PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ

FACULTAD DE CIENCIAS E INGENIERÍA

Implementación de un algoritmo genético para elaborar un conjunto de rutas óptimas para el transporte de la comunidad universitaria desde y hacia el campus principal.

Tesis para optar por el Título de Ingeniero Informático que presenta el bachiller:

JULIO RODRIGO CASTILLO HUERTA

ASESOR: ING. CÉSAR AUGUSTO AGUILERA SERPA

Lima, octubre de 2018

Dedicado a mis padres Tanya Huerta y Julio Castillo.

RESUMEN

En la actualidad, varias universidades como la PUCP no cuentan con un sistema de transporte privado para la comunidad universitaria a pesar de que existen propuestas y es un servicio pedido por un sector de la universidad. Los motivos son varios ya que se debe considerar presupuestos, logística y una planeación adecuada de las rutas de servicio. Este último punto es complicado de por sí pues es difícil poder encontrar un conjunto de rutas que cumplan con satisfacer la demanda de una manera óptima.

En primer lugar, se debe considerar que, en una ciudad de gran tamaño, realizar cualquier tipo de diseño de rutas es un trabajo que presenta muchos desafíos. Con todas las calles y avenidas a considerar, realizar un diseño de rutas eficiente y óptimo no puede ser una tarea manual.

También se debe tomar en cuenta el tamaño de la población que se desea atender. Dentro de una universidad de dimensiones similares a la PUCP, se podría estimar una población de algunos miles de usuarios, los cuales representan un desafío en la tarea de planeación de la ruta pues se debe buscar poder satisfacer a la mayoría de ellos. Al tener una población tan grande, el usar rutas no óptimas podría perjudicar a cientos de usuarios.

Finalmente, una vez determinada un conjunto de rutas, se debe también establecer la ubicación de los paraderos. Si se posee información del lugar de residencia de los miembros de la comunidad se puede planear mejor qué zonas requieren mayor cantidad de paraderos y cuales menor número.

Las herramientas informáticas han sido usadas para resolver problemas similares en el pasado con mucho éxito. Sin embargo, estas han estado más orientadas al sistema de transporte público general. En la revisión se encontró que el algoritmo PIA (Pair Insertion Algorithm) ha resuelto un problema similar de planeación de rutas de transporte público, pero que estos resultados podrían ser mejorados si se usan como población inicial de otro algoritmo como uno genético. Por esto, se propone para el presente proyecto realizar, utilizando al algoritmo PIA, la Implementación de un algoritmo genético para elaborar un conjunto de rutas óptimas para el transporte de la comunidad universitaria desde y hacia el campus principal.

FACULTAD DE
CIENCIAS E INGENIERÍA
ESPECIALIDAD DE
INGENIERÍA INFORMÁTICA

PONTIFICIA
UNIVERSIDAD CATÓLICA
DEL PERÚ

TEMA DE TESIS PARA OPTAR EL TÍTULO DE INGENIERO INFORMÁTICO

TÍTULO: Implementación de un algoritmo genético para elaborar un conjunto de rutas óptimas para el transporte de la comunidad universitaria desde y hacia el campus principal

ÁREA: Ciencias de la Computación

ASESOR: Ing. César Augusto Aguilera Serpa

ALUMNO: Julio Rodrigo Castillo Huerta

CÓDIGO: 20100795

TEMA N°: # 684

FECHA: San Miguel, 8 de junio de 2018

PONTIFICIA UNIVERSIDAD CATÓLICA DEL PERÚ
Facultad de Ciencias e Ingeniería
7 JUN. 2018
MIGUEL MEJIA PUENTE
DECANO

DESCRIPCIÓN

En la actualidad, varias universidades como la PUCP no cuentan con un sistema de transporte privado para la comunidad universitaria a pesar de que existen propuestas y es un servicio pedido por un sector de la universidad. Los motivos son varios ya que se debe considerar presupuestos, logística y una planeación adecuada de las rutas de servicio. Este último punto es complicado de por sí pues es difícil poder encontrar un conjunto de rutas que cumplan con satisfacer la demanda de una manera óptima.

En primer lugar, se debe considerar que, en una ciudad de gran tamaño, realizar cualquier tipo de diseño de rutas es un trabajo que presenta muchos desafíos. Con todas las calles y avenidas a considerar, realizar un diseño de rutas eficiente y óptimo no puede ser una tarea manual.

También se debe tomar en cuenta el tamaño de la población que se desea atender. Dentro de una universidad de dimensiones similares a la PUCP, se podría estimar una población de algunos miles de usuarios, los cuales representan un desafío en la tarea de planeación de la ruta pues se debe buscar poder satisfacer a la mayoría de ellos. Al tener una población tan grande, el usar rutas no óptimas podría perjudicar a cientos de usuarios.

Finalmente, una vez determinada un conjunto de rutas, se debe también establecer la ubicación de los paraderos. Si se posee información del lugar de residencia de los

Av. Universitaria 1801
San Miguel, Lima – Perú

Apartado Postal 1761
Lima 100 – Perú

Teléfono:
(511) 626 2000 Anexo 4801

miembros de la comunidad se puede planear mejor qué zonas requieren mayor cantidad de paraderos y cuales menor número.

Las herramientas informáticas han sido usadas para resolver problemas similares en el pasado con mucho éxito. Sin embargo, estas han estado más orientadas al sistema de transporte público general. En la revisión se encontró que el algoritmo PIA (Pair Insertion Algorithm) ha resuelto un problema similar de planeación de rutas de transporte público, pero que estos resultados podrían ser mejorados si se usan como población inicial de otro algoritmo como uno genético. Por esto, se propone para el presente proyecto realizar, utilizando al algoritmo PIA, la Implementación de un algoritmo genético para elaborar un conjunto de rutas óptimas para el transporte de la comunidad universitaria desde y hacia el campus principal.

OBJETIVO GENERAL

El objetivo general del proyecto de fin de carrera es implementar un algoritmo genético que elabore un conjunto de rutas suficientemente aceptables para el transporte de la comunidad universitaria desde y hacia el campus principal.

OBJETIVOS ESPECÍFICOS

Los objetivos específicos son:
OE1. Definir las estructuras de almacenamiento de la información de las viviendas de los integrantes de la universidad y las estructuras de datos usadas en el algoritmo.
OE2. Diseñar un algoritmo basado en PIA (Pair Insertion Algorithm) para generar la población inicial de rutas y uno genético para el mejoramiento de las rutas.
OE3. Implementar el algoritmo basado en PIA y otro genético en un lenguaje de programación.
OE4. Realizar experimentación numérica para comparar el desempeño de algoritmo genético contra el algoritmo basado en PIA.

ALCANCE

Este proyecto de fin de carrera pretende presentar una alternativa de solución al problema de planificación de rutas de transporte de alumnos hacia su campus de estudio y hogares mediante el uso de una flota de buses. Para ello, se tomarán en cuenta ciertas variables (las más importantes) que se adecuan al proyecto. Estas

Av. Universitaria 1801
San Miguel, Lima – Perú

Apartado Postal 1761
Lima 100 – Perú

Teléfono:
(511) 626 2000 Anexo 4801

variables son: ubicación de residencia de los alumnos, tamaño de la flota, rango de distancias de rutas y nivel de cobertura de estudiantes. Luego de esta identificación de variables, se procederá con la identificación de las estructuras de datos, luego se implementarán los algoritmos basado en PIA y genético. Los cuales fueron seleccionados por su desempeño positivo en trabajos similares revisados en la bibliografía. Finalmente, se implementará la aplicación y se realizarán experimentación numérica para probar el desempeño del algoritmo.

Máximo: 100 páginas

Av. Universitaria 1801
San Miguel, Lima – Perú

Apartado Postal 1761
Lima 100 – Perú

Teléfono:
(511) 626 2000 Anexo 4801

Tabla de contenido

Índice de figuras

Índice de tablas

Índice de pseudocódigos

CAPÍTULO 1: PLANTEAMIENTO DEL PROYECTO DE FIN DE CARRERA

1 PROBLEMÁTICA

La Pontificia Universidad Católica del Perú (PUCP) es una universidad privada fundada en el año 1917. En el año 2015 fue nombrada la número 18 de Sudamérica y la única universidad peruana dentro de las 500 mejores del mundo según el QS World University Rankings (*Top Universities*, 2016). A octubre de 2015 cuenta con 19,857 estudiantes en pregrado, 6,024 en postgrado, 2,825 docentes y 2,925 de personal administrativo y obrero (*Dirección de Informática - Oficina de Estadística*, 2016). Está ubicada en Lima, la cual es una ciudad cuya área metropolitana cuenta con una población de casi 10 millones de personas y un área de 281,926 km² (INEI, 2015), lo cual la convierte en la octava más grande de América y una de las más grandes del mundo.

Al estar ubicada en una urbe de este tamaño y densidad poblacional, el transporte diario de la comunidad universitaria al campus puede resultar complicado para algunos. Según el VI Informe de Percepción Sobre Calidad de Vida de Lima Cómo Vamos, el 75.6% de la población limeña usa el transporte público para dirigirse a su centro de estudios o trabajo; de este estudio también se concluyó que un 25% de la población dedica más de 2 horas diarias al transporte y que un 49% considera al transporte público como uno de los principales problemas de la ciudad (Lima Cómo Vamos, 2015). Para aliviar estos problemas, universidades limeñas como UNMSM (Aldaba, 2010), UNI (Fuertes, 2011) y UNALM (Aguilar, 2011) cuentan con buses para el transporte de sus alumnos y trabajadores, los cuales traen como beneficios adicionales mayor seguridad y menor sobrecarga de pasajeros.

En la actualidad, la PUCP no cuenta con un sistema de transporte privado para la comunidad universitaria a pesar de que existen algunas propuestas y es un

servicio pedido por un sector de la universidad. Los motivos son varios ya que se debe considerar presupuestos, logística y una planeación adecuada de las rutas de servicio. Este último punto es complicado de por sí pues es difícil poder encontrar un conjunto de rutas que cumplan con satisfacer la demanda de una manera óptima.

En primer lugar, se debe considerar que, en una ciudad de gran tamaño, realizar cualquier tipo de diseño de rutas es un trabajo que presenta muchos desafíos (JEFFREY y otros, 1997). En particular en Lima como se mencionó anteriormente, se cuenta con una población de 9,903,935 habitantes y casi 300,000 km² de superficie, comprendidos en un total de 49 distritos de dos provincias: Lima y Callao. Con todas las calles y avenidas comprendidas en este territorio, realizar un diseño de rutas eficiente y óptimo no puede ser una tarea manual ya que tomaría demasiadas horas hombre.

En segundo lugar, también se debe tomar en cuenta el tamaño de la población que se desea atender, en este caso no son los casi 10 millones habitantes de la ciudad, sino a una fracción de las aproximadamente 30,000 personas que conforman la comunidad universitaria. Se dice que es una fracción de esa pues no todos utilizan el transporte público y los hay también quienes seguirán usando el transporte público convencional. Aun considerando esto, se podría estimar una población de algunos miles de usuarios, los cuales distribuidos en toda la ciudad representan un desafío en la tarea de planeación de la ruta pues se debe buscar poder satisfacer a la mayoría de ellos. Al tener una población tan grande, un error o el usar rutas no óptimas podrían perjudicar a cientos de usuarios.

Finalmente, una vez determinada un conjunto de rutas, se debe también establecer la ubicación de los paraderos. En esta etapa se debe considerar contar con una frecuencia de paraderos tal que no sea tan concentrada pues causaría que los buses alcancen menores velocidades y por lo tanto demoren en exceso para realizar el recorrido ni tan dispersas pues los usuarios tendrían que caminar largas distancias para poder abordar el vehículo (Ceder, 2007). Si se posee información del lugar de residencia de los miembros de la comunidad

se puede planear mejor qué zonas requieren mayor cantidad de paraderos y cuales menor número.

En conclusión, se puede apreciar que son varios los problemas asociados al desarrollo de un diseño óptimo de rutas de transporte para una universidad como la del presente caso. Las herramientas informáticas han sido usadas para resolver problemas similares en el pasado con mucho éxito.

Sin embargo, estas han estado más orientadas al sistema de transporte público general. Es decir, no consideran un destino único como lo podría necesitar un sistema de transporte para una entidad privada como una universidad. Además, es importante resaltar que lo que se busca no es crear rutas que recorran todos los puntos de demandas (las viviendas de los usuarios), sino rutas que recorran un camino que pase cerca de estas. Esto también es una diferencia con otros trabajos relacionados a diseños de rutas. También se debe considerar que, debido a la gran cantidad de posibles soluciones que existen, utilizar algoritmos con soluciones exactas no sería apropiado. La bibliografía sugiere que lo mas adecuado es usar algoritmos meta heurísticos. En la revisión se encontró que el algoritmo PIA (Pair Insertion Algorithm) ha resuelto un problema similar de planeación de rutas de transporte publico, pero que estos resultados podrían ser mejorados si se usan como población inicial de un algoritmo como uno genético. Por esto, se propone para el presente proyecto de fin de carrera realizar una herramienta utilizando un algoritmo genético y PIA que ayude al proceso de diseño de rutas para la población universitaria desde y hacia la universidad tomando como apoyo trabajos pasados y adaptándolos a las características de este problema en específico y proveyendo funcionalidades adicionales para una institución de este tipo.

2 OBJETIVO GENERAL

El objetivo general del proyecto de fin de carrera es implementar un algoritmo genético que elabore un conjunto de rutas suficientemente aceptables para el transporte de la comunidad universitaria desde y hacia el campus principal.

3 OBJETIVOS ESPECÍFICOS

- OE1: Definir las estructuras de almacenamiento de la información de las viviendas de los integrantes de la universidad y las estructuras de datos usadas en el algoritmo.

- OE2: Diseñar un algoritmo basado en PIA (Pair Insertion Algorithm) para generar la población inicial de rutas y uno genético para el mejoramiento de las rutas.

- OE3: Implementar el algoritmo basado en PIA y otro genético en un lenguaje de programación.

- OE4: Realizar experimentación numérica para comparar el desempeño de algoritmo genético contra el algoritmo basado en PIA.

4 RESULTADOS ESPERADOS

Los resultados esperados por cada objetivo son los siguientes:

- Resultado esperado 1 para el objetivo específico 1 (R1O1): Modelamiento de base de datos para ubicaciones de usuarios y rutas generadas.

- Resultado esperado 2 para el objetivo específico 1 (R2O1): Estructura de datos para representar individuo de la población de soluciones del algoritmo genético, algoritmo adaptado PIA y función objetivo.

- Resultado esperado 3 para el objetivo específico 2 (R3O2): El pseudocódigo del algoritmo PIA adaptado.

- Resultado esperado 4 para el objetivo específico 2 (R4O2): El pseudocódigo del algoritmo genético para la generación de rutas óptimas.

- Resultado esperado 5 para el objetivo específico 3 (R5O3): El módulo del algoritmo PIA implementado y del algoritmo genético implementado.

- Resultado esperado 6 para el objetivo específico 3 (R6O3): Aplicación de escritorio para configurar y ejecutar el algoritmo y poder visualizar la solución.

- Resultado esperado 7 para el objetivo específico 4 (R7O4): Resultados de la experimentación numérica que compara desempeños de los algoritmos genéticos y PIA adaptado.

5 ALCANCE Y LIMITACIONES

Este proyecto de fin de carrera pretende presentar una alternativa de solución al problema de planificación de rutas de transporte de alumnos hacia su campus de estudio y hogares mediante el uso de una flota de buses. Para ello, se tomarán en cuenta ciertas variables (las más importantes) que se adecuan al proyecto. Estas variables son: ubicación de residencia de los alumnos, tamaño de la flota, rango de distancias de rutas y nivel de cobertura de estudiantes. Luego de esta identificación de variables, se procederá con la identificación de las estructuras de datos, luego se implementarán los algoritmos basado en PIA y genético. Los cuales fueron seleccionados por su desempeño positivo en trabajos similares revisados en la bibliografía (Mauttone y Urquhart, 2009). Finalmente, se desarrollará la aplicación y se realizarán experimentación numérica para probar el desempeño del algoritmo.

Para el proyecto de fin de carrera se consideraron las siguientes limitaciones:

- El proyecto no está enfocado en la planificación de tiempos de salidas y llegadas de los buses, asignación de horarios a choferes ni monitoreo de las rutas en tiempo real como lo realizan algunos sistemas de transporte

público; debido a que estas funcionalidades son ajenas al objetivo principal del proyecto, sin embargo, podrían ser materia de proyectos complementarios

- El producto del proyecto de fin de carrera está ideado para la PUCP y la ciudad de Lima. Si bien en teoría se podría aplicar a otras universidades o ciudades, se tendría que adaptar o ver si es viable previamente, pues pueden haber variaciones como universidades con más de una sede o ciudades con particularidades geográficas distintas a las de Lima.

- El tiempo de ejecución de ambos algoritmos está sujeto a las especificaciones de hardware y software del equipo donde se ejecuten. Debido a la complejidad de los algoritmos y de la cantidad de veces que se ejecutan la diferencia entre los resultados en un equipo y otro con especificaciones distintas puede ser considerable. Por lo tanto, para realizar una comparación adecuada, todos los algoritmos se ejecutarán en la misma computadora.

- Para la implementación del proyecto, se tomarán en cuenta las avenidas principales y secundarias de la ciudad de Lima. Esto debido a que sería inadecuado que un sistema de transporte de estas características recorra o tenga paraderos en calles estrechas o poco conocidas, lo cual perjudicaría a los usuarios.

- El alcance del proyecto no contempla implementar una herramienta dentro del sistema que permita marcar áreas o caminos del mapa como ‘inhabilitados’ o ‘en mantenimiento’. En caso haya avenidas en mantenimiento en la vida real se deberá modificar el archivo de rutas que se carga a la base de datos o manualmente circunnavegar las áreas afectadas después de que el algoritmo genere la ruta.

- Finalmente, el alcance del proyecto no incluye la implementación de un sistema de información complejo que se integre con alguna otra plataforma

o que permita administrar grandes conjuntos de información. Solamente se incluirá una interfaz grafica para calibrar parámetros, cargar data y visualizar resultados.

6 JUSTIFICACIÓN

Como se vio anteriormente, el problema de diseñar un conjunto de rutas para el transporte de la comunidad universitaria puede resultar muy complicado por todas las variables que maneja; por eso, el presente proyecto se ha planteado para ser útil en este escenario. Resulta conveniente contar con el resultado de la presente investigación pues ahorraría un gran esfuerzo al momento de definir las rutas por otros métodos como el manual y además se espera poder contar con mejores resultados. Los beneficios no solo se verían en la planificación del servicio de transporte publico sino que también en la aplicación de este, contar con un buen sistema de rutas ahorrará tiempo y problemas a la población de la universidad en su día a día. En este sentido, los beneficiarios no son solamente los directivos de la universidad encargados de la definición de las rutas de servicio sino también finalmente los alumnos, profesores y personal administrativo de la universidad.

Así, el proyecto de fin de carrera servirá para ayudar a la elaboración de rutas de transporte público para la comunidad de la PUCP. Sin embargo, se puede contar con productos derivados adaptando la solución a otras universidades o instituciones que requieran ofrecer un servicio de transporte desde sus locales o establecimientos; es en este sentido también que la investigación resulta conveniente. Finalmente, el proyecto además aportará un valor cognitivo o teórico pues estará disponible para que futuros proyectos o investigaciones puedan basar sus trabajos en los resultados del presente.

7 VIABILIDAD

Viabilidad técnica

En este punto es viable el proyecto pues los conocimientos técnicos básicos sobre desarrollo de algoritmos, programación de aplicaciones, creación de

interfaces gráficas, estadística y manejo de proyectos informáticos fueron cubiertos durante el transcurso de la carrera. Un punto que considerar es el diseño de algoritmos como PIA y genético que requerirán un estudio de la literatura y estado del arte existente, la cual es abundante y alcanzable gracias a la base cognitiva previamente aprendida.

Respecto a las herramientas escogidas, cada una fue escogida tomando en cuenta criterios que aseguraran mantener la viabilidad del proyecto. StarUML es una herramienta usada en los cursos de la carrera y cuenta con gran usabilidad y practicidad para el desarrollo de diagramas (StarUML, 2017). Postgres SQL fue elegido pues se cuenta con experiencia previa con esta alternativa, además su soporte multiplataforma y activa comunidad son atractivos para facilitar las tareas del proyecto (PostgreSQL, 2017); así como también el soporte para complementos como PostGIS también elegido por presentar funciones relevantes para la investigación (PostGIS, 2017). El lenguaje Java ha sido también usado durante la carrera y por lo tanto ya se posee conocimientos sobre cómo trabajar con este, es además un lenguaje bastante potente y flexible; y su gran comunidad de usuarios provee una gran fuente de referencia (Oracle, 2017). NetBeans es uno de los IDEs más populares para el lenguaje Java y es también aplicado durante el desarrollo de la carrera (NetBeans, 2017). Finalmente, el API de Google Maps fue elegido pues cuenta con una detallada información de la ciudad de Lima, además presenta una representación visualmente agradable del mapa y buenas opciones de licenciamiento (Google, 2017). En conclusión, en los aspectos técnicos del proyecto, este sí es viable.

Viabilidad temporal

El tiempo es una de las mayores limitantes del proyecto de fin de carrera; por eso el alcance definido ha sido determinado tomando en cuenta esto. Se puede apeciar, en el siguiente capítulo, el plan del proyecto con las actividades y los tiempos considerados para cada una. Con esto como sustento se puede apreciar que el presente trabajo de investigación es viable en el tiempo.

Viabilidad económica

Los principales costos en un proyecto de software o investigación son hardware, software y mano de obra. En el caso de hardware, estos costos ya han sido incurridos previamente y además la universidad cuenta con equipos de libre disponibilidad. Las licencias de software como el sistema operativo y software de pago también ya han sido adquiridas de antemano y el resto cuenta con licencias libres y gratuitas. Finalmente, la mano de obra consistiría en el tesista que realiza el trabajo sin remuneración y el asesor que recibe compensación a través de la universidad. Por lo tanto, con este esquema se puede concluir que el proyecto es viable económicamente.

Análisis de necesidades

Respecto a las necesidades, se ha identificado algunas de las más importantes. El juicio o conocimiento de algún experto en temas afines a algoritmos genéticos y proyectos de investigación están provistos por el asesor y los profesores de la especialidad en la universidad. Para la necesidad de contar con información sobre la vivienda de miembros de la comunidad universitaria se ha coordinado con un contacto dentro de la institución para proveer data y además se cuenta con un plan de contingencia de elaboración de data manual en caso esta finalmente no pueda ser provista. Otras necesidades no han sido detectadas o son irrelevantes para el análisis de viabilidad, por la tanto se concluye que el proyecto es viable también con las necesidades actuales.

CAPÍTULO 2: CONCEPTOS Y ESTADO DEL ARTE

1 ESTADO DEL ARTE

Como se mencionó anteriormente, el problema de diseño de rutas es uno que ya ha sido tratado de solucionar anteriormente con herramientas informáticas. En la presente variante la cual es la de servir a múltiples usuarios con un solo destino sin pasar directamente por sus ubicaciones no se ha encontrado mayor referencia. Sin embargo, los trabajos investigación de generación de rutas de transporte público general usando algoritmos genéticos pueden ser usados en el estado del arte pues se manejan bastantes conceptos similares. También se presentaran alternativas de software que trabajen con problemas de rutas para tomar como referencia para la aplicación de escritorio.

1.1 Investigaciones

Esta sección presenta investigaciones publicadas en publicaciones científicas (papers) o revistas académicas que trabajen el problema utilizando un algoritmo genético y en orden cronológico:

- En 1998, Pattnaik, Mohan y Tom propusieron un algoritmo genético para el diseño de una red de rutas de buses de transporte público (Pattnaik, Mohan y Tom, 1998). Su trabajo fue formulado como un problema de optimización minimizando un costo incurrido total (tomando en cuenta tanto el costo para los usuarios como para el operador). Como es usual en un algoritmo genético, primero se generó una población inicial la cual fue mejorada usando el algoritmo en sí.

  Se trabajó con dos modelos: cadena de longitud fija y cadena de longitud variable. La primera resulto ser una propuesta simple y que daba mejores soluciones, pero requería mayor tiempo computacional. Mientras que por otro lado la segunda podía manejar simultáneamente

la selección del tamaño del conjunto de rutas y del conjunto de rutas, pero requería una codificación compleja. Finalmente propusieron que un hibrido entre ambos modelos podría ser analizado para ser una solución adecuada para el problema de determinación de rutas.

- En 2002, Bielli, Caramia y Carotenuto trabajaron en un nuevo método para calcular valores de función de aptitud en algoritmos genéticos para una optimización de red de buses (Bielli, Caramia y Carotenuto, 2002). También usando la estructura de un algoritmo genético, las poblaciones iban siendo evaluadas con el cálculo de unos indicadores de desempeño obtenidos del análisis de la asignación de oferta y demanda de las redes de transporte. Así, la aptitud se hallaba en términos tanto de oferta como de demanda.

  Finalmente, en el trabajo logran mejorar una función que evalúa la aptitud añadiendo nuevos factores para la determinación del desempeño. Esta mejora logra optimizar la función aptitud en un 90%.

- En 2011, Szeto y Wu atacan el problema del diseño de rutas de buses y de frecuencia de salida de estos utilizando un algoritmo genético (Szeto y Wu, 2011). Su enfoque se basa en minimizar el número de transbordos que existe en la red de buses. En la función objetivo por tanto intentan realizar un balance entre el número de transbordos y el tiempo total de los viajes.

  Finalmente, proponen un mecanismo de control de diversidad para mejorar el desempeño del algoritmo genético. Comparado con el diseño de rutas actual de la ciudad donde realizaron el experimento el número de transferencias se redujo en un 20.9% y el tiempo de viaje total en un 22.7%.

- En 2013, Fan, Gurmu y Haile utilizan un enfoque meta-heurístico de dos niveles para resolver el problema de diseño de rutas de buses (Fan,

Gurmu y Haile, 2013). Ellos proponen un modelo de programación lineal para la solución inicial, luego se presenta una solución metaheurística en dos niveles: el primero encuentra patrones de rutas factibles y el segundo evalúa dichas rutas y las retroalimenta. Finalmente, un algoritmo genético es usado para optimizar las rutas.

- En 2014, Ciaffi, Cipriani, Petrelli y Ušpalytė trabajaron el problema de diseño de redes de buses usando un algoritmo genético cuya función objetivo era definida como la suma de los costos del operador con los costos del usuario (Francesco Ciaffia y otros, 2013). Usaron para esto una data de entrada consistente en una matriz de demanda de transporte público, características de la red de transporte y costos operativos y de usuarios. El algoritmo genético fue implementado en el lenguaje C# como un algoritmo genético paralelo mientras que la evaluación de la aptitud era calculada, luego ambos eran simulados usando el software EMME. La solución también fue capaz de determinar una frecuencia de salida de buses adecuada para la demanda.

- En 2014, Cooper, John, Lewis, Mumford y Olden publican un estudio sobre optimización de problemas de diseño de redes de transporte público a gran escala utilizando algoritmos evolucionarios multi-objetivos paralelos de modo mixto. Ellos se basan en un modelo ya existente que usa un algoritmo genético con un operador genético mejorado. Luego, este modelo es implementado dentro de un algoritmo genético multi-objetivo paralelo.

  En las conclusiones, se explica que la calidad de las soluciones es mejor cuando se utiliza una política de selección de individuos aleatoria contra solamente usar una política elitista. También se señala que el uso de algoritmos genéticos es apropiado para hallar soluciones a este tipo de problemas para una escala tamaño ciudad en un tiempo razonable.

- En 2015, Barrantes realiza un algoritmo recocido simulado para el diseño de una red de rutas de transporte público para la ciudad de Lima, distrito de cercado (Barrantes, 2015). En su proyecto de fin de carrera, compara los resultados de la implementación de este algoritmo con el algoritmo PIA. Finalmente, el recocido simulado logró un mejor desempeño que el PIA pero se debe tomar en cuenta que la solución solo trabajaba con zonas de origen y destino mas no con las calles que recorrían estas zonas, por lo tanto no se obtenía un recorrido tan detallado.

1.2 Software

Esta sección presenta algunas herramientas de software actualmente disponibles relevantes al presente proyecto de fin de carrera. No necesariamente resuelven el problema actual pero algunas características pueden ser tomadas en cuenta para el desarrollo del proyecto.

- TransCAD (Caliper, 2017): Es un software desarrollado por Caliper. Está diseñado específicamente para el uso de profesionales del sector transporte para almacenar, mostrar, manejar y analizar data de transporte. TransCad cuenta con herramientas para indicar rutas de camiones, rieles o buses que viajan de lugar a lugar. También es útil para controlar tiempos y distancias de viajes. Cuenta con versiones de $4,000 y $12,000.

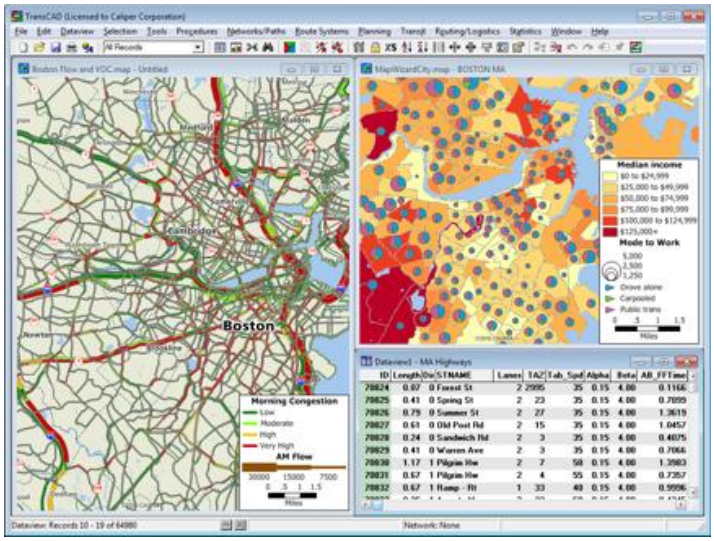

Figura 1. Captura de pantalla de sistema TransCAD

- Emme (INRO, 2017): Es un software desarrollado por INRO. Provee herramientas para el modelamiento de demanda para sistemas de transportes nacionales, regionales y urbanos. Es capaz de predecir demanda de viajes, planear el tránsito, planear el tráfico y realizar análisis económico, de emisión y ambientales.

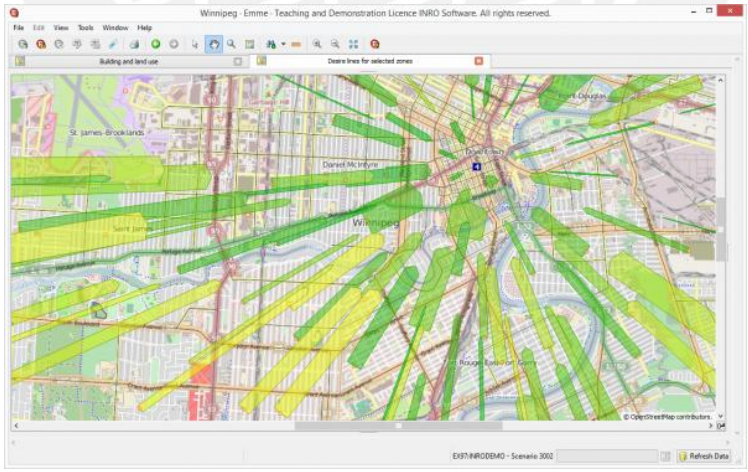

Figura 2. Captura de pantalla de sistema Emme

- Visum (PTV, 2017): Es un software desarrollado por el grupo PTV. Cuenta con la capacidad de modelar a los usuarios de las vías públicas

y sus interacciones. Puede modelar redes de transporte y demanda de viajes también como analizar cambios de tránsito y planificar servicios de transporte público. Esta última función incluye crear una red de datos de oferta y demanda, representar gráficamente rutas individuales y mostrar gráficamente el volumen de rutas por línea.

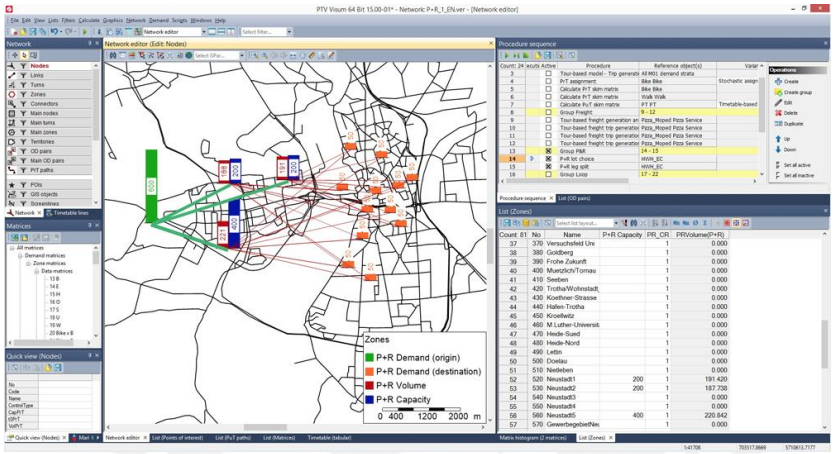

Figura 3. Captura de pantalla de sistema Visum

2 MARCO CONCEPTUAL

En la siguiente sección se definirán los términos y conceptos necesarios para entender la problemática y la solución del proyecto de fin de carrera.

2.1 Conceptos relacionados al transporte público

El transporte público es un servicio de transporte compartido de pasajeros el cual está disponible para el uso del público general, a diferencia de un taxi o el transporte privado que no es compartido por extraños sin un acuerdo previo. Las formas de transporte público incluyen los buses, monorrieles, metros, entre otros. El concepto de transporte público también aplica a largas distancias como ocurre con los aviones.

Para el problema del presente proyecto de fin de carrera, se detallará más a fondo los conceptos del sistema de transporte público en buses. Un sistema de buses se define por su flota, sus rutas y sus paraderos.

Flota

La flota es el conjunto de vehículos de una empresa (Real Academia Española, 2014), en este caso el conjunto de buses disponibles para prestar el servicio. La ruta es el camino o dirección que se toma para un propósito (Real Academia Española, 2014), en el contexto actual es el recorrido preestablecido que servirá un subconjunto de la flota, a este subconjunto de la flota que sirve a una misma ruta se le conoce como una línea.

Paraderos

Los paraderos son las paradas de autobuses y tranvías (Real Academia Española, 2014), los puntos preestablecidos dentro de la ruta donde se realiza el recojo y desembarco de pasajeros.

Es importante mencionar que en el presente proyecto de fin de carrera la solución no sería aplicada a un sistema de transporte público general, sino a uno que solo sirva a la comunidad universitario. A pesar de esto los conceptos que se aplican son los mismos con la única diferencia que los paraderos solo realizarían el recojo en la dirección de ida y solo desembarco en la dirección de vuelta.

Heurística

Una heurística, en el campo de la informática, es una técnica de resolución de problemas que permiten alcanzar un “buen” resultado en un tiempo aceptable usando un conjunto de pautas especificas del problema en lugar de dar una secuencia detallada de pasos para llegar a una solución como lo hace un algoritmo tradicional. Se usa esta alternativa principalmente cuando se trabaja con un problema demasiado complejo para un algoritmo tradicional, como los problemas NP-complejo, ya que tomaría demasiado tiempo alcanzar una solución exacta. Los algoritmos heurísticos no garantizan alcanzar la mejor

solución pero si están bien diseñados pueden alcanzar una solución suficientemente buena (Kokash, 2005).

2.2 Conceptos generales de meta-heurísticas

Un algoritmo meta-heurístico, de forma similar a una heurística, no promete alcanzar la solución más óptima o la solución exacta sino que puede hallar una solución lo suficientemente aceptable. A diferencia de la heurística, una meta-heurística no depende de la naturaleza del problema, en sí el algoritmo no conoce sobre las características del problema que trata de resolver, es por lo tanto una solución general que se puede usar para una mayor variedad de problemas aunque no siempre pueda ser la mejor forma de resolverlos (Bianchi y otros, 2009).

Para el presente proyecto de fin de carrera se propone utilizar un algoritmo genético que requerirá otro algoritmo para generar la población inicial, en este caso una adaptación del algoritmo PIA. A continuación, se presentan los conceptos de ambos.

Algoritmo genético

Un algoritmo genético es una búsqueda meta-heurística a una solución que se basa en imitar el proceso de selección natural (Schmitt, 2001). En este, se cuenta con una población de posibles soluciones, cada una con sus propias características, la cual va evolucionando hacia mejores soluciones. La población inicial puede generarse a partir de un proceso aleatorio o a partir de otro algoritmo. En cada iteración del algoritmo se evalúa a cada individuo de la solución contra una función objetivo para determinar su nivel de aptitud. Los individuos con mayor nivel de aptitud son entonces seleccionados de manera estocástica y sus características son recombinadas o mutadas. La nueva población resultante de esta etapa de recombinación o mutación es usada en la siguiente iteración del algoritmo. Normalmente el algoritmo termina cuando

se llega a un número tope de iteraciones o se ha alcanzado el nivel de aptitud deseado.

Un algoritmo genético requiere las siguientes estructuras: una representación genética y una función objetivo.

La representación genética se refiere a la forma de representar a los individuos o soluciones, en la representación genética se debe incluir la información de las características de los individuos. Un ejemplo simple podría ser un arreglo de unos y ceros indicando la presencia de ciertas características en los individuos pero las representaciones pueden llegar a ser más complejas (Whitley, 1994).

La función objetivo se usa para determinar qué tan apto es un individuo basado en lo que se desea alcanzar en la solución final. La función objetivo evalúa el desempeño de los individuos y es así como se puede determinar a los más aptos. Un ejemplo sencillo sería un cálculo basado en los valores unos y ceros del ejemplo anterior, pero también se puede llegar a tener funciones objetivos complejas como simulaciones de sistemas.

Formalmente, las etapas de un algoritmo genético, tal como se muestra en la Figura 1, son las siguientes (Schmitt, 2001):

- Inicialización: Formar la población inicial de posibles soluciones. Esta se puede generar de forma aleatoria o utilizando un algoritmo más simple que el genético. Dependiendo de la naturaleza del problema la población puede ser de cientos o miles de individuos.
- Selección: Dentro de cada iteración, el proceso de selección consiste en ejecutar la función objetivo en cada uno de los individuos para determinar su nivel de aptitud. Normalmente, los individuos más aptos son seleccionados para la siguiente iteración del algoritmo, pero también se pueden elegir otros individuos de manera aleatoria.
- Operaciones genéticas: Es la etapa en que se altera el subconjunto de la población seleccionada en la etapa anterior para obtener una población nueva, puede ser por los métodos de Crossover o Mutación. El método de Crossover consiste en escoger dos individuos de los

seleccionados en la etapa anterior y combinar las características de ambos para obtener un nuevo individuo. El proceso de Mutación consiste en alterar algunas características de un individuo para obtener uno nuevo modificado

- Terminación: Es la etapa en la que se define que el algoritmo ya no ejecutará más iteraciones. Esto se puede determinar cuándo se obtiene una solución que satisfaga ciertos criterios, se llegue a un número determinado de iteraciones, cuando las iteraciones ya no producen resultados mejores o una mezcla de las anteriores.

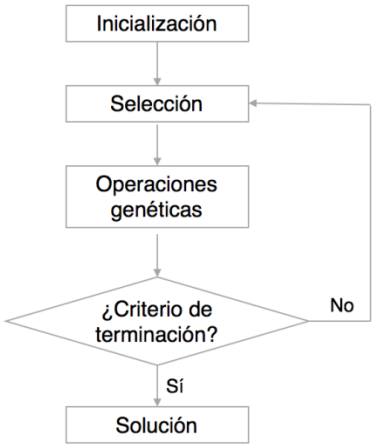

Figura 4. Flujograma de un algoritmo genético.

Algoritmo PIA (Pair Insertion Algorithm)

Es un algoritmo para diseño de rutas de transporte desarrollado por Antonio Mauttone y María E. Urquhart (Mauttone y Urquhart, 2009). Este algoritmo usa una estructura de tipo grafo en la cual cada vértice es una zona dentro de la ciudad y las aristas corresponden a las demandas de transporte entre las zonas. Las rutas se crean uniendo los vértices con mayor demanda. Es un proceso iterativo, se empieza sin ninguna ruta y en cada iteración se determina qué par de zonas tienen la mayor demanda no atendida, entonces se evalúa el costo de crear una nueva ruta directa entre las dos zonas o adherir la ruta a una ruta ya existente.

El concepto del algoritmo se basa en la satisfacción de las mayores demandas de transporte entre los vértices o zonas de la ciudad. Indican los autores que este algoritmo es un buen candidato para ser un generador de soluciones iniciales para un algoritmo de naturaleza evolutiva como lo puede ser un algoritmo genético (Mauttone y Urquhart, 2009).

3 HERRAMIENTAS, MÉTODOS, METODOLOGÍAS Y PROCEDIMIENTOS

En esta sección, se presenta las herramientas, métodos, metodologías y procedimientos que se usarán para llevar a cabo el objetivo del proyecto de fin de carrera. En la Tabla 1 se muestran por cada uno de los resultados esperados:

Tabla 1. Resultados esperados y herramientas a usar.

Resultado esperado	**Herramientas, métodos, metodologías y procedimientos**
R1O1: Modelamiento de base de datos para ubicaciones de usuarios y rutas generadas.	StarUML
R2O1: Estructura de datos para representar individuo de la población de soluciones del algoritmo genético, algoritmo adaptado PIA y función objetivo.	StarUML
R5O3: El módulo del algoritmo PIA implementado y del algoritmo genético implementado	NetBeans Java
R6O3: Aplicación de escritorio para configurar y ejecutar el algoritmo y poder	Google Maps API NetBeans

visualizar la solución.	Java
R7O4: Resultados de la experimentación numérica que compara desempeños de los algoritmos genéticos y PIA adaptado.	Microsoft Excel Prueba de Kolmogórov-Smirnov Prueba Z

3.1 Herramientas

StarUML

Es un software que permite realizar diagramas UML (Unified Modeling Language) como diagramas de Clases, de Casos de Usos, Entidad Relación, etc. Previamente contaba con licencia GNU GPL, sin embargo, desde el 2014 una versión con licencia propietaria salió disponible al público. Soporta el estándar UML 2.0. Para el caso de un DER, soporta la notación de “pata de gallo” en lugar de la notación de Chan (StarUML, 2017). Esta herramienta ayudará al modelamiento de las estructuras de datos y de la base de datos, es decir los resultados R1O1 y R2O1.

NetBeans

Es una plataforma de desarrollo de software principalmente para aplicaciones en Java pero también soporta lenguajes como PHP, C y C++. Es una herramienta multiplataforma que soporta Linux, Mac OS X y Windows ya que utiliza la máquina virtual de Java, la cual tiene soporte en estas plataformas. Permita además desarrollar la interfaz gráfica de la aplicación con herramientas visuales de Java Swing (NetBeans, 2017). Se usará en el presente proyecto de fin de carrera para la implementación de los algoritmos del resultado R5O3 y de la aplicación de escritorio del resultado R6O3.

Java

Es un lenguaje de programación de propósito general, estructurado, imperativo y orientado a objetos. Se usa tanto para aplicaciones de escritorio como para

aplicaciones Web y móviles. Una de sus principales características es su soporte multiplataforma a través de la máquina virtual de Java, la cual permite al desarrollador escribir un solo código para todos los sistemas operativos. Cuenta con una gran comunidad de desarrolladores lo cual permite obtener referencias fácilmente (Oracle, 2017). Por esta última característica, Java cuenta con una extensa documentación y librerías que facilitan la implementación de algoritmos complejos lo cual resultará útil para el resultado R5O3 y para la aplicación de escritorio que el resultado R6O3 requiere.

Google Maps API

Es una interfaz de programación de aplicación que permite acceder a funciones de Google Maps, el cual es un servicio de Google para trabajar con mapas geográficos. Ofrece vistas de satélite, de calles e incluso puede mostrar el estado actual del tráfico en algunas ciudades, así como también funciones mas avanzadas (Google, 2017). Cuenta con una versión gratuita, la cual se utilizará para integrar las funciones que ofrece esta API a la aplicación de escritorio del presente proyecto para poder mostrar gráficamente el mapa de la ciudad de Lima y las rutas generadas para poder cumplir el resultado R6O3.

Microsoft Excel

Es una herramienta de ofimática que permite trabajar con hojas de cálculo. Estas hojas proveen funciones matemáticas y estadísticas que para analizar conjuntos de datos (Microsoft, 2017). Las funciones para hallar la media, desviación estándar, varianza, entre otros serán aprovechadas para poder desarrollar la experimentación numérica del resultado esperado R7O4.

3.2 Métodos y procedimientos

Para evaluar el desempeño de los algoritmos aplicados en el proyecto, se necesitará comparar las medias de las poblaciones que conforman los resultados de varias iteraciones de los algoritmos. Para esto, se realizará una prueba de comparación de dos tratamientos, la cual está compuesta por las siguientes etapas:

Prueba de Kolmogórov-Smirnov

Es una prueba no paramétrica estadística de la igualdad de distribuciones probabilísticas continuas usada para comparar una muestra con una distribución. La prueba cuantifica una distancia entre la función de distribución de la muestra y la función de distribución de referencia; así se pueden aceptar o rechazar las hipótesis sobre si una muestra sigue la distribución esperada (Corder y Foreman, 2009). En el presente proyecto de fin de carrera, se usará esta prueba para contar con la certeza de que los resultados obtenidos sigan una distribución normal, lo cual permitirá aplicar la prueba estadística Z.

Prueba Z

Esta prueba permite determinar cual de dos muestras con distribución normal posee una mayor media, así se puede determinar si el resultado del algoritmo mejora el resultado del algoritmo PIA. Esta prueba es recomendada para cantidades de datos mayores a 30, a diferencia de la prueba T de Student.

3.3 Metodologías

Con respecto al proyecto en general se contará con la siguiente metodología.

PMBOK Quinta edición

La Guía de los Fundamentos para la Dirección de Proyectos (PMBOK) es un conjunto de guías y estándares para gestión de proyectos desarrollado por el PMI (Project Management Institute), no es en sí una metodología, pero al aplicarla en un proyecto se puede contar con una forma de trabajo que ayudará con las tareas de gestión del mismo. El PMBOK especifica en su quinta edición un total de 47 procesos repartidos en 5 agrupaciones de procesos y 10 áreas del conocimiento (Project Management Institute, 2013). Para el presente proyecto no se pretende aplicar los 47 procesos debido a las necesidades y limitaciones del proyecto. Los procesos por utilizar serán solamente los siguientes:

- Gestión de Alcance: Definir alcance y crear el EDT.

- Gestión del Tiempo: Definir las actividades, secuenciar las actividades, estimar la duración de las actividades, desarrollar el cronograma y controlar el cronograma.
- Gestión de la Comunicación: Planificar, gestionar y controlar la comunicación.
- Gestión de Riesgos: Identificar los riesgos, realizar análisis cuantitativo y cualitativo de los riesgos, planificar respuesta a los riesgos y controlar los riesgos.

Como se aprecia, solo se aplicarán procesos de 4 de las 10 áreas de conocimiento del PMBOK, en su mayoría de los grupos de proceso de Planificación y Control. Se escogieron estas por ser las más útiles para el proyecto y como indica el mismo PMBOK no se consideró aquellas que no aplican a la realidad del proyecto.

4 PLAN DE PROYECTO

Las actividades identificadas para llevar a cabo el proyecto de fin de carrera se pueden apreciar en la siguiente Estructura de Descomposición del Trabajo (EDT):

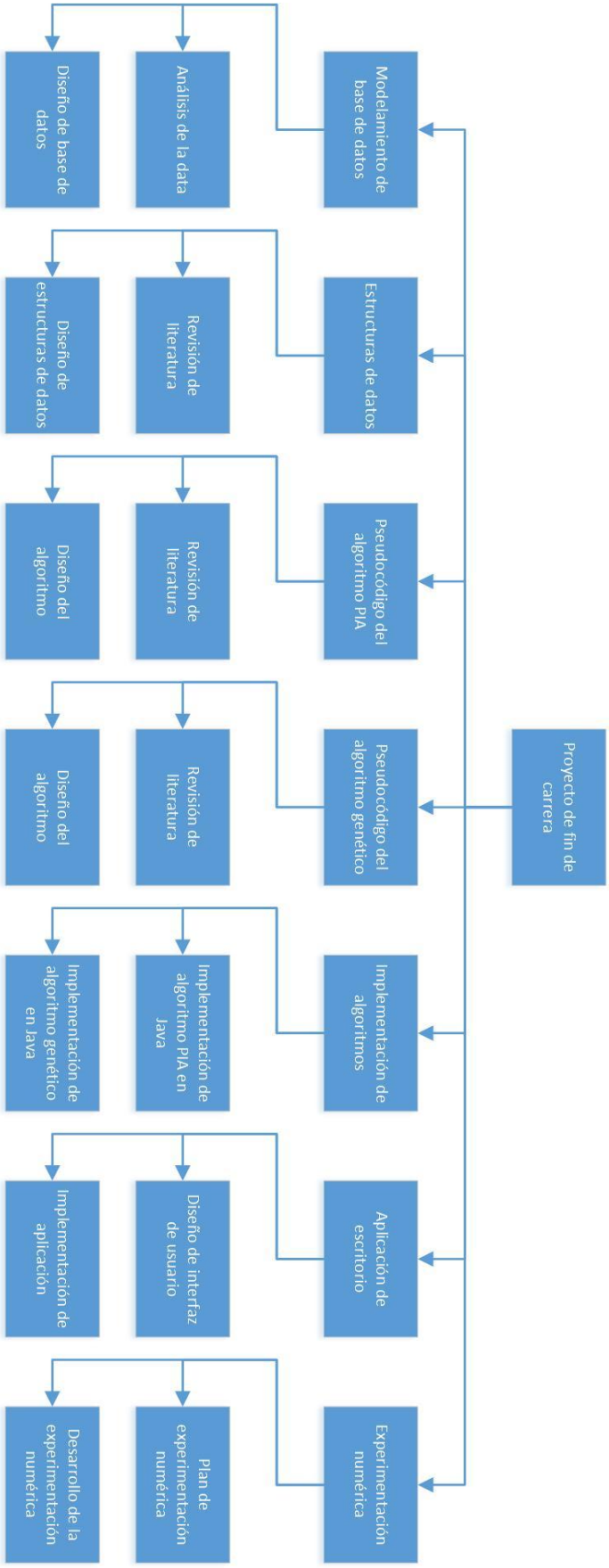

Figura 5. EDT del proyecto de fin de carrera

Los tiempos estimados para las actividades se encuentran en la Tabla 2 a continuación:

Tabla 2. Tiempos de las actividades del proyecto de fin de carrera.

Actividad	**Duración**	**Inicio**	**Fin**
Modelamiento de base de datos	3 días	Lun 14/08/17	Mie 16/08/17
Análisis de la data	1 día	Lun 14/08/17	Lun 14/08/17
Diseño de base de datos	2 días	Mar 15/08/17	Mie 16/08/17
Estructuras de datos	3 días	Jue 17/08/17	Lun 21/08/17
Revisión de literatura	1 día	Jue 17/08/17	Jue 17/08/17
Diseño de estructuras de datos	2 días	Vie 18/08/17	Lun 21/08/17
Pseudocódigo del algoritmo PIA	4 días	Mar 22/08/17	Vie 25/08/17
Revisión de literatura	1 día	Mar 22/08/17	Mar 22/08/17
Diseño del algoritmo	3 días	Mie 23/08/17	Vie 25/08/17
Pseudocódigo del algoritmo genético	5 días	Lun 28/08/17	Vie 1/09/17
Revisión de literatura	2 días	Lun 28/08/17	Mar 29/08/17
Diseño del algoritmo	3 días	Mie 30/08/17	Vie 1/09/17
Implementación de algoritmos	18 días	Lun 4/09/17	Mie 27/09/17
Implementación de algoritmo PIA en Java	6 días	Lun 4/09/17	Lun 11/09/17
Implementación de algoritmo genético en Java	12 días	Mar 12/09/17	Mie 27/09/17
Aplicación de escritorio	5 días	Jue 28/09/17	Mie 4/10/17
Diseño de interfaz de usuario	1 día	Jue 28/09/17	Jue 28/09/17
Implementación de aplicación	4 días	Vie 29/09/17	Mie 4/10/17
Experimentación numérica	5 días	Jue 5/10/17	Mie 11/10/17
Plan de experimentación numérica	2 días	Jue 5/10/17	Vie 6/10/17
Desarrollo de la experimentación numérica	3 días	Lun 9/10/17	Mie 11/10/17

En total el desarrollo del proyecto tomará desde el 14 de agosto hasta el 11 de octubre de 2017. En este tiempo, se llevará a cabo de forma paralela la documentación de los resultados obtenidos y la gestión del proyecto.

5 RIESGOS IDENTIFICADOS

Los riesgos identificados para el presente proyecto de fin de carrera junto con su nivel de impacto y medidas correctivas para mitigación se muestran en la Tabla 3 a continuación:

Tabla 3. Riesgos del proyecto de fin de carrera.

Riesgo	**Probabilidad**	**Impacto**	**Exposición = Prob x Impacto**	**Medidas correctivas para mitigación**
No contar con la información sobre la vivienda de los miembros de la comunidad universitaria para desarrollar rutas reales	media	medio	Exposición media	Trabajar con data creada manualmente, lo cual podría tomar un tiempo considerable
Pérdida total o parcial de los avances del proyecto de fin de carrera	baja	alto	Exposición media	Realizar copias de seguridad de los archivos en la nube y contar con un sistema de control de versiones
Curva de aprendizaje alta para las tecnologías a usar	media	medio	Exposición media	Conseguir referencia de las tecnologías a usar previo al inicio del desarrollo
Inadecuado diseño de los algoritmos	media	alto	Exposición alta	Recibir retroalimentación constante de un experto en el tema

Tabla 4. Escalas de probabilidad e impacto.

		Impacto		
	Exposición	*Bajo*	*Medio*	*Alto*
Probabilidad	*Baja*	Baja	Baja	Media
	Media	Baja	Media	Alta
	Alta	Media	Alta	Alta

CAPÍTULO 3: ESTRUCTURAS DE ALMACENAMIENTO DE LA INFORMACIÓN Y ESTRUCTURAS DE DATOS DEL ALGORITMO

Este capítulo muestra los resultados del primer objetivo específico, el cual se planteó como Determinar las estructuras de almacenamiento de la información de las viviendas de los integrantes de la universidad y las estructuras de datos usadas en el algoritmo. De este objetivo se obtienen dos resultados: Modelamiento de base de datos para ubicaciones de usuarios y rutas generadas y Estructura de datos para representar individuo de la población de soluciones del algoritmo genético, algoritmo adaptado PIA y función objetivo.

1 Modelamiento de base de datos para ubicaciones de usuarios y rutas generadas

La base de datos se ha diseñado para poder contar con varias simulaciones cada una con diferentes usuarios del servicio de transporte, a su vez cada simulación puede tener varias soluciones, dependiendo de la configuración que se haya especificado. Finalmente, cada solución detalla la ruta como un conjunto de aristas ordenadas indicando cuales son paraderos y cuál es el valor fitness alcanzado. En la Figura 6, se presenta el diseño de la base de datos.

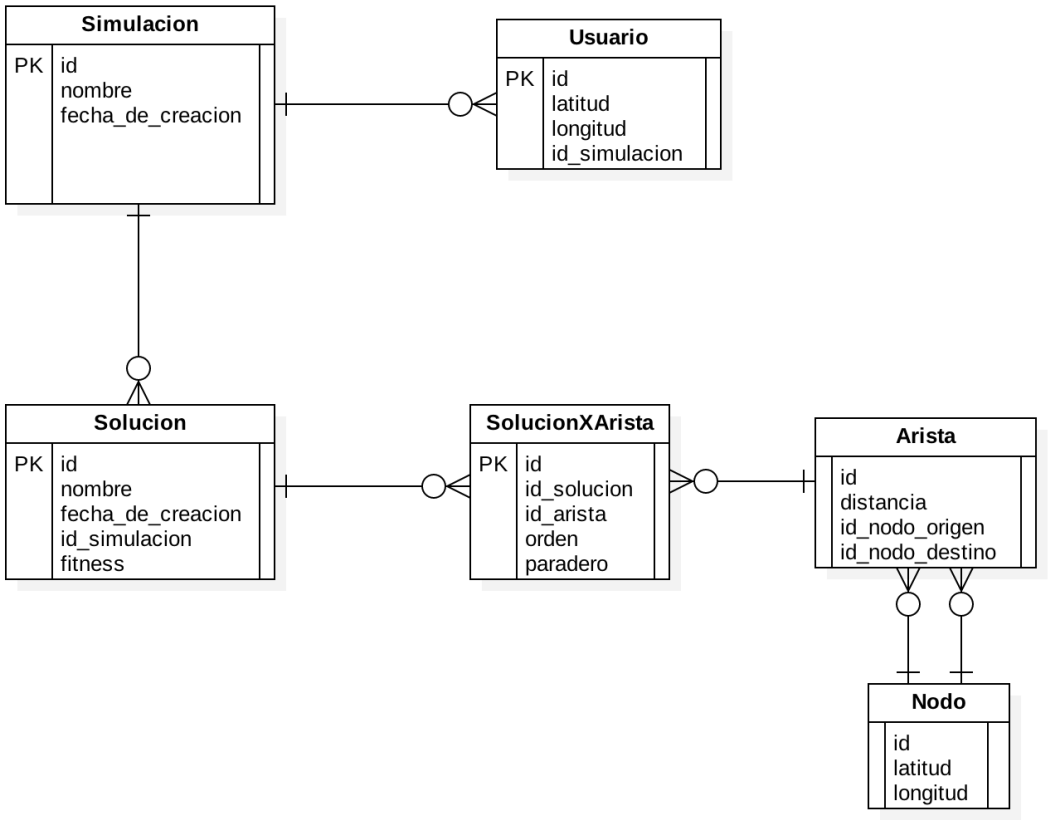

Figura 6. Diseño de la base de datos

2 Estructura de datos para representar individuo de la población de soluciones del algoritmo genético, algoritmo adaptado PIA y función objetivo

Esta sección se divide en dos partes: las estructuras de datos y la definición de la función objetivo.

2.1 Estructuras de Datos

Las estructuras de datos han sido diseñadas tomando en cuenta los conceptos relacionados a la teoría de algoritmo genéticos y teoría de grafos. También se ha tomado la consideración de que guarden correspondencia con el diseño de base de datos para poder contar con una integración lo más directa posible.

Para la representación de los cromosomas se ha considerado una estructura de datos compatible con la teoría de grafos y que además permita realizar las funciones de crossover y mutación. Por este motivo, una arista sería la representación del cromosoma en el algoritmo genético.

Clase Arista {

Nodo nodo_origen

Nodo nodo_destino

Real distancia

Booleano paradero

}

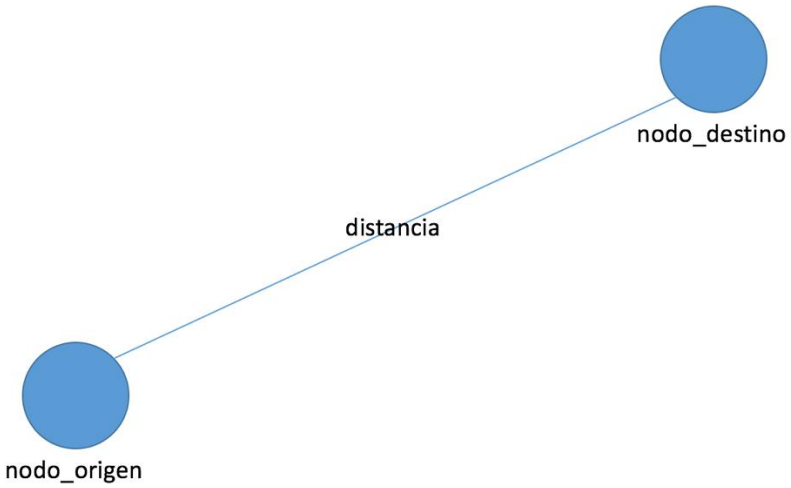

Figura 7. Representación gráfica de un cromosoma

La función de crossover en este caso consiste en reemplazar aristas de algún individuo por las de otro. La mutación consistiría en cambiar el nodo_origen o el nodo_destino. Para estas dos funciones se debe verificar que, una vez alterados los cromosomas, la ruta sigua siendo continua, si no es el caso se procederá a completar las aristas faltantes para asegurar la continuidad de la ruta.

La representación de un individuo corresponde al conjunto de rutas que atenderán a la comunidad universitaria, cada ruta es esta conformada por los

cromosomas, en este caso un arreglo de Aristas ordenadas que definan la ruta a seguir. Además, almacena el valor de la evaluación de la función objetivo.

Clase Individuo {

Lista<Ruta> rutas

Real fitness

}

Clase Ruta {

Lista<Arista> aristas

}

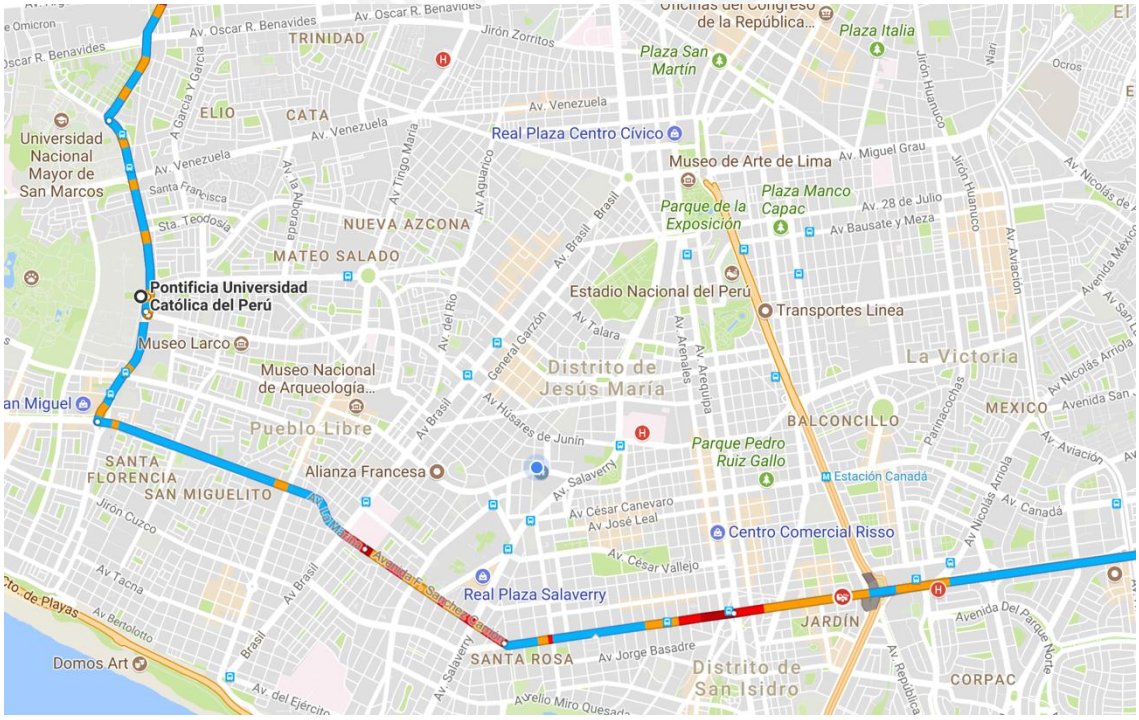

Figura 8. Representación grafica de un individuo con dos rutas

Para propósitos del algoritmo genético, esta representación se puede visualizar de como una matriz bidimensional de la siguiente manera:

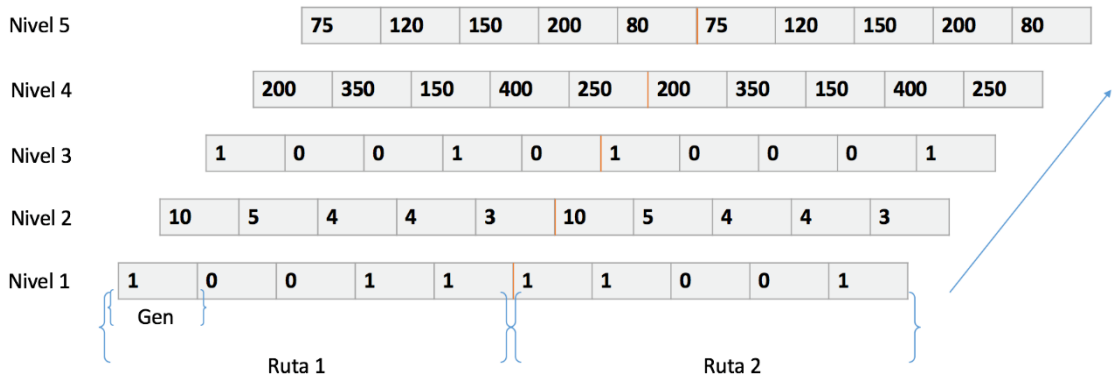

Figura 9. Representación matricial de un individuo con dos rutas

El primer nivel contiene las rutas que conforman el individuo, cada ruta es un arreglo de ceros y unos, cada uno es un gen que indica si una arista (posición en el arreglo) forma parte de la ruta. En el segundo nivel se tiene la cantidad de miembros de la comunidad que son atendidos por la arista. El tercer nivel indica si la arista posee o no un paradero con ceros y unos. El cuarto nivel tiene la información de la distancia de la arista correspondiente. Finalmente, el quinto nivel tiene la distancia promedio de los usuarios atendidos hacia el paradero.

Con esta representación se puede comprender como la mutación es cambiar el valor 1 o 0 del primer nivel de la matriz (verificando abominaciones) y como la operación crossover consiste en intercambiar rutas entre dos individuos.

2.2 Función Objetivo

El objetivo del algoritmo es encontrar la ruta que satisfaga a la mayor cantidad de usuarios de la manera más eficiente, por ello la función objetivo se plantea para que se maximice el valor de la siguiente formula, pues en el numerador se cuenta con la cantidad de usuarios atendidos y en el denominador se cuenta con el índice de desviaciones y las distancias hacia los paraderos que se desean minimizar.

$$Fo = \frac{Ua * fUa}{(Id * fId) * (Dp * fDp)}$$

Donde:

Fo: Función Objetivo

Ua: Cantidad de Usuarios atendidos

fUa: Factor de usuarios atendidos (configurable)

Id: Índice de desviaciones

FId: Factor de índice de desviaciones (configurable)

Dp: Distancia total a paraderos

fDp: Factor de distancia total a paraderos

Para calcular el Índice de desviaciones se considera que:

$$Id = \frac{Dr}{Dd}$$

Donde:

Dr: La distancia total de la ruta

Dd: La distancia en línea recta desde el inicio de la ruta hasta el final

Esto permite detectar si una ruta cuenta con muchos desvíos que terminarían causando un viaje muy largo a los usuarios.

CAPÍTULO 4: DISEÑO DE ALGORITMOS

Este capítulo muestra los resultados del segundo objetivo específico, el cual se planteó diseñar un algoritmo basado en PIA para generar la población inicial de rutas y uno genético para el mejoramiento de las rutas. De este objetivo se obtienen dos resultados: El pseudocódigo del algoritmo PIA adaptado y el pseudocódigo del algoritmo genético para la generación de rutas óptimas.

1 Pseudocódigo del algoritmo PIA adaptado

Para poder contar con la población inicial de soluciones que requiere el algoritmo genético se utilizará una variación del algoritmo PIA. La primera variación consiste en que, mientras que en el algoritmo original se escogen un par de vértices según su demanda, en este caso se escoge solo un vértice según su demanda, dado que el “segundo” vértice en este caso particular siempre es el campus principal de la universidad. La segunda variación consiste en la evaluación del costo de agregar un vértice a una ruta ya existente, para este caso la evaluación consistirá en que no se supere un valor máximo de Índice de Desviaciones (el cual se describió en el capítulo anterior) el cual será configurable por la institución. Finalmente, para poder contar con varias soluciones iniciales en vez de una sola, se plantea que para la generación de vértices de demanda del PIA se cree una subrutina que segmente en diferentes configuraciones el mapa de usuarios y que de cada segmento se obtenga un vértice en el centro de este.

Con las consideraciones anteriores, se tiene el diseño del algoritmo que se presenta a continuación. En primer lugar, se inicializa un arreglo vacío de soluciones. Se itera desde 1 hasta un valor máximo de soluciones iniciales, la cual es una constante que deberá ser calibrada según el desempeño del algoritmo. Por cada iteración se segmenta el mapa de usuarios de forma aleatoria, cada segmento es un cuadrado de tamaño y posición variable según la semilla aleatoria, de cada segmento se extra un vértice el cual está ubicado en el centro del cuadrado. Además, cada vértice contiene el valor de su

demanda, el cual es la cantidad de usuarios que habitan en el segmento. Se devuelve la lista de vértices y con esto se genera una solución con el PIA adaptado, se agrega la solución a la lista de soluciones y se retorna esta última. Por su parte, el PIA adaptado inicializa un arreglo vacío de rutas y ordena los vértices de mayor a menor demanda. El algoritmo recorre todos los vértices y por cada uno evalúa si existe alguna ruta en el arreglo de rutas que cumpla las condiciones para ser la mejor ruta para insertar el vértice. Esta evaluación consiste en insertar el vértice en las rutas existentes y evaluar si el índice de desviación excede al máximo, si no excede, el vértice puede ser insertado y se escoge la ruta con menor índice de desviación. Si no existiera una ruta para insertar se debe evaluar si ya se llegó al límite de cantidad de rutas, el cual es un valor definido por la institución. Si no se ha excedido el límite, se genera una nueva ruta del vértice al campus principal. Para la generación de rutas nuevas se usa el algoritmo Dijkstra. Finalmente se retorna el arreglo de rutas, los cuales conforman las soluciones que serán usadas en el algoritmo genético. Lo explicado anteriormente se muestra en forma de pseudocódigo a continuación.

```
1 Inicio crearSolucionesIniciales()
2       soluciones = [ ]
3       Para i desde 1 hasta MÁXIMO_SOLUCIONES_INICIALES hacer
4             vértices = segmentarMapa(numero_aleatorio)
5             solución = piaAdaptado(vértices)
6             soluciones = soluciones + solución
7       fPara
8       retornar soluciones
9 Fin crearSolucionesIniciales
```

Pseudocódigo 1. Generar Soluciones Iniciales

A continuación se explican las siguientes líneas:

- Línea 2: Se inicializa el arreglo de soluciones iniciales en un arreglo vacío.

- Línea 3 y 7: Bucle desde 1 hasta un numero máximo de soluciones iniciales configurable
- Línea 4: Se segmenta el mapa en sectores de tamaño aleatorio y se retorna un arreglo de vértices con los centroides de cada sector.
- Línea 5: Se llama a la función piaAdaptado para obtener una solución en base a los centroides hallados.
- Línea 6: Se agrega la solución de la línea 5 al arreglo de soluciones.
- Línea 8: Se retorna el arreglo de soluciones

```
1 Inicio piaAdaptado(vértices)
2       rutas = [ ]
3       vértices = ordenarDeMayorAMenorDemanda(vértices)
4       Para i desde 1 hasta vértices.tamaño hacer
5               vértice = vértices[i]
6               ruta = escogerMejorRutaParaInsertar(rutas, vértice)
7               Si ruta != vacío
8                       ruta = insertarVertice(ruta, vértice)
9               Si no
10                      Si rutas.tamaño <= MAXIMA_CANTIDAD_RUTAS
11                              rutas = rutas + generarNuevaRuta(vértice)
12                      fSi
13              fSi
14      fPara
15      retornar rutas
16 Fin piaAdaptado
```

Pseudocódigo 2. PIA Adaptado

A continuación se explican las siguientes líneas:

- Línea 2: Se inicializa el arreglo de rutas en un arreglo vacío.
- Línea 3: Cada centroide se ordena de mayor a menor según la demanda del sector al que pertenecen.
- Línea 4 y 14: Bucle desde 1 hasta el numero de centroides.

- Línea 5: Se obtiene el centroide del arreglo de centroides con el índice del bucle.
- Línea 6: Se busca cuál de las rutas es la mas apropiada para adjuntar el el centroide.
- Línea 7 y 8: Si hay una ruta apropiada para adjuntar el centroide a la ruta se ejecuta la inserción usando Dijkstra.
- Línea 9, 10 y 11: Si no hay ruta apropiada para adjuntar el centroide a la ruta y aun no se ha llegado al numero limite de rutas, se crea una nueva ruta y se agrega al arreglo de rutas
- Línea 15: Se retorna el arreglo de rutas como solución.

2 Pseudocódigo del algoritmo genético

En esta sección se muestra el diseño del algoritmo genético que mejorará las rutas generadas por el PIA adaptado. En primera instancia se inicializa el número de iteraciones en 0, se recibe la población inicial se inicializa el arreglo de los mejores individuos y se calcula el fitness de la población actual (utilizando la formulad de la función objetivo). Luego, se procede a evolucionar la población mientras no se llegue al máximo de iteraciones y no haya ocurrido una meseta. Para detectar una meseta se usa el arreglo de mejores individuos, si no se haya una mejora en varias generaciones, la solución ha llegado a una meseta. En el bucle de la evolución de la población se realizan tres etapas. Primero, se seleccionan los mejores individuos, los cuales pasaran a la siguiente generación. Segundo, se realiza la función crossover, donde se seleccionan pares de individuos y combinando características de estos se generan nuevos individuos. Tercero, se seleccionan individuos para alterar de forma aleatoria algunas de sus características. Con estas tres etapas se obtiene la nueva población, de la cual se calcula el fitness de todos sus individuos nuevamente y se selecciona al mejor para ser agregado al arreglo de mejores individuos para, finalmente, incrementar el contador de iteraciones. Cuando el bucle termina sus iteraciones, se retorna al mejor individuo del arreglo de mejores individuos, el cual representa la mejor ruta hallada.

```
1 Inicio genético(poblaciónInicial)
2       iteraciones = 0
3       población = poblaciónInicial
4       mejoresIndividuos = [ ]
5       calcularFitness(población)
6       Mientras iteraciones < MÁXIMO_ITERACIONES Y !esMeseta()
7             población = selección(población)
8             población = crossover(población)
9             población = mutación(población)
10            calcularFitness(población)
11            mejoresIndividuos = mejoresIndividuos + mejorIndividuo(población)
12            iteraciones ++
13      fMientras
14      retornar mejorIndividuo(mejoresIndividuos)
15 Fin genético
```

Pseudocódigo 3. Algoritmo genético

A continuación se explican las siguientes líneas:

- Línea 2: Se inicializa el contador de iteraciones o generaciones del algoritmo genético.
- Línea 3: Se recibe la población inicial (generada por el algoritmo PIA adaptado)
- Línea 4: Se inicializa un arreglo con los mejores individuos de cada generación para poder determinar mesetas.
- Línea 5 y 10: Se calcula el valor de la función objetivo en cada uno de los individuos de la población.
- Línea 6 y 13: Bucle del algoritmo genético, las condiciones de parada son llegar a un numero máximo de iteraciones o llegar a una meseta.
- Línea 7: Etapa de selección del algoritmo genético.
- Línea 8: Etapa de crossover o casamiento del algoritmo genético.
- Línea 9: Etapa de mutación del algoritmo genético.
- Línea 11: Se selecciona al mejor individuo de la población y se añade a la lista de mejores individuos.
- Línea 14: Se retorna al mejor individuo del arreglo de mejores individuos.

La etapa selección consiste en seleccionar a los mejores individuos, para esto se ordenan los individuos de mayor a menor fitness y se inicializa una nueva población vacía. Luego, se va insertando desde la primera posición hasta el puesto determinado por el tamaño de la población por el índice de selección, un valor entre 0 y 1 para determinar la selectividad de esta etapa, los individuos de la antigua población a la nueva. Finalmente, se retorna la nueva población.

```
1 Inicio selección(población)
2       población = ordenarDeMayorFitnessAMenor(población)
3       nuevaPoblación = [ ]
4       Para i desde 1 hasta población.tamaño * INDICE_SELECCIÓN hacer
5               nuevaPoblación = nuevaPoblación + población[i]
6       fPara
7       retornar nuevaPoblación
8 Fin selección
```

Pseudocódigo 4. Etapa de selección

A continuación se explican las siguientes líneas:

- Línea 2: Se reordena la población según el valor del fitness de los individuos de mayor a menor.
- Línea 3 a 6: Se inicializa una nueva población y se van agregando los individuos con mayor fitness desde el 1 hasta una fracción configurable del tamaño de la población.
- Línea 7: Se retorna la nueva población seleccionada.

La función crossover consiste en un bucle desde 1 hasta una cantidad determinada por el tamaño de la población por un índice de crossovers que indica la cantidad de crossovers que se realizarán en cada generación. En primer lugar, se encuentran dos padres dentro de la población y en el casamiento se elimina un ruta aleatoria del padre 1 y se agrega una ruta aleatoria del padre 2. De este casamiento sale un hijo que es agregado a la población.

```
1 Inicio crossover(población)
2       Para i desde 1 hasta población.tamaño * INDICE_CROSSOVER hacer
3             padres = encontrarPadres(población)
4             hijo = casar(padres[1], padres[2])
5             población = población + hijo
6       fPara
7 Fin crossover
```

Pseudocódigo 5. Etapa de crossover

A continuación se explican las siguientes líneas:

- Línea 2 y 6: Bucle para determinar la cantidad de individuos productos de crossover desde 1 hasta una fracción de la población.
- Línea 3: Se encuentran dos padres (dos individuos de la población)
- Línea 4: El casamiento descarta una de las rutas del padre A y agrega una de las rutas del padre B.
- Línea 5: Se agrega al hijo producto del casamiento a la población.

La etapa de mutación consiste en un bucle desde 1 hasta una cantidad determinada por el tamaño de la población por un índice de mutación que indica la cantidad de mutaciones que se realizarán en cada generación. Dentro del bucle se copia algún individuo aleatorio de la población. Luego, se selecciona alguna de sus aristas de forma aleatoria y se procede a mutar la arista. Esta operación consiste en cambiar su nodo origen o destino por algún nodo cercano que no esté incluido en la ruta. Posteriormente, se verifica que el resultado de la mutación no sea una abominación, es decir que sea una ruta no continua. Si el resultado es una abominación, se procede a completar la ruta utilizando el algoritmo Dijkstra. Finalmente, se agrega el individuo mutado a la población.

```
1 Inicio mutación(población)
2       Para i desde 1 hasta población.tamaño * INDICE_MUTACIÓN hacer
```

```
3           individuoMutado = población[numeroAleatorio()].copia
4           arista = individuo.aristas[numeroAleatorio()]
5           mutarArista(arista)
6           Si individuoMutado.esAbominación
7               completarRuta(individuo)
8           fSi
9           población = población + individuoMutado
10      fPara
11  Fin mutación
```

Pseudocódigo 6. Etapa de mutación

A continuación se explican las siguientes líneas:

- Línea 2 y 6: Bucle para determinar la cantidad de individuos productos de mutación desde 1 hasta una fracción de la población.
- Línea 3: Se encuentra un individuo aleatorio de la población y se le genera una copia.
- Línea 4: Se selecciona una de las aristas de las rutas del individuo de forma aleatoria.
- Línea 5: Se generar una alteración a la arista cambiando uno de los vértices por otro vértice cercano en el mapa.
- Lina 6 a 8: Si al alterara la arista se genera una ruta incompleta se realizan correcciones para completar la ruta
- Línea 9: Se añade el individuo mutado a la población.

CAPÍTULO 5: IMPLEMENTACIÓN

Este capítulo muestra los resultados del tercer objetivo específico, el cual se planteó como implementar el algoritmo basado en PIA y otro genético en un lenguaje de programación. Como se menciona en el capitulo de Herramientas, el lenguaje escogido fue Java, tanto para la implementación de los algoritmos como para la interfaz grafica. Siguiendo los diseños del capitulo anterior se implementaron los algoritmos siguiendo los patrones de programación orientada a objetos. La interfaz permite cargar un archivo con los potenciales usuarios del sistema de transporte y configurar el numero máximo de rutas de rutas que se desean generar. Además cuenta con la opción de habilitar o deshabilitar la ejecución del algoritmo genético con propósito de comparar los resultados entre el algoritmo PIA y Genético.

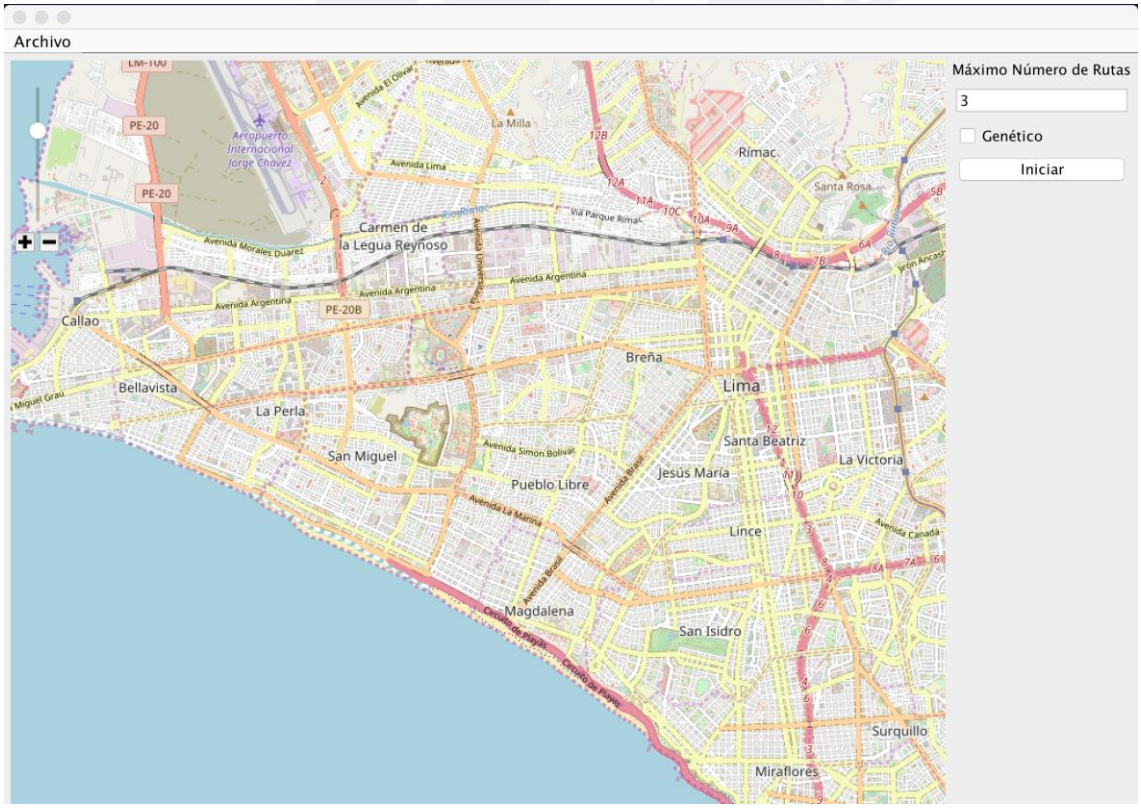

Figura 10. Pantalla de inicio de la aplicación.

En el menú de la aplicación se cuenta con una ventana de configuración del algoritmo para establecer valores como los pesos que se le dan al numero de

usuarios atendidos, desviación de las rutas y distancia a los paraderos de la función objetivo; como también valores que determinan a que distancia debe estar un usuario del paradero para que se considere como atendido la distancia mínima que debe haber entre dos paraderos.

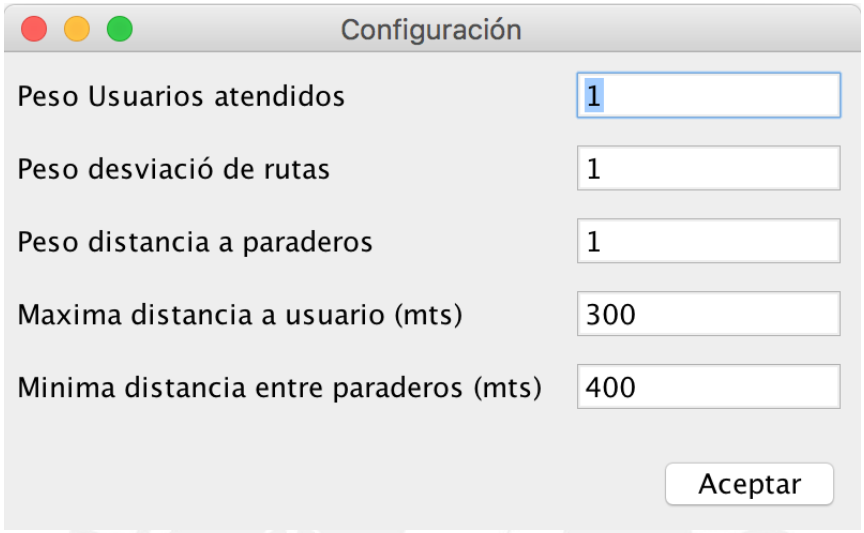

Figura 11. Pantalla de configuración de la aplicación.

Una vez terminada la ejecución de los algoritmos se mostrara las rutas en el mapa, cada una con un color distinto con la posición de los paraderos representados como puntos amarillos.

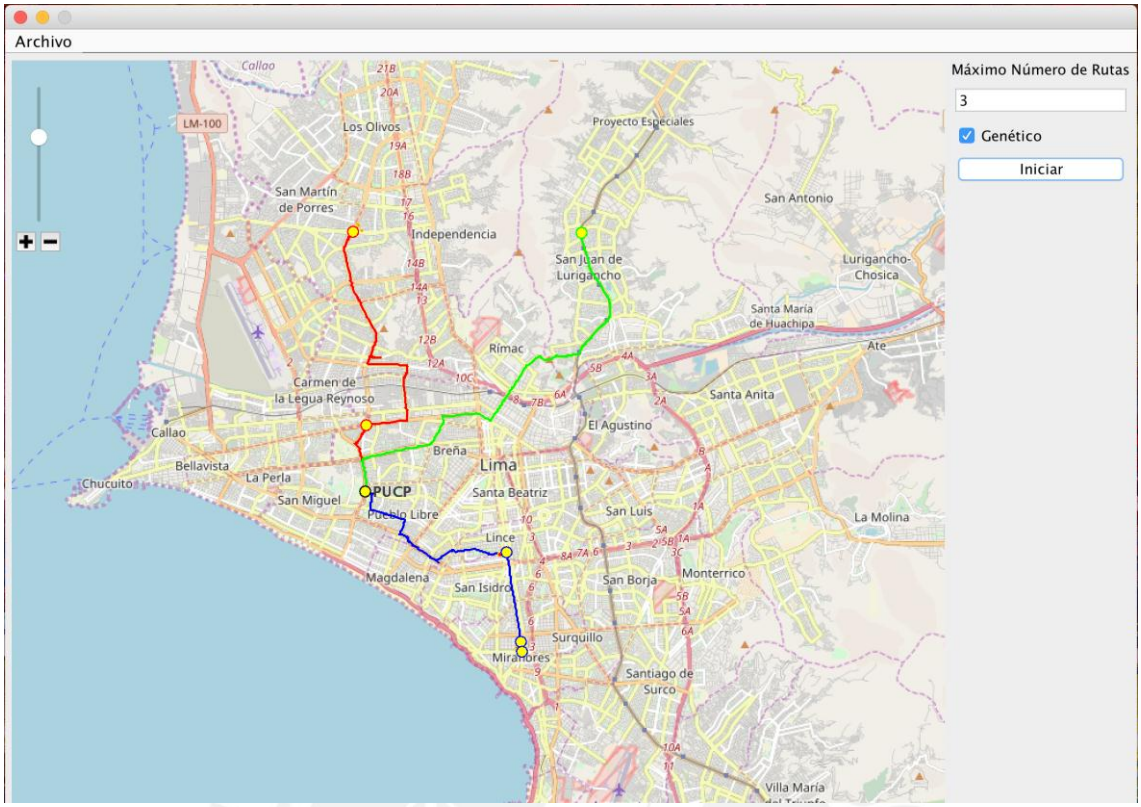

Figura 12. Pantalla de la aplicación con los resultados de las rutas y los paraderos.

Posteriormente, se puede mostrar un reporte para visualizar datos importantes del resultado de la ejecución, como lo son la cantidad de usuarios atendidos, la distancia promedio que camina cada usuario hasta el paradero y la desviación promedio de las rutas.

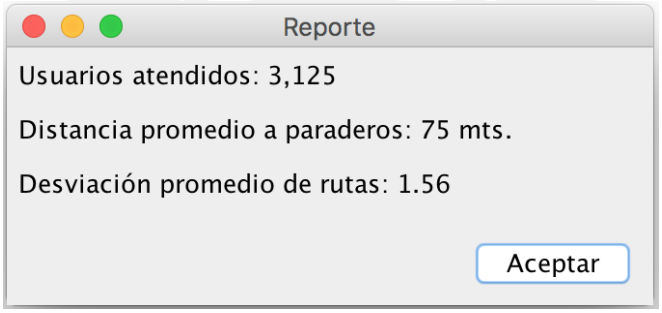

Figura 13. Pantalla de reporte de la mejor ruta.

CAPÍTULO 6: EXPERIMENTACIÓN NUMÉRICA

Este capítulo presenta los resultados de la experimentación numérica para poder comparar como mejoran los resultados del algoritmo genético respecto a los del algoritmo PIA. Los algoritmos se ejecutaron 40 veces cada uno, se muestran los valores de la función objetivo.

Tabla 5. Resultados de ejecución de los algoritmos

Ejecución	Función Objetivo PIA	Función Objetivo Genético
1	264,228.30	302,380.25
2	277,001.30	293,526.10
3	250,728.69	298,532.84
4	248,447.47	298,605.44
5	277,001.30	293,383.60
6	264,228.30	299,594.16
7	264,228.30	290,501.03
8	128,056.73	308,234.03
9	277,001.30	299,661.25
10	240,728.69	304,558.30
11	264,228.30	300,408.84
12	277,001.30	294,620.94
13	274,655.72	304,233.72
14	264,228.30	307,532.66
15	277,001.30	293,422.60
16	274,655.72	304,037.22
17	234,091.86	303,667.88
18	250,728.69	306,077.10
19	274,655.72	298,738.20
20	181,756.34	306,233.47
21	250,728.69	299,245.25
22	232,832.89	304,231.50
23	277,001.30	292,040.16
24	181,756.34	306,871.78
25	277,001.30	304,275.88
26	240,305.53	302,439.72
27	226,505.03	301,395.50
28	274,655.72	297,557.10
29	255,774.00	304,190.66

30	181,756.34	290,666.38
31	277,001.30	293,710.97
32	264,228.30	297,987.47
33	250,728.69	298,465.00
34	247,001.20	303,339.88
35	250,728.69	307,706.97
36	266,012.88	296,279.28
37	262,944.25	301,468.50
38	267,021.40	303,173.16
39	250,728.69	299,170.40
40	250,728.69	292,421.88

Para determinar si estos resultados siguen una distribución normal se usa la prueba Kolmogorov-Smirnov, esto es necesario para realizar la prueba Z.

Algoritmo PIA:

Hipótesis nula: La muestra sigue una distribución normal.

Tabla 6. Prueba Kolmogorov-Smirnov para los resultados del algoritmo PIA

Media	251,252.37	**Mínimo**	128,056.73
Desviación Estándar	32145.40	**Máximo**	277,001.30
Varianza	1033326827	**Datos**	40

Datos Ordenados (Km/h) X_i	**Probabilidad Acumulada $S_n(x_i)$**	**Probabilidad Acumulada Esperada F_i**	**Diferencias $\|S_n(x_i) - F_i\|$**
128,056.73	0.025	0.000	0.025
181,756.34	0.05	0.015	0.035
181,756.34	0.075	0.015	0.060
181,756.34	0.1	0.015	0.085
226,505.03	0.125	0.221	0.096
232,832.89	0.15	0.283	0.133
234,091.86	0.175	0.297	0.122
240,305.53	0.2	0.367	0.167
240,728.69	0.225	0.372	0.147
247,001.20	0.25	0.447	0.197
248,447.47	0.275	0.465	0.190
250,728.69	0.3	0.494	0.194

250,728.69	0.325	0.494	0.169
250,728.69	0.35	0.494	0.144
250,728.69	0.375	0.494	0.119
250,728.69	0.4	0.494	0.094
250,728.69	0.425	0.494	0.069
250,728.69	0.45	0.494	0.044
255,774.00	0.475	0.556	0.081
262,944.25	0.5	0.642	0.142
264,228.30	0.525	0.657	0.132
264,228.30	0.55	0.657	0.107
264,228.30	0.575	0.657	0.082
264,228.30	0.6	0.657	0.057
264,228.30	0.625	0.657	0.032
264,228.30	0.65	0.657	0.007
266,012.88	0.675	0.677	0.002
267,021.40	0.7	0.688	0.012
274,655.72	0.725	0.767	0.042
274,655.72	0.75	0.767	0.017
274,655.72	0.775	0.767	0.008
274,655.72	0.8	0.767	0.033
277,001.30	0.825	0.788	0.037
277,001.30	0.85	0.788	0.062
277,001.30	0.875	0.788	0.087
277,001.30	0.9	0.788	0.112
277,001.30	0.925	0.788	0.137
277,001.30	0.95	0.788	0.162
277,001.30	0.975	0.788	0.187
277,001.30	1	0.788	0.212

Máxima Diferencia	0.212	
Significancia de la prueba	0.05	95%
Valor Crítico	0.215	

Dado que la máxima diferencia es menor al valor critico se acepta la hipótesis nula que dice que la muestra sigue una distribución normal.

Algoritmo Genético:

Hipótesis nula: La muestra sigue una distribución normal.

Tabla 7. Prueba Kolmogorov-Smirnov para los resultados del algoritmo genético

Media	300114.6768	**Mínimo**	290501.03
Desviación Estándar	5050.40	**Máximo**	308234.03
Varianza	25506497.76	**Datos**	40

Datos Ordenados (Km/h) X_i	**Probabilidad Acumulada Esperada F_i**	**Diferencias $\|S_n(x_i) - F_i\|$**
290,501.03	0.028	0.003
290,666.38	0.031	0.019
292,040.16	0.055	0.020
292,421.88	0.064	0.036
293,383.60	0.091	0.034
293,422.60	0.093	0.057
293,526.10	0.096	0.079
293,710.97	0.102	0.098
294,620.94	0.138	0.087
296,279.28	0.224	0.026
297,557.10	0.306	0.031
297,987.47	0.337	0.037
298,465.00	0.372	0.047
298,532.84	0.377	0.027
298,605.44	0.383	0.008
298,738.20	0.393	0.007
299,170.40	0.426	0.001
299,245.25	0.432	0.018
299,594.16	0.459	0.016
299,661.25	0.464	0.036
300,408.84	0.523	0.002
301,395.50	0.600	0.050
301,468.50	0.606	0.031
302,380.25	0.673	0.073
302,439.72	0.677	0.052
303,173.16	0.728	0.078
303,339.88	0.738	0.063
303,667.88	0.759	0.059

304,037.22	0.781	0.056
304,190.66	0.790	0.040
304,231.50	0.793	0.018
304,233.72	0.793	0.007
304,275.88	0.795	0.030
304,558.30	0.811	0.039
306,077.10	0.881	0.006
306,233.47	0.887	0.013
306,871.78	0.910	0.015
307,532.66	0.929	0.021
307,706.97	0.934	0.041
308,234.03	0.946	0.054

Máxima Diferencia	0.098	
Significancia de la prueba	0.05	95%
Valor Crítico	0.215	

En la prueba Z se comparan las medias de las dos muestras con distribución normal.

Hipótesis nula: La media del algoritmo PIA es mayor a la del Genético.

Tabla 8. Prueba Z del algoritmo PIA con el Genético

	PIA	**Genético**
Media	251,252.37	300,114.68
Varianza	1,033,326,827	25,506,498
Observaciones	40	40
Nivel de significancia	0.05	
z	9.49708	
Valor crítico de z (una cola)	1.640	

Dado que el valor z es mayor al valor crítico se rechaza la hipótesis nula que dice que la media del algoritmo PIA es mayor a la del genético. Por lo tanto se puede afirmar que el algoritmo genético mejora los resultados que el algoritmo PIA provee como población inicial.

CAPÍTULO 6: CONCLUSIONES

1 Conclusiones

Al finalizar el presente proyecto de implementación de un algoritmo genético para elaborar un conjunto de rutas óptimas para el transporte de la comunidad universitaria desde y hacia el campus principal se puede presentar las siguientes conclusiones:

El algoritmo PIA modificado resultó ser un muy buen generador de soluciones iniciales para este tipo de problemas, ya que proveía rutas bastante directas para satisfacer las demandas mas importantes. Sin embargo, no siempre indicaba el mejor lado del camino por el que ir, es decir que en ocasiones colocaba los nodos del recorrido en el lado del sentido contrario de la ruta, esto es debido a la forma en que PIA establece sus puntos de demanda ya que solo utiliza un centro de regiones de demanda. Además, debido a que PIA se usa para establecer rutas sin paraderos no se tenia una metodología eficiente para determinar la ubicación de estos, por lo cual no llegó a alcanzar un desempeño mayor en la función objetivo

El algoritmo genético cumplió su función de proveer rutas eficientes para el transporte de la comunidad universitaria usando la población inicial que el algoritmo PIA proveyó. Se llegó a alcanzar mejoras del 20% en promedio, aunque esto depende también de los valores de configuración de la función objetivo. Esto hace que el algoritmo genético sea, además, flexible a la hora de configurar dependiendo de las necesidades de la entidad que desee generar rutas. Es decir, se le puede dar prioridad a la cantidad de usuarios atendidos, la distancia a los paraderos o las desviaciones de las rutas.

2 Recomendaciones y Trabajos Futuros

Del presente trabajo se puede recomendar, en primer lugar, la implementación del algoritmo PIA y genético para trabajos similares que incluyan planeamiento

de conjuntos de rutas ya que ambos demostraron buenos resultados para resolver el problema.

Como trabajos futuros se puede considerar una nueva versión del PIA adaptado que incluya un mecanismo para detectar ubicación de paraderos optimas ya que actualmente no cuenta con una. También se podría utilizar algunos otros algoritmos meta-heurísticos para mejorar poblaciones iniciales del algoritmo PIA.

Finalmente, los resultados de este proyecto se podrían integrar a un proyecto mas grande que también incluya la planeación de los horarios de partida de los buses, asignación de choferes y unidades de transporte o incluir algún modulo para considerar el trafico como variable para resolver el problema.

REFERENCIAS BIBLIOGRÁFICAS

Aguilar, Catherine (2011), «Siete nuevos buses para la UNALM», *Gaceta Molinera*, [en línea] <http://www.lamolina.edu.pe/gaceta/edicion2011/notas/nota180.htm> [fecha de consulta: 3 de abril de 2017].

Aldaba, Karina (2010), «San Marcos llegará más lejos», *Noticias UNMSM*, [en línea] <http://www.unmsm.edu.pe/noticias/ver/1174> [fecha de consulta: 4 de abril de 2017].

Barrantes, Walter (2015), «IMPLEMENTACIÓN DE UN ALGORITMO RECOCIDO SIMULADO PARA EL DISEÑO DE RUTAS DE TRANSPORTE PÚBLICO PARA LIMA CENTRO».

Bianchi, Leonora y otros (2009), «A survey on metaheuristics for stochastic combinatorial optimization», *Natural Computing*, vol. 8, No. 2.

Bielli, Maurizio, Massimiliano Caramia y Pasquale Carotenuto (2002), «Genetic algorithms in bus network optimization», *Transportation Research Part C: Emerging Technologies*, vol. 10, No. 1.

Caliper (2017), «TransCAD Transportation Planning Software», [en línea] <http://www.caliper.com/tcovu.htm> [fecha de consulta: 13 de abril de 2017].

Ceder, Avishai (2007), *Public Transit Planning and Operation*, CRC Press.

Corder, Gregory W y Dale I Foreman (2009), *Nonparametric Statistics for Non-Statisticians: A Step-by-Step Approach*, Test.

Dirección de Informática - Oficina de Estadística (2016), «Datos Académicos», *Dirección de Informática - Oficina de Estadística*, [en línea] <http://www.pucp.edu.pe/la-universidad/nuestra-universidad/pucp-en-cifras/datos-academicos/?seccion=comunidad-universitaria> [fecha de

consulta: 11 de abril de 2017].

Fan, Wei, Zegeye Gurmu y Elias Haile (2013), «A bi-level metaheuristic approach to designing optimal bus transit route network», *2013 IEEE International Conference on Cyber Technology in Automation, Control and Intelligent Systems, IEEE-CYBER 2013*, pÃƒÂ¡g. 308-313.

Fischer-Box, Joan (1987), «Guinness, Gosset, Fisher, and Small Samples», *Statistical Science*, vol. 2, No. 1.

Francesco Ciaffia y otros (2013), «A new methodology for the public transport network design», *The 9th International Conference «Environmental Engineering 2014»*.

Fuertes, Jose (2011), «Universidad Nacional de Ingeniería inaugura puerta de ingreso y flota de buses», *Agencia de Prensa Lima Norte*, [en línea] <http://agenciadeprensalimanorte.com/universidad-nacional-de-ingenieria-inaugura-puerta-de-ingreso-y-flota-de-buses/> [fecha de consulta: 10 de abril de 2017].

Google (2017), «API de Google Maps», [en línea] <https://developers.google.com/maps/> [fecha de consulta: 17 de mayo de 2017].

INEI (2015), «Estado de la población peruana».

INRO (2017), «Emme», [en línea] <https://www.inrosoftware.com/en/products/emme/> [fecha de consulta: 13 de abril de 2017].

JEFFREY, BRACA y otros (1997), «A computerized approach to the New York Cityschool bus routing problem», *IIE Transactions*, vol. 29, No. 8.

Kokash, Natallia (2005), «An introduction to heuristic algorithms», *Department of Informatics and Telecommunications*.

Lima Cómo Vamos (2015), «VI Informe de Percepción Sobre Calidad», Lima, Perú.

Mauttone, Antonio y María E. Urquhart (2009), «A route set construction algorithm for the transit network design problem», *Computers & Operations Research*, vol. 36, No. 8.

Mendoza, Jorge L (2010), «Statistical Concepts: A Second Course (3rd ed.).», *Organizational Research Methods*, vol. 13, No. 4.

Microsoft (2017), «Microsoft Excel», [en línea] <https://products.office.com/es/excel> [fecha de consulta: 18 de mayo de 2017].

NetBeans (2017), «Welcome to NetBeans», [en línea] <https://netbeans.org/> [fecha de consulta: 18 de mayo de 2017].

Oracle (2017), «Learn about Java technology», [en línea] <https://java.com/en/about/> [fecha de consulta: 18 de mayo de 2017].

Pattnaik, S. B., S. Mohan y V. M. Tom (1998), «Urban Bus Transit Route Network Design Using Genetic Algorithm», *Journal of Transportation Engineering*.

PostGIS (2017), «PostGIS - Spatial and Geographic Objects for PostgreSQL», [en línea] <http://postgis.net/> [fecha de consulta: 18 de mayo de 2017].

PostgreSQL (2017), «PostgreSQL», [en línea] <https://www.postgresql.org/about/> [fecha de consulta: 19 de mayo de 2017].

Project Management Institute (2013), *A Guide to the Project Management Body of Knowledge - PMBOK Guide*, Project Management Journal.

PTV (2017), «Visum», [en línea] <http://vision-traffic.ptvgroup.com/es/productos/ptv-visum/> [fecha de consulta: 13 de abril de 2017].

Real Academia Española (2014), «Diccionario de la lengua española (23.a ed.)», [en línea] <http://www.rae.es/> [fecha de consulta: 5 de mayo de 2017].

Schmitt, Lothar M. (2001), «Theory of genetic algorithms», *Theoretical Computer Science*.

StarUML (2017), «StarUML», [en línea] <http://staruml.io/> [fecha de consulta: 18 de mayo de 2017].

Szeto, W. Y. y Yongzhong Wu (2011), «A simultaneous bus route design and frequency setting problem for Tin Shui Wai, Hong Kong», *European Journal of Operational Research*, vol. 209, No. 2.

Top Universities (2016), «QS World University Rankings® 2015/16», *Top Universities*, [en línea] <https://www.topuniversities.com/university-rankings/world-university-rankings/2016> [fecha de consulta: 11 de abril de 2017].

Whitley, Darrell (1994), «A genetic algorithm tutorial», *Statistics and Computing*, vol. 4, No. 2.