithm,» *Physics Procedia*, vol. 25, págs. 1104-1109, 2012. DOI: https://doi.org/10.1016/j.phpro.2012.03.206.

[25] N. Arbin, N. S. Suhaimi, N. Z. Mokhtar y Z. Othman, «Comparative analysis between k-means and k-medoids for statistical clustering,» en *2015 3rd International Conference on Artificial Intelligence, Modelling and Simulation (AIMS)*, IEEE, 2015, págs. 117-121. DOI: 10.1109/AIMS.2015.82.

[26] P. Arora, S. Varshney et al., «Analysis of k-means and k-medoids algorithm for big data,» *Procedia Computer Science*, vol. 78, págs. 507-512, 2016. DOI: https://doi.org/10.1016/j.procs.2016.02.095.

[27] T. S. Madhulatha, «Comparison between k-means and k-medoids clustering algorithms,» en *International Conference on Advances in Computing and Information Technology*, Springer, 2011, págs. 472-481. DOI: https://doi.org/10.1007/978-3-642-22555-0_48.

[28] A. Lang y E. Schubert, «BETULA: numerically stable CF-trees for BIRCH clustering,» en *International Conference on Similarity Search and Applications*, Springer, 2020, págs. 281-296. DOI: https://doi.org/10.1007/978-3-030-60936-8_22.

[29] T. Zhang, R. Ramakrishnan y M. Livny, «BIRCH: A new data clustering algorithm and its applications,» *Data mining and knowledge discovery*, vol. 1, págs. 141-182, 1997. DOI: https://doi.org/10.1023/A:1009783824328.

[30] T. Zhang, R. Ramakrishnan y M. Livny, «BIRCH: an efficient data clustering method for very large databases,» *ACM sigmod record*, vol. 25, n.º 2, págs. 103-114, 1996. DOI: https://doi.org/10.1145/235968.233324.

[31] Z. Wang, C. Da Cunha, M. Ritou y B. Furet, «Comparison of K-means and GMM methods for contextual clustering in HSM,» *Procedia Manufacturing*, vol. 28, págs. 154-159, 2019. DOI: https://doi.org/10.1016/j.promfg.2018.12.025.

[32] C. Maugis, G. Celeux y M.-L. Martin-Magniette, «Variable selection for clustering with Gaussian mixture models,» *Biometrics*, vol. 65, n.º 3, págs. 701-709, 2009. DOI: https://doi.org/10.1111/j.1541-0420.2008.01160.x.

[33] Y. Zhang, M. Li, S. Wang et al., «Gaussian mixture model clustering with incomplete data,» *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, vol. 17, n.º 1s, págs. 1-14, 2021. DOI: https://doi.org/10.1145/3408318.

[34] S. Duarte, D. Becerra y L. F. Niño, «Un modelo de asignación de recursos a rutas en el sistema de transporte masivo TransMilenio,» *Revista Avances en Sistemas e Informática*, vol. 5, n.º 1, págs. 163-171, 2008.

[35] J. Bowers, B. Lyons y G. Mould, «Developing a resource allocation model for the Scottish patient transport service,» *Operations Research for Health Care*, vol. 1, n.º 4, págs. 84-94, 2012. DOI: https://doi.org/10.1016/j.orhc.2012.10.002.

[36] J. J. Martin Hernandez et al., «Definición de Guías de Diseño para Bases de Datos Orientadas a Grafos,» 2024.

[37] J. Castillo Ungar, «Problemas NP-completos en grafos,» 2023.

[38] S. Arenado Serrano, «Analíticas de redes y grafos,» 2023.

[39] J. E. Allauca, «Aplicación de la teoría de grafos en la Optimización de redes de transporte,» *CIENCIA INTELIGENTE*, vol. 1, n.º 1, págs. 1-14, 2023.

[40] D. S. Hochba, «Approximation algorithms for NP-hard problems,» *ACM Sigact News*, vol. 28, n.º 2, págs. 40-52, 1997. DOI: https://doi.org/10.1145/261342.571216.

[41] Y. Donoso y R. Fabregat, *Multi-objective optimization in computer networks using metaheuristics*. Auerbach Publications, 2016. DOI: https://doi.org/10.1201/9781420013627.

[42] M. Noto y H. Sato, «A method for the shortest path search by extended Dijkstra algorithm,» en *Smc 2000 conference proceedings. 2000 ieee international conference on systems, man and cybernetics.'cybernetics evolving to systems, humans, organizations, and their complex interactions'(cat. no. 0*, IEEE, vol. 3, 2000, págs. 2316-2320. DOI: https://doi.org/10.1109/ICSMC.2000.886462.

[43] B. L. Golden, T. L. Magnanti y H. Q. Nguyen, «Implementing vehicle routing algorithms,» *Networks*, vol. 7, n.º 2, págs. 113-148, 1977. DOI: https://doi.org/10.1002/net.3230070203.

[44] B. Eksioglu, A. V. Vural y A. Reisman, «The vehicle routing problem: A taxonomic review,» *Computers & Industrial Engineering*, vol. 57, n.º 4, págs. 1472-1483, 2009. DOI: https://doi.org/10.1016/j.cie.2009.05.009.

[45] D.-D. Zhu y J.-Q. Sun, «A new algorithm based on Dijkstra for vehicle path planning considering intersection attribute,» *IEEE Access*, vol. 9, págs. 19 761-19 775, 2021. DOI: 10.1109/ACCESS.2021.3053169.

[46] T. S. Jalolov, «EXPLORING THE MATHEMATICAL LIBRARIES OF PYTHON: A COMPREHENSIVE GUIDE,» *WORLD OF SCIENCE*, vol. 7, n.º 5, págs. 121-127, 2024.

[47] E. Westra, *Python geospatial development*. Packt Publishing Ltd, 2016.

[48] K. Pham, «Web Mapping with Python and Leaflet,» *The Programming Historian*, 2017. DOI: https://doi.org/10.46430/phen0070.

[49] P. Gupta y A. Bagchi, «Data Visualization with Python,» en *Essentials of Python for Artificial Intelligence and Machine Learning*, Springer, 2024, págs. 237-282. DOI: https://doi.org/10.1007/978-3-031-43725-0_7.

[50] N. J. M. Morales, V. D. P. Mieles y T. D. M. Díaz, «Los mapas interactivos, herramientas para la participación ciudadana,» *Correspondencias & análisis*, n.º 8, págs. 277-286, 2018. DOI: https://doi.org/10.24265/cian.2018.n8.14.

[51] M. Andrew-Quintana, J. M. Iturbide-Miranda, E. F. Gómez-Molina, A. Reyes-Nava, E. López-González et al., «Mapa interactivo y uso de datos a través de la implementación de una herramienta,» *Pädi Boletín Científico de Ciencias Básicas e Ingenierías del ICBI*, vol. 12, n.º Especial, págs. 109-115, 2024. DOI: https://doi.org/10.29057/icbi.v12iEspecial.12107.

[52] G. Contributors, «GeoPy Documentation,» 2014.

[53] G. Hackeling, *Mastering Machine Learning with scikit-learn*. Packt Publishing Ltd, 2017.