**Artículo Original**

**Nueva versión de la Biblioteca de Heurísticas de Construcción para Problemas de Planificación de Rutas de Vehículos**

**New version of BHCVRP**

**Ananda de la Caridad Morales Morales1,\*Isis Torres Pérez, Alejandro Rosete Suárez**

1Facultad de Ingeniería Informática. Universidad Tecnológica de La Habana “José Antonio Echeverría” (CUJAE). Calle 114 No 11901 entre 119 y 127, Marianao, La Habana, Cuba.

\*Correspondencia:amorales@ceis.cujae.edu.cu

Estedocumentoposeeuna[licenciaCreativeCommonsReconocimiento/NoComercial4.0Internacional](http://creativecommons.org/licenses/by-nc/4.0/)

**Resumen**

Los Problemas de Planificación de Rutas de Vehículos cuentan con una flota de vehículos, buscan satisfacer las demandas de clientes dispersos geográficamente y tienen como objetivo minimizar costos siempre que se cumplan con las restricciones existentes. A diario se producen nuevas situaciones prácticas en el mundo, lo que conlleva al surgimiento de nuevas variantes. Algunos de los métodos de solución para estos problemas son las heurísticas de construcción, que persiguen encontrar de manera rápida y sencilla una primera solución cercana al óptimo. Sin embargo, surge el inconveniente de que la gran mayoría de heurísticas sólo puedan ser empleadas en la variante básica. Además, en la actualidad existen pocas bibliotecas que implementen métodos heurísticos para solucionar estos problemas.

En este contexto surge la Biblioteca de Heurísticas de Construcción para Problemas de Planificación de Rutas de Vehículos (BHCVRP), para dar solución a cuatro problemas de planificación de rutas de vehículos mediante ocho heurísticas de construcción. En este trabajo se presenta una nueva versión de dicha biblioteca implementada en los lenguajes Java y Python, que resuelve las deficiencias detectadas en la versión previa. Además, se incorporan dos nuevas heurísticas de construcción, así como tres nuevas variantes que consideran ventanas de tiempo, autobuses escolares y el retorno a un punto diferente al depósito. Por último, a partir de un experimento se determina que las versiones de BHCVRP no presentan diferencias significativas.

**Palabras clave**: BHCVRP, diseño de software, heurísticas de construcción, Problema de Planificación de Rutas de Vehículos.

**Abstract**

Vehicle Routing Problems involve a fleet of vehicles, seek to satisfy the demands of geographically dispersed customers and aim to minimize costs as long as existing constraints are met. New practical situations occur daily in the world, which leads to the increase of these problems and the emergence of new variants. Some of the solution methods are construction heuristics, which aim to quickly and easily find a first solution close to the optimum. However, there is the drawback that the vast majority of heuristics can only be used in the basic variant. In addition, there are currently few libraries that implement heuristic methods to solve these problems.

In this context, BHCVRP arises with the purpose of providing solutions to four types of vehicle route planning problems by employing seven construction heuristics. This paper presents a new version of this library implemented in Java and Python programming languages, which solves some of the deficiencies found in the previous version. Two new construction heuristics and three new variants are incorporated.Finally, from an experiment it is determined that the versions of BHCVRP do not show significant differences.

**Keywords**: BHCVRP, construction heuristics, software design, Vehicle Routing Problem.

**1. Introducción**

El Problema de Planificación de Rutas de Vehículos (VRP, por las siglas en inglés de *VehicleRoutingProblem*) consiste en crear rutas eficientes para que una flota de vehículos sirva a un grupo de clientes cumpliendo unas restricciones específicas. Este problema se utiliza habitualmente en la gestión de la cadena de suministro para entregar bienes y servicios. Se diseñan varias versiones del VRP en función de factores como la naturaleza de los bienes transportados, las características de los clientes y las especificaciones de los vehículos [1]. Según [2], el VRP se introdujo por primera vez en 1959 como una forma generalizada del Problema del Viajante de Comercio (TSP). En 1964, se propuso el Algoritmo de Ahorros, el primer algoritmo eficaz para resolver VRP [3]. Posteriormente, la investigación y el trabajo en el campo de los VRPsha aumentado, dando lugar al desarrollo de modelos que incorporan características de la vida real y a la búsqueda de algoritmos eficientes para resolver estos problemas [4]. Se proponen modelos básicos para resolver situaciones prácticas, que pueden extenderse a diferentes escenarios de la logística y el transporte. El VRP tiene aplicaciones prácticas en el transporte de empleados, rutas de autobuses, recogida de desechos sólidos, distribución del periódico y correo, entre otras. Estas situaciones cotidianas pueden modelarse con las características de las diferentes variantes de VRP existentes [5]. Dentro de las variantes más conocidas se encuentran el Problema de Planificación de Rutas de Vehículos con Capacidades (CVRP) [6], el Problema de Planificación de Rutasde Vehículos con Múltiples Depósitos (MDVRP) [7], el Problema de Planificación de Rutas de Vehículos con Flota Heterogénea (HFVRP)[8], el Problema de Planificación de Rutasde Camiones y Remolques (TTRP) [9], entre otros.

Los Problemas de Planificación de Rutasde Vehículos suelen ser difíciles de resolver, especialmente cuando se trata de un gran número de clientes. Esto se debe a que los VRPs se clasifican como problemas NP-duros, conocidos por ser muy difíciles de resolver [1]. A veces, puede ser necesario renunciar a encontrar elóptimo y centrarse en construir una buena solución, aunque no se garantice que sea la mejor posible. Para hacer frente a esto, se utilizan algoritmos heurísticos, incluyendo heurísticas y metaheurísticas, como soluciones alternativas [10, 11]. Las heurísticas están diseñadas para guiar el proceso de búsqueda y recomendar el mejor curso de acción ante múltiples opciones [12, 13]. Son procedimientos destinados a proporcionar soluciones a los problemas con un buen rendimiento en términos de calidad de la solución y recursos utilizados. Las heurísticas ofrecen métodos más sencillos que brindan soluciones satisfactorias mediante algoritmos específicos para cada problema. Pueden dar la solución final a un problema o ser utilizadas en el contexto de las metaheurísticas [4, 13]. Las heurísticas se suelen clasificar en dos grupos: heurísticas de construcción y heurísticas de mejora [11, 14].

Las heurísticas de construcción se encargan de crear una solución paso a paso, mientras que las heurísticas de mejora trabajan sobre las soluciones existentes para conseguir mejoras [14]. Las heurísticas de construcción comúnmente utilizadas para resolver VRPs incluyen el Algoritmo de Ahorros, el Algoritmo de Barrido, la Heurística de Inserción Paralela de Christofides, Mingozzi y Toth, la Heurística de Inserción Secuencial de Mole &Jameson, entre otros [15]. Debido a la necesidad de resolver problemas de optimización combinatoria utilizando dichos algoritmos y para promover la reutilización de sus códigos, es beneficioso organizarlos dentro de componentes de software. Actualmente, pocas bibliotecas implementan métodos heurísticos para resolver VRPs y otros problemas de optimización. Algunos componentes de software que cumplen estos criterios sonBiCIAM[16], METSlib[17], MALLBA [18], VRPH [19], MOMHLib++ [20], BHCVRP [21], OR-Tools [22], JSprit[23], VROOM [24], entre otros.

Entre los componentes de software mencionados, se utilizan los siguientes para resolver VRP: las bibliotecas BiCIAM, VRPH y BHCVRP, y las herramientas OR-Tools, JSprit y VROOM. BHCVRP es la única que permite resolver VRPsexclusivamente mediante heurísticas de construcción. La bibliografía muestra una falta de bibliotecas heurísticas para resolver problemas de optimización combinatoria, ya que la mayor parte del desarrollo se centra en algoritmos metaheurísticos. Existen pocas adaptaciones de heurísticas a las distintas variantes de VRPs, y las características específicas de cada problema no suelen tenerse en cuenta en las versiones originales. Además, la mayoría de los componentes de software existentes para resolver estos problemas están desarrollados en los lenguajes de programación C++ y Java. Dado el crecimiento de la comunidad en Python parael trabajo con Inteligencia Artificial y problemas de optimización, sería valioso explorar cómo resolver VRPs con heurísticas utilizando este lenguaje [25, 26].

En este contexto surge BHCVRP como propuesta realizada por el proyecto de Optimización y Metaheurísticas de la Facultad de Ingeniería Informática de la Universidad Tecnológica de La Habana “José Antonio Echeverría” (CUJAE) [21]. Esta biblioteca está desarrollada en Java, contiene ocho adaptaciones de heurísticas de construcción de la literatura para cuatro variantes VRPs y presenta ciertas deficiencias relacionadas con su diseño y adaptabilidad. Su integración está restringida a aplicaciones desarrolladas en Java, limitando su interoperabilidad con otros sistemas. Además, la biblioteca presenta alto acoplamiento entre sus clases, lo que dificulta la extensión y el mantenimiento del código relacionado con las heurísticas implementadas. Este componente carece de un manejo robusto de excepciones, así como de soporte para la carga y exportación de datos en formatos estándares como JSON o CSV. Por tanto, se identifica como objetivo de este trabajo: Desarrollar una nueva versión de BHCVRP en Java y Python, que permita incorporar otras heurísticas de construcción de la literatura para diferentes variantes de problemas de planificación de rutas de vehículos.

A continuación, se presenta la estructura de la investigación. En la sección 2 se presentan las nuevas variantes VRPs y heurísticas de construcción incorporadas, así como una vista de la arquitectura de la nueva versión de BHCVRP y sus principales cambios. La sección 3 presenta y discute los resultados experimentales. Por último, la sección 4 incluye las conclusiones y recomendaciones para trabajos futuros.

**2. Materiales y Métodos**

Para este trabajo se realizó una búsqueda bibliográfica en diferentes bases de datos como Google Scholar, Scielo, ResearchGate, Scopus, Springer, IEEE Xplore, así como trabajos de diploma relacionados con el proyecto de Optimización y Metaheurísticas de la Facultad de Ingeniería Informática de la Universidad Tecnológica de La Habana “José Antonio Echeverría” (CUJAE). Se implementaron varios términos de búsqueda como “VRP”, “heurísticas de construcción”, “library to solve VRP”, entre otros. Los resultados se filtraron a artículos publicados en los últimos cinco años en español e inglés, excepto la bibliografía clásica del tema que se remonta a finales del siglo XX. Como materiales se tienen en cuenta la herramienta Excel, los lenguajes de programación Python en su versión 3.12.0 y Java en su versión 11, losIDEs de programaciónPyCharm2024.1.3 (*CommunityEdition*) y Apache NetBeans 16, el sistema operativo Windows 10 de 64 bits y la herramienta para pruebas estadísticas KEEL. A continuación, se describen los principales fundamentos teóricos a tener en cuenta para la comprensión de esta investigación.

Los modelos básicos desarrollados para VRPs se utilizan para resolver escenarios del mundo real en la industria de la logística y el transporte. Algunas aplicaciones prácticas de VRPs son el transporte obrero, las rutas de ómnibus, la recogida de desechos sólidos, la distribución del periódico y correo, entre otras. Estos escenarios cotidianos pueden representarse utilizando diversas variantes de VRPs existentes o una combinación de ellas, teniendo en cuenta sus características específicas [1]. Estainvestigaciónanaliza las variantes Problema de Planificación de Rutas de Autobuses Escolares (SBRP)[27], Problema de Planificación de Rutas de Vehículos Abiertos (OVRP)[28, 29] y Problema de Planificación de Rutas de Vehículos con Ventanas de Tiempo (VRPTW)[30], ya que son las nuevas incorporadas en BHCVRP.

La variante SBRP comprende un conjunto de autobuses para recoger a estudiantes concentrados en paradas. En esta propuesta se soluciona la variante de SBRP para una sola escuela en un entorno urbano con flota homogénea. El objetivo es minimizar la distancia total del conjunto de rutas. Una solución es factible si todas las rutas satisfacen las restricciones de distancia, capacidad máxima del autobús, máxima distancia a caminar y capacidad de la parada. Además, deben comenzar y terminar en el mismo depósito (escuela)[27].

Por otra parte, la variante OVRP se comporta del mismo modo que CVRP, con la diferencia de que el vehículo no regresa al depósito central. Su objetivo es minimizar la distancia total del conjunto de rutas. La solución se considera factible si se cumplen con las restricciones de distancia[28, 29].

La variante VRPTW presenta un comportamiento similar a la variante CVRP, adicionando las ventanas de tiempo. Su objetivo es minimizar la distancia total del conjunto de rutas cumpliendo con las demandas de los clientes y respetando sus respectivas ventanas de tiempo. La solución se considera factible si se cumplen con las restricciones de distancia, horarios de servicio y tiempo de servicio[30].

En la se presenta una vista general del paquete *data,* responsable de contener las clases que modelan un problema de planificación de rutas de vehículos. En color azul se presentan las clases que se mantienen en su estado original, en color morado las que son modificadas y en color verde las clases nuevas incorporadas.



**Fig. 1: Diagrama de clases del paquete data en la nueva versión de BHCVRP.**

En este caso la clase *BusStop*está relacionada con la variante SBRP, mientras que la clase *TimeWindow* se relaciona con la variante VRPTW. Por último, una alternativa para solucionar este tipo de problemas son los métodos aproximados como los algoritmos heurísticos. Este trabajo se centra en las heurísticas de construcción que se describen a continuación.

*Heurísticas de construcción*

Las heurísticas de construcción en el contexto deVRPs, se pueden clasificar en cuatro grupos según [31]: basadas en ahorros, de inserción o basadas en coste, ruta primero – asignardespués, y asignar primero – ruta después. Las heurísticas basadas en ahorros consisten en sustituir nodos de la ruta por otros que aún no están, manteniendo un proceso de intercambio constante mientras se cumplan las restricciones. Entre los algoritmos más conocidos de esta categoría se encuentran el Algoritmo de Ahorrosen sus versiones secuencial y paralela, así como la variante basada en *Matching*[15]. Las heurísticasde inserción se utilizan para generar un conjunto de rutas iniciales, con soluciones construidas mediante inserciones constantes de nodos. Entre las heurísticas más populares de esta categoría se encuentran Vecino más Cercano, Inserción Secuencial de Mole &Jameson(MJ), Inserción Paralela de Christofides, Mingozzi y Toth (CMT), e Inserción de Kilby[15]. La heurística de ruta primero – asignar después, determina la ruta para visitar a todos los clientes y luego la particiona en varias rutas factibles, por ejemplo, con los Algoritmos de Pétalos que muestran este comportamiento [15]. Por último, las heurísticas asignar primero – ruta después, tienen como objetivo generar un grupo de clientes pertenecientes a la misma ruta en la solución final, creando una ruta para cada grupo que visite a todos sus clientes determinando el orden en el que serán visitados. Entre los algoritmos más conocidos de esta categoría se encuentran el Algoritmo de Barrido y la Heurística de Asignación Generalizada de Fisher &Jaikumar[15]. Esta investigación se enfoca en las heurísticas de construcción: Algoritmo de Ahorros basado en *Matching* y la Heurística de Asignación Generalizada de Fisher &Jaikumar, debido a que son las nuevas que se incorporan en BHCVRP.

El Algoritmo de Ahorros basado en *Matching*considera la afectación que provoca la unión a realizar teniendo en cuenta las posibles uniones en próximas iteraciones. Para esto se considera un conjunto de ahorros potenciales entre las rutas, donde cada ahorro representa la reducción en costos al combinar dos rutas en una sola. Luego, se utiliza un *matching*de peso máximo para seleccionar las combinaciones de rutas que maximizan los ahorros totales, garantizando una solución globalmente buena[15]. Según [15], este algoritmo consta de cuatro pasos:

1. Inicialización: para cada cliente *i* construir la ruta (*0,i,0*).
2. Cálculo de ahorros: actualizar *Spq*para cada par de rutas *p* y *q* que pueda ser combinado manteniendo la factibilidad. Si ningún par de rutas puede ser combinado, terminar.
3. *Matching*: resolver un problema de *matching* de peso máximo donde se tienen las rutas de la solución actual como nodos y existe un arco entre las rutas *p* y *q* con peso *Spq* si su combinación es factible.
4. Uniones: dado el *matching* de peso máximo, combinar todo par de rutas *p* y *q* tal que *(p,q)* pertenezca al *matching*. Ir al paso 2.

Por otra parte, la Heurística de Asignación Generalizada de Fisher &Jaikumar posee dos etapas. La primera consiste en crear clústeres de clientes a visitar por cada vehículo solucionando un problema de asignación generalizada (GAP, por sus siglas en inglés)[12]. Las rutas son creadas en función de cada vehículo. En la segunda etapa el problema es tratado como un TSP. A continuación, se presentan los pasos de este algoritmo según [32]:

1. Calcular las distancias entre todos los puntos (depósito y clientes).
2. Dividir angularmente el plano donde los clientes están juntos en conos iguales al número de vehículos.
3. Escoger un cliente semilla para guiar el proceso de asignación de clientes a vehículos.
4. Calcular el costo de inserción entre clientes semillas y otros clientes.
5. Asignar clientes a vehículos teniendo en cuenta restricciones como capacidad del vehículo y demanda de los clientes.
6. Utilizar cualquier método de solución para TSP con el objetivo de obtener las rutas.

Es importante destacar que, en el proceso de resolución del GAP, la función objetivo es minimizar el costo de inserción para cada vehículo. Además, cada cliente debe ser asignado una sola vez a un solo vehículo y la demanda total para cada clúster es limitada por la capacidad del vehículo. Finalmente, se tendrán en cuenta las siguientes consideraciones: los clientes semillas serán aquellos con mayor demanda y la heurística para solucionar la segunda etapa como TSP será el Vecino más cercano por su naturaleza determinista[32].

En la se presenta el diagrama de clases del paquete *heuristic,* encargado de presentar todas las heurísticas disponibles para la construcción de rutas. En color azul se presentan las clases que se mantienen en su estado original, en color morado las que sufren modificaciones y en color verde las dos heurísticas nuevas incorporadas. Es importante destacar que en la clase abstracta *Heuristic* se define el conjunto de pasos a seguir por cada heurística, debido a la implementación del patrón de diseño *TemplateMethod*[33].



**Fig. 2: Diagrama de clases del paquete heuristic en la nueva versión de BHCVRP.**

*Modificaciones en BHCVRP*

La Biblioteca de Heurísticas de Construcción para el Problema de Planificación de Rutas de Vehículos (BHCVRP) es una biblioteca de clases que implementa heurísticas de construcción para la resolución de VRPs. Esta biblioteca fue desarrollada en [21] en el lenguaje de programación Java con una arquitectura n-capas con enfoque basado en reutilización y contiene la adaptación de ocho heurísticas clásicas para las variantes CVRP, MDVRP, HFVRP y TTRP. Las heurísticas implementadas son el Algoritmo de Barrido, dos versiones del Algoritmo de Ahorros (secuencial y paralela), la heurística del Vecino Más Cercano con Lista de Candidatos Restringidos, la Heurística de Inserción de Mole &Jameson, la Heurística CMT, la Heurística de Inserción de Kilby y un método aleatorio [21].

En su última versión, desarrollada en [34], se han incorporado tres variantes VRPs y dos heurísticas de construcción. Además, se propone una implementación en los lenguajes Java y Python, que resuelve las deficiencias anteriores.Para ello, en la se presenta el diagrama de la arquitectura de la nueva versión de BHCVRP, donde los paquetes en color azul son los que permanecen en su estado original, los paquetes en morado son los que sufren algunas modificaciones y en verde los paquetes nuevos incorporados.



**Fig. 3: Patrón n-capas con enfoque basado en reutilización de la nueva versión de BHCVRP.**

Las principales modificaciones en BHCVRP son:

* Incorporar la versión restante del Algoritmo de Ahorros, es decir, su variante basada en *Matching*, así como la Heurística de Asignación Generalizada de Fisher &Jaikumar.
* Incorporar las variantes SBRP, OVRP y VRPTW.
* Ofrecer un paquete de excepciones propias para manejar las restricciones relacionadas con capacidad, distancia, velocidad, tiempo, entre otros elementos, atendiendo a las características de las diferentes heurísticas y variantes VRPs
* Utilizar la Biblioteca de Heurísticas de Asignación para Problemas de Planificación de Rutas de Vehículos con Múltiples Depósitos (BHAVRP), con el objetivo de resolver la asignación de clientes a depósitos en la variante MDVRP eficientemente.
* Emplear un formato JSON para la carga de datos y emplear formatos como TXT, JSON, CSV y XML para la exportación de resultados.
* Adaptar el patrón de diseño *TemplateMethod*[33]para mejorar la reutilización del código y la flexibilidad de las heurísticas implementadas.
* Incorporar bibliotecas especializadas para el cálculo de distancias aproximas, distancias reales y obtención de matrices de costo y tiempo.
* Incorporar paquete *tools*para manejar el ordenamiento de la lista de capacidades, así como funcionalidades extras.

**3. Resultados y Discusión**

En esta sección, se desarrolla un experimento para validar el funcionamiento de la nueva versión de BHCVRP. Este experimento realiza una comparación de los resultados actuales obtenidos, tanto en Java como en Python, con los resultados de la versión anterior de la biblioteca en Java, teniendo en cuenta siete heurísticas y cuatro variantes VRPs.

Para la ejecución de los experimentos se seleccionan para siete variantes de problemas de planificación de rutas de vehículos instancias de la literatura [35-37].A modo resumen se tienen instancias con las siguientes características:

* Entre 50 y 600 clientes.
* Entre 3 y 150 vehículos.
* Entre 200 y 500 de capacidad.
* Para la variante MDVRP se tienen entre 2 y 8 depósitos.
* Para la variante TTRP se tienen entre 5 y 17 camiones con capacidades entre 100 y 150, así como entre 3 y 9 remolques con capacidad 100.

Para la experimentación se realizan 20 ejecuciones de las heurísticas seleccionadas debido a su naturaleza no determinista. Para ver los resultados exactos de cada instancia en las diferentes variantes con las heurísticas consultar [34]. A partir de los resultados alcanzados se obtiene que la versión anterior de BHCVRP desarrollada en [21] presenta los mejores resultados en las variantes CVRP, MDVRP y TTRP con la Algoritmo de Ahorros en su versión paralela, mientras que para HFVRP se destaca el Algoritmo de Ahorros en su versión secuencial.Para alcanzar mayor grado de certeza en la comparación de los algoritmos ya existentes en BHCVRP, se decide aplicar una prueba estadística no paramétrica de Friedman. Esta prueba se utiliza para determinar si existen diferencias significativas entre los resultados promedio alcanzados en cada heurística por cada versión de BHCVRP para las siete heurísticas en cuestión. En las ,

**Tabla *2***, y , se resumen dichos resultados.

**Tabla 1. Rankings de las heurísticas para la variante CVRP en cuanto a calidad de soluciones en distancia euclidiana para las tres versiones de BHCVRP.**

|  |  |
| --- | --- |
| Algoritmos | Rankings |
| Java anterior | Java actual | Python actual |
| Método aleatorio | 6.88 | 6.75 | 6.38 |
| Vecino más cercano | 5 | 5.38 | 5.25 |
| Algoritmo de ahorros secuencialAlgoritmo de ahorros paraleloMole &JamesonCMTAlgoritmo de barrido | 2.63**1**2.634.135.75 | 2.5**1.38**2.53.635.88 | 2.88**1.88**2.382.886.38 |

**Tabla 2. Rankings de las heurísticas para la variante HFVRP en cuanto a calidad de soluciones en distancia euclidiana para las tres versiones de BHCVRP.**

|  |  |
| --- | --- |
| Algoritmos | Rankings |
| Java anterior | Java actual | Python actual |
| Método aleatorio | 6.88 | 6.88 | 6.75 |
| Vecino más cercano | 5.38 | 5.31 | 5.06 |
| Algoritmo de ahorros secuencialAlgoritmo de ahorros paraleloMole &JamesonCMTAlgoritmo de barrido | **1.5**2.382.1345.75 | **1.75**2.3823.885.81 | 2.52.63**2.25**2.636.19 |

**Tabla 3. Rankings de las heurísticas para la variante MDVRP en cuanto a calidad de soluciones en distancia euclidiana para las tres versiones de BHCVRP.**

|  |  |
| --- | --- |
| Algoritmos | Rankings |
| Java anterior | Java actual | Python actual |
| Método aleatorio | 7 | 6.88 | 5.75 |
| Vecino más cercano | 5 | 5.25 | 5.88 |
| Algoritmo de ahorros secuencialAlgoritmo de ahorros paraleloMole &JamesonCMTAlgoritmo de barrido | 3**1.38**1.6346 | 3.251.75**1.38**3.755.75 | 4.632.63**1**2.385.75 |

**Tabla 4. Rankings de las heurísticas para la variante TTRP en cuanto a calidad de soluciones en distancia euclidiana para las tres versiones de BHCVRP.**

|  |  |
| --- | --- |
| Algoritmos | Rankings |
| Java anterior | Java actual | Python actual |
| Método aleatorio | 6.88 | 6.63 | 6 |
| Vecino más cercano | 4.75 | 4.75 | 5 |
| Algoritmo de ahorros secuencialAlgoritmo de ahorros paraleloMole &JamesonCMTAlgoritmo de barrido | **1**3.52.253.56.13 | **1.25**3.52.133.386.38 | 2.5**1.13**2.3847 |

Además, se confecciona la

**Tabla *5***, para obtener un *ranking* general de las diferentes versiones de BHCVRP. Para ello se tuvo en cuenta la unión de los *rankings* de las siete heurísticas para las cuatro variantes VRPs.

**Tabla 5. Ranking general de las versiones de BHCVRP.**

|  |  |
| --- | --- |
| Versión de BHCVRP | Ranking |
| Java anterior | 2 |
| Java actual | **1.9** |
| Python actual | 2.1 |

Para alcanzar mayor precisión, se aplican los métodos *post-hoc* Li, Finner y Holm, como se aprecia en la . Para ello se utilizan los p-valores de las versiones de BHCVRP. Se compara la versión que obtuvo mejores resultados (Java actual) con respecto a las demás versiones, para determinar si existen diferencias significativas entre ellas.

**Tabla 6. Resultados de los métodos post-hoc Li, Finner y Holm para el ranking general de Friedman de las diferentes versiones de BHCVRP.**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Versión de BHCVRP | p-valor sin ajustar | p-valor de Li | p-valor de Finner | p-valor de Holm |
| Python actual | 0.349575 | 0.49265 | 0.576947 | 0.69915 |
| Java anterior | 0.639994 | 0.639994 | 0.639994 | 0.69915 |

Al obtener un p-valor mayor que 0.05, se demuestra que no existen diferencias significativas entre las diferentes versiones de BHCVRP, a pesar de que para las instancias probadas la versión actual desarrollada en Java obtuvo los mejores resultados.

Por último, es importante destacar que resultados parciales de esta investigación han sido presentados en eventos como la Jornada Científica Estudiantil a nivel de facultad [38] y a nivel de universidad [39] obteniendo premios Relevante y Destacado, respectivamente.

**4. Conclusiones**

La nueva versión de BHCVRP resuelve las deficiencias encontradas en la versión anterior. Se incorporan dos nuevas heurísticas de construcción: el Algoritmo de Ahorro basado en *Matching* y la Heurística de Asignación Generalizada de Fisher &Jaikumar. Además, se plasman un conjunto de mejoras en la arquitectura y funcionamiento de la biblioteca como son: la creación de la primera versión que cubre los lenguajes de programación Python y Java, el tratamiento de excepciones propias relacionadas con la distancia, capacidad de los vehículos, velocidad y tiempo de ejecución, así como la aplicación del patrón de diseño *TemplateMethod* para favorecer la flexibilidad y reutilización. A partir de la comparación realizada para las diferentes versiones de BHCVRP, se aprecia que no existen diferencias significativas, a pesar de que la versión actual de Java obtuvo mejores resultados. Otros aspectos interesantes a considerar para futuras investigaciones son: aumentar el número de instancias a probar, utilizar distancias reales o aproximadas como Manhattan, Chebyshev y Haversine, aplicar técnicas de minería de datos como árboles de decisión para determinar en qué tipo de instancias es más adecuado utilizar determinada heurística, entre otras.

**Referencias**

[1] B. Golden, X. Wang, and E. Wasil, "The evolution of the Vehicle Routing Problem—A survey of VRP research and practice from 2005 to 2022," in *The Evolution of the Vehicle Routing Problem: A Survey of VRP Research and Practice from 2005 to 2022*: Springer, 2023, pp. 1-64.

[2] G. B. Dantzig and J. H. Ramser, "The truck dispatching problem," *Management science,* vol. 6, no. 1, pp. 80-91, 1959.

[3] G. Clarke and J. W. Wright, "Scheduling of vehicles from a central depot to a number of delivery points," *Operations research,* vol. 12, no. 4, pp. 568-581, 1964.

[4] M. Zirour, "Vehicle routing problem: models and solutions," *Journal of Quality Measurement and Analysis JQMA,* vol. 4, no. 1, pp. 205-218, 2008.

[5] G. D. Konstantakopoulos, S. P. Gayialis, and E. P. Kechagias, "Vehicle routing problem and related algorithms for logistics distribution: A literature review and classification," *Operational research,* vol. 22, no. 3, pp. 2033-2062, 2022.

[6] E. Arifta and F. Rakhmawati, "Analysis of Book Distribution Routes Using the Capacity Vehicle Routing Problem (CVRP) Method Using the Sweep Algorithm," *Sinkron: jurnal dan penelitian teknik informatika,* vol. 8, no. 1, pp. 360-367, 2023.

[7] F. Morsidi and I. Y. Panessai, "Overview of the Integral Impact of MDVRP Routing Variables on Routing Heuristics," *Applied Information Technology And Computer Science,* vol. 4, no. 1, pp. 1723-1738, 2023.

[8] V. R. Máximo, J.-F. Cordeau, and M. C. Nascimento, "An adaptive iterated local search heuristic for the Heterogeneous Fleet Vehicle Routing Problem," *Computers & Operations Research,* vol. 148, p. 105954, 2022.

[9] L. Accorsi and D. Vigo, "A hybrid metaheuristic for single truck and trailer routing problems," *Transportation Science,* vol. 54, no. 5, pp. 1351-1371, 2020.

[10] E.-G. Talbi, *Metaheuristics: from design to implementation*. Hoboken, New Jersey: John Wiley & Sons, Inc., 2009.

[11] F. Liu, C. Lu, L. Gui, Q. Zhang, X. Tong, and M. Yuan, "Heuristics for vehicle routing problem: A survey and recent advances," *arXiv preprint arXiv:2303.04147,* 2023.

[12] M. L. Fisher and R. Jaikumar, "A generalized assignment heuristic for vehicle routing," *Networks,* vol. 11, no. 2, pp. 109-124, 1981.

[13] M. H. Romanycia and F. J. Pelletier, "What is a heuristic?," *Computational intelligence,* vol. 1, no. 1, pp. 47-58, 1985.

[14] R. Martı, "Procedimientos metaheurısticos en optimización combinatoria," *Matemátiques, Universidad de Valencia,* vol. 1, no. 1, pp. 3-62, 2003.

[15] A. Olivera, "Heurísticas para problemas de ruteo de vehículos," *Reportes Técnicos 04-08,* 2004.

[16] A. C. Pérez-Pérez, E. Sánchez-Ansola, and A. Rosete-Suárez, "A metaheuristic solution for the school bus routing problem with homogeneus fleet and bus stop selection," *Ingeniería,* vol. 26, no. 2, pp. 233-253, 2021.

[17] Y. Koguma, "Tabu Search-based Heuristic Solver for General Integer Linear Programming Problems," *IEEE Access,* 2024.

[18] C. R. Gatica, S. M. Molina, and G. Leguizamón, "Evaluación de variantes de la metaheurística VNS para el problema de planificación de máquinas paralelas," in *XXVIII Congreso Argentino de Ciencias de la Computación (CACIC)(La Rioja, 3 al 6 de octubre de 2022)*, 2023.

[19] C. Groër, B. Golden, and E. Wasil, "A library of local search heuristics for the vehicle routing problem," *Mathematical Programming Computation,* vol. 2, pp. 79-101, 2010.

[20] K. Florios and G. Mavrotas, "Generation of the exact pareto set in multi-objective traveling salesman and set covering problems," *Applied Mathematics and Computation,* vol. 237, pp. 1-19, 2014.

[21] L. Díaz-Subi, "Nueva versión de la Biblioteca de Heurísticas de Construcción para Problemas de Planificación de Rutas de Vehículos," Tesis de diploma, Facultad de Ingeniería Informática, Instituto Superior Politécnico José Antonio Echeverría (CUJAE), 2016.

[22] F. Didier, L. Perron, S. Mohajeri, S. A. Gay, T. Cuvelier, and V. Furnon, "OR-Tools’ vehicle routing solver: a generic constraint-programming solver with heuristic search for routing problems," 2023.

[23] A. Mahmoud, T. Chouaki, S. Hörl, and J. Puchinger, "Extending JSprit to solve electric vehicle routing problems with recharging," *Procedia Computer Science,* vol. 201, pp. 289-295, March 22-25 2022.

[24] M. Karkula, J. Duda, and I. Skalna, "Comparison of capabilities of recent open-source tools for solving Capacitated Vehicle Routing Problems with Time Windows," *Carpathian Logistics Congress,* Accessed on: December 2-4, 2019

[25] M. PAREJO *et al.*, "Desarrollo de un algoritmo para seleccionar la ubicación de puntos de abastecimiento minimizando los tiempos y costos de transporte," *Revista Espacios,* vol. 41, no. 17, pp. 1-15, 2020.

[26] O. E. Calderón Calderón, "Diseño e implementación de modelo matemático para la optimización del recorrido de camiones recolectores de residuos sólidos en el municipio de León," 2020.

[27] Rita M. Newton and W. Thomas, "Design of School Bus Routes by Computer," *Socio-Economic Planning Sciences,* vol. 3, pp. 75-85, 1969.

[28] E. E. Zachariadis and C. T. Kiranoudis, "An open vehicle routing problem metaheuristic for examining wide solution neighborhoods," *Computers & Operations Research,* vol. 37, no. 4, pp. 712-723, 2010.

[29] A. F. Tangarife Álvarez, "Revisión del estado del arte del problema de ruteo abierto (OVRP)," 2017.

[30] J.-F. Cordeau and G. d. é. e. d. r. e. a. d. décisions, *The VRP with time windows*. Citeseer, 2000.

[31] J. M. Daza, J. R. Montoya, and F. Narducci, "Resolución del problema de enrutamiento de vehículos con limitaciones de capacidad utilizando un procedimiento metaheurístico de dos fases," *Revista EIA,* no. 12, pp. 23-38, 2009.

[32] N. Tok and Ş. Özkar, "Solution Of Capacitated Vehicle Routing Problem For A Food Delivery Company With Heuristic Methods," *International Review of Economics and Management,* vol. 11, no. 1, pp. 1-16, 2023.

[33] E. Gamma, R. Helm, R. Johnson, and J. Vlissides, *Design patterns: elements of reusable object-oriented software*. Pearson Deutschland GmbH, 1995.

[34] A. Morales, "Nueva versión de la bibilioteca de heurísticas de construcción para problemas de planificación de rutas de vehículos," Tesis de Diploma, Facultad de Ingeniería Informática, Universidad Tecnológica de La Habana "José Antonio Echeverría", La Habana, Cuba, 2025.

[35] E. Alba, "Networking and emerging optimization,"Available: <http://neo.lcc.uma.es/vrp/vrp-instances/>

[36] N. Christofides, "The vehicle routing problem," *Combinatorial optimization,* 1979.

[37] I. M. Chao, "A tabu search method for the truck and trailer routing problem," *Computers & Operations Research,* vol. 29, pp. 33-51, 2002.

[38] A. Morales, "Nueva versión de la Biblioteca de Heurísticas de Construcción para Problemas de Planificación de Rutas de Vehículos," in *Jornada Científica Estudiantil a nivel de facultad*, Facultad de Ingeniería Informática, 2024.

[39] A. Morales, "Nueva versión de la Biblioteca de Heurísticas de Construcción para Problemas de Planificación de Rutas de Vehículos," in *Jornada Científica Estudiantil a nivel de universidad*, CITI, CUJAE, 2024.