

**Universidad Tecnológica de la Habana “José Antonio Echeverría”**

**Facultad de Ingeniería Informática**

**Incorporación de métricas de evaluación para los algoritmos de asignación en BHAVRP**

**Gestión de Proyectos**

**Autor:**

Eric Ramos Aragón (ceramos@ceis.cujae.edu.cu)

**Tutores:**

Dr. C. Isis Torres Pérez (itorres@ceis.cujae.edu.cu)

Dr. C. Alejandro Rosete Suárez (rosete@ceis.cujae.edu.cu)

**La Habana, Cuba**

20 de febrero del 2025

# Resumen

Los problemas de optimización asociados al problema de planificación de rutas de vehículos son muy frecuentes la vida moderna, donde la logística de las cadenas de suministros es esencial. En muchos de los problemas de ruteo, el problema completo se divide en dos partes: la asignación de clientes a depósitos y luego la creación de las rutas. La asignación óptima de clientes a depósitos en un problema complejo en sí mismo, que considera restricciones de capacidad y acceso, a la vez que intenta reducir los costos de las rutas creadas. En este trabajo se presenta una biblioteca ofrecida como software libre que implementa tanto en Python como en Java un conjunto de heurísticas para la asignación de clientes a depósitos que es fácilmente integrable en soluciones académicas y empresariales debido a su flexibilidad.

# Jefe de Proyecto

El desarrollo de este proyecto estará a cargo exclusivamente del autor, quien asumirá la responsabilidad de desempeñar todos los roles. El resultado del test de Belbin arrojó que los roles preferidos por el autor son Implementador (ID), Coordinador (CO), Especialista (IS) y Finalizador (CE), mientras que los roles evitados son Investigador de Recursos (IR) y Monitor Evaluador (ME). Por otro lado, el test de MBTI indicó que el autor tiene un perfil ISTJ.

# Analista

Los requisitos funcionales definidos para el sistema son los siguientes:

* RF1: Cargar datos de problemas MDVRP desde múltiples formatos.
* RF2: Generar matrices de costo con distancias aproximadas.
* RF3: Ejecutar los algoritmos de asignación.
* RF4: Construir matrices de costo con datos reales.
* RF5: Gestionar excepciones personalizadas.
* RF6: Calcular métricas de evaluación.
* RF7: Exportar resultados a formatos Excel y CSV.

Cada uno de estos requisitos está planificado para su implementación y validación a lo largo del desarrollo del proyecto. Los requisitos funcionales RF1, RF2, RF3 y RF4 ya han sido implementados y validados, los requisitos RF5 y RF6 serán implementados y validados durante el período de tesis y RF7 será implementado y validado durante la asignatura Gestión de Proyectos.

Los requisitos de calidad definidos para el sistema son los siguientes:

* RNF1: El sistema debe ejecutar los algoritmos en menos de 2 segundos.
* RNF2: El sistema debe ser capaz de cargar datos de problemas MDVRP desde múltiples formatos.
* RNF3: El sistema debe permitir la creación de matrices de costos con distancias aproximadas.
* RNF4: El sistema debe permitir la creación de matrices de costos con distancias reales.
* RNF5: El sistema debe ser capaz de exportar los resultados.
* RNF6: El sistema debe ser capaz de ser consumido por otros sistemas en diferentes lenguajes.
* RNF7: El sistema debe ser capaz de evaluar la calidad de las asignaciones.
* RNF8: El sistema debe incorporar un paquete de excepciones personalizadas.
* RNF9: El sistema debe ser capaz de manejar grandes volúmenes de datos.

Cada uno de estos requisitos de calidad está alineado con las necesidades del proyecto y se validará en distintas etapas de su desarrollo. De estos, RNF1, RNF3, RNF4 y RNF9 ya han sido validados, mientras que RNF2, RNF6, RNF7 y RNF8 serán validados durante el período de tesis. El requisito no funcional que será validado durante la asignatura Gestión de Proyectos será el RNF5.

# Planificador

Desde el rol de planificador, se lleva a cabo la organización y planificación estructurada del proyecto con el fin de asegurar el cumplimiento de los objetivos y la correcta ejecución de cada fase. Para ello, se utilizan herramientas y metodologías de gestión de proyectos que permiten definir, asignar y dar seguimiento a las tareas.

En primer lugar, se utiliza la herramienta GanttProject para planificar las tareas restantes del proyecto. Se crea un diagrama de Gantt en el que se establecen las dependencias entre tareas, las fechas estimadas de inicio y fin, la duración de cada actividad y los responsables asignados. Asimismo, se incluye en el plan todas las tareas correspondientes a los distintos roles.

Además, se elabora el documento de gestión de riesgos, donde se identifican los riesgos potenciales del proyecto, se analiza su probabilidad e impacto y se clasifican en función de su criticidad. Como complemento a esta actividad, se desarrolla un plan de contingencia para abordar los riesgos de criticidad alta, asegurando que existan estrategias claras para mitigar su impacto en caso de materialización.

# Gestor de la configuración

Desde el rol de Gestor de la Configuración, se garantiza que el control de versiones y la gestión de cambios en el proyecto se realicen de manera estructurada y conforme a las políticas establecidas. Para ello, se documentan las prácticas utilizadas, se supervisa su cumplimiento y se verifican las responsabilidades de cada integrante del equipo en relación con la configuración del sistema.

En primer lugar, se definió y aplicó una política de gestión de ramas, estableciendo un enfoque de rama única (main), en el cual todos los cambios son desarrollados, probados y fusionados directamente en esta rama. La política incluye las siguientes normas:

* Todos los cambios deben ser probados localmente antes de ser subidos al repositorio.
* Los commits deben ser descriptivos y reflejar claramente la naturaleza de los cambios realizados.
* Se debe realizar push únicamente cuando se complete una funcionalidad o corrección relevante.
* Se utilizan GitHub y Git desde la terminal o el plugin de VS Code para gestionar el código.

Para mantener la estabilidad del sistema, se sigue un proceso estructurado ante cualquier modificación:

* Propuesta de cambio: Identificación de la necesidad de modificación.
* Evaluación del impacto: Análisis de las consecuencias en el sistema.
* Aprobación del cambio: Validación antes de la implementación.
* Implementación: Aplicación del cambio siguiendo las normas establecidas.
* Pruebas y verificación: Comprobación de que el cambio no introduce errores.
* Registro y versionado: Documentación del cambio y subida al repositorio.

Se comprobó que todas las modificaciones siguieron este procedimiento, garantizando que no se aplicaron cambios sin evaluación y aprobación previa.

Se verificó que:

* Todos los cambios fueron documentados en el control de versiones.
* Las modificaciones fueron evaluadas antes de su implementación.
* No se realizaron cambios que afectaran la estabilidad del sistema sin validación previa.

Con estas prácticas, se garantiza una gestión eficiente de la configuración, manteniendo la integridad del código y la trazabilidad de los cambios en el proyecto.