UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI

FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACIÓN, ADMINISTRACIÓN Y ECONOMÍA EMPRESARIAL

CARRERA DE LOGÍSTICA

PLAN DE INVESTIGACIÓN

Tema: "Módulo didáctico de métodos exactos y enfoques heurísticos para la resolución de problemas de ruteo de vehículos (VRP)."

Trabajo de titulación previa la obtención del

título de Ingeniero en Logística

AUTOR: Danny Alexis Benavides Caipe

TUTOR: Iván Gabriel Mafla Bolaños, Msc

Tulcán, 2020

CERTIFICADO JURADO EXAMINADOR

Certificamos que el estudiante Benavides Caipe Danny Alexis con el número de cédula 0401838305 ha elaborado el trabajo de titulación: "Módulo didáctico de métodos exactos y enfoques heurísticos de resolución de problemas de ruteo de vehículos (VRP) ".

Este trabajo se sujeta a las normas y metodología dispuesta en el Reglamento de Titulación, Sustentación e Incorporación de la UPEC, por lo tanto, autorizamos la presentación de la sustentación para la calificación respectiva.

04012544 46 IVAN GABRIEL MAFLA BOLAÑOS
Firmado digitalmente por 0401254446 IVAN GABRIEL MAFLA BOLAÑOS
Fecha: 2020.10.21 18:04:57 -05'00'

f.....

Mafla Bolaños Iván Gabriel, Msc

TUTOR

Javier Pozo
Firmado digitalmente porJavier Pozo
DN: cn=Javier Pozo gn=Javier Pozo c=Ecuador l=EC o=UPEC ou=UPEC e=eduardo.pozo@upec.edu.ec
Motivo:He revisado este documento
Ubicaci- n:
Fecha:2020-10-21 14:53+19:00

f.....

Pozo Burgos Eduardo Javier, Msc

LECTOR

Tulcán, octubre de 2020

AUTORÍA DE TRABAJO

El presente trabajo de titulación constituye requisito previo para la obtención del título de Ingeniero en la Carrera de ingeniería en logística de la Facultad de Comercio Internacional, Integración, Administración y Economía Empresarial

Yo, Benavides Caipe Danny Alexis con cédula de identidad número 0401838305 declaro: que la investigación es absolutamente original, auténtica, personal y los resultados y conclusiones a los que he llegado son de mi absoluta responsabilidad.

f.....

Benavides Caipe Danny Alexis

AUTOR

Tulcán, octubre de 2020

ACTA DE SESIÓN DE DERECHOS DEL TRABAJO DE TITULACIÓN

Yo, Benavides Caipe Danny Alexis declaro ser autor de los criterios emitidos en el trabajo de investigación: “Módulo didáctico de métodos exactos y enfoques heurísticos de resolución de problemas de ruteo de vehículos (VRP)” y eximo expresamente a la Universidad Politécnica Estatal del Carchi y a sus representantes legales de posibles reclamos o acciones legales.

f.....

Benavides Caipe Danny Alexis

AUTOR

Tulcán, octubre de 2020

AGRADECIMIENTO

Agradezco a mi familia por su apoyo constante durante mi carrera universitaria. Agradezco a mi tutor, quien me brindó su apoyo, dirección y paciencia durante todo el proceso de realización y culminación de este trabajo.

DEDICATORIA

Este trabajo se lo dedico a mis padres, como una forma de agradecimiento por todos sus esfuerzos y el apoyo que me brindaron durante toda mi carrera universitaria, además me lo dedico a mí mismo, porque su realización permitió desarrollar potencialmente mis destrezas y aplicar los conocimientos adquiridos en el aula de clase.

ÍNDICE GENERAL

ÍNDICE DE GRÁFICOS

ÍNDICE DE TABLAS

ÍNDICE DE ANEXOS

RESUMEN

El presente trabajo muestra la programación y desarrollo de un módulo didáctico de métodos de solución para problemas VRP, para lo cual se ha utilizado herramientas cuantitativas, incorporando los conocimientos adquiridos dentro de la carrera de Ingeniería en Logística de la Universidad Politécnica Estatal del Carchi, para demostrar la importancia de conocer sobre los tipos de problemas existentes, los métodos de solución y su programación, y los recursos informáticos necesarios para su resolución. Mediante este trabajo se ha podido comprobar que los métodos más adecuados son las heurísticas y meta heurísticas ya que permiten obtener soluciones válidas a un problema específico, mientras que los métodos exactos que proporcionan la respuesta óptima del problema requieren muchos recursos computacionales para su funcionamiento. El módulo didáctico permite reconocer el método más eficiente con base en el tiempo de cómputo y la respuesta operativa del problema. Para el tiempo de cómputo, el algoritmo del vecino más cercano da la mejor respuesta en dos de los tres equipos en donde se realizaron los experimentos y en el otro fue el método de Clarke y Wright lo cual permite comprobar que las heurísticas son más eficientes en lo que respecta al uso de recursos computacionales. Para la solución de los parámetros operativos del problema la mejor respuesta es la del algoritmo *brutal search*, ya que al ser un método exacto siempre obtendrá la respuesta óptima del problema. Para el escenario planteado, la solución encontrada por medio del método exacto es más eficiente en un 5% que la heurística del vecino más cercano y un 43% que el método de Clarke y Wright. Los tiempos de ejecución de las heurísticas no sobrepasan los 0,2 segundos, por lo tanto son mas eficientes que el método exacto que requiere de un tiempo de ejecución de alrededor de 400 segundos.

Palabras clave: problemas VRP, heurísticas, meta heurísticas, métodos exactos, óptimo.

ABSTRACT

This work shows the programming and development of a didactic module of solution methods for VRP problems by using quantitative tools and incorporating the knowledge acquired in the career of engineering in logistics of the Universidad Politécnica Estatal del Carchi. This process has evidenced the importance of knowing about the types of existing problems, the methods of solution, aspects related to programming and the necessary computer resources for VRP solving. During the research, it has been possible to verify that the heuristics and meta heuristics are the most suitable methods because they allow obtaining valid solutions to a specific problem, while the exact methods require many computational resources to provide the optimal answer to the problem efficiently. The didactic module helped to identify the most efficient methods based on the computation time and the operational response of the problem. For the computational time, the nearest neighbor algorithm provided the best answer in two of the three computers where the experiments were made, while in the other, the Clarke and Wright method performed better. These tests allowed verifying that the heuristics are more efficient in terms of the use of computational resources. For the solution related to operational parameters of the problem, the brutal search algorithm produced the best answer because it is an exact method that will always obtain the optimal answer to the problem. In the proposed scenario, the solution found through the exact method was 5% more efficient than the one of the nearest neighbor algorithm and 43% more efficient than the Clarke and Wright's. The execution times of the heuristics did not exceed 0.2 seconds; therefore, they are more efficient than the exact method that required an execution time of around 400 seconds.

Key words: VRP problems, heuristics, meta heuristics, exact methods, optimum.

INTRODUCCIÓN

Para formular problemas de ruteo vehicular (VRP) es necesario tener conocimientos previos sobre la definición de los aspectos más relevantes que afectan el proceso de transporte y distribución de productos dentro de una empresa, esto sugiere que la persona encargada de realizar esta actividad deberá establecer las dimensiones sobre las cuales se va a trabajar, además de conocer las formulaciones matemáticas para los tipos de VRP que ya se han planteado y tener la capacidad de modificarlos según convenga.

En el presente trabajo se analiza definiciones relacionadas con el proceso de distribución en logística, el transporte, tipos de problemas que se han formulado a través del tiempo, métodos de solución, lenguajes de programación, teorías relacionadas con el desarrollo del conocimiento por parte del programador y enfocado a fines didácticos para los estudiantes que desean conocer sobre el proceso y el funcionamiento de los algoritmos para solución de VRP. Se ha utilizado un tipo de investigación experimental, la cual permite aplicar los conocimientos adquiridos en el aula de clase y transformarlos a un lenguaje de programación funcional que sea capaz de dar una solución a problemas VRP.

La generación de un módulo didáctico permitirá reforzar los conocimientos adquiridos en el aula de clase y promover la aplicación y desarrollo de los mismos, que beneficien a la innovación y creación de herramientas tecnológicas que contribuyan a la solución de problemas y toma de decisiones.

I. PROBLEMA

1.1 PLANTEAMIENTO DEL PROBLEMA

Ecuador ha tenido un importante desarrollo logístico que lo ubica entre los 10 primeros puestos en el índice de desempeño logístico de Latinoamérica y el Caribe en 2018 (Competitividad, 2018). El país se encuentra en este lugar por la importante inversión en infraestructura para el transporte hecha en los últimos 10 años, impulsando el comercio nacional e internacional que se moviliza por tierra, mar y aire, pero la falta de industrias dificulta y retrasa el desarrollo económico, ya que, las personas que no tienen conocimientos básicos de cómo empezar un negocio y cómo hacer para que las cosas funcionen de tal manera que formen un sistema conjunto de cooperación que haga a ese negocio rentable, simplemente se orientan a la "suerte" y buscan emprender sin un pensamiento de innovación, desarrollo y aportación al progreso tecnológico e industrial del país. Si todo sale bien, esa empresa será un apoyo importante para su propietario, pero son pocas las personas que se arriesgan a la superación o expansión de su compañía.

En todo ámbito es aplicable la logística, pero es mucho más importante para las empresas que desean controlar cada una de sus actividades y generar un beneficio haciendo óptimos sus procesos. Todos los procesos de la empresa son importantes y dentro de toda esta cadena debe estar presente siempre la logística como el eje transversal que los conecta.

La programación de rutas o ruteo es una acción relevante en empresas de sector comercial, es decir, que se dedican específicamente a la distribución y entrega de sus productos. Esta tarea es difícil, ya que se requiere de un amplio conocimiento del área a cubrir. Básicamente, las personas que se encargan de esto, son quienes están en contacto directo con los clientes y se trasladan en el camión repartidor para generar estas rutas de su propia experiencia. La manera de establecer estas rutas es empírica y rutinaria, ya que conocen muy bien la ubicación exacta de cada negocio a ser atendido y deciden el orden en que visitaran cada una.

En la ciudad de Tulcán existen varias empresas dedicadas a la comercialización de productos en general, y por la cantidad de puntos de entrega que deben cubrir es complicado que utilicen un algoritmo para establecer las rutas que van a seguir a diario en la ciudad. Además las empresas casi siempre prefieren tomar la dirección exacta del destino y que el repartidor sea el encargado de identificarlo. En una empresa que acaba de iniciar sus funciones como distribuidora sería un problema para las personas que ejecutan el reparto para encontrar la ruta

óptima hacia los destinos, a menos que tengan un conocimiento amplio de las calles de la ciudad y puedan movilizarse y ubicarse sin ningún problema.

Es necesario un algoritmo para calcular rutas, que permita optimizar todos los costos provenientes del proceso de distribución, que es uno de los más caros para las empresas, ya que implica varios lineamientos que necesitan de un aporte económico para ser ejecutados como, por ejemplo, combustible, depreciación del vehículo, desgaste de llantas, mantenimiento, permisos necesarios para la circulación, entre otros. Muchas empresas no cuentan con el asesoramiento necesario para la utilización de este tipo de algoritmos que es de mucha ayuda para evaluar su proceso de distribución, comparar rutas de entrega en tiempo real, calcular costos y optimizarlos, pero para esto debe existir un modelo que se adapte a sus necesidades y al mismo tiempo sea amigable con el usuario, permitiendo hacer más sencilla su utilización.

Para los estudiantes de la carrera de logística es una necesidad el conocer múltiples métodos de solución para problemas de ruteo, ya que en la etapa laboral encontrarán varios ejercicios de este tipo que deberán ser resueltos. Lo ideal sería contar con aplicaciones que integren múltiples métodos de solución para problemas de ruteo en donde las personas puedan interactuar y con los resultados obtenidos determinar el algoritmo que se adapte mejor a sus necesidades y que optimice este proceso.

1.2. FORMULACIÓN DEL PROBLEMA

¿Cómo desarrollar un módulo didáctico de métodos exactos y enfoques heurísticos para la resolución de problemas de ruteo de vehículos (VRP).?

1.3. JUSTIFICACIÓN

La necesidad de establecer rutas que optimicen el proceso de distribución de una empresa es fundamental y los resultados se verán reflejados en los costos en los que incurre la empresa. Una aplicación que integre varios modelos de solución para problemas de ruteo y que también permita interactuar entre modelos para buscar la mejor solución, será de mucha utilidad para que las empresas empiecen a añadir procesos de planificación más sofisticados y tecnológicos, además de utilizar sus recursos eficientemente obteniendo el mayor provecho posible y generando más ganancias.

Los trabajadores que se movilizan en el vehículo repartidor casi siempre utilizan su conocimiento para generar estas rutas, pero solo las generan de forma empírica, es decir, sin

una base técnica, solo se basan en experiencia propia que ellos adquieren probando cual ruta será la mejor para atender más pronto a todos los clientes, sin saber si el camino que ellos deciden tomar es o no el correcto.

Para los estudiantes de la carrera de logística es muy importante contar con los recursos tecnológicos adecuados que permitan establecer sistemas para facilitar la toma de decisiones mediante pruebas de varios escenarios que podrían generarse en el caso de un problema de ruteo, además de que una aplicación de este tipo mejoraría el proceso de aprendizaje y reforzaría el conocimiento mediante la aplicación y experimentación, y también tendría un beneficio sobre la experiencia que adquieran los estudiantes para utilizar herramientas tecnológicas en su vida laboral.

1.4. OBJETIVOS Y PREGUNTAS DE INVESTIGACIÓN

1.4.1. Objetivo General

Desarrollar un módulo didáctico de métodos exactos y enfoques heurísticos para la resolución problemas de ruteo de vehículos (VRP).

1.4.2. Objetivos Específicos

Establecer los parámetros de dimensionamiento del modelo matemático para problemas de planificación y diseño de rutas.

Desarrollar los diagramas de flujo de los algoritmos de solución de problemas VRP a programar en el módulo.

Diseñar, escribir y programar los códigos de programación de los algoritmos de solución de problemas VRP y la interfaz del módulo didáctico.

Identificar el método de solución adecuado para el problema VRP planteado a partir de los resultados obtenidos en el módulo.

1.4.3. Preguntas de Investigación

¿Cuáles son los parámetros de dimensionamiento del modelo matemático para problemas de planificación y diseño de rutas?

¿Cómo desarrollar los diagramas de flujo de los algoritmos de solución de problemas VRP a programar en el módulo?

¿Cómo diseñar, escribir y programar los códigos de programación de los algoritmos de solución de problemas VRP y la interfaz del módulo didáctico?

¿Cuál es el método de solución adecuado para el problema VRP planteado a partir de los resultados obtenidos en el módulo?

II. FUNDAMENTACIÓN TEÓRICA

2.1. ANTECEDENTES INVESTIGATIVOS

Una primera investigación, es la de Ocaña y Ramírez (2012), quienes realizaron un trabajo relacionado con el desarrollo de un modelo matemático a partir de la heurística de Clarke and Wright, y que además cuenta con una restricción de ventanas de tiempo. Este proyecto cuenta con un análisis documental apoyado por la información real de la empresa y se añaden imágenes que describen las variables a considerar para su desarrollo.

La metodología utilizada tiene relación con la aplicación del conocimiento, describiendo de manera objetiva los problemas y posibles soluciones que pueden aplicarse a cada interrogante para el cumplimiento de los objetivos planteados. Esta investigación tiene un aporte en cuanto al diseño del modelo matemático, que se basa en un algoritmo ya existente lo cual permitirá una fácil comprensión sobre el uso y la constitución del mismo. En este sentido el proyecto será de mucha utilidad para analizar los resultados documentados por los autores y los beneficios obtenidos después de haber realizado el trabajo.

Una segunda investigación, es la realizada por Hernández (2016) en la ciudad de Bogotá, la cual se relaciona con el desarrollo de una herramienta que permite facilitar el proceso de ruteo, teniendo en cuenta que la empresa tiene varios depósitos. En el desarrollo del proyecto se pueden identificar los problemas por los cuales atraviesan las empresas distribuidoras día a día y se proponen varias soluciones con diferentes escenarios buscando la optimización de procesos a través de una meta heurística híbrida.

Para el desarrollo del trabajo antes mencionado, el autor realiza una investigación exhaustiva en libros y escritos de reconocidos autores. El aporte de este proyecto a la presente investigación va desde la propuesta y diseño del modelo meta heurístico, hasta la bibliografía utilizada que proporcionará un vasto conocimiento en lo que se refiere al ruteo con múltiples depósitos.

Un tercer proyecto de investigación realizado por Herazo (2012) en la ciudad de Barranquilla, presenta la modelación matemática del problema de ruteo integrando algunas restricciones encontradas en problemas reales como, flota heterogénea de vehículos, ventanas de tiempo y múltiples depósitos. En el desarrollo de este trabajo el autor analiza varios problemas de ruteo de vehículos y desarrolla un modelo matemático para la planificación de rutas con las restricciones antes mencionadas.

La metodología utilizada para esta investigación tiene que ver con la aplicación del conocimiento partiendo de la definición del concepto de logística, transporte, tipos de problemas de ruteo, definición del algoritmo que ayudan a comprender el alcance del trabajo, además de analizar y explicar los diferentes métodos de solución para cada problema de ruteo.

Este proyecto tendrá un aporte importante en el presente proyecto de investigación ya que el objetivo es muy similar en los dos casos, el diseño de un modelo matemático para un problema de ruteo incluyendo todas las restricciones, pero en el presente trabajo se intenta programar varios algoritmos incluyendo también el beneficio para el usuario, es decir, que sea de fácil utilización y sencillo para aplicar y que se adapte a la mayoría de casos que pudieran identificarse en el ruteo.

2.2. MARCO TEÓRICO

2.2.1. Transporte

Como lo afirma Fuentes, 1960 (como se citó en Islas y Zaragoza, 2003):

> es, de todas las operaciones que efectúa el ser humano, una de las más necesarias y la más uniforme a la vez. No hay uno solo de nuestros actos, que no implique en su elaboración, en su realización, el desplazamiento de personas, de cosas, de pensamiento. (p. 19)

Desde tiempos inmemoriales las personas utilizan el transporte como medio para trasladar bienes y personas por grandes y pequeñas distancias con el fin de ir desde un origen hacia un destino para cumplir algún objetivo en específico. Con la aparición del intercambio de bienes este deseo de trasladarse se convierte en una necesidad, ya que los lugares donde se daba el comercio, llamados mercados, estaban alejados de los productores que debían llevar su mercadería hacia estos lugares para así poder intercambiarlos por otro bien que satisfaga alguna necesidad que tuvieran, a esto se le llama trueque.

El transporte conecta a las personas con las actividades que vayan a realizar, en un lugar específico y en el momento que lo requieran, pero surgen problemas al momento de desempeñar esta función, debido a que varias personas quieren ir a diferentes lugares al mismo tiempo generando así el tráfico o congestión vehicular, esto trae efectos negativos incluso en la salud de las personas porque genera estrés. Además de provocar contaminación al medio ambiente, también genera mucho ruido lo cual molesta a las personas que se movilizan y residen en el lugar de tránsito de vehículos. A esto se le suma el problema que se genera para las empresas

del sector terciario, es decir, las que se dedican a la comercialización y distribución de sus productos, ya que los vehículos deben cubrir una gran cantidad de puntos de entrega en vías congestionadas lo cual genera un gran retraso e insatisfacción del servicio para el cliente.

2.2.2. Logística

La palabra logística se menciona por primera vez en la segunda guerra mundial por miembros del ejército de Estados Unidos. Acabada esta guerra los grandes profesionales que gestionaron la logística durante la guerra prestan sus servicios a las empresas como asesores para la implementación de un sistema logístico dentro de todas sus actividades. Desde este acontecimiento la logística ha tenido avances enormes en las técnicas y estrategias utilizadas para la optimización de procesos dentro de una empresa, además ha ido de la mano con la electrónica, la informática, la modelación matemática que ayudan a comprender y desarrollar nuevas maneras más eficientes de lograr los objetivos de una compañía (Meserón, 2007).

El *Council of Logistics Management*, 1986 (como se citó en Shouyan, Ning y Xu, 2019) define a la logística como:

> el proceso de planificación, implementación y control eficiente del flujo efectivo de costes y almacenaje de materiales, inventarios en curso y productos terminados, así como la información relacionada desde el punto de origen al punto de consumo con el fin de atender a las necesidades del cliente. (párr, 5)

Para una empresa debe ser vital planificar sus actividades a fin de optimizarlas maximizando las ganancias, además de implementar estrategias que le permitan utilizar todos sus recursos con el mínimo desperdicio. Para esto es necesario estar siempre realizando controles en los procesos realizados con el fin de identificar las falencias del sistema y en este punto también se hace necesario el flujo de información para dar a conocer sobre los problemas identificados y tomar una decisión en cuanto a la deficiencia para así poder corregirla o reducirla.

Figura 1. Descripción gráfica de las actividades de la cadena logística *(hazlogística, 2018)*.

Existen varios tipos de logística dentro de una empresa que rigen y controlan sus procesos las cuales según la revisión realizada a la publicación de Nuño (2017) son:

- Logística de aprovisionamiento: controla el flujo de suministros necesarios para el desempeño de las funciones de la empresa.
- Logística de producción: en una empresa productora, es necesario tomar decisiones que minimicen los costos implicados en el proceso productivo, y de esta manera generar más ganancias.
- Logística de distribución: cuando se tiene el producto terminado en la bodega es necesario comercializar para recuperar el capital invertido, haciendo llegar el producto hasta el cliente para satisfacer su necesidad.
- Logística inversa: es necesario retirar productos que hayan sufrido algún tipo de daño o, en productos alimenticios se hayan caducado, y darles un adecuado tratamiento en la empresa.

2.2.3. Logística de distribución

La logística de distribución se encarga del flujo de productos terminados hasta el punto de venta, y luego al consumidor final, cabe recalcar que el proceso de transporte de las mercancías es el más costoso dentro de la cadena de suministros. La logística de distribución es la respuesta a la necesidad de situar el producto en el momento adecuado, cantidad adecuada, lugar adecuado y con el menor coste posible. Es importante la identificación de todos los costos inmersos en este proceso, ya que si se cuenta con un vehículo propio se debe calcular su depreciación, desgaste

de llantas, consumo de combustible, mantenimiento y otros costos presentes, si el camión es rentado se debe tener en cuenta el flete que depende de la distancia a recorrer, seguro, características del vehículo requerido, entre otros.

Se debe tener cuenta varias premisas importantes para evaluar la calidad de servicio en el transporte, como rapidez, confiabilidad, seguridad, higiene, cumplimiento de las fechas y horas de entrega, información y control del transporte. Pero sin embargo la planeación de este proceso es muy complicada, ya que no se podrá predecir cuál será el flujo del tránsito, que sería el problema más común para el retraso de las entregas, además de que el vehículo puede ir a mantenimiento constantemente pero aun si así fuera no está exento de sufrir una descompostura, es por eso que en el transporte se deben considerar planes alternativos para poder dar solución a algún inconveniente que pudiera suscitarse.

Dentro de una empresa comercializadora es importante la red de distribución que hará llegar el producto a todos los puntos de venta con los que la empresa ha hecho negociaciones para adquirir su producto. El ruteo es un método que se utiliza para poder establecer la manera óptima de llegar a cada punto de entrega, la desventaja es que no es tan sencillo realizar este tipo de procesos.

En algunas empresas se hace uso de algoritmos diseñados especialmente para este problema que cuentan con varias restricciones como el tamaño de la flota vehicular, número de depósitos, ventanas de tiempo, entre otras, que generarán un resultado más completo y exacto. Pero existen varios problemas ya definidos y cada uno tiene algoritmos diferentes para su resolución que encuentran la mejor manera de resolverlo.

2.2.4. Problemas VRP (Problema de ruteo de vehículos)

Los problemas VRP tienen gran importancia en el ámbito logístico, ya que requieren ser optimizados, lo cual debe ser profundamente estudiado. A lo largo del tiempo este tipo de problemas han evolucionado integrando varias restricciones tales como: número de vehículos, demanda, cantidad de depósitos, capacidades, ventanas de tiempo, entre otros (Rocha, González y Orjuela, 2011).

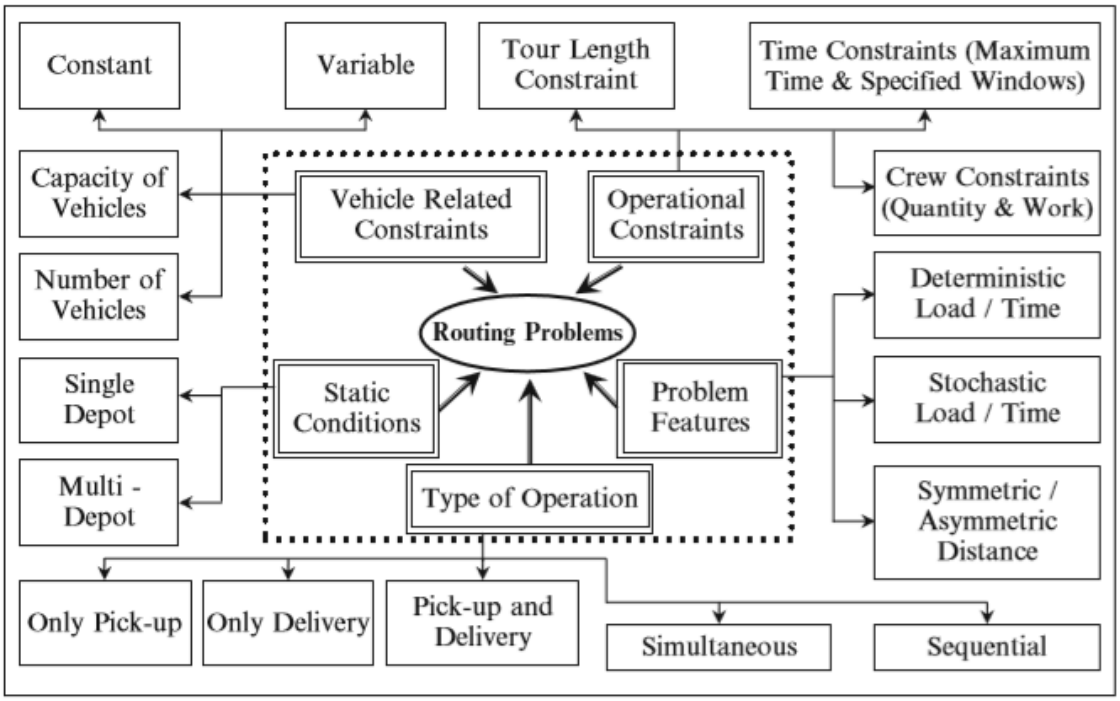


Figura 2. Dimensiones del problema de ruteo (Anbuudayasankar, Mohapatra y Ganesh, 2018, p. 4).

En 1956 Flood presenta el primer problema de tipo VRP denominado problema del agente viajero o TSP (*Travelling Salesman Problem*), en el cual se describe como un vendedor tiene que visitar cierta cantidad de ciudades en un solo viaje, pasando una sola vez por cada una y terminando el recorrido en la ciudad de origen. En 1959 aparece una variación al TSP descrito anteriormente, con el nombre de TSP generalizado publicado por Dantzing y Ramser en donde se modela el despacho de combustible a diversas estaciones de servicio, desde una terminal (Rocha et al. 2011). En 1960 surge la primera referencia del TSP múltiple o m-TSP generado por Miller, Tucker y Zemlin, este problema considera a varios clientes y varios camiones, esto quiere decir que existen también varias rutas una para cada camión, y el objetivo es que cada camión visite una sola vez a cada cliente, y vuelva al depósito. Para el problema m-TSP se le asigna una cierta demanda a cada cliente y a los camiones una capacidad de carga, restricciones que se pueden evidenciar en la vida real y que son tomadas en cuenta en la toma de decisiones al momento de buscar soluciones para un problema de ruteo. En 1969 Tillman desarrolla un modelo TSP probabilístico o PTSP que busca minimizar el costo del recorrido pasando por una serie de nodos, con una probabilidad de que los clientes requieran o no ser atendidos por el servicio (Rocha et al. 2011).

Los problemas TSP y todas sus variedades son la base para la formulación de problemas VRP como por ejemplo un m-TSP es muy similar a un CVRP (*capacited VRP*) que integra la capacidad de carga de la flota vehicular y es considerada como una restricción para la modelación, donde a cada cliente se le asigna una demanda, y se considera vehículos iguales que salen desde un mismo origen. En este tipo de problema se busca minimizar el costo que representa el transporte de mercancías desde un origen a un destino, tomando en cuenta varios nodos y considerando que la sumatoria de las demandas de todos los nodos en una sola ruta que será cubierta por un vehículo determinado debe ser igual o menor a la capacidad del vehículo. Existen dos tipos de CVRP, un modelo simétrico en donde el costo que representa ir de un centro de consumo a otro es igual para todos los casos, y si esto no se cumple es un modelo asimétrico.

A partir de los modelos TSP generalizados se desprenden dos grandes categorías que son: VRP homogéneos y VRP heterogéneos. Rocha et al. (2011) afirman:

> El VRP homogéneo se refiere a características comunes en las que todos los nodos manejan el mismo recurso como distancia, ventanas de tiempo, retornos y entregas fraccionadas. Por su parte, el VRP heterogéneo se refiere a componentes desiguales en las que cada nodo maneja recursos distintos bien sea flota de vehículos, depósitos, viajes y componentes estocásticos en algunos casos. (p. 38)

A continuación, se presentan los modelos VRP homogéneos principales, a partir de la revisión del trabajo realizado por Rocha et al. (2011):

DCVRP: desarrollado por Toth y Vigo, es un VRP en donde se integran restricciones de capacidad de los vehículos y distancias de recorrido.

VRPTW (1967): presentado por Pullen y Webb, es un problema en el cual el tiempo juega un papel crucial para cumplir con la entrega a los clientes, ya que se establecen ventanas de tiempo, es decir, intervalos en los que el cliente estará disponible para poder atender la recepción del producto. El hecho de que los vehículos lleguen a la hora acordada sería un sistema perfecto sin retrasos ni adelantos en el tiempo. Cuando un vehículo se retrasa deberá pasar al siguiente punto de entrega, por que el cliente ya no está disponible para recibir la mercancía, y cuando se adelanta deberá esperar la hora para cumplir con el servicio.

VRPB (1985): realizado por Golden, Baker, Alfaro y Scahffer, toma en cuenta retornos, esto puede darse en ciertos casos, por la falta de atención que genera una mala toma de pedidos y

puede ocasionar este tipo de molestias, aunque también se podría analizar desde el punto de vista de la logística inversa, en donde los clientes retornan productos en mal estado. Aunque este modelo intenta establecer dos rutas, la primera sería la del recorrido normal para entregar los pedidos y la segunda se refiere a pasar por los puntos en donde se han generado devoluciones, en ambos casos la demanda no debe exceder la capacidad del vehículo.

SDVRP (1989): modelado por Dror y Trudeau, es un problema en donde se considera la posibilidad de que un cliente sea visitado por más de un vehículo, ya que la cantidad de pedido ha superado la capacidad del vehículo.

A continuación, se detallan algunos de los principales modelos VRP heterogéneos, a partir de la revisión del trabajo realizado por Rocha et al. (2011) :

VRPHF (1994): desarrollado por Golden, Assad, Levy y Gheysens, en este tipo de problemas se toma en cuenta la variación en costo y el tipo de vehículo a utilizar, ya que se considera que se cuenta con un número ilimitado de vehículos con diferentes características, entonces se toman decisiones sobre la ruta y el tipo de flota que se va a utilizar.

PVRP (1991): efectuado por Russell y Gribbin, toma en cuenta el periodo de planeación, para los problemas VRP típicos el periodo de planeación es de un solo día, y para este tipo de problemas se pueden considerar el número de días que sea necesario.

Multi-trip VRP (1990): planteado por Feischman, describe una situación en donde un vehículo puede cubrir varias rutas planeadas, aunque para llevar a cabo esta tarea hay que tener en cuenta la disponibilidad de vehículos y conductores.

MCVRP (1998): modelado por Guan y Zhu, consiste en llevar cantidades de más de un objeto a la vez en el mismo viaje.

MOVRP (1995): presentado por Boffe, utiliza varios objetivos que se refieren a aspectos del VRP como rutas, nodos y arcos, y recursos que afectan a los costos y beneficios, nivel de servicio, mantenimiento de vehículos, entre otros.

SVRP (1983): realizado por Stewart y Golden, es un modelo estocástico que depende de algunas variables aleatorias, como por ejemplo demanda de los clientes.

2.2.5. Métodos de solución

Mediante una revisión documental se han podido determinar tres grandes grupos de métodos de solución: métodos exactos, heurísticas y meta heurísticas (Rocha et al. 2011).

2.2.5.1. Métodos exactos

Los métodos exactos se utilizan para problemas de hasta 50 depósitos porque requieren de una gran capacidad de procesamiento computacional y toma mucho tiempo el poder obtener una respuesta. A continuación, se realizará un breve análisis de los principales métodos exactos usados para la resolución de problemas VRP.

En los métodos de búsqueda directa de árbol se realiza una búsqueda en cada nodo del árbol de acuerdo a varios criterios específicos propios de cada método (Rocha et al. 2011). El algoritmo de asignación de cota inferior desarrollado por Laporte, Mercure y Norbert en 1986, trata de establecer cotas inferiores para disminuir el número de vehículos a utilizar en la resolución de un problema VRP. El algoritmo de ramificación y acotamiento realizado por Litle, Morty, Sweeney y Karel en 1968, consiste en recorrer todos los nodos del árbol de arriba hacia abajo, estableciendo un programa lineal en cada nodo que permite evaluar si se lo puede conservar o no, es decir, que si un nodo no tiene una respuesta factible este es eliminado del modelo. El árbol del centro de k-grados modelado por Christofides en 1981, describe un número fijo de vehículos a utilizar, una solución factible agrupa las aristas en cuatro conjuntos que son: las aristas que no pertenecen a la solución, las aristas que forman el árbol, las aristas que inciden en el primer vértice, las aristas que no inciden en el primer vértice (Rocha et al. 2011).

La programación dinámica propuesta por Elion, Watson-Gandy y Christofides en 1971, se caracteriza por contar con un número fijo de vehículos, y trata de calcular el costo mínimo posible utilizando estos vehículos, pero esto se realiza dentro de cada subproblema teniendo en cuenta la longitud que deben recorrer los vehículos para pasar por todos los vértices del subconjunto, luego se suman los costos de todos los subconjuntos para tener el costo total (Rocha et al. 2011).

La programación lineal y entera comprende técnicas analizadas a continuación. El método de partición y generación de columnas desarrollado por Balinski y Quandt en 1964, consiste en la realización de tablas con coeficientes binarios para una ruta, si y solo si un determinado depósito pertenece a esa ruta específica y además se utiliza el mismo principio, pero esta vez sí y solo si

esa ruta pertenece a la solución óptima del sistema. La formulación de flujo de vehículos de dos y tres índices realizadas por Fisher y Jaikumar en 1978, comprenden los caminos y vehículos que unen un depósito con otro (Rocha et al. 2011). En la formulación de dos índices se toma en cuenta solo el camino que une un nodo con otro, la formulación con tres índices añade el tipo de vehículo que se ha utilizado, aunque cabe recalcar que los vehículos no deben ser iguales.

2.2.5.2. Heurísticas

Las heurísticas son métodos para aumentar el conocimiento, que proporcionan una respuesta válida que se acerca a la respuesta óptima del problema, tomando en cuenta una exploración limitada del espacio de búsqueda (Rocha et al. 2011). La primera heurística aplicable a los problemas VRP y que tuvo resultados efectivos fue propuesta por Clarke y Wright en 1964. La mayoría de las heurísticas para dar solución a problemas VRP fueron desarrolladas entre 1960 y 1990. Estos métodos se han utilizado para problemas en donde se tiene un solo depósito, y se trata de buscar la mejor ruta que se adapte mejor al problema. Las heurísticas pueden clasificarse en tres grupos: métodos constructivos, métodos de dos fases y heurísticas de mejora (Rocha et al. 2011).

Dentro de los métodos constructivos se encuentran los algoritmos de ahorro y las heurísticas de inserción. El algoritmo de ahorro desarrollado por Clarke y Wright es usado en problemas donde el número de vehículos a utilizar es una variable de decisión, además calcula el mayor ahorro en distancias al utilizar arcos. Puede darse casos en los que se tengan dos soluciones posibles para una ruta, de ser así se podrá combinar las dos soluciones y se podrá utilizar la solución que se genere de esta combinación. Este algoritmo puede calcular buenas rutas al inicio, pero hacia el final ya no (Rocha et al. 2011). Es por eso que Laporte, Toth y Vigo generalizaron este problema integrando un parámetro de forma que penaliza la unión de rutas con clientes muy lejanos (Rocha et al. 2011).

Existe otro tipo de algoritmo de ahorro que se denomina ahorro basado en coincidencia, que consiste en la combinación de rutas, siempre que sea factible, mediante la utilización de un grafo para optimizar las distancias de recorrido.

Por otra parte, las heurísticas de inserción se utilizan cuando no se cuenta con un número específico de vehículos. Este método consiste en generar rutas con un subconjunto de clientes, luego se selecciona un cliente que no esté en el subconjunto y lo inserta al modelo recalculando las rutas integrando al nuevo cliente (Rocha et al. 2011).

La heurística de inserción secuencial de Mole y Jameson considera los costos que significarían integrar a otro cliente en la ruta, y también recalcula las distancias de las rutas sin reordenar los nodos que se encuentran dentro de ella (Rocha et al. 2011).

La heurística de inserción paralela desarrollada por Christofides, Mingozzi y Toth, consiste en dos fases, la primera es crear las rutas aplicando el algoritmo de inserción secuencial, para generar rutas compactas manteniendo los clientes iniciales dentro de cada ruta, pero sin dejar de un lado el número total de rutas a generar, por último, se crean las rutas y se añaden los clientes que faltan en ellas (Rocha et al. 2011).

Como afirman Rocha et al. (2011): "En los métodos de dos fases se encuentran los métodos de asignación elemental, el algoritmo de ramificación y acotamiento truncados, el algoritmo de los pétalos, el método de rutear primero y asignar después y los procedimientos de búsqueda local" (p. 46).

Dentro de los métodos de asignación elemental se encuentran el algoritmo de barrido, el algoritmo basado en asignación generalizada y la heurística basada en localización. El algoritmo de barrido consiste en girar una semirrecta para definir agrupamientos de clientes hasta que la capacidad del vehículo sea violada. En algunos casos este proceso requiere una fase de optimización posterior. El algoritmo basado en asignación generalizada no utiliza un método geométrico para establecer los bloques de clientes, sino formula un problema de asignación generalizada, y consta de dos fases, la primera es definir los vértices semilla para construir los agrupamientos, luego se asignan los vértices a cada agrupamiento sin exceder la capacidad de los vehículos (Rocha et al. 2011). La heurística basada en localización establece rutas iniciales y luego se añaden vértices asignados a cada ruta en una segunda etapa. La inserción se realiza en cada paso para las rutas iniciales, teniendo en cuenta que se debe tener el menor costo de inserción (Rocha et al. 2011).

La ramificación y acotamiento truncados es un método propuesto por Christofides, Mingozzi y Toth en 1979, que tiene la forma de un árbol en donde cada rama representa un conjunto de rutas, se implementan niveles que se pueden ir descartando si la ruta no es la mejor, aunque este proceso podría ser más sencillo si se mantienen pocas rutas en cada nivel (Rocha et al. 2011).

El algoritmo de los pétalos realizado por Ryan, Hjorring y Glover en 1993, consiste en establecer un conjunto de rutas, de tal manera que un cliente puede ser visitado varias veces,

pero por diferentes rutas, y se procede a seleccionar un subconjunto de rutas que pase una sola vez por cada cliente (Rocha et al. 2011).

Los métodos de ruteo primero y asignación después modelados por Beasley en 1983, consisten en establecer una ruta que pase por todos los nodos sin tomar en cuenta ninguna restricción y luego a partir de esta ruta gigante se forman soluciones factibles, haciendo subconjuntos de nodos tomando en cuenta la capacidad del vehículo (Rocha et al. 2011).

Los procedimientos de búsqueda local se basan en establecer soluciones vecinas a parte de la solución primaria del problema, luego se evalúa que ruta genera menor costo y si es necesario se la reemplaza, este proceso se repite hasta que el sistema no se pueda mejorar más (Rocha et al. 2011).

2.2.5.3. Meta heurísticas

Osman y Kelly, 1996 (como se citó en Vélez y Montoya, 2007) afirman:

> Los procedimientos meta heurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos clásicos no son ni efectivos ni eficientes. Las metas heurísticas proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de la inteligencia artificial, la evolución biológica y la mecánica estadística. (p. 4)

Dentro de las meta heurísticas se encuentran el reconocido simulado, redes neuronales, búsqueda Tabú, algoritmos genéticos, algoritmos de hormiga y búsqueda de vecindades (Rocha et al. 2011).

Los algoritmos genéticos inspirados en la teoría de la evolución darwiniana, desarrollados por Holland en 1975, operan de la siguiente manera para dar solución a un problema VRP, primero se generan soluciones iniciales que representan cada viaje, luego para cruzar dos soluciones, se toma en cuenta una subruta que no necesariamente debe iniciar y terminar en el depósito, si la ruta no fuera factible se particiona, de esta manera se genera una descendiente es decir una copia modificada de las soluciones iniciales (Rocha et al. 2011).

Los algoritmos de hormigas realizados por Colorni, Dorigo y Maniezzo en 1991, están inspirados en la estrategia que utilizan las colonias de hormigas para la recolección de alimentos. Cuando una hormiga encuentra una fuente de alimento desprende una feromona que

depende de la longitud del camino y la calidad de la comida. Para la solución de los problemas VRP la aplicación sería la siguiente; primero se coloca una hormiga en cada nodo, luego se utiliza un método probabilístico que asigna una probabilidad igual a cero si el nodo ya ha sido visitado y un valor diferente de cero para el caso contrario. La hormiga visita los nodos con mayor probabilidad. Este proceso finaliza si se obtiene un valor menor a una cota preestablecida, de no darse este caso se recalcula y la hormiga sigue construyendo soluciones (Rocha et al. 2011).

La búsqueda Tabú efectuada por Glover en 1986, consiste en realizar una búsqueda local aceptando soluciones que mejoran en términos del costo la solución, de tal manera que en cada iteración la solución se mueve a otra mejor dentro de un subconjunto de soluciones cercanas. Como esta solución nueva no necesariamente es la mejor, se debe crear una memoria de corto plazo de tal manera que se prohíbe volver a soluciones ya visitadas, estas soluciones prohibidas se llaman soluciones Tabú y las soluciones nuevas que se encuentran se denominan soluciones movidas Tabú (Rocha et al. 2011). Para la solución de problemas VRP el procedimiento es el siguiente:

Se inicia con un algoritmo de búsqueda local que se asemeja al del vecino más cercano en donde se establece una solución inicial seleccionando los nodos más cercanos. Luego se seleccionan dos nodos los cuales son eliminados de la soloucion actual cambiando su posición y por ende la dirección de la ruta. El movimiento que se realizó con los arcos seleccionados se convierte en Tabú, es decir, se bloquea y los nodos ya no pueden cambiar de posición hasta que se haya terminado todas las iteraciones, esto sucede con cada par de nodos que se selecciona para cada iteración. El tamaño de la lista Tabú es un criterio bajo el cual se decide que movimiento se desbloquea después de un cierto número de iteraciones. Para finalizar se establece un número determinado de iteraciones a realizar, si no se ha producido mejoría en la solución actual entonces la respuesta es la misma, si se ha mejorado la solución entonces la respuesta se actualiza al movimiento que ha producido la mejor solución (Rocha et al. 2011).

2.2.6. Distancia entre nodos

Para realizar el cálculo de las distancias entre los nodos utilizando las coordenadas de cada uno, se considera utilizar el método de distancias euclidianas, el cual es similar al teorema de Pitágoras cuya fórmula es: $d = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$, en donde (x, y) representan las coordenadas de latitud y longitud respectivamente de cada punto. El uso de este método obtiene

un resultado que asume conexiones directas entre los puntos sin considerar las irregularidades del terreno ni la estructura esférica de la tierra, por lo tanto, el resultado tiende a ser incongruente en comparación con la distancia real (Anbuudayasankar et al. 2014).

Para tener un cálculo más exacto de la distancia se utiliza la fórmula de Harvesine, que toma en cuenta el radio de la tierra lo cual hace que el resultado coincida con la distancia real, aunque se tiene algo de variación debido a las irregularidades del terreno y a la composición de la vía. Esta fórmula de igual manera asume conexiones directas entre los nodos, pero se considera que esta, es un arco que se calcula con base en el radio de la Tierra, por lo que esta fórmula ha sido utilizada para medir distancias marítimas que iban a recorrer los barcos en sus viajes y está definida de la siguiente manera:

$$d = 2r \arcsin\left(\sqrt{sin^2\left(\frac{\emptyset_2 - \emptyset_1}{2}\right) + \cos(\emptyset_1)\cos(\emptyset_2)\, sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

En donde:

r = radio de la esfera (terrestre).

$\emptyset$ = latitud.

λ = longitud (Mangesh, 2013).

2.2.7. Visual Basic for Apps (VBA)

Según Kenton (2019):

> *Visual Basic* para Aplicaciones (VBA) es parte del software heredado de *Microsoft Corporation* (NASDAQ: MSFT) , *Visual Basic*, que la compañía creó para ayudar a escribir programas para el sistema operativo Windows. *Visual Basic* para aplicaciones se ejecuta como un lenguaje de programación interno en aplicaciones de Microsoft Office (MS Office, Office) como Access, Excel , PowerPoint, Publisher, Word y Visio.
>
> VBA es una herramienta basada en eventos, lo que significa que puede usarla para indicarle a la computadora que inicie una acción o una serie de acciones. Para hacer esto, crea macros personalizadas, abreviatura de macroinstrucciones, escribiendo comandos en un módulo de edición. (párr, 1-3)

VBA va más allá de las funciones que se pueden ingresar en una hoja de cálculo ya que permite personalizar las acciones a ejecutar, y realizar operaciones más complejas y extensas automatizando el proceso de resolución para agilizar el trabajo mediante códigos de programación.

2.2.8. Rutina y subrutina

Una rutina es una tarea específica y repetitiva compuesta por varias líneas de programación que se ejecutan dentro de un programa. Una sub-rutina es una serie de instrucciones dentro de un programa para ejecutar una tarea, que se compone de líneas de programación dentro y que son invocadas desde el programa, una vez ejecutadas regresan al lugar de donde fueron llamadas.

2.2.9. Teoría general de los sistemas

Johansen (1993) afirma:

> La teoría general de sistemas a través del análisis de las totalidades y las interacciones internas de estas y las externas con su medio, es una poderosa e importante herramienta que permite la explicación de los fenómenos que suceden en la realidad y también hace posible la predicción de la conducta futura de esa realidad. (p. 14)

La teoría general de sistemas en la aplicación científica, busca la explicación de la realidad con un enfoque general e interdisciplinario, es decir que la ciencia como tal que ha sido dividida en varias partes para la explicación de fragmentos de la realidad específicos. Debería establecerse principios y conceptos básicos que se puedan asociar y relacionar con las demás ciencias realizando un estudio desde una perspectiva generalista integrando todos los conocimientos que se pueda obtener de las diferentes disciplinas.

El módulo didáctico a desarrollarse busca la integración de los conocimientos obtenidos en el aula de clase por los docentes de cada asignatura, buscando una relación coherente que permita consolidarlos y ordenarlos de tal manera que faciliten su utilización y entendimiento. Es necesario resaltar los aspectos más importantes que puedan contribuir a la realización de este proyecto, con el fin de crear una herramienta que pueda servir como apoyo didáctico para todas las disciplinas estudiadas.

2.2.10. Teoría de optimización combinatoria

La optimización combinatoria ejecuta algoritmos, con ayuda de programas computacionales, para la resolución de problemas en los cuales el número de soluciones viables es casi imposible de enumerar, pero este tipo de algoritmos encuentra la mejor respuesta del conjunto de soluciones posibles (Anbuudayasankar et al. 2018).

2.2.11. Cibernética de segundo orden

La cibernética de segundo orden permite al individuo no solo ser observador de fenómenos sino también participar y generar conocimiento de ellos, por ejemplo, dentro de las actividades académicas, los docentes deben dialogar con sus estudiantes ya que podrá existir una retroalimentación para ambos y de esta manera generar nuevo conocimiento que despierta la necesidad de auto corregirse y auto educarse (Santiago, 2012).

Con relación a este trabajo, el operador del módulo didáctico también forma parte del sistema. Para llegar a obtener el resultado esperado que optimice el problema planteado, es muy importante que tenga claro las pautas que debe seguir para manejar el programa y conseguir de esta manera un conocimiento sólido con respecto al uso de la tecnología en beneficio de sí mismo. Se pueden tener varias opciones de respuestas que pueden aportar a la construcción del conocimiento, aunque el programa podría no tener en algún caso la respuesta adecuada a un problema planteado ya que los métodos de solución incorporados pueden ser inadecuados. De esta manera el operario podría modificar el sistema de tal manera que beneficie sus propios fines, además de permitirle relacionar varios tipos de datos para poder satisfacer la necesidad de aprender y obtener una respuesta válida para el problema propuesto.

2.2.12. Teoría de la complejidad computacional

La teoría de la complejidad computacional establece que existen dos tipos de problemas computacionales, P (polinomial) y NP (no polinomial). Esta teoría trata de establecer qué problemas pueden ser resueltos mediante la capacidad computacional existente y cuáles no, de esta manera se reconoce a los problemas de tipo P como los que se pueden resolver mediante un algoritmo y se obtiene una respuesta exacta al problema, es decir, la respuesta óptima, al contrario de los problemas NP que requieren de gran capacidad computacional para su resolución y en realidad aún no se ha podido dar una respuesta concreta (Anbuudayasankar et al. 2014).

Para resolver problemas VRP se toma en cuenta el tamaño de los mismos, ya que este determina la complejidad del modelo, entre más nodos se tenga, mayor será el esfuerzo para resolver dicho problema. Por esta razón los problemas de ruteo vehicular se ubican en los NP (no polinomial), ya que son modelos que pueden ser resueltos en un tiempo polinomial por máquinas no deterministas, pero este tipo de máquinas no existen en la actualidad, aún el equipo computacional más potente podría tardar años en resolver un problema de VRP que quiera encontrar una solución óptima, y como alternativa se consideran las heurísticas y meta heurísticas que pueden obtener una respuesta válida para un problema planteado y además la calcula en un tiempo computacional razonable, aunque en algunos casos requiere de gran capacidad de procesamiento, pero al utilizar un método exacto la máquina podría estar probando todas la alternativas posibles hasta hallar la respuesta óptima del problema.

III. METODOLOGÍA

3.1. ENFOQUE METODOLÓGICO

3.1.1. Enfoque

La presente investigación tiene un enfoque cuantitativo, ya que se espera determinar un módulo que integre varios modelos de solución para problemas VRP, además se tomarán mediciones con respecto al tiempo, capacidad de procesamiento en cada una, y calidad de la respuesta obtenida, deseando establecer el método más idóneo para dar solución a estos problemas.

3.1.2. Tipo de Investigación

Experimental, ya que se considerará el tiempo de procesamiento de cada método de solución y además se compararán los resultados identificando el mejor. Además, es necesario realizar varias pruebas con el código de programación en distintos equipos con distintas características, debido a que este trabajo se orienta a la aplicación práctica del conocimiento y el uso de herramientas tecnológicas para la resolución de problemas.

3.2. HIPÓTESIS

¿Se puede desarrollar un módulo didáctico de solución de problemas de diseño de rutas que integre múltiples métodos de solución?

3.3. DEFINICIÓN Y OPERACIONALIZACIÓN DE VARIABLES

Tabla 1. Operacionalización de variables.

VARIBLE	DIMENSIÓN	SUB-DIMENSIÓN	INDICADORES	ÍTEMS
Módulo	Sistema didáctico para solución de problemas de VRP.	Modularización, Líneas de instrucción, rutinas, sub-rutinas, complejidad del código.	Se desarrollará el módulo variando la complejidad del código de programación y se probará el funcionamiento del módulo en ordenadores con distintas características de hardware, software, procesador, memoria.	¿En cuántos ordenadores se probó el funcionamiento del módulo? 1. Ninguno 2. Uno 3. Dos 4. Tres

VARIBLE	DIMENSIÓN	SUB-DIMENSIÓN	INDICADORES	ÍTEMS
				5. Más de 3
				Tiempo de ejecución del programa/tipo de ordenador: 1. Menor a un s (Óptimo) 3. Entre 1 y 10 s (Aceptable) 4. Entre 10 s y un minuto (Regular) 5. Más de un minuto (Ineficiente)
Nivel de integración	Mecanismos de solución del problema que funcionando de forma conjunta sobre los mismos datos de entrada producen la solución más adecuada.	Métodos exactos, heurísticas, metaheurísticas.	Se programarán diferentes métodos y se compararán las variables de VRP permitidas, las soluciones obtenidas en tiempo de cómputo y parámetros VRP (distancia, costo y tiempo de viaje) para establecer el método más adecuado.	Número de métodos exactos Número de heurísticas Número de metaheurísticas Si se tiene al menos un método de solución en cada enfoque entonces nivel de integración óptimo para fines didácticos. Número de restricciones permitidas por método, al menos cada método debería permitir tener velocidad promedio de circulación, costo fijo, costos variables, posición del depósito, número de nodos (5 nodos mínimo).

VARIBLE	DIMENSIÓN	SUB-DIMENSIÓN	INDICADORES	ÍTEMS
				Solución VRP/método: ¿Cuál fue la Distancia recorrida? ¿Cuál fue el costo total de la ruta? ¿Cuál fue el tiempo total de la ruta a la velocidad establecida? ¿Cuál fue el mejor método en tiempo de cómputo? ¿Cuál fue el mejor método en parámetros VRP?

3.4. MÉTODOS UTILIZADOS

3.4.1. Análisis Estadístico

Se realizará un análisis estadístico simple para determinar la eficiencia de los distintos métodos de solución con base en tiempos promedio de procesamiento para cada método, uso promedio de recursos computacionales y estimación de los errores cometidos en las soluciones encontradas respecto a las soluciones óptimas de los problemas.

IV. RESULTADOS Y DISCUSIÓN

4.1. RESULTADOS

4.1.1. Parámetros de dimensionamiento del modelo matemático

Para el planteamiento de un problema de ruteo vehicular se necesita examinar el proceso distributivo real del caso que se está analizando, para extraer todas las variables que pueda considerarse dentro de su resolución, o establecer las más relevantes, ya que considerar todas las variables genera mucha complejidad en el modelo y hay otras que no se puede definir con valores exactos porque son variables aleatorias que están fuera del control de los operarios. El proceso de distribución de una empresa es uno de los más costosos, ya que se considera flota vehicular, operarios, capacidad vehicular, número de nodos a visitar, distancia a recorrer, combustible necesario, tiempo de llegada a cada nodo, ventanas de tiempo, entre otros que podrían definirse por el encargado de realizar la planificación VRP y cada uno de estos aspectos representa un costo para la operación del transporte y distribución de productos. El proceso más complejo puede ser, pasar el problema identificado con sus restricciones a un lenguaje matemático, es decir, establecer la función objetivo que simule el sistema real y restringir las operaciones de acuerdo a las características reales por las cuales la distribución está limitada. Para realizar esta actividad se requiere de un gran conocimiento matemático, además de un pensamiento analítico que permita construir dichas ecuaciones para llegar a la solución del problema. Para la realización de este trabajo se ha tomado en cuenta el problema VRP puro el cual solo optimiza la distancia recorrida, es decir, busca visitar todos los nodos con la menor distancia posible.

El modelo matemático del problema VRP está representado por la siguiente ecuación:

$$(1)\ \sum_{i=1}^{n}\sum_{j=1}^{n}\sum_{k=1}^{NV} d_{ij}x_{ij}^{k}$$

Sujeto a las siguientes restricciones:

$$(2)\ \sum_{i=1}^{n}\sum_{k=1}^{NV} x_{ij}^{k} = 1 \quad (j = 2, \dots\dots, n)$$

$$(3)\ \sum_{j=1}^{n}\sum_{k=1}^{NV} x_{ij}^{k} = 1 \quad (i = 2, \dots\dots, n)$$

$$(4)\ \sum_{i=1}^{n} x_{ip}^{k} - \sum_{j=1}^{n} x_{pj}^{k} = 0 \quad (k = 1, \dots\dots, nv) \quad (p = 1, \dots\dots, n)$$

$$(5)\ \sum_{i=1}^{n} Q_i \left(\sum_{j=1}^{n} x_{ij}^{k} \leq p_k \quad (k = 1, \dots\dots, NV\right)$$

$$(6)\ \sum_{i=1}^{n} t_i^{k} \sum_{j=1}^{n} x_{ij}^{k} + \sum_{i=1}^{n}\sum_{j=1}^{n} t_{ij}^{k}\, x_{ij}^{k} \leq T_k \quad (k = 1, \dots\dots, NV)$$

$$(7)\ \sum_{j=2}^{n} x_{ij}^{k} \leq 1 \quad (k = 1, \ldots\ldots, NV)$$

$$(8)\ \sum_{i=2}^{n} x_{ij}^{k} \leq 1 \quad (k = 1, \ldots\ldots, NV)$$

$$(9)\ x_{ij}^{k} = 0\ o\ 1 \quad para\ todo\ i, j, k$$

$$(10)\quad x \in S$$

Donde:

El nodo 1 es el depósito.

n = número de nodos.

NV = número de vehículos.

P_k = capacidad del vehículo k.

T_k = tiempo máximo permitido para una ruta del vehículo k.

Q_i = demanda del nodo i (Q_1= 0).

t_i^k = tiempo requerido por el vehículo k para entregar o recoger en el nodo i (t_i^k = 0).

t_{ij}^k = tiempo de viaje para el vehículo k desde el nodo i hasta el nodo j.

d_{ij} = menor distancia desde el nodo i hasta el nodo j.

$x_{ij}^k = \begin{Bmatrix} 1 \\ 0 \end{Bmatrix}$ si el arco (i,j) ya ha sido atravesado por el vehículo k es igual a 1, caso contrario es 0.

x = matriz con los componentes $x_{ij} \equiv \sum_{k=1}^{NV} x_{ij}^k$, especificando las conexiones independientemente del tipo de vehículo.

La ecuación (1) establece que la distancia total debe ser minimizada. Las ecuaciones (2) y (3) aseguran que cada nodo es visitado solo por un vehículo. El conjunto de ecuaciones (4) representa la continuidad de la ruta. El conjunto de ecuaciones (5) son las limitaciones de capacidad de los vehículos. El conjunto de ecuaciones (6) son las limitaciones de tiempo. Las ecuaciones (7) y (8) aseguran que no se exceda el número de vehículos disponibles. (Pathumnakul, 1996, p. 5-7)

Las ecuaciones de capacidad de flota vehicular y tiempo han sido añadidas para simular un sistema más real, ya que para el problema de encontrar la distancia mínima solo es necesario

trabajar con la función objetivo correspondiente a la ecuación (1) y las restricciones (2), (3) y (4) para asegurar que los nodos sean visitados una sola vez y también que la ruta sea continua.

Para la realización del módulo didáctico se han establecido algunas variables similares que se utilizan en los diagramas de flujo de cada algoritmo, teniendo en cuenta la misma función objetivo del problema VRP antes expuesto, es decir, solo se quiere optimizar la distancia recorrida.

Las variables definidas para la realización de los diagramas de flujo son:

Número de depósitos (Nd), se toma en consideración para el desarrollo del módulo didáctico la presencia de un depósito dentro del problema de ruteo.

Número de nodos (N), representa la cantidad total de nodos involucrados en el problema de ruteo (N=10).

Número de vehículos (N_v), proporciona la cantidad total de vehículos disponibles para la solución del problema, dentro de este parámetro se deben considerar dos aspectos: cantidad fija o variable.

Capacidad de los vehículos (x), se debe determinar la capacidad de carga de los vehículos ya sea en unidades o peso, pero especificando solo uno de estos. Dentro del módulo se considera una capacidad suficiente para realizar el recorrido por todos los nodos con un solo vehículo.

Distancia (d_{ij}), describe la separación que existe entre dos nodos.

Tiempo de viaje (T_v), es el tiempo total requerido para realizar la visita a todos los nodos del problema.

Origen (i), establece el nodo inicial de una iteración.

Destino (j), representa el nodo final de una iteración.

Conjunto de nodos visitados (C), contiene a todos los nodos que ya se han visitado mediante la aplicación del algoritmo de resolución.

Conjunto de nodos no visitados (Ĉ), abarca a los nodos que aún no han sido visitados.

Conjunto (S) de nodos conectados al nodo i, contiene a los nodos que están conectados directamente con el nodo inicial de una iteración.

Permutaciones (p), es el número de opciones que se generan a partir de una ruta inicial.

Ahorro (S), representa el valor del ahorro entre un par de nodos obtenido con la fórmula: $S = d_i + d_j - d_{ij}$, que se utiliza solo para el algoritmo de Clarke and Wright. La explicación de la formula se realizará en el punto: "Diagrama de flujo heurística de Clarke y Wright"

Ruta (r), es la ruta parcial generada por la ejecución de un algoritmo.

Nota: Las variables usadas dentro de los diagramas de flujo no cuentan con subíndices, ya que el programa Visio en donde fueron desarrollados no cuentan con la opción de añadirlos. Por ejemplo si en un diagrama de flujo se encuentra la variable dij esta es igual a d_{ij}.

4.1.2. Programación de métodos de solución y heurísticas del módulo

4.1.2.1. Heurística del Vecino más Cercano, proceso analítico

El proceso de funcionamiento de esta heurística es:

1. Definir el nodo inicial (i).
2. Establecer dos conjuntos que representen los nodos visitados (C) y los que aún no se han visitado (Ĉ).
3. Pasar el nodo de origen a C.
4. Encontrar el menor arco que conecte al nodo i con algún nodo de destino (j).
5. Si todos los nodos han sido visitados, finaliza el algoritmo, caso contrario establecer el nuevo nodo i y repetir el paso 4.

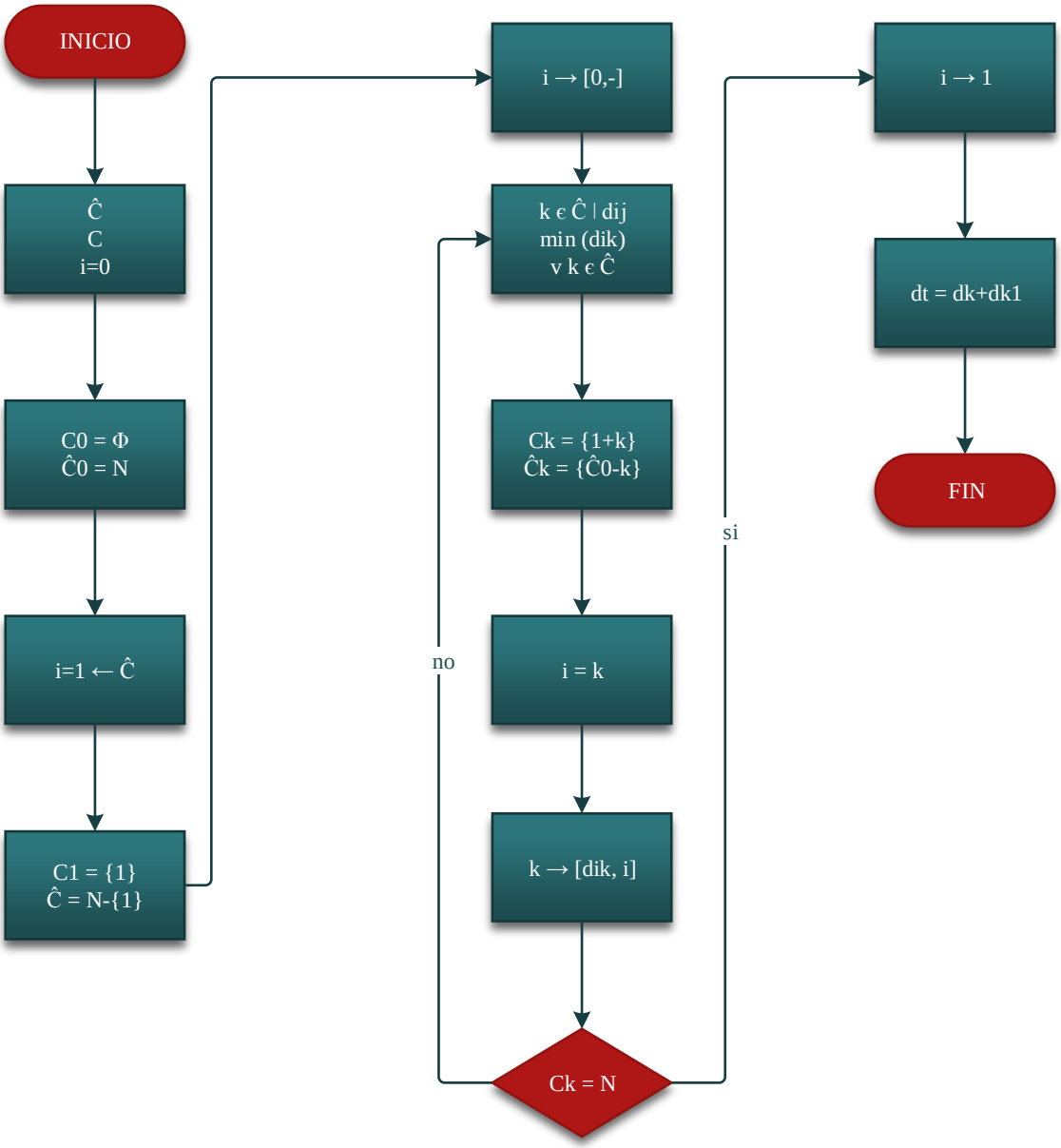


Figura 3. Diagrama de flujo heurística del vecino más cercano, proceso analítico.

4.1.2.1.1. Código de programación en VBA

Para iniciar con la programación del algoritmo es necesario definir algunas variables con las que se podrá establecer los parámetros de funcionamiento del modelo, luego se elige la hoja de trabajo donde se encuentra la matriz de distancias sobre la cual se ejecutará la heurística. Se debe tener en cuenta la posición de la matriz de distancias y el nodo considerado como depósito para definir las celdas de donde se tomará cada valor, además de establecer la condición de buscar el mínimo valor de la distancia de un nodo a otro. Se deben definir las celdas donde se imprimirán los valores de la ruta y distancias resultantes de cada iteración. Adicionalmente se

puede establecer un costo por la unidad de distancia y calcular el costo de las distancias parciales y también la distancia total de la ruta que se haya encontrado.

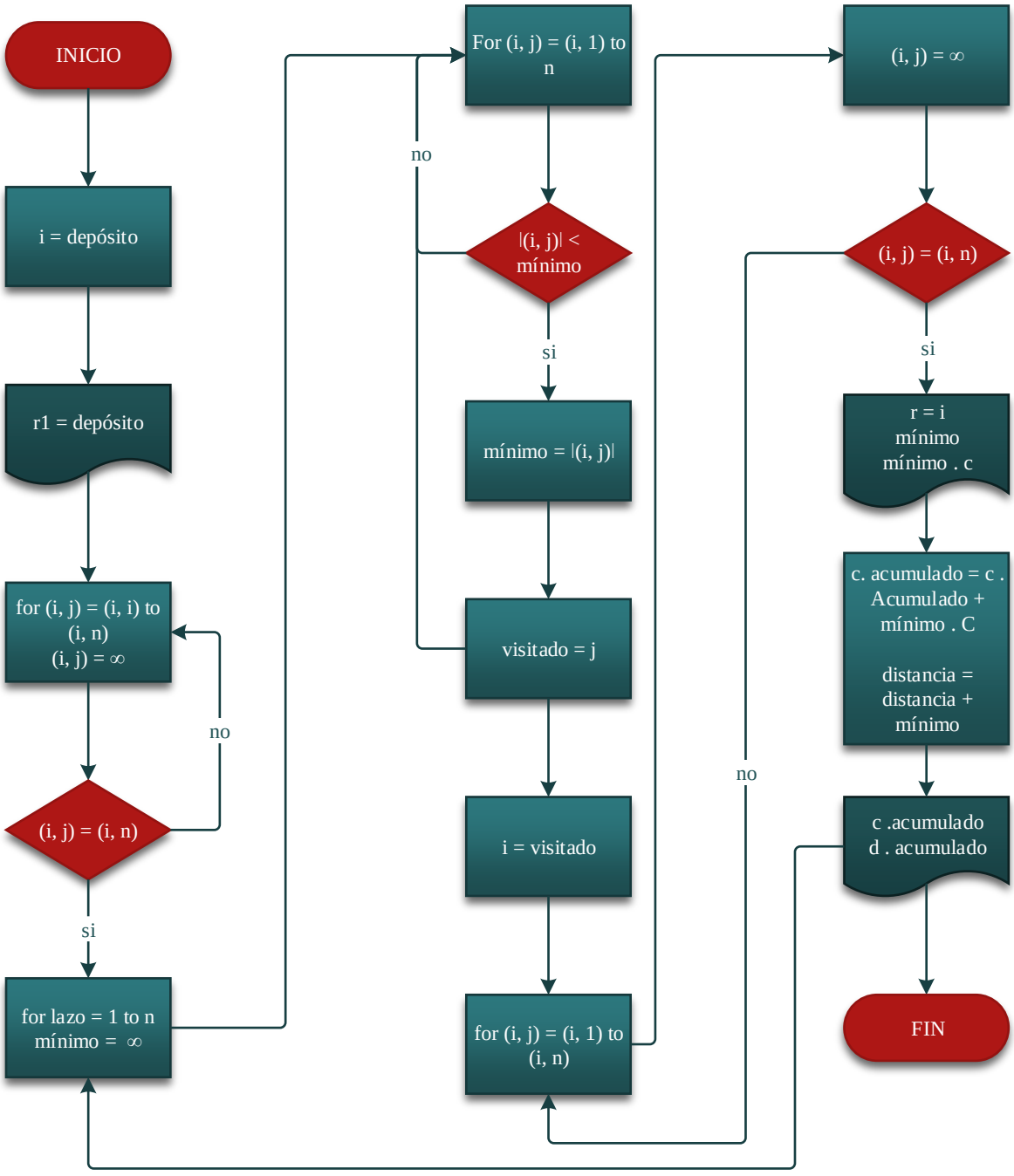


Figura 4. Diagrama de flujo heurística del vecino más cercano, pseudo código.

Sub vecino_mas_cercano()

Dim i As Integer

```
Dim j As Integer
Dim minimo As Double
Dim lazo As Integer
Dim visitado As Integer
Dim fila As Integer
Set h = Sheets("VRP")
```

Establecimiento de la hoja de trabajo y la celda que contiene el depósito.

```
i = h.Cells(16, 2) + 2
For fila = 3 To 12
```

Se considera un valor muy alto de los nodos hacia el depósito para que no sea visitado.

```
h.Cells(fila, i) = 10000
Next fila
For lazo = 1 To 9
```

Se prueba el valor de la distancia entre cada i y cada j y de la misma manera de cada j a cada i. El mínimo es un valor muy alto que se optimizará.

```
minimo = 1000
For j = 3 To 12
```

Se actualiza el valor mínimo.

```
If h.Cells(i, j) < minimo Then
minimo = h.Cells(i, j)
visitado = j
End If
Next j
i = visitado
For fila = 3 To 12
```

Las celdas de los nodos visitados se llenan con un valor muy alto.

```
h.Cells(fila, visitado) = 10000
```

```
Next fila

h.Cells(17, 2 + lazo) = visitado - 2

h.Cells(18, 2 + lazo) = minimo

h.Cells(19, 2 + lazo) = minimo * 0.75

Next lazo

h.Cells(17, 2) = h.Cells(16, 2)

h.Cells(18, 2) = 0

h.Cells(19, 2) = 0

Range("L18").Formula = "=SUM(B18:K18)"

Range("L19").Formula = "=SUM(B19:K19)"

For fila = 3 To 12

For lazo = 3 To 12

h.Cells(lazo, fila) = h.Cells(lazo, fila + 12)

Next lazo

Next fila

End Sub
```

Se imprime el nodo visitado, la distancia parcial entre cada nodo y el costo.

Suma de distancias y costos.

Se reestablecen los valores en la matriz de distancias.

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	INICIO										FIN	TOTAL	SEGUNDOS
2	1	6	5	4	3	8	7	9	2	10	1		0,0625
3	DISTANCIA P	25,5114586	58,7647274	38,3635798	20,9523735	173,838015	49,7124805	50,0967628	36,7698788	26,7309809	155,330472	636,070729	
4	COSTO PARC	7,65343759	17,6294182	11,5090739	6,28571204	52,1514044	14,9137442	15,0290288	11,0309636	8,01929427	46,5991415	190,821219	
5	COSTO TOTA	290,821219											

Figura 5. Resultado de la ejecución, heurística del vecino más cercano (Módulo didáctico).

4.1.2.2. Método exacto *brutal search,* proceso analítico

Brutal search (búsqueda por fuerza bruta), es un método exacto que establece y prueba todos los ciclos o permutaciones posibles que puedan darse en un conjunto de nodos obteniendo respuestas parciales que se guardan y se comparan con las demás respuestas obtenidas para conseguir la solución óptima del problema.

A partir de la solución encontrada con la heurística del vecino más cercano se establece el siguiente proceso:

1. Calcular el número de ciclos o permutaciones a generarse. El número de permutaciones se calcula con la siguiente fórmula:

$$P = (N - 1)!$$

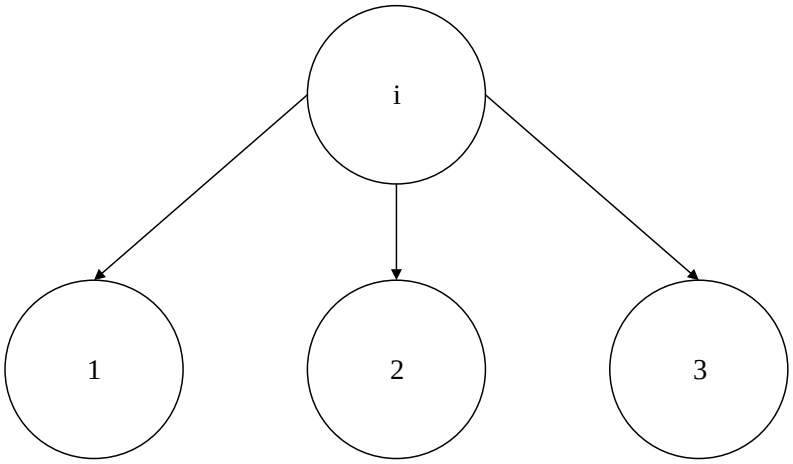


Figura 6. Nodos para la demostración de la fórmula para el cálculo de permutaciones.

El número de nodos está representado por la letra N y este indica también el número de permutaciones por ciclo. Cuando ya se define el nodo inicial entonces el número de opciones que se tienen para conectar el siguiente nodo son $(N - 1)$, de la misma forma si se conecta el nodo inicial con el primer nodo entonces se resta un nodo más $(N - 2)$, y así sucesivamente hasta terminar con todos los nodos y formar todas las permutaciones. De este análisis se puede obtener la ecuación resultante:

$$(N - 1)(N - 2)(N - 3) \ldots .1 = (N - 1)!$$

De tal manera que el uno representa al depósito, que es un nodo fijo y se realiza las permutaciones con los demás nodos.

Por ejemplo:

$$N = 10$$

$$P = (10 - 1)!$$

$$P = 362\,880$$

El número obtenido representa las opciones que se pueden generar teniendo en cuenta nueve nodos, ya que el depósito no se lo considera porque es el mismo para todas las opciones.

2. Definir la ruta inicial a partir de la heurística del vecino más cercano.
3. Generar todas las permutaciones.
4. Calcular las nuevas distancias para cada permutación.
5. Si se encuentra una distancia menor, entonces la nueva solución es la ruta de la permutación que genera la menor distancia.

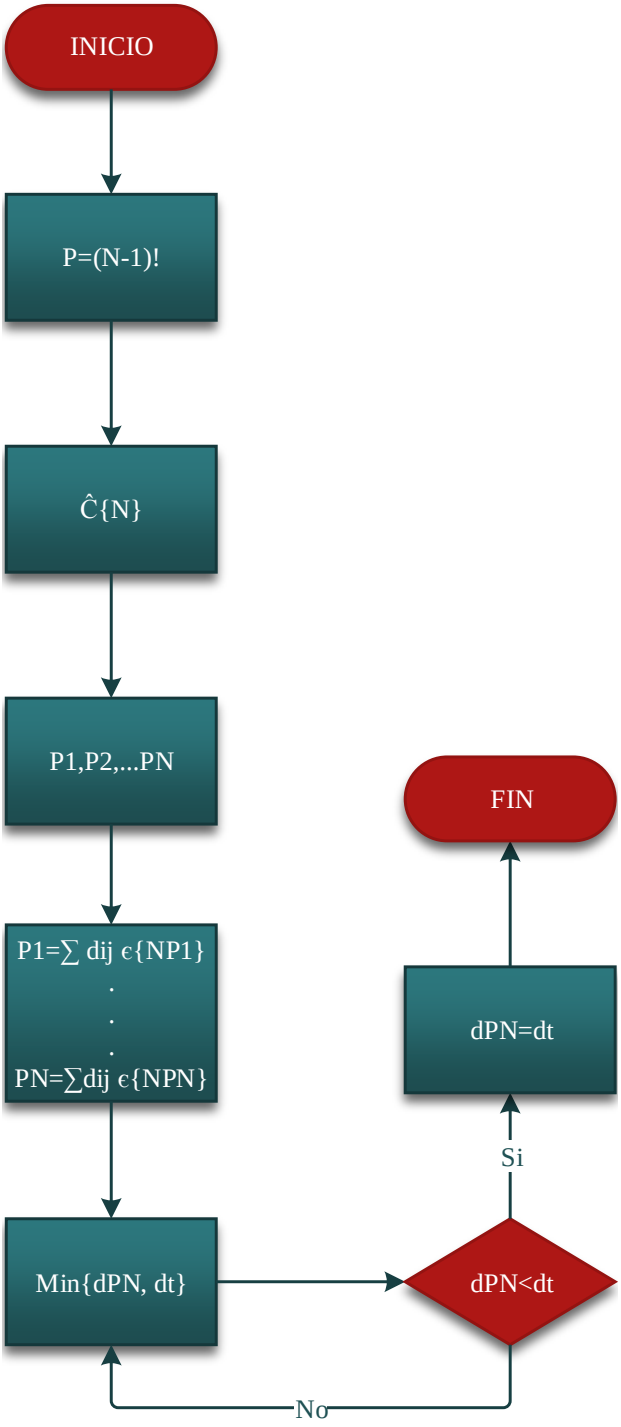


Figura 7. Diagrama de flujo búsqueda por fuerza bruta, proceso analítico.

4.1.2.2.1. Cálculo del número de permutaciones en VBA

Para el cálculo del número de permutaciones se define la función factorial en donde se aplica la fórmula antes descrita en los pasos del algoritmo y luego se establece la celda que contendrá el valor total de permutaciones.

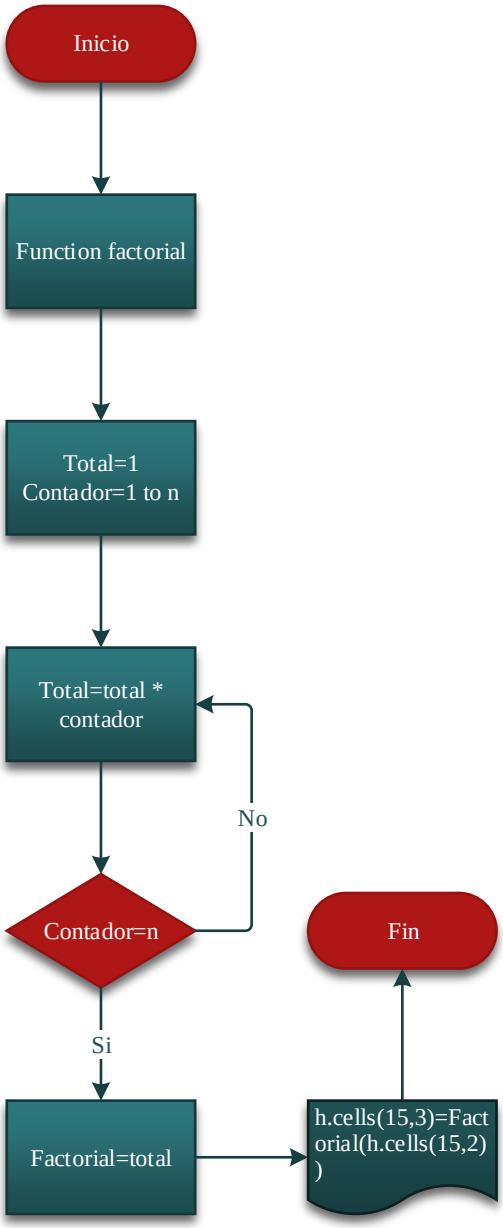


Figura 8. Cálculo del número de permutaciones, pseudo código.

```
Function Factorial(numero)
    Dim total, contador As Integer
    total = 1
    For contador = 1 To numero
        total = total * contador
    Next
```

Cálculo del factorial con la fórmula k*k+1

```
    Factorial = total
End Function
Sub permutaciones#()
Set h = Sheets("VRP")
h.Cells(15, 2) = h.Cells(12, 2)
h.Cells(15, 3) = Factorial(h.Cells(15, 2))
End Sub
```

Impresión en la celda definida

4.1.2.2.2. Permutaciones en VBA

Para generar todas las permutaciones posibles se utiliza el algoritmo empuja e inserta creado por eloycaballero (2018), el cual toma el último nodo y lo inserta al inicio de la ruta recorriendo los demás, es decir, que el último nodo que se visita en la ruta inicial, es el primero en la nueva ruta generada repitiendo este proceso hasta generar el número total de permutaciones definido en el paso anterior.

```
Option Explicit
Sub R_Permutacion()
Worksheets("Hoja1").Range("$A$2:$I$1048576").Clear
'Dim tInicio As Double, dUracion As Double
'tInicio = Timer
Dim DatString() As Variant
Dim ene As Long
ene = Worksheets("Hoja1").Range("a1").CurrentRegion.Count
If ene > 9 Then
    MsgBox "Solo hasta 9 elementos"
```

Limpieza de la zona de escritura.

Control de tiempo.

Cadena a permutar.

Número de elementos a permutar.

Se tolera máximo nueve elementos.

```
    Exit Sub
End If
If ene <= 1 Then
    MsgBox "Mínimo dos elementos"
    Exit Sub
End If
ReDim DatString(1 To ene)
Dim i As Long, j As Long, k As Long, m As Long, n As Long
For i = 1 To ene
    DatString(i) = Worksheets("Hoja1").Cells(1, i).Value
Next i
Dim ciclos As Long
ciclos = Fkii(ene)
Dim FilCiclo, ColCiclo As Long
Dim NewElment As Variant
Dim MPer() As Variant, MPas() As Variant
ReDim MPer(1 To ciclos, 1 To ene)
ReDim MPas(1 To ciclos, 1 To ene)
Dim SubNiv As Long
MPer(1, 1) = DatString(2)
MPer(1, 2) = DatString(1)
```

Se tolera mínimo dos elementos.

Carga de elementos a permutar.

Filas y columnas de cada ciclo.

Nuevo elemento a empujar.

Matriz de resultados y de paso.

Subnivel de modulación de la matriz de paso.

Se carga los dos primeros niveles de permutación.

```
MPer(2, 1) = DatString(1)
MPer(2, 2) = DatString(2)
For i = 3 To ene
   FilCiclo = Fkii(i)
   ColCiclo = i
   NewElment = DatString(i)
   For j = 1 To FilCiclo
      SubNiv = 1 + ((j - 1) \ i)
         For k = 1 To ColCiclo
            MPas(j, k) = MPer(SubNiv, k)
         Next k
   Next j
For m = 1 To ciclos
For n = 1 To ene
MPer(m, n) = MPas(m, n)
Next n
Next m
Dim jmod As Long
   For j = 1 To FilCiclo
   jmod = j Mod i
   If jmod = 0 Then
```

Se hará tantos ciclos como elementos a permutar.

Se crea la matriz de paso para cada fila del ciclo.

Se alimenta la matriz de permutaciones con la matriz de paso.

Para cada fila del ciclo se hace el empuje e inserción.

```
    jmod = i
    End If
        For k = ColCiclo To jmod + 1 Step -1
            MPer(j, k) = MPer(j, k - 1)
        Next k
        MPer(j, jmod) = NewElment
    Next j
Next i
For m = 1 To ciclos
    For n = 1 To ene
        Worksheets("Hoja1").Cells(m + 1, n) = MPer(m, n)
    Next n
Next m
'dUracion = Timer - tInicio
'Debug.Print dUracion
MsgBox ciclos y " Permutaciones generadas"
End Sub
```

Se empuja desde la última para atrás.

Se introduce el nuevo elemento en el espacio generado al empujar.

Impresión de resultados.

Function Fkii(arg As Long) As Long

4.1.2.2.3 Cálculo de distancias en VBA

	A	B	C	D	E	F	G	H	I	J
1										
2		6	5	4	3	8	7	9	2	10
3		10	2	9	7	8	3	4	5	6
4		2	10	9	7	8	3	4	5	6
5		2	9	10	7	8	3	4	5	6
6		2	9	7	10	8	3	4	5	6
7		2	9	7	8	10	3	4	5	6
8		2	9	7	8	3	10	4	5	6
9		2	9	7	8	3	4	10	5	6
10		2	9	7	8	3	4	5	10	6
11		2	9	7	8	3	4	5	6	10
12		10	9	2	7	8	3	4	5	6
13		9	10	2	7	8	3	4	5	6
14		9	2	10	7	8	3	4	5	6
15		9	2	7	10	8	3	4	5	6
16		9	2	7	8	10	3	4	5	6
17		9	2	7	8	3	10	4	5	6
18		9	2	7	8	3	4	10	5	6
19		9	2	7	8	3	4	5	10	6
20		9	2	7	8	3	4	5	6	10
21		10	9	7	2	8	3	4	5	6
22		9	10	7	2	8	3	4	5	6
23		9	7	10	2	8	3	4	5	6

Figura 9. Ejecución del algoritmo empuja e inserta y generación de permutaciones (Módulo didáctico).

Para el cálculo de las nuevas distancias se toma en cuenta la matriz de distancias original y se imprimen los resultados para su comparación que definirá el resultado óptimo del problema.

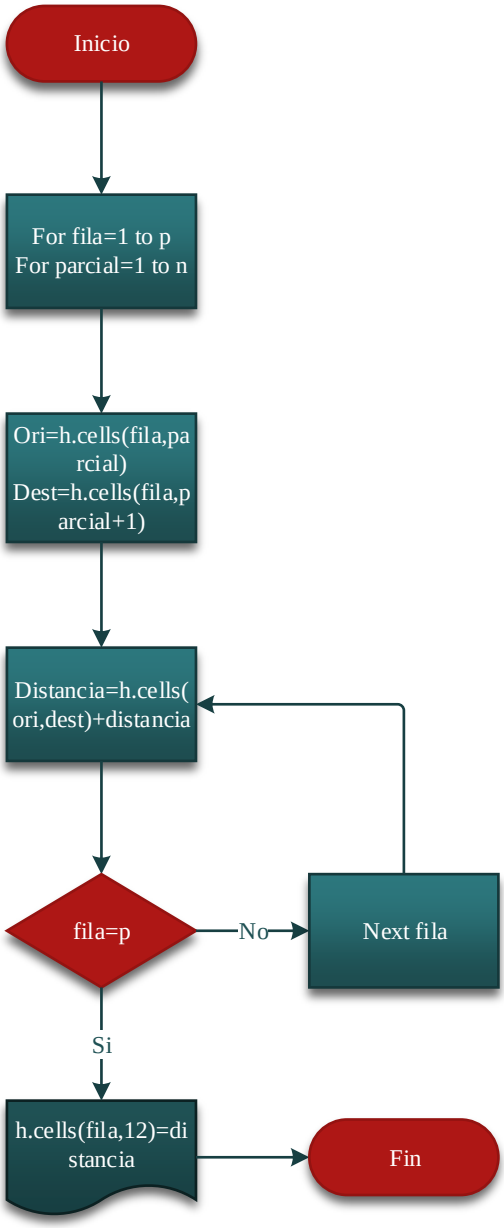


Figura 10. Cálculo de distancias en VBA, pseudo código.

```
Sub ciclos()

Dim fila As Integer

Dim parcial As Integer

Dim ori As Integer

Dim dest As Integer

Dim distancia As Integer
```

```
For fila = 1 To 30000
distancia = 0
For parcial = 1 To 10
ori = Worksheets("Hoja1").Cells(fila, parcial)
dest = Worksheets("Hoja1").Cells(fila, parcial + 1)
distancia = Worksheets("VRP").Cells(ori + 2, dest + 2) + distancia
Next parcial
Worksheets("Hoja1").Cells(fila, 12) = distancia
Next fila
End Sub
```

Búsqueda de valores en la matriz de distancias.

Sumatoria de distancias.

Impresión de la sumatoria.

4.1.2.3. Diagrama de flujo heurística de Clarke y Wright

1. Establecer una solución inicial.
2. Establecer rutas de ida y vuelta desde el origen hacia cada nodo.
3. Calcular los ahorros entre cada par de nodos con la fórmula: $d_1 + d_2 - d_{12}$

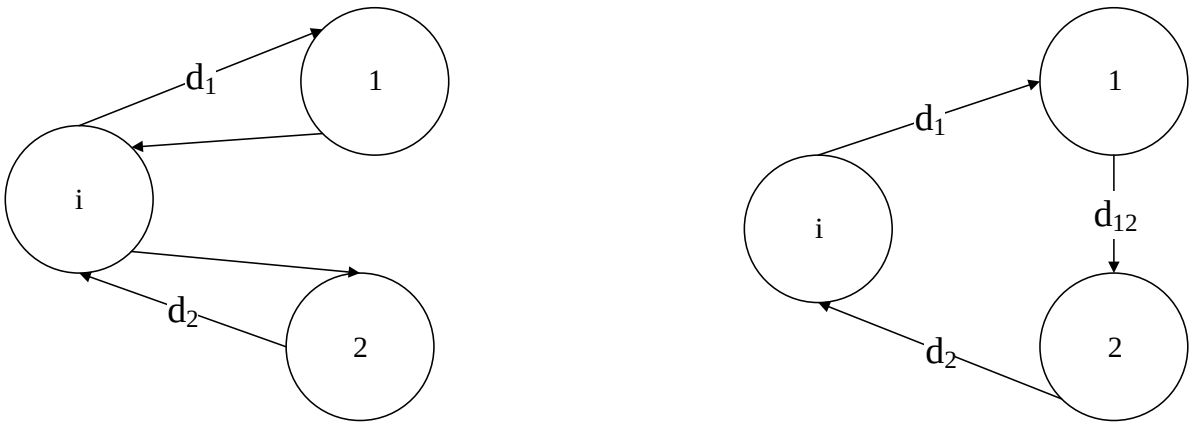


Figura 11. Gráfico ilustrativo del cálculo del ahorro entre un par de nodos.

Para obtener la fórmula para el cálculo del ahorro que produce la combinación de ambas rutas se resta la solución uno a la solución dos:

$$2d_1 + 2d_2 - d_1 - d_2 - d_{12}$$

$$d_1 + d_2 - d_{12} > 0$$

Si se cumple eso entonces hay ahorro y se reemplaza la ruta. El nodo i representa el depósito, el uno y el dos son el primer y segundo nodos respectivamente seleccionados para calcular el ahorro. Las distancias siempre se consideran entre el depósito y el nodo seleccionado, por ejemplo, la d_{12} es la distancia desde el deposito hasta el nodo uno, mientras que la d_{12} que es la distancia entre los nodos que se especifican.

4. Determinar el mayor ahorro entre un par de nodos y establecer una ruta parcial.
5. Evaluar la optimización de distancia de la ruta encontrada.
6. Definir la ruta como parte de la solución final.
7. Buscar el siguiente mayor ahorro entre un par de nodos, repetir el paso 4.

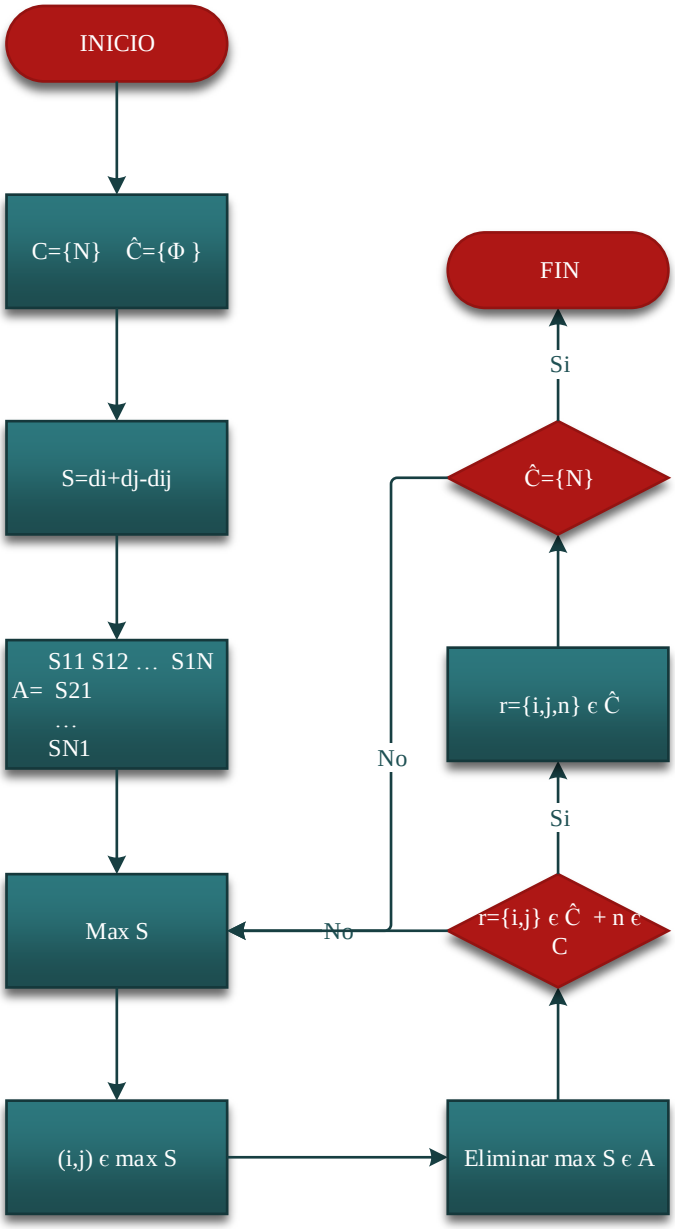


Figura 12. Diagrama de flujo heurística de Clarke y Wright, proceso analítico.

4.1.2.3.1. Matriz de ahorros en VBA

Para el cálculo de matriz de ahorros se considera la siguiente fórmula: $S = d_i + d_j - d_{ij}$.

Para generalizar lo mostrado en la figura 11 se considera que la letra i y j son los nodos seleccionados para el cálculo del ahorro.

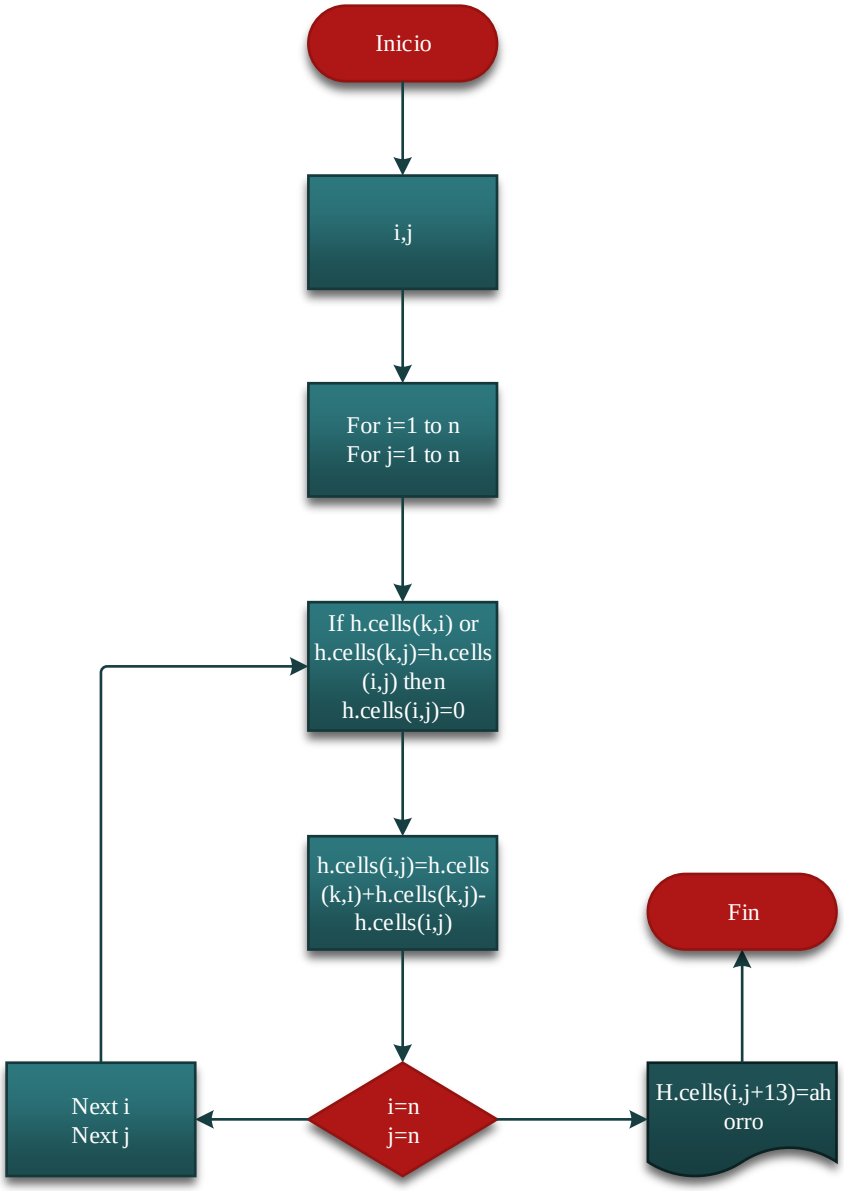


Figura 13. Matriz de ahorros en VBA, pseudo código.

```
Sub ahorros()

Dim i As Integer

Dim j As Integer

Set h = Sheets("Hoja1")

For i = 3 To 12

For j = 3 To 12
```

Hacer cero los valores de ahorros de cada nodo con el depósito.

```
If i = j Or h.Cells(3, j) = h.Cells(i, j) Or h.Cells(3, i) = h.Cells(i, j) Then

h.Cells(i, j + 13) = 0

Else

h.Cells(i, j + 13) = h.Cells(3, i) + h.Cells(3, j) - h.Cells(i, j)

End If

Next j

Next i

End Sub
```

Cálculo del ahorro con la fórmula di+dj-dij.

NODOS	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	2,09537971	4,57116946	5,14350864	0,12097151	137,01313	83,1107622	214,062951	271,834685
3	0	2,09537971	0	225,769498	153,763812	49,0757624	7,77865556	0,42500583	0,90247568	0,27461766
4	0	4,57116946	225,769498	0	157,068162	47,3537942	11,1337677	0,01539802	2,59381547	1,57717969
5	0	5,14350864	153,763812	157,068162	0	45,6043942	11,4427249	0,02516573	3,24047617	2,20871097
6	0	0,12097151	49,0757624	47,3537942	45,6043942	0	0,48613525	2,12494787	0,32874977	0,77069982
7	0	137,01313	7,77865556	11,1337677	11,4427249	0,48613525	0	66,7162805	129,814406	129,833936
8	0	83,1107622	0,42500583	0,01539802	0,02516573	2,12494787	66,7162805	0	84,1832121	86,3496744
9	0	214,062951	0,90247568	2,59381547	3,24047617	0,32874977	129,814406	84,1832121	0	213,887911
10	0	271,834685	0,27461766	1,57717969	2,20871097	0,77069982	129,833936	86,3496744	213,887911	0

Figura 14. Matriz de ahorros (Módulo didáctico).

Por ejemplo:

$$i = 2, j = 3$$

$$S = 143{,}235\ km + 130{,}148\ km - 271{,}288\ km$$

$$S = 2{,}1\ km$$

El valor obtenido representa un ahorro en kilómetros si se combina la ruta entre el depósito y los nodos dos y tres.

4.1.2.3.2. Posibles rutas en VBA

Se toma en cuenta el máximo ahorro obtenido en el paso anterior y se definen los nodos a los cuales pertenece dicho ahorro, se procede a eliminarlo de la matriz conjuntamente con la fila y

columna que forman la intersección y se busca un nuevo máximo para establecer la siguiente ruta.

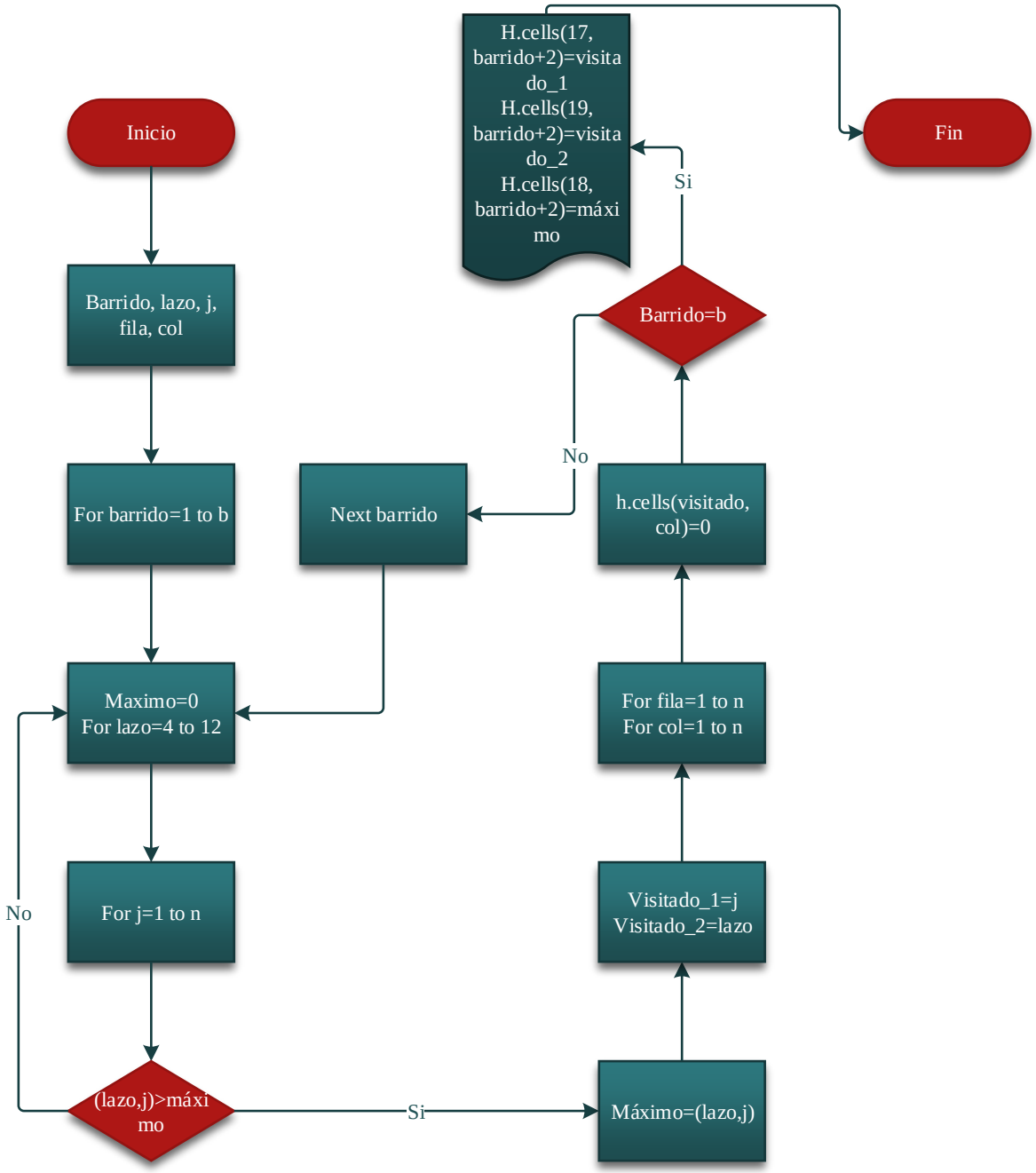


Figura 15. Posibles rutas en VBA, pseudo código.

```
Sub ruta()

Dim i As Integer

Dim j As Integer
```

```
Dim lazo As Integer
Dim dep As Integer
Dim visitado_1 As Integer
Dim visitado_2 As Integer
Dim maximo As Integer
Dim fila_1 As Integer
Dim fila_2 As Integer
Dim fila_3 As Integer
Dim col As Integer
Dim barrido As Integer
Set h = Sheets("Hoja2")
For barrido = 1 To 9
maximo = 0
For lazo = 4 To 12
For j = 3 To 12
If h.Cells(lazo, j) > maximo Then
maximo = h.Cells(lazo, j)
visitado_1 = j
visitado_2 = lazo
End If
Next j
```

Se define el mayor ahorro de la matriz.

Se actualiza el mayor ahorro.

```
Next lazo

For fila = 3 To 12

h.Cells(fila, visitado_1) = 0

Next fila

For fila_2 = 3 To 12

h.Cells(visitado_2, fila_2) = 0

Next fila_2

For col = 3 To 12

h.Cells(visitado_1, col) = 0

Next col

For fila_3 = 3 To 12

h.Cells(fila_3, visitado_2) = 0

Next fila_3

Next barrido

h.Cells(17, barrido + 2) = visitado_1 - 2

h.Cells(19, barrido + 2) = visitado_2 - 2

h.Cells(18, barrido + 2) = maximo

For matriz = 1 To 2

For nodos = 1 To 4

suma_ruta_p = suma_ruta_p + h.Cells(16 + matriz, 2 + nodos)

Next nodos
```

Se realiza una búsqueda del mayor ahorro dentro de toda la matriz y se elimina la fila y la columna a la cual pertenece dicho ahorro.

Impresión de los nodos visitados y el ahorro.

```
h.Cells(17, 7) = suma_ruta_p

Next matriz

Select Case suma_ruta_p

Case 44

h.Cells(17, 7) = 10

Case 45

h.Cells(17, 7) = 9

Case 46

h.Cells(17, 7) = 8

Case 47

h.Cells(17, 7) = 7

Case 48

h.Cells(17, 7) = 6

Case 49

h.Cells(17, 7) = 5

Case 50

h.Cells(17, 7) = 4

Case 51

h.Cells(17, 7) = 3

Case Else

h.Cells(17, 7) = 2
```

Se realiza una sumatoria de los nodos estableciendo casos para cada uno de los resultados que pudiera obtenerse y de esta manera definir el nodo que falta visitar.

```
    End Select
For fila = 3 To 12
For lazo = 3 To 12
Worksheets("Hoja2").Cells(lazo, fila) = Worksheets("Hoja1").Cells(lazo, fila + 13)
Next lazo
Next fila
End Sub
```

Se reestablecen los valores originales en la matriz de ahorros.

Todas las rutas encontradas deben salir del depósito definido como el nodo uno y terminar en el mismo, teniendo en cuenta que solo se van a generar rutas entre pares de nodos.

4.1.3. Código de programación completo del módulo

```
Sub hoja_TSP()
Worksheets.Add.Name = "VECINO MÁS CERCANO"
Set h = Sheets("VECINO MÁS CERCANO")
Worksheets("VECINO MÁS CERCANO").Cells(1, 1) = "INICIO"
Worksheets("VECINO MÁS CERCANO").Cells(1, 11) = "FIN"
Worksheets("VECINO MÁS CERCANO").Cells(1, 12) = "TOTAL"
Worksheets("VECINO MÁS CERCANO").Cells(3, 1) = "DISTANCIA PARCIAL"
Worksheets("VECINO MÁS CERCANO").Cells(4, 1) = "COSTO PARCIAL"
Worksheets("VECINO MÁS CERCANO").Cells(5, 1) = "COSTO TOTAL"
Worksheets("VECINO MÁS CERCANO").Cells(1, 13) = "SEGUNDOS"
Worksheets("VECINO MÁS CERCANO").Range("A1:M1").Interior.Color = RGB(127, 219,
231)
```

```
Worksheets("VECINO MÁS CERCANO").Range("A3:A5").Interior.Color = RGB(127, 219, 231)

For fila = 3 To 12

For lazo = 3 To 12

Worksheets("DISTANCIAS").Cells(lazo, fila) = Worksheets("DISTANCIAS").Cells(lazo, fila + 12)

Next lazo

Next fila

Range("L3").Formula = "=SUM(B3:K3)"

Range("L4").Formula = "=SUM(B4:K4)"

End Sub

Sub vecino_mas_cercano()

Dim i As Integer

Dim j As Integer

Dim minimo As Double

Dim lazo As Integer

Dim visitado As Integer

Dim fila As Integer

Dim t As Double

Dim segundos1 As Single

Dim segundos2 As Single

segundos1 = Timer()
```

```
Set h = Sheets("VECINO MÁS CERCANO")

i = Worksheets("DISTANCIAS").Cells(15, 2) + 2

For fila = 3 To 12

Worksheets("DISTANCIAS").Cells(fila, i) = 10000

Next fila

For lazo = 1 To 9

minimo = 1000

For j = 3 To 12

If Worksheets("DISTANCIAS").Cells(i, j) < minimo Then

minimo = Worksheets("DISTANCIAS").Cells(i, j)

visitado = j

End If

Next j

i = visitado

For fila = 3 To 12

Worksheets("DISTANCIAS").Cells(fila, visitado) = 10000

Next fila

Worksheets("VECINO MÁS CERCANO").Cells(2, 1 + lazo) = visitado - 2

Worksheets("VECINO MÁS CERCANO").Cells(3, 1 + lazo) = minimo

Worksheets("VECINO MÁS CERCANO").Cells(4, 1 + lazo) = minimo *
Worksheets("DISTANCIAS").Cells(18, 2)

Next lazo
```

```
Worksheets("VECINO MÁS CERCANO").Cells(2, 1) = Worksheets("DISTANCIAS").Cells(15, 2)

Worksheets("VECINO MÁS CERCANO").Cells(2, 11) = Worksheets("DISTANCIAS").Cells(15, 2)

For fila = 3 To 12

For lazo = 3 To 12

Worksheets("DISTANCIAS").Cells(lazo, fila) = Worksheets("DISTANCIAS").Cells(lazo, fila + 12)

Next lazo

Next fila

ori = Worksheets("VECINO MÁS CERCANO").Cells(2, 10)

dest = Worksheets("VECINO MÁS CERCANO").Cells(2, 11)

distancia = Worksheets("DISTANCIAS").Cells(ori + 2, dest + 2)

Worksheets("VECINO MÁS CERCANO").Cells(3, 11) = distancia

Worksheets("VECINO MÁS CERCANO").Cells(4, 11) = Worksheets("VECINO MÁS CERCANO").Cells(3, 11) * Worksheets("DISTANCIAS").Cells(18, 2)

Worksheets("VECINO MÁS CERCANO").Cells(5, 2) = Worksheets("VECINO MÁS CERCANO").Cells(4, 12) + Worksheets("DISTANCIAS").Cells(19, 2)

Worksheets("VECINO MÁS CERCANO").Range("A2:K2").Interior.Color = RGB(191, 191, 191)

Worksheets("VECINO MÁS CERCANO").Range("B3:L4").Interior.Color = RGB(191, 191, 191)

Worksheets("VECINO MÁS CERCANO").Range("B5").Interior.Color = RGB(191, 191, 191)

Worksheets("VECINO MÁS CERCANO").Range("M2").Interior.Color = RGB(191, 191, 191)
```

```
segundos2 = Timer()

Worksheets("VECINO MÁS CERCANO").Cells(2, 13) = segundos2 - segundos1

MsgBox ("TIEMPO UTILIZADO PARA CORRER EL CÓDIGO:" y vbNewLine y segundos2
- segundos1 y " SEGUNDOS" y vbCrLf y "ALGORITMO FINALIZADO")

End Sub

Function Factorial(numero)

   Dim total, contador As Integer

   total = 1

   For contador = 1 To numero

      total = total * contador

   Next

   Factorial = total

End Function

Sub permutaciones()

Dim minimo As Integer

Dim i As Integer

Dim fila As Integer

Dim parcialx As Integer

Dim parcialy As Integer

Set h = Sheets("DISTANCIAS")

Worksheets("DISTANCIAS").Cells(16,                2)                =
Factorial(Worksheets("DISTANCIAS").Range("B3").End(xlDown).Value - 1)
```

```
End Sub

Sub hoja_BSEARCH()

Worksheets.Add.Name = "BRUTALSEARCH"

End Sub

Sub Ruta_parcial()

Dim lazo As Integer

For lazo = 1 To 9

Worksheets("BRUTALSEARCH").Cells(1, lazo) = Worksheets("VECINO MÁS CERCANO").Cells(2, 1 + lazo)

Next lazo

End Sub

Sub InsertarFilas()

Dim numeroFilasAInsertar As Integer

Dim contadorFilas As Integer

Set h = Sheets("BRUTALSEARCH")

numeroFilasAInsertar = InputBox("Introduzca el numero de filas", "Aplicacion filas")

ActiveCell.EntireRow.Select

For contadorFilas = 1 To numeroFilasAInsertar

    Selection.Insert Shift:=xlToDown

Next contadorFilas

End Sub

Sub InsertarColumnas()
```

```
Dim numeroColumnasAInsertar As Integer

Dim contadorColumnas As Integer

numeroColumnasAInsertar = InputBox("Introduzca el numero de Columnas", "Aplicacion columnas")

ActiveCell.EntireColumn.Select

For contadorColumnas = 1 To numeroColumnasAInsertar

   Selection.Insert Shift:=xlToRight

Next contadorColumnas

End Sub

Sub inicio_fin()

Dim fila As Integer

Dim segundos1 As Single

Dim segundos2 As Single

segundos1 = Timer()

Worksheets("BRUTALSEARCH").Cells(1, 1) = "INICIO"

Worksheets("BRUTALSEARCH").Cells(1, 11) = "FIN"

Worksheets("BRUTALSEARCH").Cells(1, 12) = "DISTANCIA"

Worksheets("BRUTALSEARCH").Cells(1, 13) = "TIEMPO"

Worksheets("BRUTALSEARCH").Cells(1, 15) = "ÓPTIMO"

Worksheets("BRUTALSEARCH").Cells(1, 14) = "COSTO TOTAL"

Worksheets("BRUTALSEARCH").Cells(1, 16) = "FILA"

Worksheets("BRUTALSEARCH").Cells(3, 15) = "RUTA"
```

```
Worksheets("BRUTALSEARCH").Cells(1, 17) = "SEG_PERM"

Worksheets("BRUTALSEARCH").Cells(1, 18) = "SEG_IN_FIN"

Worksheets("BRUTALSEARCH").Cells(1, 19) = "SEG_CICLOS"

Worksheets("BRUTALSEARCH").Range("O3").Interior.Color = RGB(255, 176, 17)

Worksheets("BRUTALSEARCH").Range("A1:S1").Interior.Color = RGB(255, 176, 17)

For lazo = 2 To 30000

Worksheets("BRUTALSEARCH").Cells(lazo, 1) = Worksheets("DISTANCIAS").Cells(15, 2)

Worksheets("BRUTALSEARCH").Cells(lazo, 11) = Worksheets("DISTANCIAS").Cells(15, 2)

Next lazo

Worksheets("BRUTALSEARCH").Range("K2:K362882").Interior.Color = RGB(191, 191, 191)

Worksheets("BRUTALSEARCH").Range("A2:A362882").Interior.Color = RGB(191, 191, 191)

segundos2 = Timer()

Worksheets("BRUTALSEARCH").Cells(2, 18) = segundos2 - segundos1

MsgBox ("TIEMPO UTILIZADO PARA CORRER EL CÓDIGO:" y vbNewLine y segundos2 - segundos1 y " SEGUNDOS")

End Sub

Sub ciclos()

Dim fila As Integer

Dim parcial As Integer

Dim ori As Integer
```

```
Dim dest As Integer

Dim distancia As Double

Dim segundos1 As Single

Dim segundos2 As Single

segundos1 = Timer()

For fila = 2 To 30000

distancia = 0

For parcial = 1 To 10

ori = Worksheets("BRUTALSEARCH").Cells(fila, parcial)

dest = Worksheets("BRUTALSEARCH").Cells(fila, parcial + 1)

distancia = Worksheets("DISTANCIAS").Cells(ori + 2, dest + 2) + distancia

Next parcial

Worksheets("BRUTALSEARCH").Cells(fila, 12) = distancia

Worksheets("BRUTALSEARCH").Cells(fila, 13) = distancia /
Worksheets("DISTANCIAS").Cells(17, 2)

Worksheets("BRUTALSEARCH").Cells(fila, 14) = distancia *
Worksheets("DISTANCIAS").Cells(18, 2) + Worksheets("DISTANCIAS").Cells(19, 2)

Next fila

Worksheets("BRUTALSEARCH").Range("O2").Formula = "=MIN(L2:L30000)"

For fila = 2 To 30000

If Worksheets("BRUTALSEARCH").Cells(fila, 12) =
Worksheets("BRUTALSEARCH").Cells(2, 15) Then

Worksheets("BRUTALSEARCH").Cells(2, 16) = fila
```

```
End If

Next fila

For lazo = 1 To 11

Worksheets("BRUTALSEARCH").Cells(3, 15 + lazo) = Worksheets("BRUTALSEARCH").Cells(Worksheets("BRUTALSEARCH").Cells(2, 16), lazo)

Next lazo

Worksheets("BRUTALSEARCH").Range("L2:N362882").Interior.Color = RGB(191, 191, 191)

Worksheets("BRUTALSEARCH").Range("O2:S2").Interior.Color = RGB(191, 191, 191)

Worksheets("BRUTALSEARCH").Range("P3:Z3").Interior.Color = RGB(191, 191, 191)

segundos2 = Timer()

Worksheets("BRUTALSEARCH").Cells(2, 19) = segundos2 - segundos1

MsgBox ("TIEMPO UTILIZADO PARA CORRER EL CÓDIGO:" y vbNewLine y segundos2 - segundos1 y " SEGUNDOS" y vbCrLf y "ALGORITMO FINALIZADO")

End Sub

Function Fkii(arg As Long) As Long

Dim i As Long

   Fkii = 1

   For i = 1 To arg

      Fkii = Fkii * i

   Next i

End Function
```

```
Sub R_Permutacion()
Dim segundos1 As Single
Dim segundos2 As Single
segundos1 = Timer()
Worksheets("BRUTALSEARCH").Range("$A$2:$I$1048576").Clear
Dim DatString() As Variant
Dim ene As Long
ene = Worksheets("BRUTALSEARCH").Range("a1").CurrentRegion.Count
If ene > 9 Then
   MsgBox "Solo hasta 9 elementos"
   Exit Sub
End If
If ene <= 1 Then
   MsgBox "Mínimo dos elementos"
   Exit Sub
End If
ReDim DatString(1 To ene)
Dim i As Long, j As Long, k As Long, m As Long, n As Long
For i = 1 To ene
   DatString(i) = Worksheets("BRUTALSEARCH").Cells(1, i).Value
Next i
```

```
Dim ciclos As Long

ciclos = Fkii(ene)

Dim FilCiclo, ColCiclo As Long

Dim NewElment As Variant

Dim MPer() As Variant, MPas() As Variant

ReDim MPer(1 To ciclos, 1 To ene)

ReDim MPas(1 To ciclos, 1 To ene)

Dim SubNiv As Long

MPer(1, 1) = DatString(2)

MPer(1, 2) = DatString(1)

MPer(2, 1) = DatString(1)

MPer(2, 2) = DatString(2)

For i = 3 To ene

   FilCiclo = Fkii(i)

   ColCiclo = i

   NewElment = DatString(i)

   For j = 1 To FilCiclo

      SubNiv = 1 + ((j - 1) \ i)

         For k = 1 To ColCiclo

            MPas(j, k) = MPer(SubNiv, k)

         Next k
```

```
    Next j
For m = 1 To ciclos
For n = 1 To ene
MPer(m, n) = MPas(m, n)
Next n
Next m
Dim jmod As Long
    For j = 1 To FilCiclo
    jmod = j Mod i
    If jmod = 0 Then
    jmod = i
    End If
        For k = ColCiclo To jmod + 1 Step -1
            MPer(j, k) = MPer(j, k - 1)
        Next k
        MPer(j, jmod) = NewElment
    Next j
Next i
For m = 1 To ciclos
    For n = 1 To ene
        Worksheets("BRUTALSEARCH").Cells(m + 1, n) = MPer(m, n)
```

```
    Next n
Next m
Worksheets("BRUTALSEARCH").Range("A1:I362881").Interior.Color = RGB(191, 191, 191)
segundos2 = Timer()
Worksheets("BRUTALSEARCH").Cells(1, 16) = segundos2 - segundos1
MsgBox ciclos y ("PERMUTACIONES GENERADAS" y vbCrLf y "TIEMPO UTILIZADO PARA CORRER EL CÓDIGO:" y vbNewLine y segundos2 - segundos1 y " SEGUNDOS")
End Sub
Sub hoja_cyw()
Worksheets.Add.Name = "CYW"
For fila = 3 To 12
For lazo = 3 To 12
Worksheets("DISTANCIAS").Cells(lazo, fila) = Worksheets("DISTANCIAS").Cells(lazo, fila + 12)
Next lazo
Next fila
End Sub
Sub ahorros()
Dim i As Integer
Dim j As Integer
Dim dep As Integer
Dim segundos1 As Single
```

```
Dim segundos2 As Single

segundos1 = Timer()

dep = Worksheets("DISTANCIAS").Cells(15, 2) + 2

Worksheets("CYW").Range("B2:M2").Interior.Color = RGB(50, 209, 29)

Worksheets("CYW").Range("B3:B12").Interior.Color = RGB(50, 209, 29)

Worksheets("CYW").Range("B16").Interior.Color = RGB(50, 209, 29)

Worksheets("CYW").Range("B19:B22").Interior.Color = RGB(50, 209, 29)

Worksheets("CYW").Range("M2") = "SEG_AHORROS"

Worksheets("CYW").Range("B22") = "SEG_RUTAS"

For i = 3 To 12

For j = 3 To 12

If i = j Or Worksheets("DISTANCIAS").Cells(dep, j) = Worksheets("DISTANCIAS").Cells(i, j) Or Worksheets("DISTANCIAS").Cells(dep, i) = Worksheets("DISTANCIAS").Cells(i, j) Then

Worksheets("CYW").Cells(i, j + 13) = 0

Else

Worksheets("CYW").Cells(i, j + 13) = Worksheets("DISTANCIAS").Cells(dep, i) + Worksheets("DISTANCIAS").Cells(dep, j) - Worksheets("DISTANCIAS").Cells(i, j)

End If

Next j

Next i

Worksheets("CYW").Cells(2, 2) = "NODOS"

For lazo = 1 To 10
```

```
Worksheets("CYW").Cells(2, 2 + lazo) = lazo

Worksheets("CYW").Cells(2 + lazo, 2) = lazo

Next lazo

For fila = 3 To 12

For lazo = 3 To 12

Worksheets("CYW").Cells(lazo, fila) = Worksheets("CYW").Cells(lazo, fila + 13)

Next lazo

Next fila

Worksheets("CYW").Range("C3:L12").Interior.Color = RGB(191, 191, 191)

Worksheets("CYW").Range("P3:Y12").Interior.Color = RGB(191, 191, 191)

Worksheets("CYW").Range("M3").Interior.Color = RGB(191, 191, 191)

Worksheets("CYW").Range("C22").Interior.Color = RGB(191, 191, 191)

segundos2 = Timer()

Worksheets("CYW").Range("M3") = segundos2 - segundos1

MsgBox ("TIEMPO UTILIZADO PARA CORRER EL CÓDIGO:" y vbNewLine y segundos2
- segundos1 y " SEGUNDOS")

End Sub

Sub ruta()

Dim i As Integer

Dim j As Integer

Dim lazo As Integer

Dim dep As Integer
```

```
Dim visitado_1 As Integer

Dim visitado_2 As Integer

Dim maximo As Double

Dim fila_1 As Integer

Dim fila_2 As Integer

Dim fila_3 As Integer

Dim col As Integer

Dim barrido As Integer

Dim ruta As Integer

Dim ori As Integer

Dim dest As Integer

Dim distancia As Double

Dim segundos1 As Single

Dim segundos2 As Single

segundos1 = Timer()

Set h = Sheets("CYW")

For barrido = 1 To 9

maximo = 0

For lazo = 4 To 12

For j = 3 To 12
```

```
If h.Cells(lazo, j) > maximo Then

maximo = h.Cells(lazo, j)

visitado_1 = j

visitado_2 = lazo

End If

Next j

Next lazo

h.Cells(17, barrido + 2) = visitado_1 - 2

h.Cells(18, barrido + 2) = visitado_2 - 2

h.Cells(20, barrido + 2) = maximo

For fila = 3 To 12

h.Cells(fila, visitado_1) = 0

Next fila

For fila_2 = 3 To 12

h.Cells(visitado_2, fila_2) = 0

Next fila_2

For col = 3 To 12

h.Cells(visitado_1, col) = 0

Next col

For fila_3 = 3 To 12

h.Cells(fila_3, visitado_2) = 0
```

```
Next fila_3

Next barrido

h.Cells(18, 7) = 1

For matriz = 1 To 2

For nodos = 1 To 4

suma_ruta_p = suma_ruta_p + h.Cells(16 + matriz, 2 + nodos)

Next nodos

h.Cells(17, 7) = suma_ruta_p

Next matriz

Select Case suma_ruta_p

Case 44

h.Cells(17, 7) = 10

Case 45

h.Cells(17, 7) = 9

Case 46

h.Cells(17, 7) = 8

Case 47

h.Cells(17, 7) = 7

Case 48

h.Cells(17, 7) = 6
```

```
Case 49

h.Cells(17, 7) = 5

Case 50

h.Cells(17, 7) = 4

Case 51

h.Cells(17, 7) = 3

Case Else

h.Cells(17, 7) = 2

End Select

h.Cells(16, 2) = "INICIO"

h.Cells(19, 2) = "FIN"

h.Cells(20, 2) = "AHORRO"

h.Cells(21, 2) = "DISTANCIA"

For lazo = 1 To 5

h.Cells(16, 2 + lazo) = Worksheets("DISTANCIAS").Cells(15, 2)

Next lazo

For lazo = 1 To 4

h.Cells(19, 2 + lazo) = Worksheets("DISTANCIAS").Cells(15, 2)

Next lazo

For parcial = 3 To 7

distancia = 0
```

```
For fila = 16 To 18
ori = Worksheets("CYW").Cells(fila, parcial)
dest = Worksheets("CYW").Cells(fila + 1, parcial)
distancia = Worksheets("DISTANCIAS").Cells(ori + 2, dest + 2) + distancia
Next fila
Worksheets("CYW").Cells(21, parcial) = distancia
Next parcial
For fila = 8 To 11
For lazo = 17 To 20
Worksheets("CYW").Cells(lazo, fila) = ""
Next lazo
Next fila
For fila = 3 To 12
For lazo = 3 To 12
Worksheets("CYW").Cells(lazo, fila) = Worksheets("CYW").Cells(lazo, fila + 13)
Next lazo
Next fila
Worksheets("CYW").Cells(21, 7) = Worksheets("CYW").Cells(21, 7) - 1
Worksheets("CYW").Range("C16:G21").Interior.Color = RGB(191, 191, 191)
segundos2 = Timer()
Worksheets("CYW").Range("C22") = segundos2 - segundos1
```

```
MsgBox ("TIEMPO UTILIZADO PARA CORRER EL CÓDIGO:" y vbNewLine y segundos2 - segundos1 y " SEGUNDOS" y vbCrLf y "ALGORITMO FINALIZADO")

End Sub

Sub hoja_resultados()

Worksheets.Add.Name = "RESULTADOS"

End Sub

Sub resultados_tsp()

Dim lazo As Integer

Worksheets("RESULTADOS").Range("A1:L1").Merge

Worksheets("RESULTADOS").Cells(1, 1) = "VECINO MÁS CERCANO"

Worksheets("RESULTADOS").Range("A1").HorizontalAlignment = xlCenter

Worksheets("RESULTADOS").Cells(2, 1) = "RUTA"

Worksheets("RESULTADOS").Cells(3, 1) = "DISTANCIA"

Worksheets("RESULTADOS").Cells(4, 1) = "COSTO TOTAL"

Worksheets("RESULTADOS").Cells(5, 1) = "TIEMPO"

Worksheets("RESULTADOS").Cells(3, 3) = "KM"

Worksheets("RESULTADOS").Cells(4, 3) = "UM"

Worksheets("RESULTADOS").Cells(5, 3) = "HORAS"

Worksheets("RESULTADOS").Range("E4") = "T_EJECUCIÓN"

Worksheets("RESULTADOS").Range("G4") = "SEGUNDOS"

For lazo = 1 To 11
```

```
Worksheets("RESULTADOS").Cells(2, 1 + lazo) = Worksheets("VECINO MÁS CERCANO").Cells(2, lazo)

Next lazo

Worksheets("RESULTADOS").Cells(3, 2) = Worksheets("VECINO MÁS CERCANO").Cells(3, 12)

Worksheets("RESULTADOS").Cells(4, 2) = Worksheets("VECINO MÁS CERCANO").Cells(5, 2)

Worksheets("RESULTADOS").Cells(5, 2) = Worksheets("RESULTADOS").Cells(3, 2) / Worksheets("DISTANCIAS").Cells(17, 2)

Worksheets("RESULTADOS").Range("F4") = Worksheets("VECINO MÁS CERCANO").Range("M2")

Worksheets("RESULTADOS").Range("A1").Interior.Color = RGB(127, 219, 231)

Worksheets("RESULTADOS").Range("E4").Interior.Color = RGB(127, 219, 231)

Worksheets("RESULTADOS").Range("A2:A5").Interior.Color = RGB(127, 219, 231)

Worksheets("RESULTADOS").Range("B2:L2").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("B3:B5").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("C3").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("C5").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("F4:G4").Interior.Color = RGB(191, 191, 191)

End Sub

Sub resultados_bsearch()

Dim lazo As Integer

Worksheets("RESULTADOS").Range("A7:L7").Merge
```

```
Worksheets("RESULTADOS").Cells(7, 1) = "BRUTAL SEARCH"

Worksheets("RESULTADOS").Range("A7").HorizontalAlignment = xlCenter

Worksheets("RESULTADOS").Cells(8, 1) = "RUTA"

Worksheets("RESULTADOS").Cells(9, 1) = "DISTANCIA"

Worksheets("RESULTADOS").Cells(10, 1) = "COSTO TOTAL"

Worksheets("RESULTADOS").Cells(11, 1) = "TIEMPO"

Worksheets("RESULTADOS").Cells(9, 3) = "KM"

Worksheets("RESULTADOS").Cells(10, 3) = "UM"

Worksheets("RESULTADOS").Cells(11, 3) = "HORAS"

Worksheets("RESULTADOS").Range("E10") = "T_EJECUCIÓN"

Worksheets("RESULTADOS").Range("G10") = "SEGUNDOS"

For lazo = 1 To 11

Worksheets("RESULTADOS").Cells(8, 1 + lazo) = Worksheets("BRUTALSEARCH").Cells(3, 15 + lazo)

Next lazo

Worksheets("RESULTADOS").Cells(9, 2) = Worksheets("BRUTALSEARCH").Cells(Worksheets("BRUTALSEARCH").Cells(2, 16), 12)

Worksheets("RESULTADOS").Cells(10, 2) = Worksheets("BRUTALSEARCH").Cells(Worksheets("BRUTALSEARCH").Cells(2, 16), 14)

Worksheets("RESULTADOS").Cells(11, 2) = Worksheets("BRUTALSEARCH").Cells(Worksheets("BRUTALSEARCH").Cells(2, 16), 13)
```

```
Worksheets("RESULTADOS").Range("F10")                                    =
Worksheets("BRUTALSEARCH").Range("Q2")                                   +
Worksheets("BRUTALSEARCH").Range("R2")                                   +
Worksheets("BRUTALSEARCH").Range("S2")

Worksheets("RESULTADOS").Range("A7").Interior.Color = RGB(255, 176, 17)

Worksheets("RESULTADOS").Range("E10").Interior.Color = RGB(255, 176, 17)

Worksheets("RESULTADOS").Range("A8:A11").Interior.Color = RGB(255, 176, 17)

Worksheets("RESULTADOS").Range("B8:L8").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("B9:B11").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("C9").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("C11").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("F10:G10").Interior.Color = RGB(191, 191, 191)

End Sub

Sub resultados_cyw()

Dim fila As Integer

Dim lazo As Integer

Worksheets("RESULTADOS").Range("A13:F13").Merge

Worksheets("RESULTADOS").Cells(13, 1) = "C Y W"

Worksheets("RESULTADOS").Range("A13").HorizontalAlignment = xlCenter

Worksheets("RESULTADOS").Range("A14:A17").Merge

Worksheets("RESULTADOS").Cells(14, 1) = "RUTAS"

Worksheets("RESULTADOS").Range("A14").VerticalAlignment = xlCenter

Worksheets("RESULTADOS").Cells(18, 1) = "DISTANCIA"
```

```
Worksheets("RESULTADOS").Cells(19, 1) = "COSTO TOTAL"

Worksheets("RESULTADOS").Cells(20, 1) = "TIEMPO"

Worksheets("RESULTADOS").Cells(18, 3) = "KM"

Worksheets("RESULTADOS").Cells(19, 3) = "UM"

Worksheets("RESULTADOS").Cells(20, 3) = "HORAS"

Worksheets("RESULTADOS").Range("E19") = "T_EJECUCIÓN"

Worksheets("RESULTADOS").Range("G19") = "SEGUNDOS"

For fila = 1 To 4

For lazo = 1 To 5

Worksheets("RESULTADOS").Cells(13 + fila, 1 + lazo) = Worksheets("CYW").Cells(15 + fila, 2 + lazo)

Next lazo

Next fila

Worksheets("RESULTADOS").Range("B18").Formula = "=SUM(CYW!C21:G21)"

Worksheets("RESULTADOS").Cells(19, 2) = Worksheets("RESULTADOS").Cells(18, 2) * Worksheets("DISTANCIAS").Cells(18, 2) + Worksheets("DISTANCIAS").Cells(19, 2)

Worksheets("RESULTADOS").Cells(20, 2) = Worksheets("RESULTADOS").Cells(18, 2) / Worksheets("DISTANCIAS").Cells(17, 2)

Worksheets("RESULTADOS").Range("F19") = Worksheets("CYW").Range("M3") + Worksheets("CYW").Range("C22")

Worksheets("RESULTADOS").Range("A13").Interior.Color = RGB(50, 209, 29)

Worksheets("RESULTADOS").Range("E19").Interior.Color = RGB(50, 209, 29)

Worksheets("RESULTADOS").Range("A14:A20").Interior.Color = RGB(50, 209, 29)
```

```
Worksheets("RESULTADOS").Range("B14:F16").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("B17:E17").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("B18:B20").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("C18").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("C20").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("F19:G19").Interior.Color = RGB(191, 191, 191)

End Sub

Sub mejora()

Worksheets("RESULTADOS").Range("M1:N1").Merge

Worksheets("RESULTADOS").Cells(1, 13) = "MEJORA"

Worksheets("RESULTADOS").Range("M1").HorizontalAlignment = xlCenter

Worksheets("RESULTADOS").Cells(2, 13) = "B_SEARCH"

Worksheets("RESULTADOS").Cells(2, 14) = "C Y W"

Worksheets("RESULTADOS").Cells(3, 13) = ((Worksheets("RESULTADOS").Cells(9, 2) -
Worksheets("RESULTADOS").Cells(3, 2)) / Worksheets("RESULTADOS").Cells(9, 2)) *
100

Worksheets("RESULTADOS").Cells(4, 13) = ((Worksheets("RESULTADOS").Cells(10, 2) -
Worksheets("RESULTADOS").Cells(4, 2)) / Worksheets("RESULTADOS").Cells(10, 2)) *
100

Worksheets("RESULTADOS").Cells(5, 13) = ((Worksheets("RESULTADOS").Cells(11, 2) -
Worksheets("RESULTADOS").Cells(5, 2)) / Worksheets("RESULTADOS").Cells(11, 2)) *
100
```

```
Worksheets("RESULTADOS").Cells(3, 14) = ((Worksheets("RESULTADOS").Cells(18, 2) - Worksheets("RESULTADOS").Cells(3, 2)) / Worksheets("RESULTADOS").Cells(18, 2)) * 100

Worksheets("RESULTADOS").Cells(4, 14) = ((Worksheets("RESULTADOS").Cells(19, 2) - Worksheets("RESULTADOS").Cells(4, 2)) / Worksheets("RESULTADOS").Cells(19, 2)) * 100

Worksheets("RESULTADOS").Cells(5, 14) = ((Worksheets("RESULTADOS").Cells(20, 2) - Worksheets("RESULTADOS").Cells(5, 2)) / Worksheets("RESULTADOS").Cells(20, 2)) * 100

Worksheets("RESULTADOS").Cells(8, 13) = "V+CERCANO"

Worksheets("RESULTADOS").Cells(8, 14) = "C Y W"

Worksheets("RESULTADOS").Cells(9, 13) = ((Worksheets("RESULTADOS").Cells(3, 2) - Worksheets("RESULTADOS").Cells(9, 2)) / Worksheets("RESULTADOS").Cells(3, 2)) * 100

Worksheets("RESULTADOS").Cells(10, 13) = ((Worksheets("RESULTADOS").Cells(4, 2) - Worksheets("RESULTADOS").Cells(10, 2)) / Worksheets("RESULTADOS").Cells(4, 2)) * 100

Worksheets("RESULTADOS").Cells(11, 13) = ((Worksheets("RESULTADOS").Cells(5, 2) - Worksheets("RESULTADOS").Cells(11, 2)) / Worksheets("RESULTADOS").Cells(5, 2)) * 100

Worksheets("RESULTADOS").Cells(9, 14) = ((Worksheets("RESULTADOS").Cells(18, 2) - Worksheets("RESULTADOS").Cells(9, 2)) / Worksheets("RESULTADOS").Cells(18, 2)) * 100

Worksheets("RESULTADOS").Cells(10, 14) = ((Worksheets("RESULTADOS").Cells(19, 2) - Worksheets("RESULTADOS").Cells(10, 2)) / Worksheets("RESULTADOS").Cells(19, 2)) * 100
```

```
Worksheets("RESULTADOS").Cells(11, 14) = ((Worksheets("RESULTADOS").Cells(20, 2) - Worksheets("RESULTADOS").Cells(11, 2)) / Worksheets("RESULTADOS").Cells(20, 2)) * 100

Worksheets("RESULTADOS").Cells(17, 13) = "V+CERCANO"

Worksheets("RESULTADOS").Cells(17, 14) = "B_SEARCH"

Worksheets("RESULTADOS").Cells(18, 13) = ((Worksheets("RESULTADOS").Cells(3, 2) - Worksheets("RESULTADOS").Cells(18, 2)) / Worksheets("RESULTADOS").Cells(3, 2)) * 100

Worksheets("RESULTADOS").Cells(19, 13) = ((Worksheets("RESULTADOS").Cells(4, 2) - Worksheets("RESULTADOS").Cells(19, 2)) / Worksheets("RESULTADOS").Cells(4, 2)) * 100

Worksheets("RESULTADOS").Cells(20, 13) = ((Worksheets("RESULTADOS").Cells(5, 2) - Worksheets("RESULTADOS").Cells(20, 2)) / Worksheets("RESULTADOS").Cells(5, 2)) * 100

Worksheets("RESULTADOS").Cells(18, 14) = ((Worksheets("RESULTADOS").Cells(9, 2) - Worksheets("RESULTADOS").Cells(18, 2)) / Worksheets("RESULTADOS").Cells(9, 2)) * 100

Worksheets("RESULTADOS").Cells(19, 14) = ((Worksheets("RESULTADOS").Cells(10, 2) - Worksheets("RESULTADOS").Cells(19, 2)) / Worksheets("RESULTADOS").Cells(10, 2)) * 100

Worksheets("RESULTADOS").Cells(20, 14) = ((Worksheets("RESULTADOS").Cells(11, 2) - Worksheets("RESULTADOS").Cells(20, 2)) / Worksheets("RESULTADOS").Cells(11, 2)) * 100

Worksheets("RESULTADOS").Range("O3:O5") = "%"

Worksheets("RESULTADOS").Range("O9:O11") = "%"

Worksheets("RESULTADOS").Range("O18:O20") = "%"

Worksheets("RESULTADOS").Range("M1:N2").Interior.Color = RGB(223, 115, 218)
```

```
Worksheets("RESULTADOS").Range("M8:N8").Interior.Color = RGB(223, 115, 218)

Worksheets("RESULTADOS").Range("M17:N17").Interior.Color = RGB(223, 115, 218)

Worksheets("RESULTADOS").Range("M3:N5").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("M9:N11").Interior.Color = RGB(191, 191, 191)

Worksheets("RESULTADOS").Range("M18:N20").Interior.Color = RGB(191, 191, 191)

End Sub

Sub solucion()

If Worksheets("RESULTADOS").Cells(3, 2) < Worksheets("RESULTADOS").Cells(9, 2) And Worksheets("RESULTADOS").Cells(3, 2) < Worksheets("RESULTADOS").Cells(18, 2) Then

MsgBox ("OPERATIVAMENTE LA MEJOR OPCIÓN ES VECINO MÁS CERCANO")

ElseIf Worksheets("RESULTADOS").Cells(9, 2) < Worksheets("RESULTADOS").Cells(3, 2) And Worksheets("RESULTADOS").Cells(9, 2) < Worksheets("RESULTADOS").Cells(18, 2) Then

MsgBox ("OPERATIVAMENTE LA MEJOR OPCIÓN ES BRUTAL SEARCH")

ElseIf Worksheets("RESULTADOS").Cells(18, 2) < Worksheets("RESULTADOS").Cells(3, 2) And Worksheets("RESULTADOS").Cells(18, 2) < Worksheets("RESULTADOS").Cells(9, 2) Then

MsgBox ("OPERATIVAMENTE LA MEJOR OPCIÓN ES CLARKE Y WRIGHT")

End If

If Worksheets("RESULTADOS").Cells(4, 6) < Worksheets("RESULTADOS").Cells(10, 6) And Worksheets("RESULTADOS").Cells(4, 6) < Worksheets("RESULTADOS").Cells(19, 6) Then

MsgBox ("EN TIEMPO DE CÓMPUTO LA MEJOR OPCIÓN ES VECINO MÁS CERCANO")
```

```
ElseIf Worksheets("RESULTADOS").Cells(10, 6) < Worksheets("RESULTADOS").Cells(4, 6) And Worksheets("RESULTADOS").Cells(10, 6) < Worksheets("RESULTADOS").Cells(19, 6) Then

MsgBox ("EN TIEMPO DE CÓMPUTO LA MEJOR OPCIÓN ES BRUTALSEARCH")

ElseIf Worksheets("RESULTADOS").Cells(19, 6) < Worksheets("RESULTADOS").Cells(4, 6) And Worksheets("RESULTADOS").Cells(19, 6) < Worksheets("RESULTADOS").Cells(10, 6) Then

MsgBox ("EN TIEMPO DE CÓMPUTO LA MEJOR OPCIÓN ES CLARKE Y WRIGHT")

End If

End Sub
```

La programación del módulo se realizó en cuatro rutinas, las cuales comprenden los tres algoritmos seleccionados y la comparación de resultados. Dentro de cada rutina para cada método lo primero que se realizó fue la interpretación analítica del algoritmo y su diagrama de flujo, para realizar la programación. Esta actividad puede llevar un tiempo, ya que es necesario realizar varios experimentos y revisiones para asegurar su correcto funcionamiento. Para el algoritmo *brutal search* el cual tiene más subrutinas que los otros, se inició por la programación para la generación de las permutaciones, luego se añadió el depósito al inicio y fin de cada ruta y al final se calculan las distancias y se identifica la ruta óptima. Para la heurística Clarke and Wright la primera actividad es el cálculo de ahorros entre cada par de nodos, para luego definir las rutas cuyos nodos generen los mayores ahorros. Para mejorar la presentación de los datos se programó una hoja de trabajo específica, que contiene los rótulos, cálculos de distancia, tiempo de viaje, costo y el tiempo de ejecución. Para la verificación de la respuesta se creó una hoja de resultados en donde se cargan los datos antes obtenidos por cada método y se calculan porcentajes de mejora de cada uno con respecto a los otros, y al final se puede ver la solución tanto de manera operativa como también en tiempo de ejecución del código.

4.1.4. Descripción analítica del módulo

El módulo didáctico de métodos exactos y enfoques heurísticos para la resolución de problemas VRP, ha sido elaborado para demostrar la importancia del uso de herramientas tecnológicas al realizar cálculos que podrían llevar un tiempo y esfuerzo considerables si son resueltos

analíticamente. Dicho módulo reúne varios métodos de solución a problemas VRP con fines didácticos orientados al reforzamiento del conocimiento para estudiantes que examinan dichos ejercicios dentro de su vida académica. Es importante mostrar que los métodos de solución como heurísticas y meta heurísticas que han sido utilizados en este programa pueden ser modificados según los requerimientos del usuario, con conocimientos sobre lenguaje de programación y matemático para interpretar, cambiar y diseñar partes del código que optimicen las soluciones obtenidas.

El programa se ha desarrollado en *visual basic for apps* dentro de Excel, obteniendo una interfaz más amigable para los usuarios. Ha sido diseñado para trabajar con un máximo de 10 nodos debido a que está orientado al apoyo didáctico en el aprendizaje y también a un problema relacionado con el método exacto *brutal search* que requiere gran capacidad de procesamiento.

Dentro de la interfaz del programa se han añadido varios botones que permiten realizar paso a paso la resolución del problema planteado, estos se muestran a continuación:

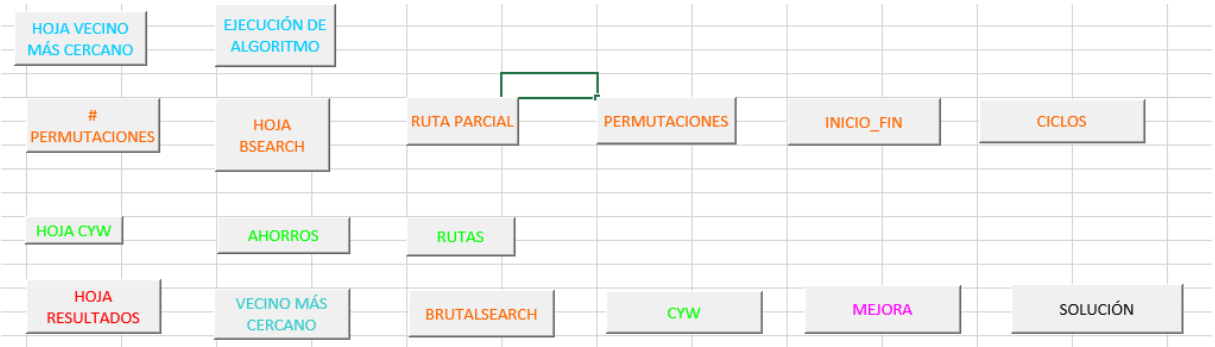


Figura 16. Botones de acción para cada algoritmo (Módulo didáctico).

Cada grupo de botones pertenece a un método de solución específico, y cada botón describe la función que realiza.

Este programa no requiere el uso de herramientas adicionales ni de una instalación previa, a excepción del caso que no se esté instalado Excel. Es necesario aclarar que las celdas que están con fondo blanco son en las que se puede cambiar valores y las que se encuentran de color gris no.

4.1.4.1 Coordenadas

La primera parte del manejo del módulo se refiere al ingreso de las coordenadas de cada uno de los nodos a considerar dentro del problema a resolver.

	A	B	C	D
1	NODO	NOMBRE	LONGITUD (X)	LATITUD (Y)
2	1	Ibarra, ECU	-78,10613251	0,365090013
3	2	Tumbaco, ECU	-78,388863	-0,21361
4	3	Tulcán, ECU	-77,727959	0,81173
5	4	Julio Andrade, ECU	-77,71851349	0,717989981
6	5	San Gabriel, ECU	-77,82717133	0,583999991
7	6	El Chota, ECU	-78,06693268	0,472900003
8	7	Cayambe, ECU	-78,1455307	0,042319998
9	8	Otavalo, ECU	-78,25688171	0,236149997
10	9	Guayllabamba, ECU	-78,34963989	-0,052990001
11	10	Quito, ECU	-78,50879669	-0,205620006
12				
13				
14				
15				
16				
17				
18				
19				
20				
21				
22				
23				

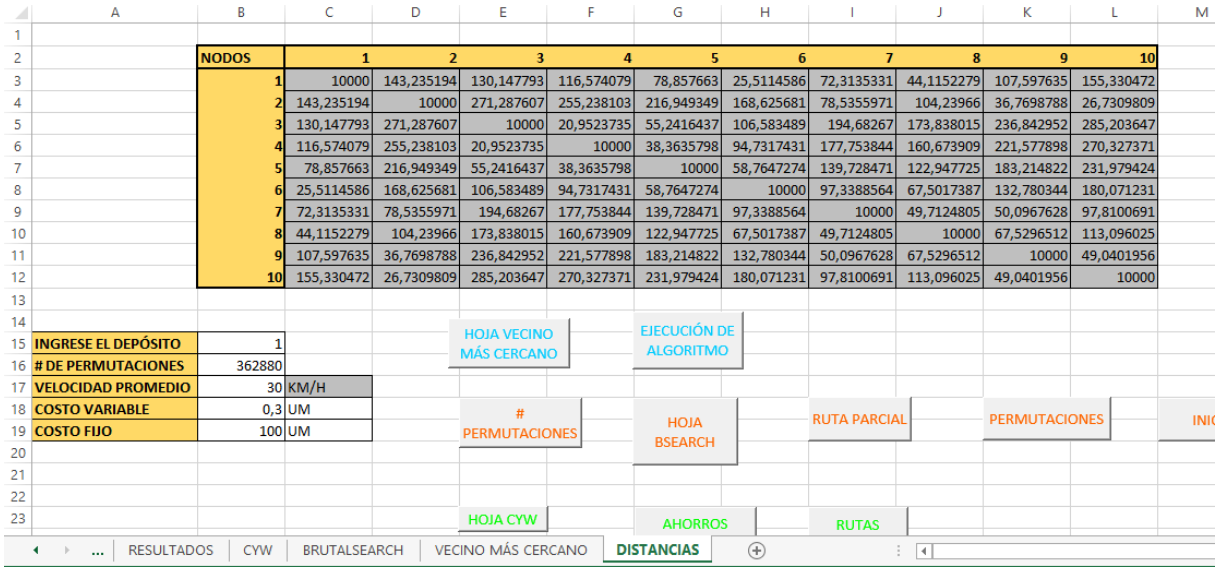
COORDENADAS DISTANCIAS

Figura 17. Ingreso de coordenadas (Módulo didáctico).

Como se puede ver en la figura 17, la tabla de las coordenadas se encuentra en una hoja independiente y se ingresan los datos de latitud y longitud de cada punto, los cuales se pueden obtener a través de plataformas en internet como mapas virtuales. Adicionalmente también se debe digitar el nombre del lugar.

4.1.4.2 Distancias

La siguiente pestaña contiene la matriz de distancias calculadas a partir de los datos previamente ingresados, además muestra algunos campos con los que se podrá obtener información adicional de cada algoritmo.



NODOS	1	2	3	4	5	6	7	8	9	10
1	10000	143,235194	130,147793	116,574079	78,857663	25,5114586	72,3135331	44,1152279	107,597635	155,330472
2	143,235194	10000	271,287607	255,238103	216,949349	168,625681	78,5355971	104,23966	36,7698788	26,7309809
3	130,147793	271,287607	10000	20,9523735	55,2416437	106,583489	194,68267	173,838015	236,842952	285,203647
4	116,574079	255,238103	20,9523735	10000	38,3635798	94,7317431	177,753844	160,673909	221,577898	270,327371
5	78,857663	216,949349	55,2416437	38,3635798	10000	58,7647274	139,728471	122,947725	183,214822	231,979424
6	25,5114586	168,625681	106,583489	94,7317431	58,7647274	10000	97,3388564	67,5017387	132,780344	180,071231
7	72,3135331	78,5355971	194,68267	177,753844	139,728471	97,3388564	10000	49,7124805	50,0967628	97,8100691
8	44,1152279	104,23966	173,838015	160,673909	122,947725	67,5017387	49,7124805	10000	67,5296512	113,096025
9	107,597635	36,7698788	236,842952	221,577898	183,214822	132,780344	50,0967628	67,5296512	10000	49,0401956
10	155,330472	26,7309809	285,203647	270,327371	231,979424	180,071231	97,8100691	113,096025	49,0401956	10000

INGRESE EL DEPÓSITO	1	
# DE PERMUTACIONES	362880	
VELOCIDAD PROMEDIO	30	KM/H
COSTO VARIABLE	0,3	UM
COSTO FIJO	100	UM

Figura 18. Matriz de distancias (Módulo didáctico).

Como se puede ver en la figura 18, se tiene las distancias entre cada par de nodos, las cuales se han calculado con el uso de la fórmula de Harvesine, la cual es muy utilizada para calcular las distancias a recorrer en viajes marítimos y considera una superficie esférica utilizando el radio de la Tierra para realizar el cálculo y trazar rutas directas entre dos puntos. Esta fórmula está expresada de la siguiente forma:

$$d = 2r \arcsin\left(\sqrt{sin^2\left(\frac{\emptyset_2 - \emptyset_1}{2}\right) + \cos(\emptyset_1)\cos(\emptyset_2)\, sin^2\left(\frac{\lambda_2 - \lambda_1}{2}\right)}\right)$$

En donde:

r = radio de la esfera.

$\emptyset$ = latitud.

λ = longitud (Mangesh, 2013).

Por ejemplo:

$$P1 = (\emptyset 1 = -78{,}106, \lambda 1 = 0{,}365)\ Ibarra$$

$$P2 = (\emptyset 2 = -78{,}388, \lambda 2 = -0{,}213)\ Tumbaco$$

$$r = 6371\ km$$

$$d = 2 * 6371 * \arcsin\left(\sqrt{sin^2\left(\frac{(-78{,}388) - (-78{,}106)}{2}\right) + \cos(-78{,}106)\cos(-78{,}388)\, sin^2\left(\frac{(-0{,}213) - (0{,}365)}{2}\right)}\right)$$

$$d(Ibarra - Tumbaco) = 143{,}2\ km$$

4.1.4.3 Datos adicionales

Se requiere definir los siguientes parámetros:

- El nodo que representará el depósito
- El número de permutaciones que se calculará después con el uso de los botones que se observan bajo la matriz de distancias
- La velocidad promedio de circulación del vehículo, que permite calcular el tiempo de viaje
- El costo por kilómetro recorrido (costo variable)
- El costo fijo

Una vez realizado el proceso anterior, se debe establecer el método de solución a utilizar y esto dependerá de las necesidades del usuario. Para la demostración del funcionamiento del programa se utilizarán todos los métodos programados y se consideran los siguientes datos:

INGRESE EL DEPÓSITO	1	
# DE PERMUTACIONES	362880	
VELOCIDAD PROMEDIO	30	KM/H
COSTO VARIABLE	0,3	UM
COSTO FIJO	100	UM

Figura 19. Tabla de datos (Módulo didáctico).

4.1.4.4 Algoritmo del vecino más cercano

Cada uno de los algoritmos cuenta con una hoja de trabajo independiente en la cual se cargan algunos parámetros que han sido ya programados y que facilitan la interpretación de la respuesta además de los tiempos de ejecución de cada proceso que integra el algoritmo. En la figura 19 se muestran los datos necesarios para hacer el cálculo del tiempo de viaje y también el costo total de cada ruta en todos los métodos.

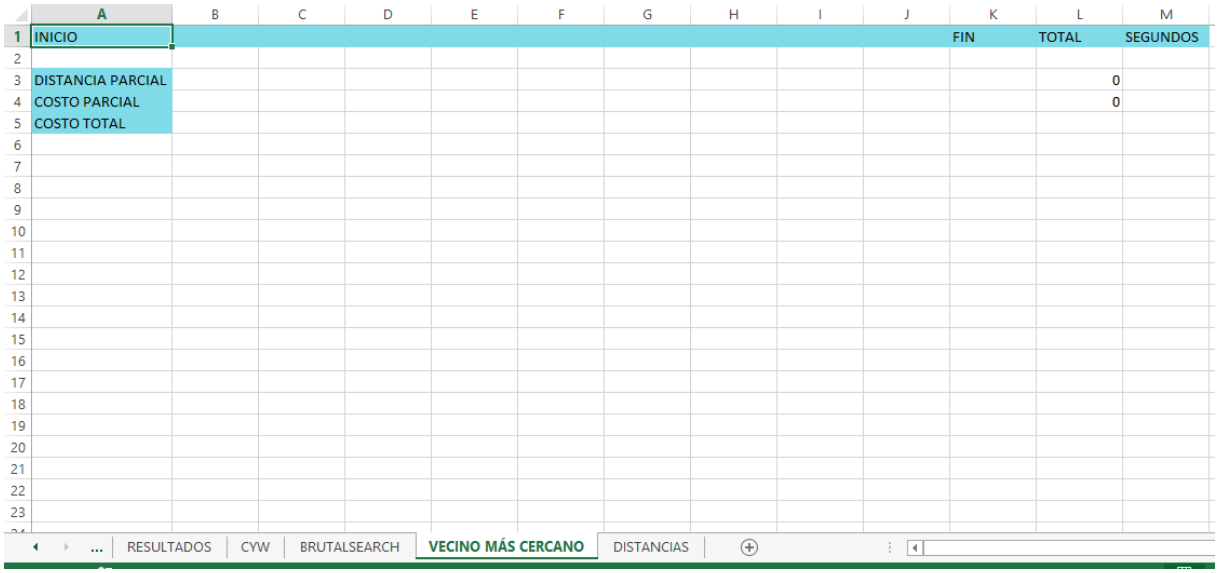


Figura 20. Hoja de solución del algoritmo vecino más cercano (Módulo didáctico).

Como se puede ver en la figura 20, se han cargado, el inicio y final de la ruta a calcularse, además del espacio para la suma de distancias y costos ya antes definidos.

Luego de la generación de la hoja de trabajo realizada con el botón "HOJA VECINO MÁS CERCANO", se prosigue a la ejecución del algoritmo del vecino más cercano y se lo puede ejecutar desde el botón denominado "EJECUCIÓN DEL ALGORITMO".

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	INICIO										FIN	TOTAL	SEGUNDOS
2	1	6	5	4	3	8	7	9	2	10	1		0,203125
3	DISTANCIA PARCIAL	25,5114586	58,7647274	38,3635798	20,9523735	173,838015	49,7124805	50,0967628	36,7698788	26,7309809	155,330472	636,070729	
4	COSTO PARCIAL	7,65343759	17,6294182	11,5090739	6,28571204	52,1514044	14,9137442	15,0290288	11,0309636	8,01929427	46,5991415	190,821219	
5	COSTO TOTAL	290,821219											
6													

Figura 21. Solución del algoritmo vecino más cercano (Módulo didáctico).

Como se puede ver en la figura 21, este algoritmo muestra una ruta obtenida que podría ser o no la óptima del problema, ya que este método es una heurística y podría arrojar varias respuestas válidas para el problema. Este método funciona buscando el nodo más cercano al nodo previo, como su nombre lo dice a su vecino más cercano, y los conecta formando un ciclo que parte del nodo definido como depósito y vuelve al mismo.

En esta hoja también se pueden ver las distancias entre cada par de nodos del resultado y su costo, obteniendo la sumatoria final, y también el costo total que es la suma del costo total parcial más el costo fijo antes definido.

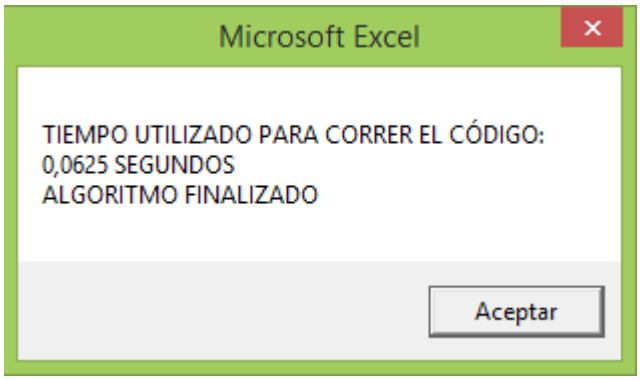


Figura 22. Notificación de finalización del algoritmo y tiempo de ejecución (Módulo didáctico).

El tiempo de cómputo para el algoritmo del vecino más cercano se encuentra en un rango óptimo, esto quiere decir que el código se ejecuta y obtiene la respuesta en menos de un segundo, aunque esto depende del equipo que se esté utilizando. En este caso, la descripción analítica del módulo ha sido desarrollada en una computadora con procesador *Intel(R) Celeron(R) CPU N2840 @ 2,16GHz*, por lo tanto todos los tiempos mostrados son con base en esta máquina. Al finalizar cada algoritmo se muestra el mensaje de la figura 22, en la pestaña de distancias que es donde se encuentran todos los datos que se consideran para cada método.

4.2.4.5 Algoritmo *brutal search*

Se inicia generando el número de permutaciones o cambios que se pueden generar, y como ya se ha mencionado anteriormente, si el número de nodos es muy grande el número de permutaciones crece exponencialmente, lo cual requiere de mayor uso de recursos computacionales y mayor tiempo de ejecución. Para el caso planteado de 10 nodos se obtiene el siguiente número de permutaciones.

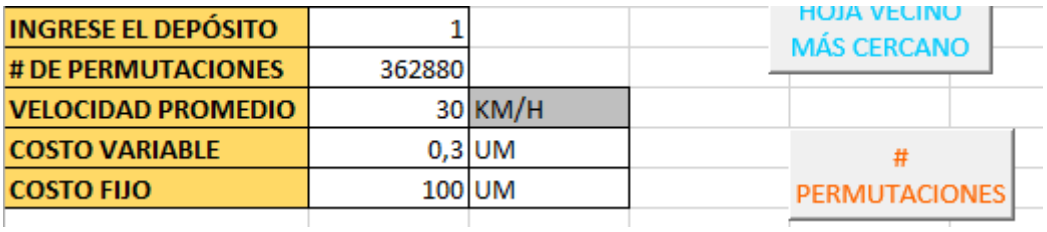


Figura 23. Cálculo del número de permutaciones (Módulo didáctico).

El número de permutaciones se obtiene al presionar el botón "# PERMUTACIONES".

Luego, al igual que el anterior algoritmo se genera la hoja de trabajo en donde se puede tener dos opciones, si no se ha realizado previamente el algoritmo del vecino más cercano podría

digitar el número de los nodos sin tomar en cuenta el depósito para que se generen las permutaciones o si ya se tiene la respuesta previa se la puede cargar y utilizarla para la ejecución de este algoritmo.

El botón denominado “RUTA PARCIAL” carga la respuesta obtenida en el algoritmo anterior y se generan las permutaciones a partir de ella.

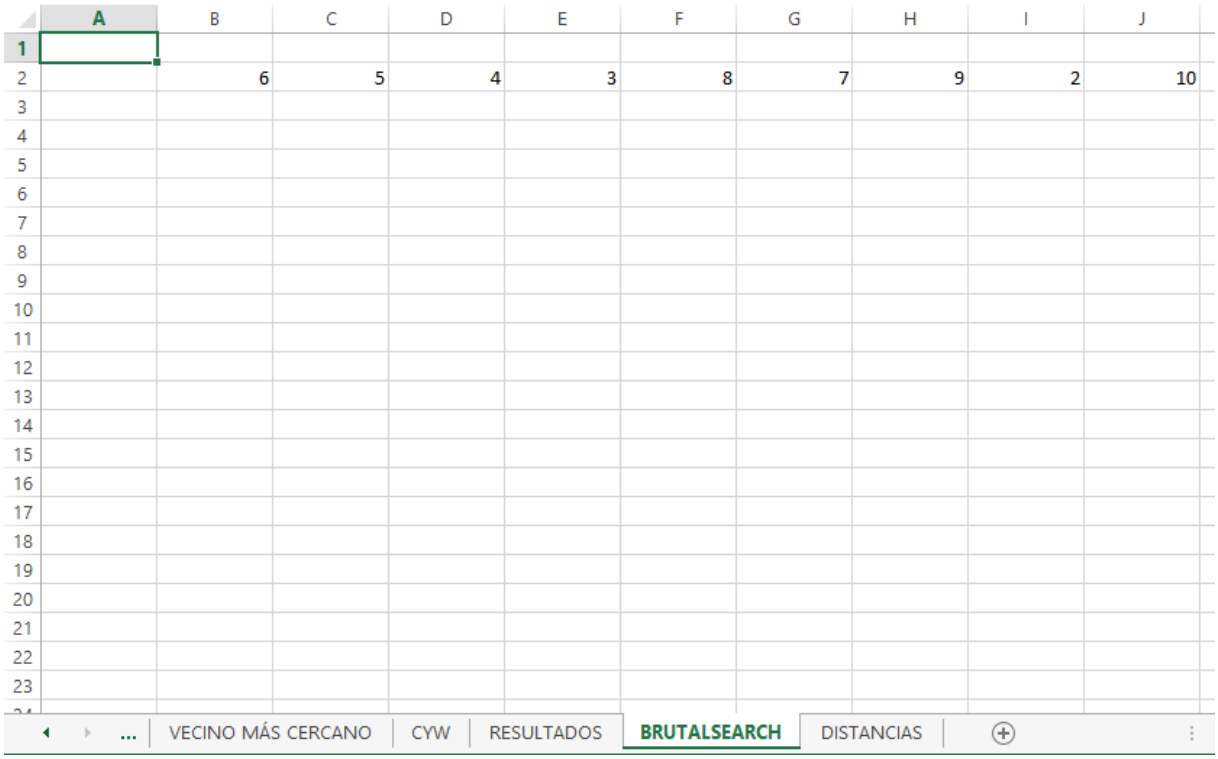


Figura 24. Se carga la ruta antes encontrada en el algoritmo del vecino más cercano (Módulo didáctico).

Como se observa en la figura 24, se ha cargado la respuesta anterior, pero como se mencionó también se podría digitar una ruta aleatoria para generar las permutaciones, teniendo en cuenta la posición en la que están los nodos dentro de las celdas y sin tomar en cuenta el depósito.

La generación de permutaciones es el proceso que más tiempo requiere en el módulo ya que es un método exacto, es decir, que busca la solución óptima del problema. Al finalizar este proceso se obtiene una ventana emergente con el número de permutaciones que se han generado y la hoja de trabajo muestra todas las rutas encontradas.

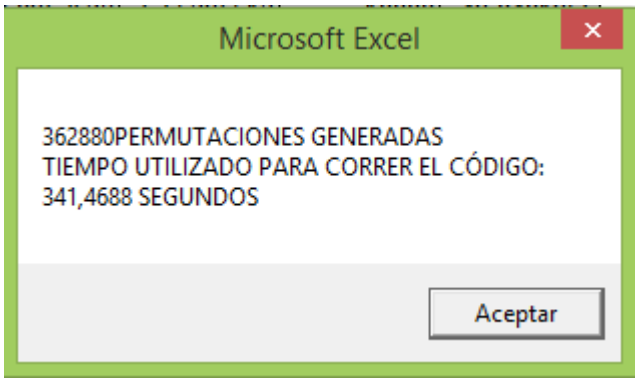


Figura 25. Notificación de la generación de permutaciones y tiempo de ejecución (Módulo didáctico).

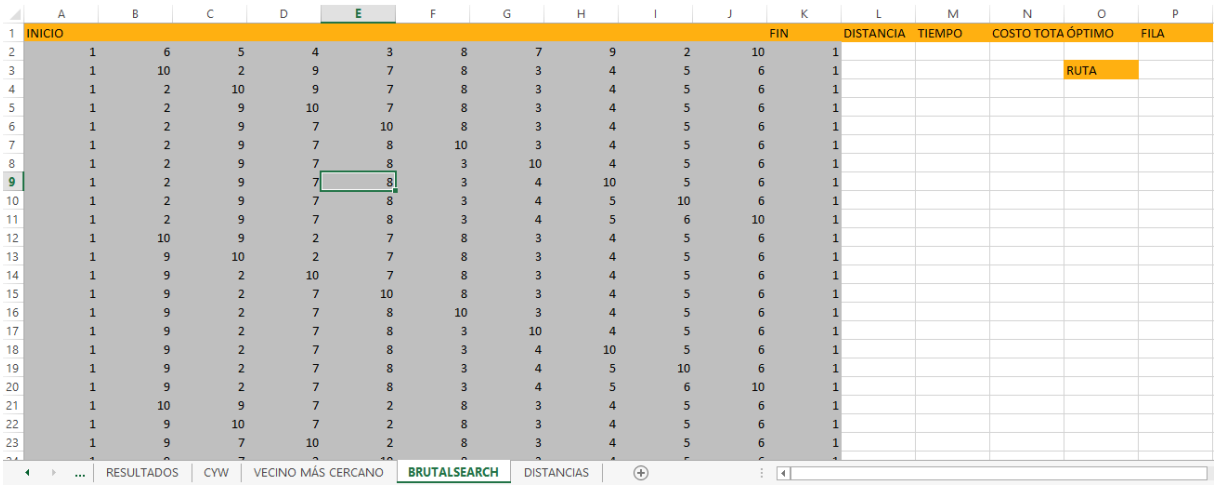
Para el código que genera todas las permutaciones, el tiempo de cómputo sobrepasa los 60 segundos lo cual significa que es deficiente. El número de permutaciones debe ser igual al calculado anteriormente, si no es así será necesario revisar el código ya que puede haber surgido un problema en las variables y parámetros definidos.

	A	B	C	D	E	F	G	H	I	J
1										
2		6	5	4	3	8	7	9	2	10
3		10	2	9	7	8	3	4	5	6
4		2	10	9	7	8	3	4	5	6
5		2	9	10	7	8	3	4	5	6
6		2	9	7	10	8	3	4	5	6
7		2	9	7	8	10	3	4	5	6
8		2	9	7	8	3	10	4	5	6
9		2	9	7	8	3	4	10	5	6
10		2	9	7	8	3	4	5	10	6
11		2	9	7	8	3	4	5	6	10
12		10	9	2	7	8	3	4	5	6
13		9	10	2	7	8	3	4	5	6
14		9	2	10	7	8	3	4	5	6
15		9	2	7	10	8	3	4	5	6
16		9	2	7	8	10	3	4	5	6
17		9	2	7	8	3	10	4	5	6
18		9	2	7	8	3	4	10	5	6
19		9	2	7	8	3	4	5	10	6
20		9	2	7	8	3	4	5	6	10
21		10	9	7	2	8	3	4	5	6
22		9	10	7	2	8	3	4	5	6
23		9	7	10	2	8	3	4	5	6

Figura 26. Permutaciones generadas (Módulo didáctico).

La hoja de trabajo contiene las rutas encontradas y como se puede ver en la figura 26, cada solución es diferente, de esta manera este algoritmo comprueba todas las posibles soluciones para encontrar la que ya no se pueda mejorar, es decir, la óptima.

Para añadir el inicio y fin de cada ruta se presiona el botón “INICIO_FIN”, en la pestaña “DISTANCIAS”.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	INICIO										FIN	DISTANCIA	TIEMPO	COSTO TOTA	ÓPTIMO	FILA
2	1	6	5	4	3	8	7	9	2	10	1					
3	1	10	2	9	7	8	3	4	5	6	1				RUTA	
4	1	2	10	9	7	8	3	4	5	6	1					
5	1	2	9	10	7	8	3	4	5	6	1					
6	1	2	9	7	10	8	3	4	5	6	1					
7	1	2	9	7	8	10	3	4	5	6	1					
8	1	2	9	7	8	3	10	4	5	6	1					
9	1	2	9	7	8	3	4	10	5	6	1					
10	1	2	9	7	8	3	4	5	10	6	1					
11	1	2	9	7	8	3	4	5	6	10	1					
12	1	10	9	2	7	8	3	4	5	6	1					
13	1	9	10	2	7	8	3	4	5	6	1					
14	1	9	2	10	7	8	3	4	5	6	1					
15	1	9	2	7	10	8	3	4	5	6	1					
16	1	9	2	7	8	10	3	4	5	6	1					
17	1	9	2	7	8	3	10	4	5	6	1					
18	1	9	2	7	8	3	4	10	5	6	1					
19	1	9	2	7	8	3	4	5	10	6	1					
20	1	9	2	7	8	3	4	5	6	10	1					
21	1	10	9	7	2	8	3	4	5	6	1					
22	1	9	10	7	2	8	3	4	5	6	1					
23	1	9	7	10	2	8	3	4	5	6	1					

RESULTADOS | CYW | VECINO MÁS CERCANO | BRUTALSEARCH | DISTANCIAS

Figura 27. Completado de rutas con inicio y fin y definición de celdas para el resultado (Módulo didáctico).

Como se puede apreciar en la figura 31, se han añadido los campos antes mencionados, además de algunos adicionales los cuales se calculan presionando el botón “CICLOS” y también se obtendrá la ruta óptima, aunque es importante mencionar que dentro de VBA solo se admiten aproximadamente 30000 cálculos del total de número de permutaciones generadas, de esta manera la respuesta mínima que se obtenga podría ser optimizada aún más si se prueban las demás rutas, este proceso requiere de un ligero ajuste en el código de programación para que el cálculo continúe la secuencia con los siguientes 30000 datos y así sucesivamente hasta generar todas las respuestas.

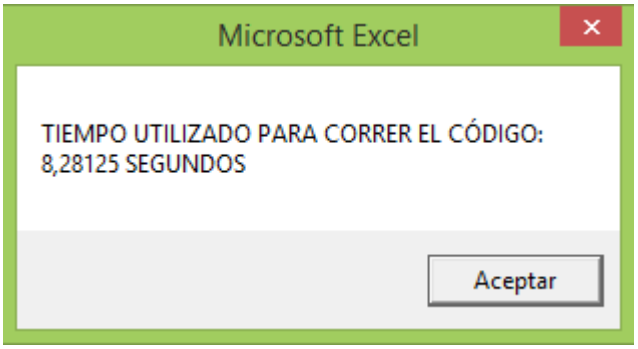
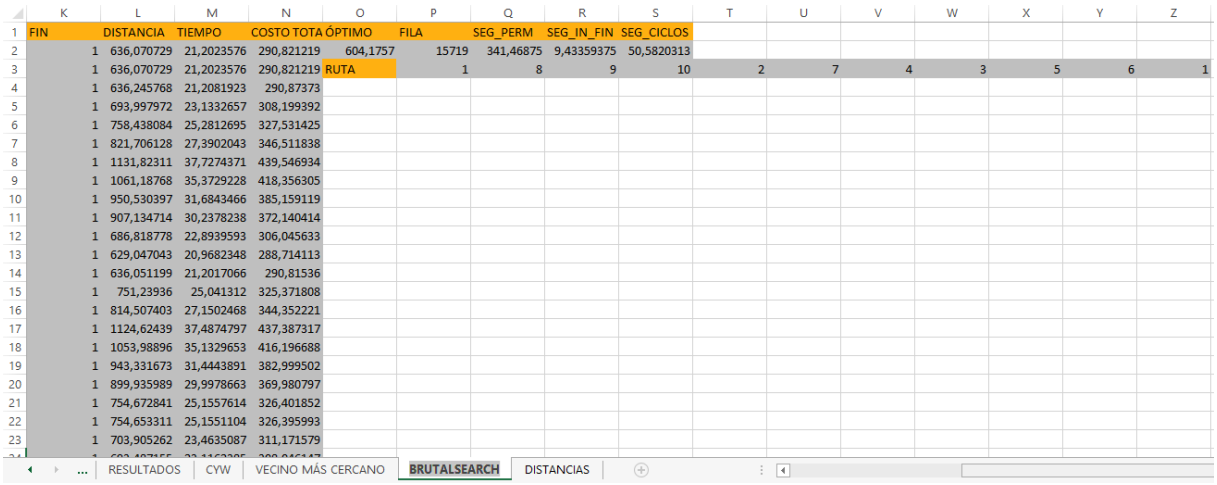


Figura 28. Tiempo de ejecución para la subrutina "INICIO_FIN" (Módulo didáctico).

El tiempo mostrado en la figura 28 está dentro de un rango aceptable, ya que se encuentra entre uno y diez segundos.



	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
1	FIN	DISTANCIA	TIEMPO	COSTO TOTA	ÓPTIMO	FILA	SEG_PERM	SEG_IN_FIN	SEG_CICLOS							
2	1	636,070729	21,2023576	290,821219	604,1757	15719	341,46875	9,43359375	50,5820313							
3	1	636,070729	21,2023576	290,821219	RUTA	1	8	9	10	2	7	4	3	5	6	1
4	1	636,245768	21,2081923	290,87373												
5	1	693,997972	23,1332657	308,199392												
6	1	758,438084	25,2812695	327,531425												
7	1	821,706128	27,3902043	346,511838												
8	1	1131,82311	37,7274371	439,546934												
9	1	1061,18768	35,3729228	418,356305												
10	1	950,530397	31,6843466	385,159119												
11	1	907,134714	30,2378238	372,140414												
12	1	686,818778	22,8939593	306,045633												
13	1	629,047043	20,9682348	288,714113												
14	1	636,051199	21,2017066	290,81536												
15	1	751,23936	25,041312	325,371808												
16	1	814,507403	27,1502468	344,352221												
17	1	1124,62439	37,4874797	437,387317												
18	1	1053,98896	35,1329653	416,196688												
19	1	943,331673	31,4443891	382,999502												
20	1	899,935989	29,9978663	369,980797												
21	1	754,672841	25,1557614	326,401852												
22	1	754,653311	25,1551104	326,395993												
23	1	703,905262	23,4635087	311,171579												

RESULTADOS | CYW | VECINO MÁS CERCANO | BRUTALSEARCH | DISTANCIAS

Figura 29. Cálculo de distancias, tiempos, costos y ruta óptima (Módulo didáctico).

Las distancias y tiempos de viaje para cada opción han sido calculados y también se ha obtenido la ruta óptima de los primeros ciclos, como se observa en la figura 29. Es importante tener en cuenta que podría encontrarse una mejor respuesta y de esta manera el algoritmo finaliza.

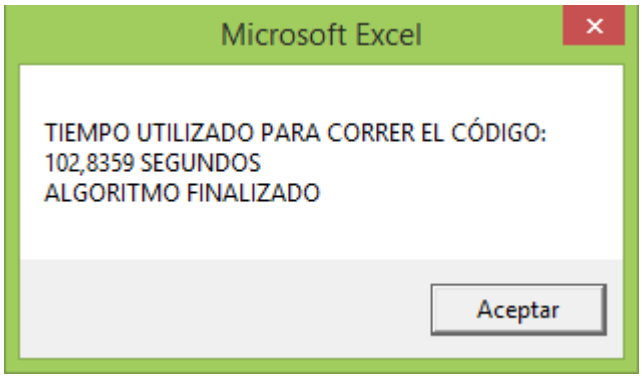


Figura 30. Tiempo de ejecución de la subrutina "CICLOS" (Módulo didáctico).

Para el proceso del cálculo de distancias para cada ciclo y la búsqueda de la ruta óptima, el tiempo de ejecución del código es mayor a 60 segundos, por lo tanto, también es deficiente.

4.1.4.6 Algoritmo de Clarke y Wright

El funcionamiento de este algoritmo inicia con el establecimiento de rutas parciales desde cada nodo hasta el depósito de ida y vuelta y se trata de comprobar qué rutas se podría combinar de tal manera que se pueda recorrer un par de nodos en menor distancia que la que se obtuvo en la solución inicial. Dentro del programa se toma en cuenta que si el ahorro calculado en la matriz es positivo entonces la ruta entre ese par de nodos puede ser una posible solución al problema. Este algoritmo resultará en varias rutas las cuales representarán los mayores ahorros.

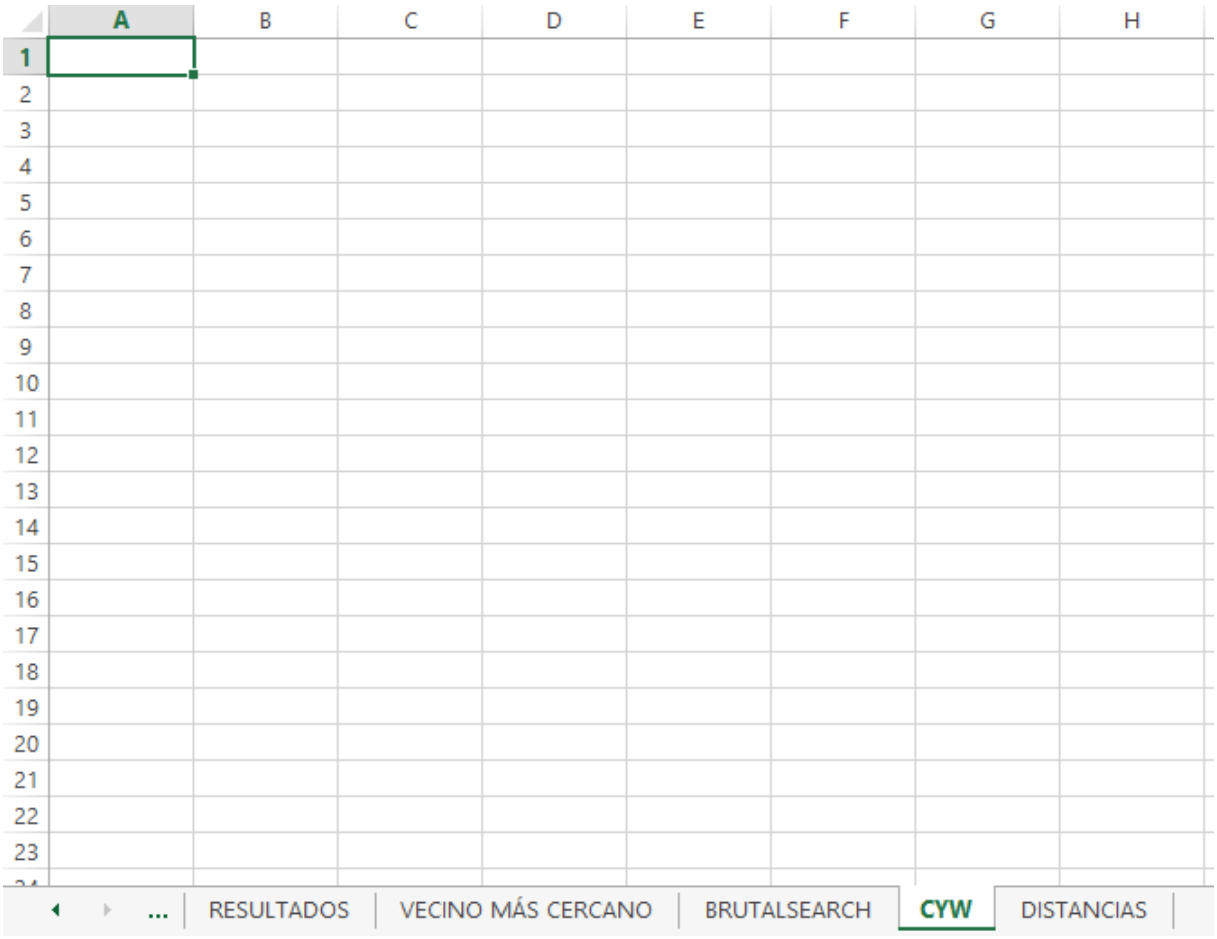
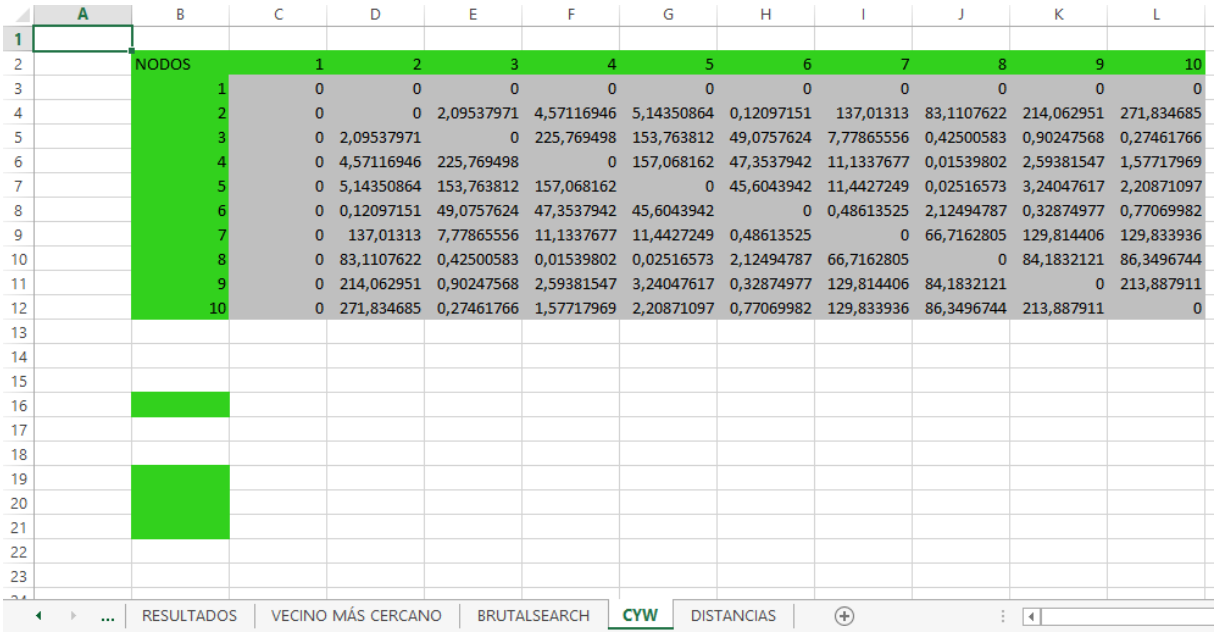


Figura 31. Hoja para resolución del algoritmo de ahorros (Módulo didáctico).

Se genera la hoja de trabajo denominada “CYW”.

Para generar la matriz de ahorros se procede al utilizar el siguiente botón denominado “AHORROS”.



NODOS	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	2,09537971	4,57116946	5,14350864	0,12097151	137,01313	83,1107622	214,062951	271,834685
3	0	2,09537971	0	225,769498	153,763812	49,0757624	7,77865556	0,42500583	0,90247568	0,27461766
4	0	4,57116946	225,769498	0	157,068162	47,3537942	11,1337677	0,01539802	2,59381547	1,57717969
5	0	5,14350864	153,763812	157,068162	0	45,6043942	11,4427249	0,02516573	3,24047617	2,20871097
6	0	0,12097151	49,0757624	47,3537942	45,6043942	0	0,48613525	2,12494787	0,32874977	0,77069982
7	0	137,01313	7,77865556	11,1337677	11,4427249	0,48613525	0	66,7162805	129,814406	129,833936
8	0	83,1107622	0,42500583	0,01539802	0,02516573	2,12494787	66,7162805	0	84,1832121	86,3496744
9	0	214,062951	0,90247568	2,59381547	3,24047617	0,32874977	129,814406	84,1832121	0	213,887911
10	0	271,834685	0,27461766	1,57717969	2,20871097	0,77069982	129,833936	86,3496744	213,887911	0

Figura 32. Matriz de ahorros (Módulo didáctico).

Los ahorros entre cada par de nodos se calculan con la formula $S = di + dj - dij$, en donde la letra i representa el nodo inicial y j el final, y d es la distancia tomada desde la matriz de distancias. En la figura 32 se aprecia el cálculo de todos los ahorros generados entre cada par de nodos.

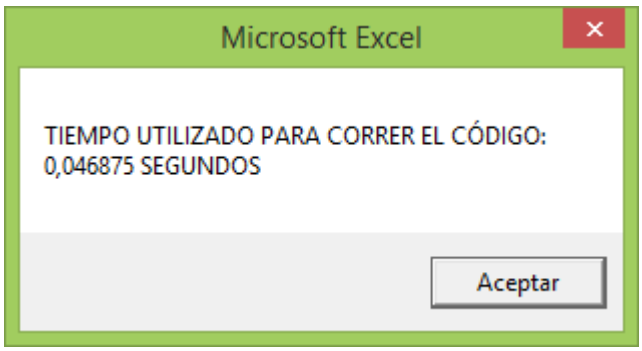
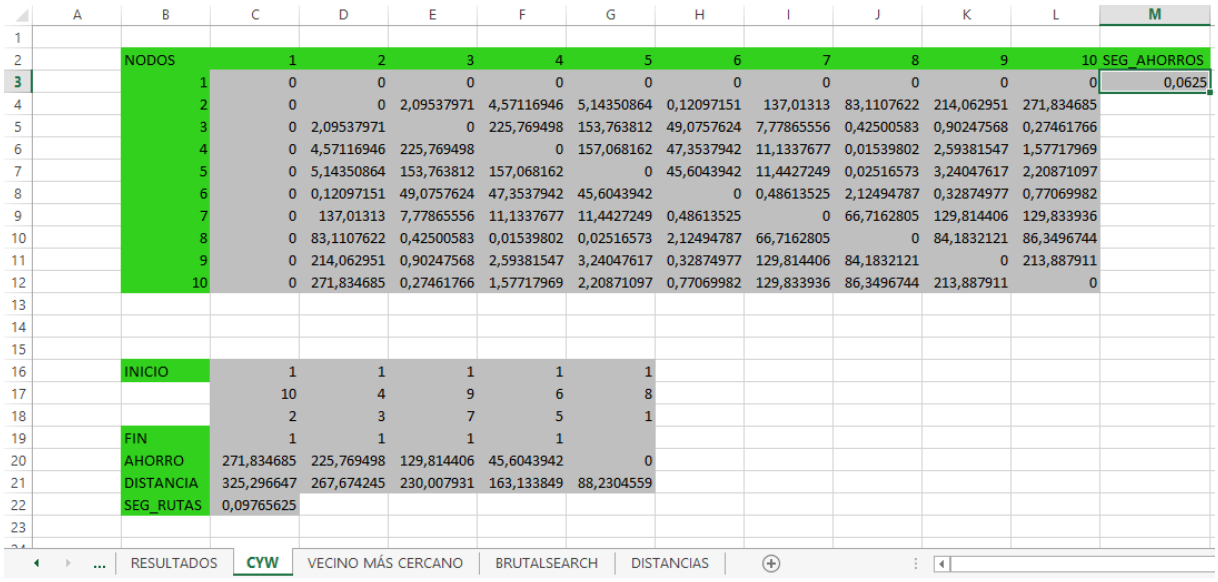


Figura 33. Tiempo de cálculo de ahorros (Módulo didáctico).

Como se puede ver en la figura 33, para el cálculo de ahorros se ha requerido un tiempo de ejecución del código menor a un segundo, lo que está dentro de un rango óptimo.



	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2		NODOS	1	2	3	4	5	6	7	8	9	10	SEG_AHORROS
3		1	0	0	0	0	0	0	0	0	0	0	0,0625
4		2	0	0	2,09537971	4,57116946	5,14350864	0,12097151	137,01313	83,1107622	214,062951	271,834685	
5		3	0	2,09537971	0	225,769498	153,763812	49,0757624	7,77865556	0,42500583	0,90247568	0,27461766	
6		4	0	4,57116946	225,769498	0	157,068162	47,3537942	11,1337677	0,01539802	2,59381547	1,57717969	
7		5	0	5,14350864	153,763812	157,068162	0	45,6043942	11,4427249	0,02516573	3,24047617	2,20871097	
8		6	0	0,12097151	49,0757624	47,3537942	45,6043942	0	0,48613525	2,12494787	0,32874977	0,77069982	
9		7	0	137,01313	7,77865556	11,1337677	11,4427249	0,48613525	0	66,7162805	129,814406	129,833936	
10		8	0	83,1107622	0,42500583	0,01539802	0,02516573	2,12494787	66,7162805	0	84,1832121	86,3496744	
11		9	0	214,062951	0,90247568	2,59381547	3,24047617	0,32874977	129,814406	84,1832121	0	213,887911	
12		10	0	271,834685	0,27461766	1,57717969	2,20871097	0,77069982	129,833936	86,3496744	213,887911	0	
13													
14													
15													
16		INICIO	1	1	1	1	1						
17			10	4	9	6	8						
18			2	3	7	5	1						
19		FIN	1	1	1	1							
20		AHORRO	271,834685	225,769498	129,814406	45,6043942	0						
21		DISTANCIA	325,296647	267,674245	230,007931	163,133849	88,2304559						
22		SEG_RUTAS	0,09765625										
23													

Figura 34. Generación de rutas (Módulo didáctico).

La solución se genera presionando el botón “RUTAS”, y de esta manera se imprimen los pares de nodos, como se muestra en la figura 34, donde se ha obtenido el mayor ahorro teniendo en cuenta que cada ruta encontrada inicia y termina en el depósito, y al final se muestra la distancia de cada recorrido. De esta manera el algoritmo finaliza.

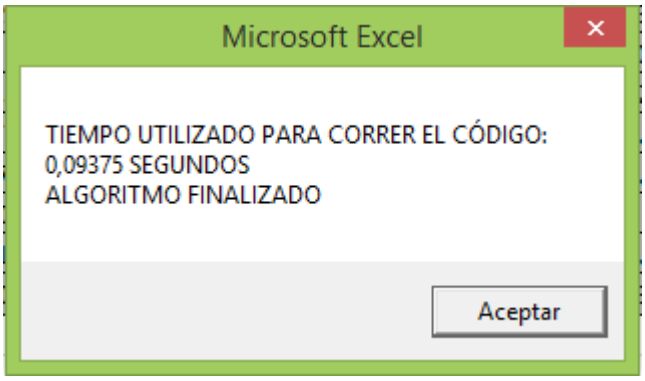
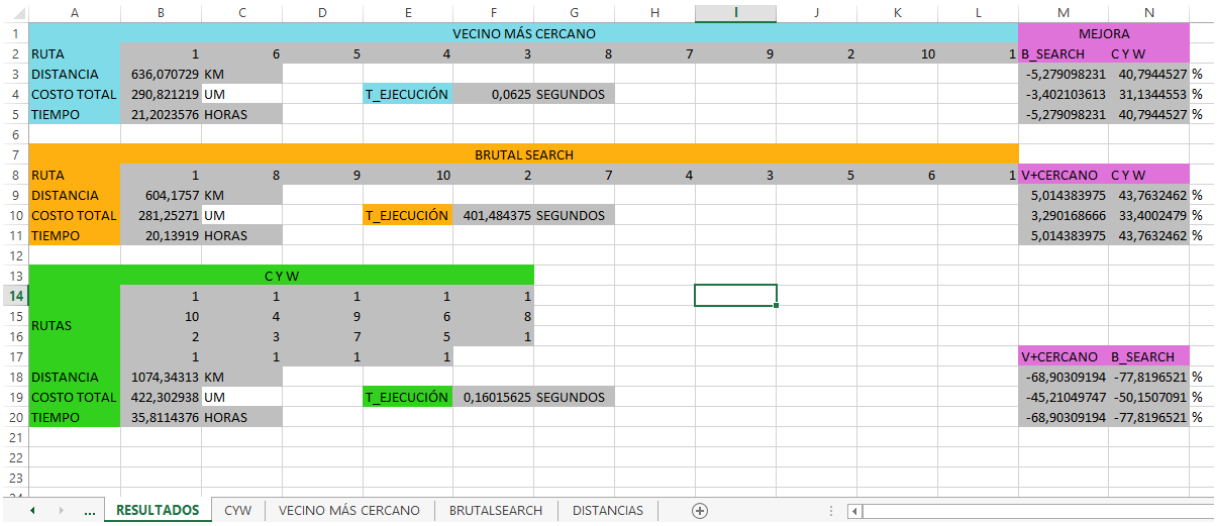


Figura 35. Tiempo para la determinación de las rutas (Módulo didáctico).

Para la identificación de las rutas posibles, el código de programación se ejecuta en un tiempo óptimo, ya que este proceso requiere menos de un segundo, como se observa en la figura 35.

4.1.4.7 Comparación de resultados

En la última parte del módulo didáctico se tiene una comparación de resultados en una hoja que se crea con el mismo procedimiento de los algoritmos. Primero se genera la hoja de resultados y luego cada botón carga el resultado del algoritmo que se ha descrito con su nombre.



1	VECINO MÁS CERCANO												MEJORA		
2	RUTA	1	6	5	4	3	8	7	9	2	10	1	B_SEARCH	C Y W	
3	DISTANCIA	636,070729	KM										-5,279098231	40,7944527	%
4	COSTO TOTAL	290,821219	UM		T_EJECUCIÓN	0,0625	SEGUNDOS						-3,402103613	31,1344553	%
5	TIEMPO	21,2023576	HORAS										-5,279098231	40,7944527	%
6															
7	BRUTAL SEARCH														
8	RUTA	1	8	9	10	2	7	4	3	5	6	1	V+CERCANO	C Y W	
9	DISTANCIA	604,1757	KM										5,014383975	43,7632462	%
10	COSTO TOTAL	281,25271	UM		T_EJECUCIÓN	401,484375	SEGUNDOS						3,290168666	33,4002479	%
11	TIEMPO	20,13919	HORAS										5,014383975	43,7632462	%
12															
13	C Y W														
14	RUTAS	1	1	1	1	1									
15		10	4	9	6	8									
16		2	3	7	5	1									
17		1	1	1	1								V+CERCANO	B_SEARCH	
18	DISTANCIA	1074,34313	KM										-68,90309194	-77,8196521	%
19	COSTO TOTAL	422,302938	UM		T_EJECUCIÓN	0,16015625	SEGUNDOS						-45,21049747	-50,1507091	%
20	TIEMPO	35,8114376	HORAS										-68,90309194	-77,8196521	%
21															
22															
23															

... RESULTADOS | CYW | VECINO MÁS CERCANO | BRUTALSEARCH | DISTANCIAS

Figura 36. Hoja de resultados (Módulo didáctico).

Se muestran las rutas que se generan con cada algoritmo, además de la distancia total del recorrido, el tiempo de viaje y el costo que se expresa en unidades monetarias (UM) para que sea aplicable a cualquier tipo de moneda.

Las soluciones que se obtengan en un problema se pueden comparar y analizar de mejor manera en la hoja de resultados, ya que es un resumen de la aplicación de los algoritmos y esto puede contribuir a la toma de decisiones en caso de ser un problema real, además de incluir el cálculo del porcentaje de mejora para cada algoritmo con respecto a los otros, en donde, un valor negativo quiere decir que el resultado del algoritmo ha aumentado con respecto a los otros y un valor positivo muestra que si hay reducción en los resultados. En este caso, al ser una aplicación didáctica, se puede notar la optimización de un algoritmo a otro y también analizar más opciones que puedan considerar mayor cantidad de vehículos, capacidad, distancia máxima, entre otros, que pueden irse añadiendo por parte del usuario ingresando al código de programación y modificándolo para considerar las restricciones que crea necesarias.

Como se muestra en la figura 36, cada algoritmo tiene un tiempo de ejecución total, que es la suma de los tiempos de todos los procesos antes mencionados. El algoritmo del vecino más cercano tiene un tiempo de ejecución total óptimo, ya que este método proporciona una respuesta en menos de un segundo. Para el método *brutal search*, se muestra un tiempo de ejecución total deficiente, que sobrepasa los 60 segundos y el método de Clarke and Wright de igual forma, tiene un tiempo de ejecución total óptimo, ya que es menor a un segundo. A partir de los tiempos antes mostrados se puede identificar que las heurísticas requieren tiempos

menores para su ejecución y el método exacto, al proporcionar la solución óptima el problema aumenta el tiempo de ejecución exponencialmente ya que prueba todas las opciones posibles.

Para finalizar la ejecución del módulo es importante definir la mejor opción y se da en dos variables, la que proporciona la mejor solución operativa y la que tiene menor tiempo de ejecución, esto se obtiene al presionar el último botón denominado “SOLUCIÓN”

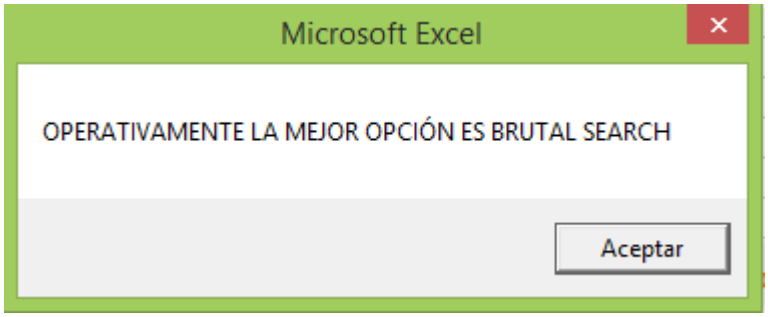


Figura 37. Mejor opción operativa (Módulo didáctico).

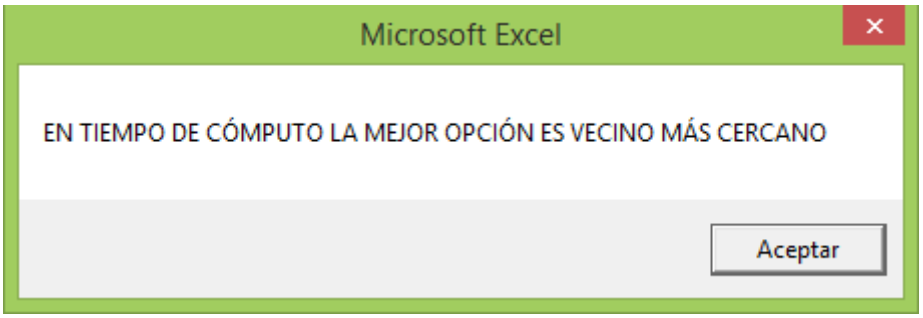


Figura 38. Mejor opción en tiempo de ejecución (Módulo didáctico).

Aunque el resultado puede ser evidente a simple vista es necesario comprobar esta respuesta con el botón solución para estar seguro de la elección que se realice. Además observando la sección de mejora de los algoritmos se podría decidir entre optimizar los recursos operativos o los computacionales, pero esto depende del juicio del usuario.

4.1.5. Análisis de resultados de cada algoritmo

Para el desarrollo del módulo y los experimentos realizados en un escenario ficticio con la matriz de distancias reales se ha utilizado una computadora con un procesador *AMD-FX 8370 Eight-Core Processor 4GHz,* la cual funciona muy bien para las heurísticas teniendo tiempos de ejecución muy bajos con los datos de 10 nodos utilizados. El uso de heurísticas es más frecuente para la resolución de problemas reales ya que proporcionan una respuesta válida que puede no ser la mejor pero que es adecuada al problema en cuestión. Para fines didácticos y de

experimentación, en el presente trabajo se ha tomado en cuenta un método exacto que consiste en la generación de permutaciones la cuales han sido realizadas con un algoritmo ya desarrollado denominado inserta y empuja que trabajó con un máximo de nueve nodos para realizar las permutaciones, ya que para un problema con más nodos el tiempo de ejecución del algoritmo, aumentaría exponencialmente. El tiempo promedio de ejecución para los nueve nodos es de dos minutos, la siguiente parte del algoritmo consiste en la sumatoria de distancias para las nuevas rutas encontradas pero se tiene una limitante en el programa ya que la variable a utilizar denominada fila se declara como entera (*integer*) y esta solo admite un máximo de 30000 filas a calcular y esto representa aproximadamente el 9% del total de permutaciones generadas y el tiempo de ejecución es 30 segundos. Una de las opciones para continuar probando todas las nuevas rutas es seguir con la serie desde 30001 hasta 60001 y así hasta poder terminar con todas las permutaciones y de esta manera también comprobar la mejor respuesta para el problema. Para el procesador es mucho mejor ejecutar el algoritmo por partes ya que se intentó hacer el cálculo para el total de permutaciones y surgió un problema de desbordamiento dentro de la aplicación con lo cual el computador colapsó en el proceso, y tuvo que ser reiniciado para continuar con el trabajo. Por esta razón, en problemas reales, se trabaja con heurísticas y meta heurísticas ya que el número de nodos es mucho más grande que el que se ha considerado en este trabajo, y de igual forma se obtiene una respuesta que se acerca a la óptima del problema y que funciona muy bien para la empresa y reduce sus costos de distribución.

Para cada VRP planteado existe un método definido que optimiza la respuesta y esto depende del parámetro que se busque optimizar como costo, distancia, tiempo entre otros, que se establecen según la necesidad del programador o de la empresa, además que se toma en consideración todas las restricciones del caso aplicadas a un problema real, de esta manera se asegura que la respuesta sea la que se necesita.

Para el desarrollo de este trabajo se toma en cuenta una matriz de distancias con fines didácticos, es decir, que el escenario es ficticio pero las distancias entre los nodos son reales. Para el primer caso en la ejecución de la heurística del vecino más cercano se puede obtiene una ruta parcial cuya distancia total de recorrido es 636,07 km. Hasta el momento se cree que esta es la mejor ruta que se puede encontrar ya que el módulo ha considerado todos los pasos del algoritmo y ha determinado que está es la mejor respuesta, considerando que el depósito se encuentra en el nodo uno. Al ejecutar el método brutal search se ha encontrado una respuesta que reduce

considerablemente la distancia obtenida con la heurística anterior, y por lo tanto se la toma como la solución óptima del problema.

VECINO MÁS CERCANO										
RUTA	1	6	5	4	3	8	7	9	2	10	1
DISTANCIA	636,070729 KM										

Figura 39. Solución heurística del vecino más cercano (Módulo didáctico).

BRUTAL SEARCH										
RUTA	1	8	9	10	2	7	4	3	5	6	1
DISTANCIA	604,1757 KM										

Figura 40. Solución optimizada por el algoritmo de búsqueda por fuerza bruta (Módulo didáctico).

De esta manera se puede establecer un algoritmo que se complementa con un método exacto y una heurística y que además optimiza la distancia total recorrida, lo que indica que para fines didácticos el uso de heurísticas puede generar cualquier cantidad de respuestas válidas, pero si se quiere encontrar la mejor solución es necesario utilizar un método exacto, aunque técnicamente es más complejo.

Para el siguiente método utilizado que es la heurística de Clarke y Wright se ha obtenido una respuesta combinando solo pares de nodos, es decir, que se ha modificado el algoritmo de tal manera que solo toma en cuenta los nodos que generan el mayor ahorro en cada caso y forma una ruta considerando que parte desde el depósito y llega al mismo. Para el buen uso de la heurística de ahorros se debería tener restricciones como capacidad de vehículos, distancia máxima, entre otros que podrían permitir al método realizar la combinación entre más nodos. Para este caso práctico con fines didácticos el funcionamiento del algoritmo da como resultado cinco rutas.

INICIO	1	1	1	1	1
	10	4	9	6	8
	2	3	7	5	1
FIN	1	1	1	1	
AHORRO	271,834685	225,769498	129,814406	45,6043942	0
DISTANCIA	325,296647	267,674245	230,007931	163,133849	88,2304559

Figura 41. Rutas generadas (Módulo didáctico).

Los tres métodos pueden utilizarse con mismo un fin, que es buscar una ruta que recorra todos los nodos con la menor distancia recorrida posible, aunque si se analiza el uso del último algoritmo, es evidente que la distancia aumentaría ya que las rutas se componen de dos nodos

y el depósito esto significaría que el vehículo recorre cada ruta por separado, pero en otro escenario se podría considerar que se dispone de más vehículos y de esta manera cada ruta sería cubierta por uno y así la distancia individual que recorre cada uno sería mucho menor. El hecho de realizar las entregas de esta forma podría también aumentar los costos del viaje en total, y para los casos anteriores el vehículo debería tener la capacidad necesaria para abastecer a todos los nodos y eso también mostraría un aumento en los costos dependiendo de la demanda y el tipo de vehículo que se requiera.

Este trabajo pretende mostrar la importancia de realizar experimentos para encontrar la mejor solución a un problema propuesto y analizar las opciones que surgen con base en la modificación de los parámetros operativos para facilitar la toma de decisiones, además del uso de herramientas tecnológicas que facilitan la resolución de VRP, aunque esto requiera un conocimiento matemático e informático, para la elección del programa a utilizarse o en su defecto la elaboración de uno propio con las características que se requiera.

4.1.6. Aceptación de la hipótesis

Si se pudo generar un módulo didáctico de métodos exactos y enfoques heurísticos para solución de problemas VRP, que contiene dos heurísticas y un método exacto, es decir, que tiene tres niveles de integración. Cada algoritmo genera una respuesta válida para el mismo problema y se puede comparar tanto con base en los parámetros operativos como también en tiempo de ejecución del código.

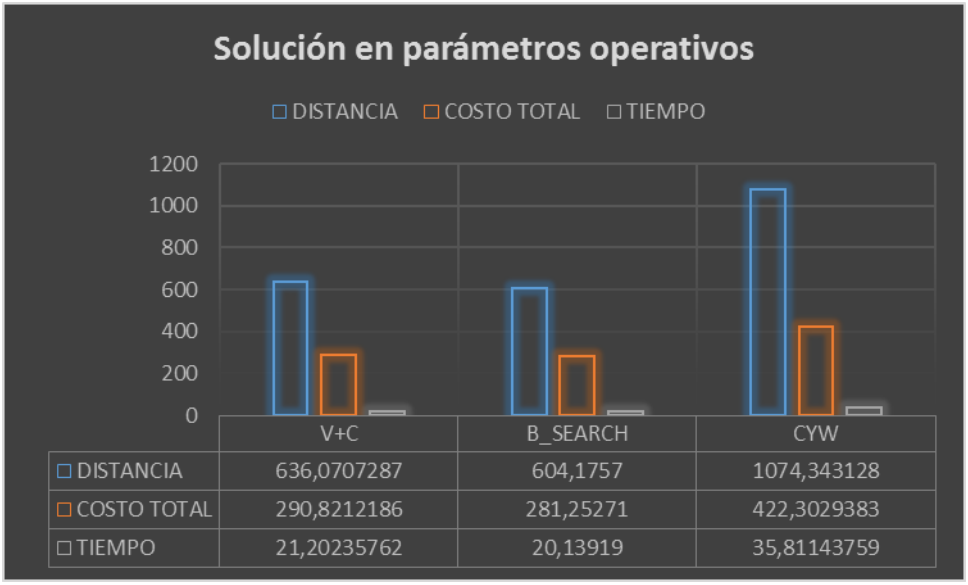


Figura 42. Solución en parámetros operativos.

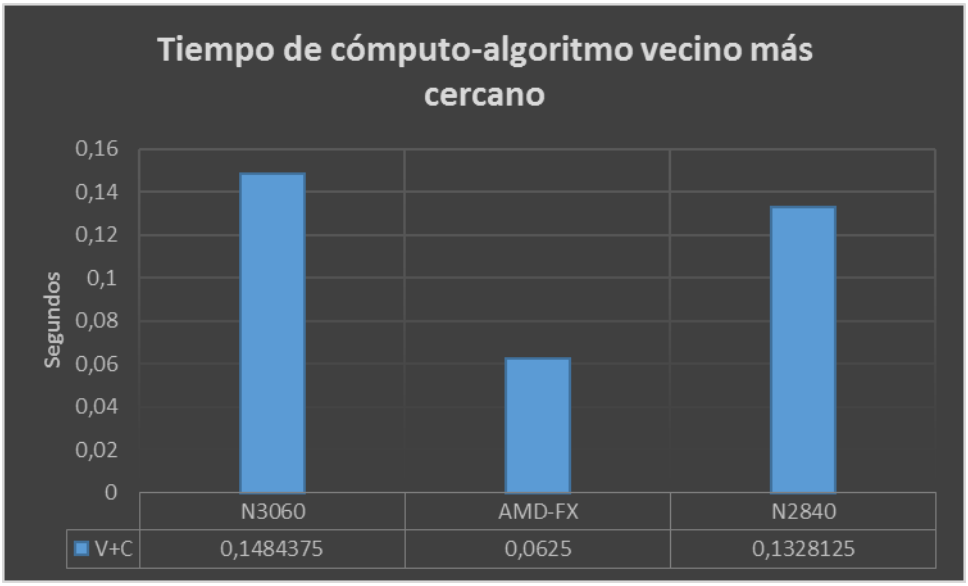


Figura 43. Tiempo de cómputo del algoritmo vecino más cercano.

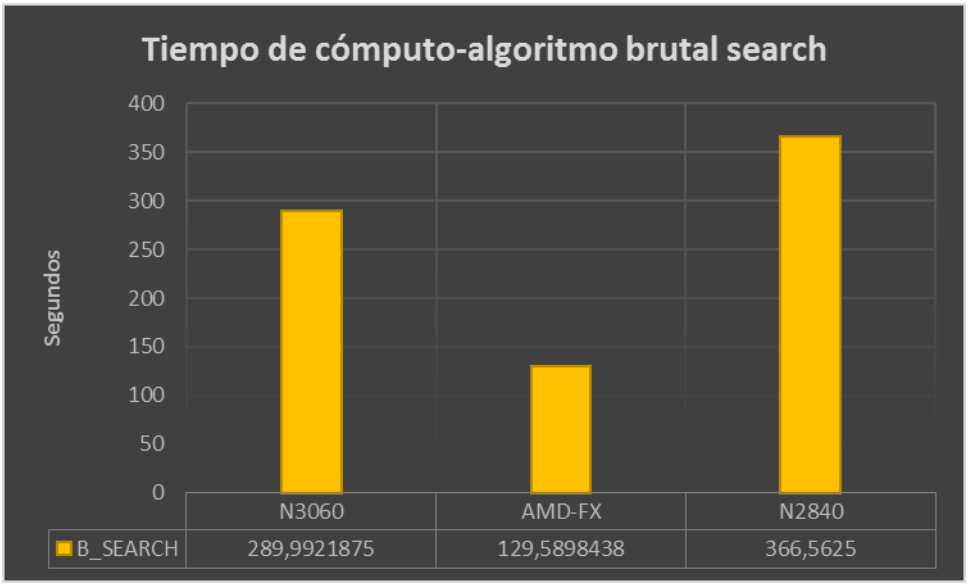


Figura 45. Tiempo de cómputo del algoritmo brutal search.

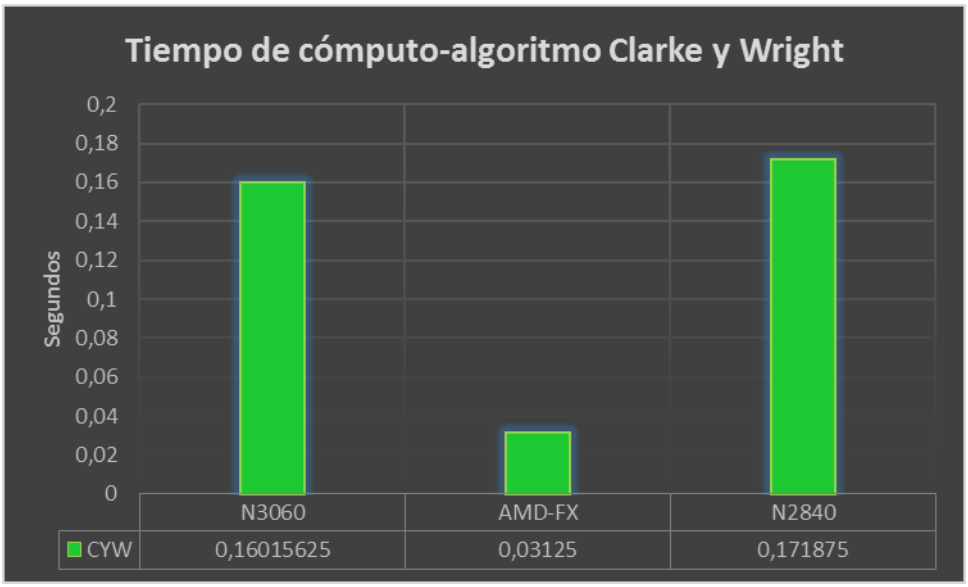


Figura 44. Tiempo de cómputo del algoritmo de Clarke y Wright.

La figura 42 muestra los resultados generados por cada algoritmo y es evidente notar que los métodos del vecino más cercano y *brutal search*, tienen soluciones similares y esto se debe a que ambos buscan una sola ruta que pase por todos los nodos. La heurística de Clarke y Wright tiene una respuesta completamente diferente, ya que busca realizar varias rutas de acuerdo al mayor ahorro generado entre un par de nodos. Esta solución podría tener diferentes interpretaciones dependiendo del juicio del usuario, considerando que se requeriría contar con una flota vehicular mayor, es decir, que para los métodos anteriores solo se necesitaba un solo vehículo y para el último se necesitarían cinco. Además es importante tener en cuenta que los costos aumentarían, debido a que para cada vehículo, los costos variables y el costo fijo serían específicos.

Las figuras 43, 44 y 45 presentan los tiempos de ejecución en segundos para cada uno de los algoritmos y para cada tipo de procesador que se utilizó en la experimentación. Los resultados para el método *brutal search* son los más altos, debido a que es un método exacto y como ya se ha mencionado requiere de más recursos para encontrar la solución, lo que no ocurre con las heurísticas. Los algoritmos restantes tienen tiempos de ejecución que varían para cada tipo de procesador, pero siempre son bajos en comparación con el método exacto.

La solución en parámetros operativos será la misma en cualquier equipo, aunque esto no aplica para los tiempos de ejecución. Por ejemplo, en el experimento realizado con el procesador *AMD-FX 8370 Eight-Core Processor 4GHz*, el algoritmo con el menor tiempo de ejecución fue el de Clarke y Wright, y en los otros dos fue el del vecino más cercano, de esta manera se puede ver claramente que las heurísticas requieren de menos recursos computacionales.

4.2. DISCUSIÓN

Para la ejecución de los algoritmos programados en el módulo didáctico desarrollado en el presente trabajo se debe tener en cuenta una restricción muy importante que limita la posibilidad de trabajar con problemas demasiado grandes, ya que los métodos exactos buscan una solución óptima para un problema determinado y esto requiere de gran capacidad de procesamiento, porque el algoritmo sigue funcionando hasta que haya encontrado la única solución que representa el óptimo para ese problema, en caso de haberla. Debido a este inconveniente se ha determinado trabajar con un máximo de 10 nodos en la presente investigación ya que, al ser para fines didácticos, se puede apreciar que los equipos de cómputo pueden procesar de mejor manera los datos ingresados y obtener una respuesta en un tiempo aceptable.

Para las heurísticas, el cálculo es mucho más rápido ya que estas arrojan respuestas que pueden ser las mejores pero que también son una de muchas que se pueden encontrar durante su ejecución. Se pueden también utilizar algoritmos modificados que mejoran el tiempo de procesamiento y la respuesta obtenida para que sea de mayor beneficio para el interesado.

En el presente trabajo se ha utilizado un método exacto denominado búsqueda por fuerza bruta que consiste en cambiar el orden de los nodos hasta que se pueda obtener la respuesta óptima del problema, pero se ha combinado con la heurística del vecino más cercano para formar un algoritmo más completo que busca la ruta óptima partiendo de una ruta parcial. Para los 10 nodos considerados en el problema el número de permutaciones o cambios que se pueden generar en la ruta es extremadamente alto y se debería probar todas, ya que esto además de optimizar la respuesta, contribuye a verificar si la respuesta parcial es la mejor o al menos una de las mejores. En este punto se debe tener en cuenta la capacidad de procesamiento con que se cuenta ya que se podría generar todas las respuestas posibles pero el tiempo de procesamiento sería muy alto y esta es la principal razón por la cual se prefiere el uso de heurísticas que aseguran conseguir una de las mejores respuestas para un problema. Por otro lado, en esta investigación se ha podido probar algunas de las permutaciones generadas, pero la capacidad computacional no es suficiente para poder probarlas todas, lo cual genera un problema al hablar de la posibilidad de considerar el aumento del número de nodos porque sería necesario conseguir una computadora muy avanzada tecnológicamente incluso para fines didácticos, pero para evidenciar el funcionamiento del algoritmo las permutaciones probadas ya pueden establecer una respuesta parcial que pueda mejorar la antes obtenida. En el caso de un problema de ruteo real el uso de este método exacto sería algo muy costoso, porque como se menciona se

requiere de una capacidad de procesamiento para resolver un problema con una cantidad mayor de nodos, y es por eso que se desarrollan métodos en donde se pueden considerar grupos de nodos, pero se opta por utilizar heurísticas que para problemas más grandes podrían necesitar de más tiempo de procesamiento pero se obtendría una respuesta, que podría ser comparable con la solución de un método exacto pero se debería considerar establecer grupos muy pequeños de nodos que se puedan procesar en un tiempo razonable y verificar que tan cerca de la respuesta óptima está la solución obtenida con la heurística.

La heurística desarrollada por Clarke y Wright o conocida también como heurística de ahorros, es una de las más utilizadas para resolver problemas reales en empresas y esto se puede evidenciar en los antecedentes de este trabajo, aunque se considera modificarlo para poder obtener una mejor respuesta. Las modificaciones que se puedan hacer al algoritmo dependen de las restricciones que se encuentren en un sistema real, y que se puedan programar para el cálculo de las rutas. En esta investigación se toma en cuenta la heurística de ahorros como un método que proporciona una solución muy diferente a los algoritmos del vecino más cercano y de búsqueda por fuerza bruta, dado que estas últimas buscan conectar todos los nodos recorriendo la menor distancia posible y la heurística de ahorros calcula una ruta de ida y vuelta para cada nodo con el depósito y luego evalúa el ahorro que podría se generar conectando pares de nodos o más con el depósito. El algoritmo puro conecta todos los nodos con mayor ahorro al depósito y un algoritmo modificado puede conectar menos nodos dependiendo de la restricción que se haya programado. Para fines didácticos se puede trabajar con el algoritmo puro para que se conozca su funcionamiento, y que luego el programador lo modifique según convenga, además que se podría considerar combinar dos heurísticas para formar una meta heurística que compruebe y mejore la respuesta antes obtenida.

El uso de software de ruteo simplifica considerablemente el trabajo analítico del programador porque es necesario conocer sobre el uso del programa y hacer un levantamiento de información preciso para obtener una respuesta que modele la realidad de una empresa específica. En muchos de los casos incluso se genera un gráfico que contiene la ruta trazada dentro de un mapa real para que sea más comprensible, en otros casos solo se obtienen las rutas generadas con los nodos que han de ser conectados y es preciso interpretar la información obtenida. Aunque el elemento humano debe estar siempre presente, ya que al encontrar una solución y analizar la respuesta se podría identificar algún problema fuera de las restricciones programadas que impidan la implementación de la ruta, en ese caso se debe seguir considerando tantas restricciones como sea posible siempre teniendo en cuenta la capacidad computacional.

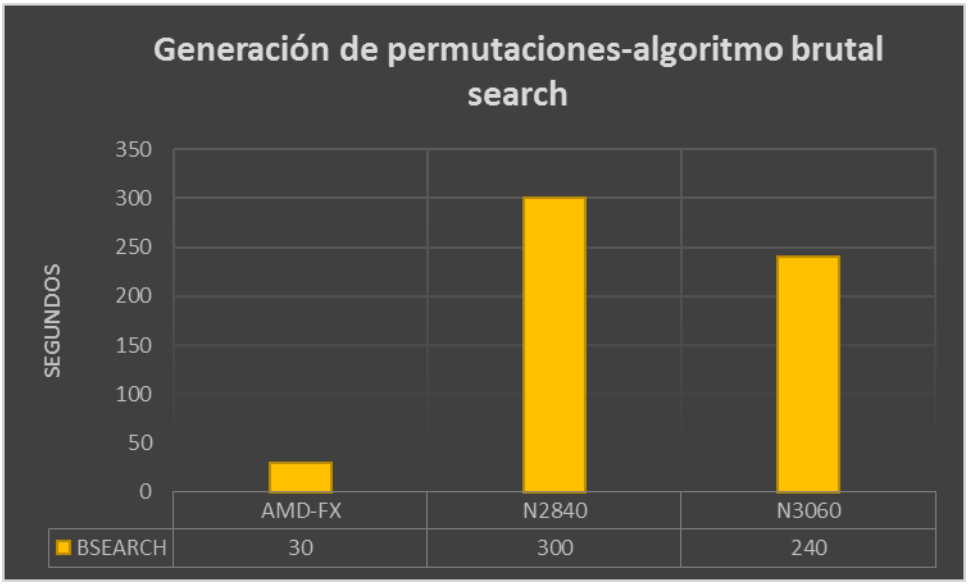


Figura 46. Tiempo de procesamiento para la generación de permutaciones.

Es complicado definir los recursos óptimos para el correcto funcionamiento de los algoritmos de ruteo, especialmente para los métodos exactos ya que al aumentar un nodo al problema el número de cálculos requeridos se incrementa exponencialmente y por lo tanto requiere de una mayor capacidad de procesamiento. En el presente trabajo se han realizado pruebas en algunos equipos de cómputo en los cuales se ha tomado como referencia el tiempo de procesamiento para obtener la respuesta. Como ya se mencionó en el análisis de resultados, para el procesador *AMD-FX 8370 Eight-Core Processor 4GHz* el tiempo requerido es 30 segundos para el método exacto programado en el módulo. Se toma como referencia solamente este algoritmo ya que busca la respuesta óptima del problema, es decir, que la solución encontrada ya no se puede mejorar más, lo que no pasa con los demás métodos que buscan una solución aceptable pero que podría mejorarse. Las pruebas se han considerado realizarlas en quipos que se pueden encontrar dentro de instituciones educativas y equipos portátiles con los que cuentan los estudiantes, para establecer de alguna manera los requisitos mínimos que se requieren para que el módulo obtenga la respuesta, aunque utilice más tiempo de procesamiento de datos, es asi que se considera un procesador *Intel(R) Celeron(R) CPU N2840 @ 2,16GHz* el cual requiere un tiempo de procesamiento de cinco minutos para generar todas las posibles soluciones que se pueden obtener del problema. La diferencia en tiempo de ejecución es notoria, lo que podría deberse al número de núcleos que tiene cada procesador que determinan el número de subprocesos que se pueden llevar a cabo. El primero cuenta con ocho núcleos y el segundo tan solo con dos, es por esto que el tiempo de procesamiento varía tanto, además que los equipos portátiles, que es el segundo caso, por lo general cuentan con procesadores que tienen una

menor cantidad de núcleos que los equipos de mesa. Ahora bien, cinco minutos de espera para obtener todas las posibles rutas aún está dentro de un rango de tiempo razonable, pero es necesario revisar también el estado del equipo a utilizar ya que podría ser que no sea muy actual y a lo largo del tiempo se le ha dado un uso intenso, por ende el procesador no se encuentre en la mejor forma por lo cual requerirá más tiempo, aunque en realidad el número de núcleos del procesador determina el tiempo que le llevará ejecutar el algoritmo. Se ha realizado un experimento más en un procesador *Intel(R) Celeron(R) CPU N3060 @ 2,16GHz* el cual tiene las mismas características que el anteriormente mencionado, pero es un equipo que ha estado en constante mantenimiento para optimizar sus funciones y por lo tanto se tiene una disminución en el tiempo de procesamiento que es aproximadamente de un minuto en comparación con el procesador *N2840*. En consecuencia se comprueba que el estado del equipo también juega un papel importante ya que un equipo que está saturado de aplicaciones y archivos disminuye su rendimiento.

Para la toma de decisiones es importante definir cuál es la mejor respuesta al problema planteado, de tal manera que se evalúan las respuestas de los algoritmos para determinar la que genera un mayor beneficio en cuanto a costo, distancia y tiempo. Dentro del módulo didáctico se cuenta con el porcentaje de mejora que se calcula para cada algoritmo con respecto a los demás y de esta manera facilita el reconocimiento de la mejor opción. Se espera que el método exacto proporcione la mejor respuesta, pero el tiempo de ejecución es más elevado por lo tanto se podría considerar algunas de las otras soluciones para agilizar el proceso, entonces solo se verifica la comparación de los resultados de los algoritmos aplicados y la mejora que cada uno produce respecto de la respuesta del otro y se tomaría una decisión. Para las empresas, es vital definir este tipo de procesos con rapidez ya que podría requerirse una respuesta inmediata a un problema y se podría tomar en consideración solo realizar el cálculo de las heurísticas y compararlo, teniendo en consideración que la respuesta de las heurísticas puede ser la óptima del problema o una solución factible. En todo caso el método exacto se tomaría en cuenta como una respuesta de apoyo con la cual se puede comparar el resultado de las heurísticas y observar la coincidencia que tenga con la respuesta óptima del problema.

V. CONCLUSIONES Y RECOMENDACIONES

5.1. CONCLUSIONES

Establecer las variables para un problema de ruteo vehicular es un proceso complejo que requiere de gran capacidad de análisis por parte del diseñador del modelo, tratando de considerar los aspectos más relevantes de la realidad del proceso de transporte y distribución en una empresa.

El manejo e interpretación del lenguaje matemático es esencial al formular un problema de ruteo vehicular, ya que es necesario establecer una ecuación que modele la realidad analizada y también formular restricciones que se presenten.

Para el responsable del proceso de ruteo vehicular es importante conocer sobre el manejo de programas informáticos que contribuyan a la solución del problema, y además conocer el funcionamiento de los métodos de solución programados, ya que esto define la calidad de respuesta para el problema.

Dentro del modulo didáctico generado la mejor solución para los parámetros operativos del problema es la de brutal search, que obtiene una respuesta de 604,2 km de recorrido la cual se mejora 5% con respecto a la heurística del vecino más cercano, y un 44% con respecto al método de Clarke y Wright, y es importante aclarar que la reducción en la distancia afecta directamente a los parámetros de tiempo y costo los que también se optimizan.

El tiempo de ejecución tiene una notable y gran diferencia en lo que se refiere al método utilizado y no se puede definir cual de las dos heurísticas es la mejor, ya que en los experimentos se obtienen tiempos similares que no sobrepasan un segundo, lo que deja en última posición al método exacto *brutal search* el cual requiere de un promedio de 263 segundos para obtener una respuesta.

Los métodos exactos requieren de una gran capacidad de procesamiento, porque proporcionan la respuesta óptima a un problema planteado, es decir, prueban todas las posibles soluciones que se hayan encontrado y las evalúa para definir la mejor. Este proceso puede realizarse en un tiempo razonable cuando el número de nodos es bajo, pero el aumento de un nodo sugiere un incremento exponencial en el tiempo de procesamiento.

Las heurísticas y meta heurísticas son métodos de solución que proporcionan una respuesta válida para un problema en un tiempo razonable y por esta razón son más utilizados dentro de los programas informáticos para resolución de problemas VRP.

El módulo didáctico desarrollado servirá como apoyo para demostrar el uso y el beneficio que tienen las herramientas tecnológicas en la realización de un trabajo, además de contribuir al mejor entendimiento del funcionamiento de los algoritmos de ruteo.

5.2. RECOMENDACIONES

Desarrollar aptitudes de programación en los estudiantes para que puedan generar, utilizar y modificar herramientas informáticas que les permitan desarrollar sus actividades laborales en el futuro.

Para programar un algoritmo matemático, es necesario tener el diagrama de flujo para identificar los pasos y facilitar la generación del código.

Contar con una computadora de gran capacidad de procesamiento de datos, permite realizar cálculos de manera rápida aunque para resolver problemas VRP, es recomendable trabajar con heurísticas que requieren menos tiempo de ejecución, y si se requiere trabajar con un método exacto se debe tener en cuenta que el número de nodos sea pequeño para que la máquina pueda calcular la respuesta en un tiempo razonable.

VI. REFERENCIAS BIBLIOGRÁFICAS

Anbuudayasankar, S., Mohapatra, S., y Ganesh, K. (2018). *Models for practical ruting problems in logistics.* Suiza: Springer International Publishing Switzerland.

Competitividad, C. N. (2018). *http://www.competitividad.org.do/.* Recuperado el 20 de Septiembre de 2020, de http://www.competitividad.org.do/: http://www.competitividad.org.do/wp-content/uploads/2018/07/%C3%8Dndice-de-Desempe%C3%B1o-Log%C3%ADstico-2018-Final.pdf

eloycaballero. (2018). *eloycaballero.com.* Recuperado el 10 de Enero de 2020, de eloycaballero.com: https://eloycaballero.com/permutaciones-en-hoja-de-calculo/

hazlogística. (2018). *hazlogística*. Recuperado el 23 de Octubre de 2018, de hazlogística: https://hazlogistica.com/2018/04/09/actividades-de-la-cadena-logistica/

Herazo, N. (2012). *Modelación matemática del problema de ruteo de vehículos con restricciones de múltiples depósitos, flota heterogénea de vehículos y ventanas de tiempo.* Barranquilla: Corporación Universitaria de la Costa. Recuperado el 18 de Octubre de 2018

Hernández, Y. (2016). *Diseño de un sistema de ruteo de vehículos con múltiples depósitos en empresas de transporte de carga por carretera.* Bogotá: Universidad Distrital Francisco José de Caldas. Recuperado el 16 de Octubre de 2018

Islas, V., y Zaragoza, M. (2003). *Análisis de los sistemas de transporte.* México: Sanfandilla.

Johansen, O. (1993). *Introducción a la teoría general de sistemas.* México: LIMUSA. Recuperado el 25 de Octubre de 2019

Kenton, W. (2019). *investopedia*. Recuperado el 20 de Noviembre de 2019, de investopedia: https://www.investopedia.com/terms/v/visual-basic-for-applications-vba.asp

Mangesh, N. (2013). *Landmark based shortest path detection by using Dijkestra Algorithm and Harvesine Formula.* Amravati: IJERA.

Meserón, S. (2007). *http://evoluciondelalogistica.blogspot.com/*. Obtenido de http://evoluciondelalogistica.blogspot.com/: http://evoluciondelalogistica.blogspot.com/2007/12/resumen-de-la-historia-de-logstica.html

Nuño, P. (2017). *https://www.emprendepyme.net/*. Recuperado el 6 de Septiembre de 2020, de https://www.emprendepyme.net/: https://www.emprendepyme.net/tipos-de-logistica-empresarial.html

Ocaña, R., y Ramírez, C. (2012). *Diseño de un modelo matemático para resolver problemas de ruteo vehicular capacitado con ventanas de tiempo, con la aplicación del algoritmo de Clarke y Wright. Caso de estudio: Empresa de servicios de courier ciudad de Guayaquil.* Guayaquil, Ecuador: Escuela Politécnica Superior del Litoral. Recuperado el 15 de Octubre de 2018

Pathumnakul, S. (1996). *Solving multi depot vehicle routing problem for Iowa recycled paper by Tabu Search heuristic.* Iowa.

Rocha, L., González, C., & Orjuela, J. (2011). Una revisión del estado del arte del problema de ruteo de vehículos: Evolución hitórica y métodos de solución. *Ingeniería*, *16*, 35-55.

Santiago, P. (2012). Aporte de la cibernética de segundo orden como estrategia pedagógica en la educación universitaria. *Aporte de la cibernética de segundo orden como estrategia pedagógica en la educación universitaria.* Nueva Granada.

Shouyan, S., Ning, J., y Xu, Z. (2019). *gestiopolis.com*. Recuperado el 6 de Septiembre de 2020, de gestiopolis.com: https://www.gestiopolis.com/la-gestion-de-inventarios-como-parte-de-la-logistica-empresarial/

Vélez, M., y Montoya, J. (2007). *Metaheurísticos: una alternativa para la solución de problemas combinatorios en administración de operaciones.* Recuperado el 11 de Noviembre de 2019, de Scielo: http://www.scielo.org.co/scielo.php?script=sci_arttext&pid=S1794-12372007000200009

VII. ANEXOS

Anexos 1 **Acta de predefensa de informe de investigación**

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI
FACULTAD DE COMERCIO INTERNACIONAL, INTEGRACIÓN, ADMINISTRACIÓN Y ECONOMÍA EMPRESARIAL
CARRERA DE LOGISTICA Y TRANSPORTE

ACTA

DE LA SUSTENTACIÓN DE PREDEFENSA DEL INFORME DE INVESTIGACIÓN DE:

NOMBRE: BENAVIDES CAIPE DANNY ALEXIS **CÉDULA DE IDENTIDAD:** 0401838305
NIVEL/PARALELO: 0 **PERIODO ACADÉMICO:** UNIO-SEPTIEMBRE 202

TEMA DE INVESTIGACIÓN: "MÓDULO DIDÁCTICO DE MÉTODOS EXACTOS Y ENFOQUES HEURÍSTICOS DE RESOLUCIÓN DE PROBLEMAS DE RUTEO DE VEHÍCULOS (VRP) ".

Tribunal designado por la dirección de esta Carrera, conformado por:

PRESIDENTE: MSC. LÓPEZ RUANO JUAN CARLOS
LECTOR: MSC. POZO BURGOS EDUARDO JAVIER
ASESOR: MSC. MAFLA BOLAÑOS IVÁN GABIEL

De acuerdo al artículo 21: Una vez entregados los requisitos para la realización de la pre-defensa el Director de Carrera integrará el Tribunal de Pre-defensa del informe de investigación, fijando lugar, fecha y hora para la realización de este acto:

EDIFICIO DE AULAS: 0 **AULA:** 0
FECHA: lunes, 14 de septiembre de 2020
HORA: 10H04

Obteniendo las siguientes notas:

1) Sustentación de la predefensa:	7.00
2) Trabajo escrito	3.00
Nota final de PRE DEFENSA	**10.00**

Por lo tanto: **APRUEBA** ; debiendo acatar el siguiente artículo:

Art. 24.- De los estudiantes que aprueban el Plan de Investigación con observaciones. - El estudiante tendrá el plazo de 10 días laborables para proceder a corregir su informe de investigación de conformidad a las observaciones y recomendaciones realizadas por los miembros Tribunal de sustentación de la pre-defensa.

Para constancia del presente, firman en la ciudad de Tulcán el lunes, 14 de septiembre de 2020

0401145065 JUAN CARLOS LOPEZ RUANO

MSC. LÓPEZ RUANO JUAN CARLOS
PRESIDENTE

04012544 46 IVAN GABRIEL MAFLA BOLAÑOS

Javier Pozo

MSC. MAFLA BOLAÑOS IVÁN GABIEL
TUTOR

MSC. POZO BURGOS EDUARDO JAVIER
LECTOR

Adj.: Observaciones y recomendaciones

Anexos 2 **Certificado del Abstract emitido por parte del centro de idioma**

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI
FOREIGN AND NATIVE LANGUAGE CENTER

Informe sobre el Abstract de Artículo Científico o Investigación.

Autor: Benavides Caipe Danny Alexis
Fecha de recepción del abstract: 04 de octubre de 2020
Fecha de entrega del informe: 06 de octubre de 2020

El presente informe validará la traducción del idioma español al inglés si alcanza un porcentaje de: 9 – 10 Excelente.

Si la traducción no está dentro de los parámetros de 9 – 10, el autor deberá realizar las observaciones presentadas en el ABSTRACT, para su posterior presentación y aprobación.

Observaciones:

Después de realizar la revisión del presente abstract, éste presenta una apropiada traducción sobre el tema planteado en el idioma Inglés. Según los rubrics de evaluación de la traducción en Inglés, ésta alcanza un valor de 9, por lo cual se validad dicho trabajo.

Edison Boanerges Peñafiel Arcos
COORDINADOR CENTRO DE IDIOMAS
(06) 2224079 - 2224080
Ext. 1480
Calle Antisana y Av. Universitaria
www.upec.edu.ec

Revisor: Ing. Edison Peñafiel Arcos

EDISON BOANERGES
PEÑAFIEL ARCOS

2

UNIVERSIDAD POLITÉCNICA ESTATAL DEL CARCHI
FOREIGN AND NATIVE LANGUAGE CENTER

ESSAY EVALUATION SHEET

NAME: Benavides Caipe Danny Alexis **DATE:** 04 de octubre de 2020
TOPIC: Modelo didáctico de métodos exactos y enfoques heurísticos para la resolución de problemas de ruteo de vehículos (VRP).

MARKS AWARDED **QUANTITATIVE AND QUALITATIVE**

VOCABULARY AND WORD USE	Use new learnt vocabulary and precise words related to the topic	Use a little new vocabulary and some appropriate words related to the topic	Use basic vocabulary and simplistic words related to the topic	Limited vocabulary and inadequate words related to the topic
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
WRITING COHESION	Clear and logical progression of ideas and supporting paragraphs.	Adequate progression of ideas and supporting paragraphs.	Some progression of ideas and supporting paragraphs.	Inadequate ideas and supporting paragraphs.
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
ARGUMENT	The message has been communicated very well and identify the type of text	The message has been communicated appropriately and identify the type of text	Some of the message has been communicated and the type of text is little confusing	The message hasn't been communicated and the type of text is inadequate
	EXCELLENT: 2 ■	GOOD: 1,5 ☐	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
CREATIVITY	Outstanding flow of ideas and events	Good flow of ideas and events	Average flow of ideas and events	Poor flow of ideas and events
	EXCELLENT: 2 ☐	GOOD: 1,5 ■	AVERAGE: 1 ☐	LIMITED: 0,5 ☐
SCIENTIFIC SUSTAINABILITY	Reasonable, specific and supportable opinion or thesis statement	Minor errors when supporting the thesis statement	Some errors when supporting the thesis statement	Lots of errors when supporting the thesis statement
	EXCELLENT: 2 ☐	GOOD: 1,5 ■	AVERAGE: 1 ☐	LIMITED: 0,5 ☐

TOTAL/AVERAGE
9 - 10: EXCELLENT
7 - 8,9: GOOD
5 - 6,9: AVERAGE
0 - 4,9: LIMITED

TOTAL 9

1