DESARROLLO DE UN ALGORITMO HÍBRIDO PARA RESOLVER EL PROBLEMA DE LOCALIZACIÓN-RUTEO (LRP)

ANGÉLICA PATRICIA NIER OBREGÓN
JENIFER ALEXANDRA NIÑO HERNÁNDEZ

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE FÍSICO-MECÁNICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2015

DESARROLLO DE UN ALGORITMO HÍBRIDO PARA RESOLVER EL PROBLEMA DE LOCALIZACIÓN-RUTEO (LRP)

ANGÉLICA PATRICIA NIER OBREGÓN (2093188)
JENIFER ALEXANDRA NIÑO HERNÁNDEZ (2093225)

Tesis de grado en modalidad "Trabajo de investigación" presentado como requisito parcial para optar al título de Ingeniero Industrial.

Director
HENRY LAMOS DIAZ
Ph.D. Física – Matemática

Codirector
JULIANA NIÑO
Estudiante de Maestría en Ingeniería Industrial

GRUPO DE INVESTIGACIÓN OPTIMIZACIÓN Y ORGANIZACIÓN DE SISTEMAS PRODUCTIVOS, ADMINISTRATIVOS Y LOGÍSTICOS (OPALO)

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE FÍSICO-MECÁNICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2015

AGRADECIMIENTOS

A Dios por brindarme la sabiduría, la fortaleza y el crecimiento tanto personal como profesional y por la oportunidad de culminar esta etapa.

A mis padres por el apoyo, la motivación y esfuerzo durante toda mi carrera, este logro fue posible gracias a ustedes y por ustedes.

A mi hermana por el apoyo incondicional, mi motivación y por el ejemplo que ha sido para mí.

A nuestro director Henry por el apoyo y el conocimiento aportado a lo largo de este proceso.

A mi compañera de proyecto Angélica por toda la paciencia, la motivación, los conocimientos y sabiduría, la bonita energía y sobre todo la buena disposición en todo momento.

A Miguel Ángel Cruz que desde el primer momento que comencé esta carrera fue mi apoyo, motivador incansable y mi inspiración.

A mis amigos, a ustedes les agradezco su compañía y su valioso aporte durante todo este proceso de aprendizaje, aprendí cosas valiosas de cada uno de ustedes e hicieron que el transcurso por esta carrera fuera inolvidable.

JENIFER ALEXANDRA NIÑO HERNÁNDEZ

AGRADECIMIENTOS

A Dios y a mi familia.

ANGÉLICA PATRICI A NIER OBREGÓN

CONTENIDO

Pág.

LISTA DE TABLAS

Pág.

LISTA DE FIGURAS

Pág.

ANEXOS

RESUMEN

TÍTULO: DESARROLLO DE UN ALGORITMO HIBRIDO PARA RESOLVER EL PROBLEMA DE LOCALIZACION-RUTEO (LRP).*

AUTORES: NIER OBREGÓN, Angélica Patricia
NIÑO HERNÁNDEZ, Jenifer Alexandra**

PALABRAS CLAVE: Localización, ruteo, colonia de hormigas, heurística, algoritmo, búsqueda local iterativa.

DESCRIPCIÓN:

El problema conjunto de localización de instalaciones y enrutamiento de vehículos, con capacidad limitada en depósitos y un vehículo por depósito (LRP, por sus siglas en inglés), es definido como un caso especial del problema de ruteo de vehículos (VRP), donde se determina de forma simultánea la localización de depósitos y las rutas de distribución. Dado un conjunto potencial de depósitos, con sus costos de apertura y capacidad, y demandas esperadas de los clientes para determinado horizonte de tiempo, se tiene como objetivo minimizar el costo total dado por la ubicación y distribución, donde cada cliente es visitado exactamente una vez y asignado a un único depósito, sin exceder su capacidad.

En este proyecto, se propone un algoritmo híbrido que emplea optimización por colonia de hormigas (ACO, por sus siglas en inglés), con el uso de tres colonias (selección de depósitos, asignación de clientes y VRP), para generar una solución inicial que es mejorada con búsqueda local iterativa (ILS, por sus siglas en inglés) que consta de cuatro estructuras de vecindad y una perturbación.

El algoritmo ACO+ILS es implementado en MATLAB y con un banco de pruebas propuesto se emplea un diseño factorial fraccionado 2^{k-2} con el fin de determinar el efecto y la mejor combinación en valores de los factores. Los resultados del algoritmo propuesto son comparados con los del ACO tradicional mostrando mejoras significativas frente a este en cuanto a disminución costo total del sistema.

* Proyecto de grado

** Facultad de ingenierías fisicomecánicas. Escuela de estudios industriales y empresariales. Director: PhD Henry Lamos Díaz

ABSTRACT

TITLE: HIBRID ALGORITHM FOR SOLVING THE ROUTING-LOCATION PROBLEM (LRP).*

AUTORS: NIER OBREGÓN, Angélica Patricia
NIÑO HERNÁNDEZ, Jenifer Alexandra**

KEYWORDS:
Location, routing, ant colony, heuristic, algorithm, iterated local search.

DESCRIPTION:

The location routing problem (LRP) is defined as a special case of the vehicle routing problem (VRP), which is to simultaneously determine locating facilities and distribution routes. Given a potential container assembly with its opening costs and capacity, and expected customer demand for a given time horizon, the objective is to minimize the total cost given the location and distribution, where each customer is visited exactly once and assigned to a single facility, without exceeding their capacity.

In this research work, a hybrid algorithm that uses Ant Colony Optimization (ACO), with the use of three colonies (location selection, customer assignment and VRP) to generate an initial solution, is proposed; that is enhanced with the iterated local search (ILS) consisting of four structures neighborhood and disturbance.

ACO+ILS algorithm is implemented in Matlab and a test is proposed. A fractional factorial design 2^{k-2} is used to determine the effect and the best combination of values of the factors utilized. The results of the proposed algorithm are compared with the traditional ACO and significant improvements over this are showed. These results are presented in terms of the reduction of total system cost.

* Degree Project
** Faculty of Engineering physicomechanical. School of Industrial and Business Studies. Directed by Henry Lamos Díaz PhD

CUMPLIMIENTO DE OBJETIVOS

Tabla 1. Cumplimiento de objetivos

OBJETIVOS	CUMPLIMIENTO
Realizar una revisión de la literatura de los métodos de solución del problema de localización y ruteo (LRP).	Capítulo 2
Formular el modelo de programación matemática del LRP.	Subcapítulo 1.3 Subcapítulo 2.2.1
Construir el algoritmo de búsqueda local iterativa para la solución del LRP.	Subcapítulo 3.5
Desarrollar un framework en base a la metahurística de optimización Colonia de Hormigas (ACO) y la búsqueda local iterativa para resolver el problema LRP.	Capítulo 3
Implementar en MATLAB el framework para la solución del problema LRP.	Anexo E
Validar la herramienta desarrollada mediante experimentos numéricos que permitan medir su eficiencia.	Capítulo 4 Anexo F
Realizar un artículo de carácter publicable con los resultados del proyecto de investigación.	Anexo G

INTRODUCCIÓN

En el mundo, se presenta un nivel de globalización de mercados que da al transporte un grado significativo de importancia en un escenario que supone el aumento en el intercambio de bienes y servicios. Teniendo en cuenta que en la actualidad no compiten las empresas de manera individual, sino que esta competencia se da entre cadenas de abastecimiento[1], se opta por su gestión en conjunto de manera que se abarquen las tareas logísticas tradicionales enfocados a la reducción de costos totales integrando estrategias y tácticas con un poder suficiente para la toma de decisiones (localización de centros de distribución, asignación de clientes a depósitos, determinación de rutas, etc.), lo cual implica grados de complejidad avanzados que han generado la aparición de nuevos sistemas de apoyo informático[2].

La optimización de redes logísticas, es un proceso que ofrece pautas para los procesos de toma de decisiones con afectaciones a corto, mediano y largo plazo, cuyas características tienen que ver con combinación de variables (costo, distancias, demandas, etc.), y además pueden emplearse como herramientas de simulación para ensayar distintos escenarios según opciones específicas de las compañías que la apliquen[3].

Es así que el diseño de redes de transporte es un problema importante para la gestión de la cadena de suministro, pues ofrece un gran potencial en la reducción

[1] REVISTA ESCUELA DE ADMINISTRACIÓN DE NEGOCIOS, 2008, Las Pymes: costos en la cadena de abastecimiento. 2008. [en línea]. [citado 21 Marzo 2014]. Available from: http://www.redalyc.org/pdf/206/20611455002.pdf

[2] VIRTUAL PRO, 2005, La evolución del concepto “Logística” al de “Cadena de suministros” y más allá. [en línea].2005. [citado 21 Marzo 2014]. Available from: http://www.revistavirtualpro.com/files/TIE02_200702.pdf

[3] PALMAS, 2007, Introducción a la optimización de la red logística en el aceite de palma. [en línea].2007. [citado 21 Marzo 2014]. Available from: http://publicaciones.fedepalma.org/index.php/palmas/article/download/1319/1319

de costos y además contribuye a mejorar la calidad del servicio en cuanto a disminución de tiempos de entrega. El Problema conjunto de localización de instalaciones y ruteo de vehículos (LRP por sus siglas en inglés) se define como un caso especial del Problema de ruteo de vehículos (VRP por sus siglas en inglés), en el que la ubicación de depósitos se determina simultáneamente con las rutas de distribución[4]. LRP es un problema NP-Hard, ya que contiene dos problemas NP-Hard: localización de instalaciones y ruteo de vehículos (Nagy y Salhi, 2007)[5]. La idea de hacer esta combinación se remonta casi cincuenta años, y sin embargo todavía existen académicos que prefieren ignorar esta combinación por diferentes razones. Una de estas está relacionada con el horizonte de planeación, que es diferente para estas decisiones, es decir, la localización de instalaciones es una decisión estratégica, mientras que, el enrutamiento de vehículos es una decisión táctica; no obstante se ha logrado demostrar el beneficio al tratar una planificación a largo plazo, pero permitiendo cambiar las rutas[6].

El presente documento busca estudiar los métodos de solución que han sido utilizados en el LRP, con el fin de desarrollar un algoritmo aplicado a este problema con múltiples almacenes con limitaciones de capacidad y un vehículo por depósito. La estructura del documento se presenta de la siguiente manera: en la sección dos se encuentra la revisión de la literatura respecto al LRP, así como las metaheurísticas de optimización Colonia de Hormigas(ACO) y Búsqueda Local Iterativa (ILS); en la sección tres se presenta el desarrollo de framework y en la cuatro se encuentran los resultados obtenidos.

[4] MOHAMMAADI, M.. Multi-objective invasive weed optimization for stochastic green hub location routing problema with simultaneous pick-ups and deliveries. 2013.
[5] NAGY, G., & SALHI, S. Location-routing: Issues, models and methods. European Journal of Operational Research, 177(2), 649–672. 2007.
[6] Ibíd., p. 660.

1 ANTECEDENTES DEL PROYECTO

Se hace una revisión de proyectos de grado a partir del catálogo bibliográfico de la universidad, sin encontrar antecedentes explícitamente relacionados con el problema conjunto de localización de instalaciones y ruteo de vehículos (LRP, por sus siglas en inglés). Por tanto, la búsqueda de antecedentes, se basa en los proyectos que han abordado el problema por separado, es decir, el problema de localización de instalaciones, y por otro lado, el problema de ruteo de vehículos. Para el primer problema se presenta un trabajo relacionado con este, mientras que para el segundo, se encuentra un mayor número de proyectos.

1.1 PROYECTOS RELACIONADOS CON EL PROBLEMA DE LOCALIZACIÓN DE INSTALACIONES.

- Diseño de una propuesta de localización para los centros de distribución (CEDIs) de la empresa Comertex S.A. a partir de un modelo matemático[7].

Descripción: diseño de un modelo matemático basado en programación lineal entera binaria y que utiliza el software GAMS. El modelo se realizó con el fin de determinar la mejor solución posible al problema de localización de los centros de distribución de la empresa Comertex S.A., integrando características propias de esta. Se hizo una recolección de información del estado de la empresa y se analizó con el objetivo de adecuar los datos para la formulación matemática y establecer los parámetros y restricciones para encontrar la solución.

[7] NIÑO, Edna. PELÁEZ, Johanna. Diseño de una propuesta de localización para los centro de distribución (CEDIs) de la empresa Comertex S.A. a partir de un modelo matemático. Bucaramanga.. Universidad Industrial de Santander. 2012.

Alcance: Generar una propuesta de localización de centros de distribución de Comertex S.A. mediante un modelo matemático a través del software GAMS. Se elige la solución más adecuada en cuanto a costos, analizando además de los resultados, los factores cualitativos y cuantitativos que no se incorporan en el modelo matemático. Se identifican los grupos de los productos que se almacenaran en cada centro de distribución, así como sus correspondientes proveedores, segmento de clientes a atender y la capacidad estimada.

Aporte al proyecto: El aporte de este proyecto es principalmente la formulación matemática para el problema de localización de instalaciones; y que permite tener una perspectiva del problema, debido a la utilización de información existente en la empresa y su aplicación a un caso cercano a la realidad en cuanto a las redes de distribución.

1.2 PROYECTOS RELACIONADOS CON EL PROBLEMA DE RUTEO DE VEHÍCULOS.

- Métodos heurísticos para la solución de problemas de ruteo de vehículos con capacidad (CVRP)[8].

Descripción: En este proyecto se realizó una herramienta de software llamada H-CVRP creada en Matlab®, que utiliza dos métodos heurísticos: el vecino más cercano y de ahorros de *Clarke and Wright;* para resolver el problema de ruteo de vehículos con capacidad CVRP. El primer método implementado consiste en inspeccionar en la vecindad de cada nodo, y el segundo consiste en adicionar a la fórmula del cálculo de los ahorros, un parámetro de forma de la ruta que evita la formación de rutas circulares. Esta herramienta busca optimizar el tiempo de asignación de las rutas y obtener una solución factible del problema.

[8] CONTRERAS, Claudia. DIAZ, María. Métodos heurísticos para la solución de problemas de ruteo de vehículos con capacidad (CVRP). Bucaramanga. Universidad Industrial de Santander.

Alcance: Desarrollo de una herramienta en Matlab® con el respectivo manual, basada en el algoritmo del vecino más cercano y el algoritmos de ahorros de *Clarke and Wright;* que presenta la solución al CVRP.

Aporte al proyecto: el aporte consiste en que el trabajo brinda conceptos básicos sobre el problema CVRP, que permiten comprender su formulación matemática. También presenta un manual básico para la programación de un toolbox en Matlab®.

- Un método evolutivo de colonia de hormigas para la solución del problema de ruteo de vehículos con demandas estocásticas[9].

Descripción: Elaboración de un algoritmo EACO para la solución del problema de ruteo de vehículos con demandas estocásticas VRPSD, utilizando el software Matlab®; y se realiza con el fin de mejorar la calidad de la solución. Se aplica la metaheurística Sistema de Hormigas con la estrategia de actualización global para disminuir la explotación. Los agentes se ubicaron de manera aleatoria en la localización de clientes, en la construcción de rutas, para mejorar la capacidad de exploración. Se selecciona el operador de mutación para aumentar la capacidad de exploración, y la heurística de búsqueda local 2-opt para explorar las mejores soluciones, con el fin de mejorar el rendimiento del algoritmo.

Alcance: Desarrollo del algoritmo evolutivo de colonia de hormigas para resolver el problema de ruteo de vehículos con demandas estocásticas y diseño de una toolbox en el software Matlab® para la solución del modelo y que mejore el rendimiento del algoritmo AS tradicional en ocho instancias comparadas.

[9] ACOSTA, Francia. OSORIO, Daniela Un método evolutivo de colonia de hormigas para la solución del problema de ruteo de vehículos con demandas estocásticas. Bucaramanga. Universidad Industrial de Santander.

Aporte del proyecto: Este trabajo aporta principalmente con la modelación matemática del VRPSD, y el desarrollo del algoritmo basado en la metaheurística ACO. Aunque se presenta una variación del problema en cuanto a los datos de entrada respecto al presente trabajo, el desarrollo del algoritmo evolutivo muestra una representación de la utilización de esta metaheurística.

- Desarrollo de un algoritmo híbrido para la resolución del problema de vehículos con entrega y recogida simultaneas (VRPSPD)[10].

Descripción: Propuesta de un algoritmo híbrido para la resolución del problema de ruteo de vehículos con entrega y recolección simultaneas, utilizando un enfoque de optimización de enjambres de partículas, combinado con la técnica heurística de construcción de rutas *cheapest insertion heuristic* y una heurística 2-opt de búsqueda local. La primera heurística se aplica para decodificar la solución y la segunda para re-optimizar soluciones.

Alcance: Aplicación de Matlab® que permita dar una alternativa de solución al problema de ruteo de vehículos con entrega y recolección simultáneas mediante el uso de los algoritmos “optimización de enjambre de partículas” y “Heurística de búsqueda local 2-opt”.

Aporte al proyecto: El aporte se basa esencialmente en la utilización de un enfoque de enjambre de partículas, siendo una de las técnicas de optimización dentro de la inteligencia de enjambres, al igual que la optimización basada en colonia de hormigas; que pretende resolver el VRP con entrega y recolección simultáneas. Además, este es un algoritmo híbrido con la heurística de búsqueda local de 2-opt.

[10] CRUZ, Camilo. GONZÁLEZ, Ludy. Desarrollo de un algoritmo híbrido para la resolución del problema de vehículos con entrega y recogida simultáneas (VRPSPD). Trabajo de grado Ingeniería Industrial Bucaramanga. Universidad Industrial de Santander. Facultad de Ingenierías físico-mecánicas. 2013.

1.3 PLANTEAMIENTO DEL PROBLEMA

El problema conjunto de localización de instalaciones y enrutamiento de vehículos (LRP), se puede definir como un caso especial del problema de ruteo de vehículos (VRP) donde la ubicación y número óptimo de depósitos se determina simultáneamente con la búsqueda de rutas de distribución.[11]

El presente trabajo, considera un LRP estático con flota de vehículos homogénea en el cual se busca determinar los depósitos que se deben abrir, la asignación de clientes a cada depósito y la construcción de las rutas de distribución para satisfacer las demandas de los clientes; el problema principal que se aborda es la localización de depósitos y como sub-problema, el ruteo de vehículos. Así, el objetivo general es minimizar la suma de los costos asociados con la apertura de depósitos y los costos de operación proporcionales a las distancias recorridas. Para esto las siguientes restricciones se deben cumplir:

- La demanda de cada cliente debe ser satisfecha.
- Cada cliente es atendido exactamente por un vehículo.
- Cada ruta comienza y termina en el mismo depósito.
- La longitud de una ruta y el número de vehículos tiene un número límite predefinido.
- La suma de las demandas de los clientes asignados a cada depósito abierto no debe exceder su capacidad.
- No se permite el flujo entre depósitos.

[11] MOHAMMANDI, M.. Multi-objetive invasive weed optimization for stochastic Green hub location routing problema with simultaneous pick-ups and deliveries. 2013. C44, C61, L91.

1.4 JUSTIFICACIÓN DEL PROBLEMA

En los últimos años se ha hecho más evidente que los avances tecnológicos, la innovación y el mundo globalizado, son causas de una economía más competitiva. Por lo tanto, las empresas buscan, entre otras soluciones, disminuir los costos de su cadena productiva y buscar soluciones más eficientes que les permitan permanecer en el mercado y obtener cierta ventaja respecto a sus competidores. La acertada toma de decisiones logísticas puede llevar al éxito de las empresas.

Revisión de literatura científica permite afirmar que cuando se da solución al problema de localización omitiendo la asignación de rutas se aumentan los costos de distribución del sistema, por tanto, la importancia de considerar el problema de forma simultánea y contribuir significativamente a minimizar los costos totales en la cadena de suministro, además, la adecuada proximidad al mercado y tiempos de entrega, aumentan la calidad del servicio favoreciendo los niveles de ventas.

En este trabajo se presenta un algoritmo de optimización desarrollado para resolver LRP con múltiples depósitos *capacitados* y un vehículo por depósito usando la metaheurística de Optimización Colonia de Hormigas (ACO) junto con los métodos de búsqueda local iterativa (ILS). Con la presente herramienta se busca mejorar las decisiones en cuanto a localización de depósitos y la asignación de rutas obteniendo soluciones factibles al problema.

Se decide trabajar con un método de solución metaheurística, pues en la implementación que han tenido los métodos exactos para resolver LRP, la solución óptima está limitada para problemas que se consideran de mediana escala.

Al tratarse de un proyecto de investigación, se aportará al proceso de generación de conocimiento contribuyendo así al contenido de la visión de la Universidad Industrial de Santander que considera la investigación como uno de los ejes articuladores de sus funciones misionales.

1.5 OBJETIVOS

1.5.1 Objetivo general. Desarrollar un algoritmo para la solución del problema de localización- ruteo (LRP) con múltiples depósitos con limitaciones de capacidad y un vehículo por depósito; utilizando el método de optimización Colonia de Hormigas (ACO) y la búsqueda local iterativa (ILS).

1.5.2 Objetivos específicos

- Realizar una revisión de la literatura de los métodos de solución del problema de localización y ruteo (LRP).
- Formular el modelo de programación matemática del LRP.
- Construir el algoritmo de búsqueda local iterativa para la solución del LRP.
- Desarrollar un framework en base a la metahurística de optimización Colonia de Hormigas (ACO) y la búsqueda local iterativa para resolver el problema LRP.
- Implementar en MATLAB el framework para la solución del problema LRP.
- Validar la herramienta desarrollada mediante experimentos numéricos que permitan medir su eficiencia.
- Realizar un artículo de carácter publicable con los resultados del proyecto de investigación.

1.6 RESULTADOS ESPERADOS

Al finalizar el proyecto, se tendrán los siguientes documentos como resultado del mismo:

- Documento que recopile literatura concerniente a los métodos de solución al problema de localización de depósitos y ruteo de vehículos (LRP), el método de solución metaheurístico optimización colonia de hormigas (ACO), Búsqueda local iterativa (ILS) y algoritmos híbridos que se han desarrollado para dar solución al LRP.
- Programa elaborado en MATLAB donde se implemente el framework basado en la metaheurística híbrida de optimización colonia de hormigas (ACO) y búsqueda local iterativa (ILS) para la solución del LRP junto con los respectivos experimentos numéricos que validaran su eficiencia.
- Artículo de carácter publicable donde se presentarán los procedimientos y resultados obtenidos con el trabajo de investigación.

2 MARCO TEÓRICO

2.1 GENERALIDADES

2.1.1 Instalación o facilidad. El término es usado en los problemas de localización para definir un objeto cuya posición espacial es optimizada teniendo en cuenta la forma como interactúa con objetos pre-existentes en el problema, ejemplo de esto son almacenes, centros de distribución, cajeros, redes bancarias, depósitos de materiales, etc. Las instalaciones tienen tres propiedades principales: costo (costo fijo relacionado con el emplazamiento de las instalaciones y costo variable directamente relacionado con la distancia entre cliente y facilidad), número y tipo (características tales como capacidad, servicio y consideraciones de estructura).[12]

2.1.2 Localización. La localización hace referencia al lugar físico donde se ubicará la instalación o depósito, el conjunto de ubicaciones es llamado espacio de solución y puede ser representado de forma continua, discreta o sobre redes como lo se describe a continuación[13]:

- Espacio continuo: se emplea en los modelos de generación de sitios por el algoritmo de búsqueda, donde se parte de las coordenadas en un espacio euclidiano de dos dimensiones en el caso más típico.
- Espacio discreto: se especifica un conjunto finito de posibles lugares para ubicar las instalaciones. Como la solución se obtiene partiendo de un conjunto de facilidades candidatos, estos son modelos llamados de selección.

[12]ALVAREZ, Cesar. Optimización de enjambre de partículas (PSO) aplicada al problema de la P-mediana. Bucaramanga. Universidad Industrial de Santander. 2013.p.25
[13] Ibid.,p.26

- Representación en redes: comúnmente llamada grafo, empleado en redes viales, telecomunicaciones, oleoductos, etc. Este espacio pude ser continuo, cuando los posibles lugares de ubicación yacen tanto en vértices como en cualquier punto sobre los arcos del grafo, o discreto cuando únicamente los vértices son candidatos para ubicar las instalaciones.

2.1.3 Cliente. Este término se usa para denotar los objetos que requieren o demandan acceso a un producto o servicio, de los clientes se debe conocer su comportamiento, demanda y distribución.[14]

- Distribución: los clientes se distribuye uniformemente en el espacio o en los vértices de una red.
- Demanda: a cada cliente se le asigna un valor que representa la cantidad de servicio o producto que requiere. Esta demanda puede ser estocástica o determinista.
- Comportamiento: se refiere a las preferencias del cliente por seleccionar la instalación que le presta el servicio u ofrece el producto.

2.1.4 Ruteo de vehículos. El VRP es una generalización del problema del agente viajero (TSP), introducido por primera vez por Dantzig y Ramser (1959), es un problema de optimización combinatoria clasificado como NP-hard por su complejidad. Consiste en el diseño de rutas óptimas para entrega o recolección de productos de uno o varios depósitos a un conjunto de clientes dispersos geográficamente. Con VRP, se plantea resolver objetivos como:

- Minimizar el costo de transporte total: costo asociado a la distancia y el uso de vehículos.
- Minimizar el número de vehículos requeridos para servir a todos los clientes.

[14] Ibid., p.27

- Minimizar la penalización dada por un servicio parcial a clientes. Situación que se presenta cuando la capacidad del vehículo no es suficiente para suplir la demanda total del cliente.
- Combinación del peso de estos objetivos, conocidos como problemas multi-objetivo.[15]

2.1.5 Metaheurística. Una metaheurística es un conjunto de conceptos algorítmicos que se pueden utilizar para definir los métodos heurísticos aplicables a un amplio conjunto de problemas diferentes[16]. En otras palabras, una metaheurística puede ser vista como un método heurístico de propósito general diseñado para guiar una heurística fundamental de un problema específico hacia regiones prometedoras del espacio de búsqueda que contienen soluciones de alta calidad[17].

2.2 PROBLEMA CONJUNTO DE LOCALIZACIÓN DE INSTALACIONES Y ENRUTAMIENTO DE VEHÍCULOS (LRP)

Actualmente, incluso las pequeñas y medianas empresas son conscientes que su éxito en el futuro puede depender de las decisiones de ubicación y distribución en que se basa el diseño de su gestión logística[18]. Las investigaciones recientes en el diseño de redes de logística han demostrado que el costo de distribución general

[15] ACOSTA, Francia, OSORIO, Daniela. Un método evolutivo de colonia de hormigas para la solución del problema de ruteo de vehículos con demandas estocásticas. Bucaramanga. Universidad Industrial de Santander. 2013. p. 36.
[16] DORIGO Marco, BIRATTARI Mauro, and Thomas St¨utzle. Ant Colony Optimization. Artificial Ants as a Computational Intelligence Technique. IEEE COMPUTATIONAL INTELLIGENCE MAGAZINE. NOVEMBER 2006
[17] DORIGO Y STÜTZLE. Ant Colony Optimization. Bradford Books. MIT Press, Cambridge, MA. 2004.
[18] BELENGUER, J.-M., BENAVENT, E., PRINS, C., PRODHON, C., & WOLFLER-CALVO, R. A. branch-and-cut method for the capacitated location-routing problem. Computers and Operations Research, 38(6), 931–941. 2011.

puede ser excesivo si las decisiones de enrutamiento se ignoran cuando se localizan los depósitos[19].

Se considera que la primera publicación sobre el LRP, la realizó Maranzana en 1964, cuando indicó que la ubicación de fábricas, almacenes y puntos de suministro en general, con frecuencia se ve influenciada por los costos de transporte; aunque incorpora la trayectoria más corta, en lugar del problema enrutamiento de vehículos, en un problema de localización[20].

Watson-Gandy y Dohrn (1973), son considerados como los primeros autores en plantear claramente el LRP a través de un modelo no lineal que incluía la disminución de distancias entre clientes y puntos de venta. Salhi y Rand (1989), mostraron los beneficios que se obtienen al combinar las decisiones de ruteo de vehículos y localización de depósitos, teniendo en cuenta que su consideración por separado puede llevar a soluciones subóptimas[21].

El problema de localización de instalaciones y enrutamiento de vehículos se puede definir según Bruns (1998), como la planificación de ubicación con aspectos de planificación de recorrido tomados en cuenta. Por lo tanto, el problema es jerarquizado, es decir, se debe resolver el problema principal de localización, pero para lograr el objetivo es necesario resolver el sub-problema de enrutamiento simultáneamente. Debido a que si se tratan estos problemas por separado es posible encontrar soluciones sub-óptimas y además representa un mayor costo en la cadena logística.

---

[19] MIN. Hokey, JAYARAMAN. Vaidyanathan, SRIVASTA. Rajest. Combined Location-Routing Problems: A Synthesis and Future Research Directions. European Journal of Operational Research. 1997. p. 2.

[20] NAGY., Óp. cit. p. 667.

[21] PRODHON, C., & PRINS, C. A survey of recent research on location-routing problems. European Journal of operational research. 2014

El problema conjunto de localización de instalaciones y enrutamiento de vehículos (LRP) se formula como se presenta por Yu, Lin, Lee, y Ting (2010): dado un conjunto de clientes con una demanda conocida y un conjunto de posibles sitios de depósito, se busca determinar la ubicación de los depósitos y las rutas de los vehículos de depósitos a clientes para la entrega de pedidos. El objetivo es minimizar la suma de los costos asociados a la localización de los depósitos y la distribución a los clientes. Hay un costo fijo asociado con la apertura de un depósito en cada sitio potencial, y un costo de distribución asociado con el enrutamiento de vehículos que incluye el costo de instalación de rutas y el costo de transporte, que es proporcional a la distancia total recorrida por los vehículos. Cada cliente es asignado a un depósito del que se enviará un vehículo para cumplir con su demanda. Una ruta vehicular parte y termina en el mismo depósito. Puede haber otras limitaciones prácticas, tales como limitaciones en el tiempo de viaje total de una ruta, la distancia total recorrida por un vehículo, la capacidad de un vehículo, o la capacidad de un depósito[22].

Nagy y Salhi realizan una revisión de la literatura hasta el año 2007, para el Problema conjunto de Localización de Instalaciones y Ruteo de Vehículos, y presentan una clasificación de este problema, la cual permite tener una perspectiva de las diferentes maneras en que ha sido abordado el tema, además incluye una lista de artículos en los cuales es aplicado LRP. La clasificación se estructura de acuerdo a ocho aspectos que son: la estructura jerárquica, tipo de datos de entrada, periodo de planificación, método de solución, función objetivo, espacio de soluciones y número de depósitos.

Para la solución de los problemas determinísticos, existen métodos exactos y métodos de solución heurísticos. Laporte y Norbert (1981), presentan el primer algoritmo exacto para el LRP general utilizando un algoritmo *branch-and-*

[22] YU, V. F., LIN, S.-W., LEE, W., & TING, C.-J. A simulated annealing heuristic for the capacitated location routing problema. Computers & Industrial Engineering 58. 2010. 288–299

bound, donde se debe seleccionar solo un depósito, se utiliza un número fijo de vehículos y se resuelven casos con hasta 50 clientes. Laporte ha aportado estudios en los años 1983 y 1988, en donde se han resuelto óptimamente los casos generales de la ubicación y enrutamiento con hasta 40 ubicaciones de los depósitos potenciales y 80 clientes. En Laporte, Nobert y Arpin (1986), la solución a un LRP con limitaciones de capacidad del vehículo es obtenido por un método *branch and cut*. Las restricciones de eliminación de sub recorridos y las limitaciones de la cadena de restricción garantizan que cada ruta comience y termine en la misma instalación. Los métodos exactos proporcionan importantes conocimientos sobre los problemas, pero debido a la complejidad de la localización y enrutamiento, sólo pueden abordar instancias relativamente pequeñas.

Laporte, et al. (1986), realizan la primera formulación matemática para el LRP con limitaciones de capacidad para depósitos y vehículos, la cual se puede definir de la siguiente manera:

Sea $G = (N, E, C)$ un grafo, donde $N = \{1, \dots, n\}$ es el conjunto de nodos (que representa los sitios), E es el conjunto de aristas no dirigidas (i, j) y $C = c_{ij}$ es la matriz simétrica de rutas de menor costo asociadas con los bordes C, y satisface $c_{ij} \leq c_{ik} + c_{kj}$ para todo $i, j, k \in N$. Teniendo en cuenta que (i, j) y C_{ij} sólo se definen si $i < j$. Sea $R \subset N$ un conjunto de depósitos potenciales. El número P de dichos nodos utilizados como depósitos en la solución óptima debe estar entre dos límites preestablecidos $\underline{P} \geq 1$ y $\dot{P} \leq |R|$. El costo de utilización de nodo r como un depósito es igual a g_r. Hay m_r vehículos idénticos basados en el depósito r, cada uno con la misma capacidad de D, pero con diferentes costes fijo F_r. Para cada nodo de $N - R$, es decir, los nodos que no eran sitios potenciales de depósito, que asocian un requisito de servicio no negativo $d_i (< D)$. Teniendo en cuenta que si alguno de los posibles lugares de depósito r no se utilizan como depósitos, no es necesario seguir examinando. Los posibles sitios de depósito no

tienen ningún requisito servicio como los nodos de $N - R$. También tenga en cuenta que, dado que $d_i < D$, nunca habrá una necesidad de un nodo a ser visitado por más de un vehículo para satisfacer sus requerimientos de servicio.

El problema consiste en la selección de depósitos (donde $\underline{P} < |R|$, para determinar cuántos vehículos se asignan a cada depósito seleccionado, y establecer las rutas de los vehículos de tal manera que:

- Cada ruta inicia y termina en el mismo almacén
- Cada cliente es atendido por un solo vehículo en una sola visita.
- La suma de todas las demandas atendidas por un vehículo no debe ser superior a D.
- El número de nodos utilizados como depósitos está entre los límites: $\underline{P}$ y $\dot{P}$: $1 \leq \underline{P} \leq P \leq \dot{P} \leq |R|$.
- Para cada nodo r utilizado como depósito, el número de vehículos se encuentra entre: $\underline{m}_r$ y $\dot{m}_r$: $1 \leq \underline{m}_r \leq m_r \leq \dot{m}_r$
- El costo total se debe minimizar.

Formulación

Se define:

S: Un subconjunto de $N - R$.

L: Un número arbitrariamente grande.

t: $\begin{cases} El\ menor\ entero\ mayor\ o\ igual\ a\ t\ si\ t > 0 \\ 1\ de\ otra\ manera. \end{cases}$

x_{ij} Variable que indica el número de veces que el borde (i, j) es usado en la solución óptima.

x_{ij} no es definida si $i > j$, si $i, j \in R$ o si $d_i + d_j > D$.

x_{ij} debe interpretarse x_{ji} siempre que $i > j$.

y_r: variable binaria que indica si el nodo r se utiliza como un depósito $(y_r = 1)$ o no $(y_r = 0)$

Z: costo total del sistema.

Función objetivo:

Minimizar Z: $\sum_{i,j \in N} Cijxij + \sum_{r \in R}(gr\ yr + fr\ mr)$ *(1)*

Sujeto a:

$\sum_{i<k} x_{ik} + \sum_{k<j} x_{kj} = 2 \qquad (k \in N - R)$ *(2)*

$\sum_{i<r} x_{ir} + \sum_{r<j} x_{rj} = 2\, m_r \qquad (r \in R)$ *(3)*

$\sum_{i,j \in S} x_{ik} < |s| - \left\lceil \frac{\sum_{k \in S} d_k}{D} \right\rceil \qquad (S \subset N - R, |S| \geq 3)$ *(4)*

$x_{i_1 i_2} + 3x_{i_2 i_3} + x_{i_3 i_4} \leq 4 \qquad (i_1, i_4 \in R;\ N - R;\ i_2, i_3 \in N - R)$ *(5)*

$x_{i_1 i_2} + x_{i_{h-1} i_h} + 2 \sum_{i.j \in \{i_2, \ldots, i_{h-1}\}} x_{ij} \leq 2h - 5$

$(h \geq 5; i_1, i_h \in R;\ i_2, \ldots, i_{h-1} \in N - R)$ *(6)*

$y_r \leq m_r \leq L y_r \qquad (r \in R)$ *(7)*

$\underline{m}_r \leq m_r \leq \dot{m}_r$ *(8)*

$\underline{P} \leq \sum_{r \in R} y_r \leq \dot{P}$ *(9)*

$y_r = 0{,}1 \qquad (r \in R)$ *(10)*

$$x_{ij} = \begin{cases} 0{,}1 & (i, j \in N - R) \\ 0{,}1{,}2 & (i\ o\ j \in R) \end{cases}$$ (11)

En esta formulación, la función objetivo se define como la suma de los costos de viaje, costos de operación de depósitos y los costos asociados al uso de vehículos. La restricción (2) especifica que cada sitio no utilizado como un depósito debe ser visitado una sola vez por un determinado vehículo.

La restricción (3) asegura que la solución no contiene ningún subtour disjuntos de R o subtours con un peso total superior a D.

Las restricciones (4) y (5) aseguran que cada ruta inicia y termina en el mismo depósito.

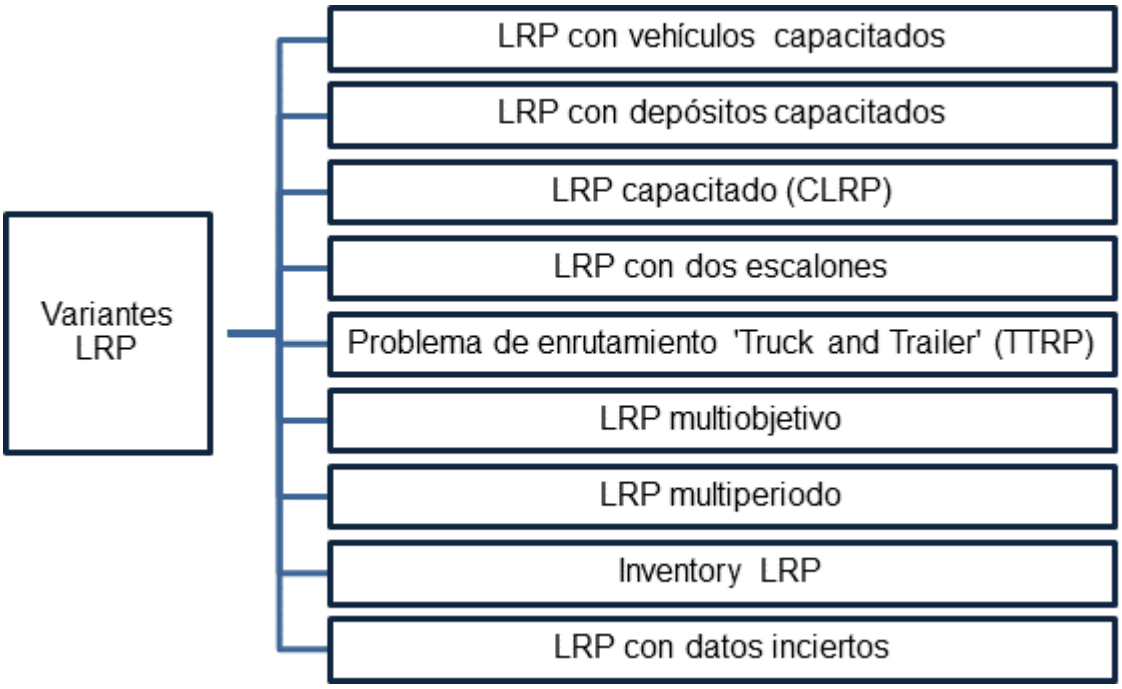
La restricción (6) significa que ningún vehículo puede pasarse a un nodo que fue asignado a un depósito; por otra parte, si un nodo se utiliza como un depósito, que debe tener al menos un vehículo asignado a éste.

La restricción (7) indica que el número de vehículos asignados a cualquier depósito dado debe estar entre los límites pre-especificados. Del mismo modo, la restricción (8) limita el número total de depósitos.

La restricción (9) Asegura que haya al menos P depósitos abiertos en la solución óptima. Por último, la restricción (10) indica el número de veces que se utiliza la arista (i, j) en la solución óptima; $x_{ij} = 2$ corresponde a un viaje de retomo entre i y j.

Prodhon y Prins (2014) presentan las siguientes variantes del problema conjunto de localización de instalaciones y ruteo de vehículos:

Figura 1. Variantes del LRP



En la siguiente tabla se mencionan los diferentes métodos de solución empleados para LRP en sus diferentes variantes, en resumen al estado del arte presentado en los subcapítulos posteriores.

Tabla 2. Métodos de solución LRP

MÉTODOS DE SOLUCIÓN LRP		
Exactos	- Branch and Price - Branch and cut - Branch and bound	
Aproximados	- Optimización colonia de hormigas (ACO) - Búsqueda local iterativa (ILS) - Búsqueda Tabú (TS) - Búsqueda en entorno variable (VNS) - Recocido simulado - Algoritmo genético - Algoritmo Clarke and Wright - *Adaptive epsilon-constraint algorithm (AECA)*	
	Algoritmos híbridos	- Algoritmo genético con gestión de la población (MA/PM) - Búsqueda local evolutiva (ELS) + *Randomized extended Clarke and Wright Algorithm (RECWA).* - Procedimiento codicioso de búsqueda local adaptativa aleatorizado (GRASP)+ Búsqueda local evolutiva (ELS). - Algoritmo genético (GA)+ Búsqueda local iterativa (ILS). - Heurística de Lin y Kemighan + Búsqueda tabú (TS).

2.2.1 Formulación matemática LRP. El problema conjunto de localización de instalaciones y enrutamiento de vehículos (LRP) se puede definir como sigue: dado un conjunto de clientes con una demanda conocida y un conjunto de posibles sitios de instalación del depósito, se busca determinar la ubicación de los depósitos y las rutas de los vehículos de los depósitos a los clientes.

Existe un costo fijo F_j asociado con la apertura de un depósito en cada sitio potencial y un costo variable relacionado con las rutas que atienden la demanda de cada cliente. El objetivo es minimizar el costo total, compuesto por el costo de localización de instalaciones y el costo de distribución. Cada cliente es asignado a un depósito que envía un solo vehículo para cumplir con su demanda. Una ruta de vehículo debe partir y terminar en el mismo depósito[23]. La capacidad de un

[23] LEE W., LIN Y., TING C. & YUV. A simulated annealing heuristic for the capacitated location routing problem. Computers & Industrial Engineering 58. 2010 288–299.

depósito abierto también representa la capacidad del vehículo para la ruta asociada con el depósito[24].

En este trabajo se aborda el LRP considerando múltiples depósitos con limitaciones de capacidad y un solo vehículo asignado por depósito. La formulación matemática para la solución de este problema, está basada en Wu, Low, y Bai (2002) y Yu et al. (2010).

Sea $G = (V, E)$ un grafo no dirigido, donde V es un conjunto de nodos que comprenden un subconjunto J de m sitios candidatos de depósito y un subconjunto $I = V \setminus J$ de n clientes. E es un conjunto de aristas que conectan cada par de nodos en V.

Los conjuntos, parámetros, y variables utilizadas en el modelo matemático se definen a continuación[25]:

Conjuntos

$\boldsymbol{I}$ Conjunto de todos los clientes

$\boldsymbol{J}$ Conjunto de todos los sitios potenciales de depósito

$\boldsymbol{K}$ Conjunto de rutas que equivale al total de depósitos seleccionados

Parámetros

$\boldsymbol{N}$ Número de clientes

$\boldsymbol{C_{ij}}$ Matriz de costos asociados a la distancia entres los puntos i, j. $i, j \in I \cup J$

$\boldsymbol{F_j}$ Costo fijo por instalación del depósito j

$\boldsymbol{R_j}$ Capacidad máxima del depósito j

$\boldsymbol{d_i}$ Demanda del cliente i

[24]ALBAREDA-SAMBOLA. M., DÍAZ. J. & FERNÁNDEZ. E. A compact model and tight bounds for a combined location-routing problem. Computers & Operations Research 32. 2005

[25]BAI J., LOW C. & WU T. Heuristic solutions to multi-depot location-routing problems. Computers & Operations Research 29. 2002

Variables de decisión

$\boldsymbol{x_{ijk}}$

$= \begin{cases} 1, & \text{Si el nodo } i \text{ precede inmediatamente el nodo } j \text{ en la ruta } k \\ 0, & \text{En caso contrario} \end{cases} \qquad (i, j \in I \cup J)$

$\boldsymbol{y_j} = \begin{cases} 1, & \text{Si se selecciona el depósito } j \\ 0, & \text{En caso contrario} \end{cases}$

$\boldsymbol{z_{ji}} = \begin{cases} 1, & \text{Si el cliente } i \text{ se asigna al depósito } j \\ 0, & \text{En caso contrario} \end{cases}$

$\boldsymbol{U_{lk}}$: Variable auxiliar para la eliminación de las restricciones del sub − tour en la ruta k

$\boldsymbol{Z}$: Costo total

En el modelo no se contemplan los costos variables asociados al uso de vehículos y almacenamiento en depósitos.

Función Objetivo

$$Min\ Z = \sum_{j \in J} F_j\, y_j \ + \ \sum_{i \in I \cup J} \sum_{j \in I \cup J} \sum_{k \in K} C_{ij} x_{ijk}\ , \qquad (12)$$

La función objetivo minimiza la suma del costo fijo por seleccionar un depósito y el costo de las rutas, que se define como la suma de los costos de cada viaje en la ruta.

Sujeto a:

$$\sum_{k \in K} \sum_{j \in I \cup J} x_{ijk} \ = \ 1, \quad i \in I \qquad (13)$$

Cada cliente es visitado por exactamente una ruta.

$$U_{lk} - U_{ik} + Nx_{lik} \leq N - 1, \quad l,i \in I,\ k \in K, \qquad (14)$$

Conjunto de restricciones para la eliminación de sub-tour. $U_{lk}\ y\ U_{ik}$ Son números arbitrarios reales. Las variables reales U determinan una cantidad estrictamente creciente a lo largo de la ruta, es decir, $U_{ik} \geq U_{lk} + 1$ si i es visitado inmediatamente después que l. Esta desigualdad fue propuesta por Miller, Tucker y Zemlin (1960)[26].

$$\sum_{i \in I \cup J} x_{jik} - \sum_{i \in I \cup J} x_{ijk} = 0, \quad k \in K,\ i \in I \cup J, \qquad (15)$$

$$\sum_{j \in J} \sum_{i \in I} x_{jik} \leq 1, \quad k \in K, \qquad (16)$$

Garantizan la continuidad de cada ruta. Cada ruta termina en el depósito donde inició.

$$\sum_{i \in I} d_i\ z_{ij} - R_j y_j \leq 0, \quad j \in J, \qquad (17)$$

Restricción de capacidad para los depósitos. La demanda de los clientes asignados al depósito j no puede superar la capacidad de este.

$$\sum_{u \in I} x_{juk} + \sum_{u \in V \setminus \{i\}} x_{uik} \leq 1 + z_{ji}\ ,\ \forall j \in J,\ i \in I, k \in K, \qquad (18)$$

Especifica que un cliente se puede asignar a un depósito sólo si se abre una ruta que los conecta.

$$x_{ijk} \in \{0,1\}, \quad \forall i \in I,\ j \in J,\ k \in K, \qquad (19)$$

$$y_j \in \{0,1\}, \quad \forall j \in J, \qquad (20)$$

[26] OLIVERA. A. Heurísticas para Problemas de Ruteo de Vehículos. Instituto de Computación, Facultad de Ingeniería, Universidad de la República, Montevideo, Uruguay. 2004 [en linea] [citado 15 de febrero de 2015]. Disponible en. www.fing.edu.uy/inco/pedeciba/bibliote/reptec/TR0408.pdf

$$z_{ij} \quad \in \{\ 0,1\}, \quad \forall\ i\ \in\ I,\ \ j\ \in\ J, \qquad (21)$$

Requisitos binarios en las variables de decisión.

$$U_{lk} \quad \geq \quad 0, \qquad l\ \in\ I,\ k\ \in\ K, \qquad (22)$$

La variable auxiliar U_{lk}, toma valores positivos.

2.2.2 Métodos exactos en la solución de LRP. Algunos métodos exactos se han tratado a partir de la publicación de Nagy y Salhi (2007), como lo exponen Alka, Berger y Ralphs (2009) quienes describen un algoritmo '*branch and price'* para la solución de LRP, con limitaciones de capacidad tanto en depósitos como en vehículos (CLRP), con aplicación de la descomposición 'Dantzing Wolfe' dado un conjunto de ubicaciones de instalaciones candidatos y un conjunto de ubicaciones de clientes.

Belenger, Benavent y Prins (2011), proponen un enfoque exacto basado en un algoritmo '*Branch and cut'* para solucionar CLRP, este método se basa en un modelo lineal cero-uno reforzado por nuevas familias de desigualdades válidas. Los experimentos computacionales, demuestran que éste puede resolver problemas con un máximo de 50 clientes. Con este mismo enfoque, 'branch and cut', Karaoglan, Aliparmak, Kara y Dengiz (2011) resuelven un LRP con entrega y recogida simultánea (LRPSPD por sus siglas en inglés), empleando recocido simulado para obtener límites superiores. Éste resuelve casos con un máximo de 88 clientes y 8 depósitos en tiempo computacional razonable.

2.2.3 Heurísticas y metaheurísticas para solución de LRP. Debido a que el LRP es *NP-Hard*, la mayor parte de las investigaciones utilizan heurísticas para su solución[27] con el objetivo de encontrar soluciones de calidad en un tiempo

[27] CHING-JUNG Ting & CHIA-HoChen. A multiple ant colony optimization algorithm for the capacitated location routing problem. Int. J. Production Economics 141 2013. 34–44.

razonable, dado que en la mayoría de los casos el tiempo computacional requerido por métodos exactos resulta inaceptable[28]. En Nagy y Salhi (2007), se presenta una clasificación para la heurística de acuerdo a diferentes métodos de solución estos son: secuenciales, basados en agrupación, iterativos y jerárquicos.

Wu et al, 2002, plantean un método iterativo, que se basa en una búsqueda tabú combinada y el marco de recocido simulado, pero con una estructura de vecindad sencilla. Se consideran los puntos finales de los tours como la entrada a la fase de localización, se investigan todos los pares de clientes, y también se incluye la duración de los viajes en el cálculo de los costos variables del modelo de ubicación. Además, estos costos variables se ajustan para tener en cuenta una flota heterogénea.

Albareda-Sambola et al. (2005), encuentran una solución inicial a través de un modelo de programación lineal. Esta solución es el límite inferior para el método de solución y además el punto de partida de la heurística de búsqueda tabú (TS). La heurística TS consiste en una serie de iteraciones de intensificación / diversificación. Se propone un nuevo límite inferior que tiene dos términos que se derivan de la estructura de cada uno de los dos componentes principales del problema original: El primero se refiere principalmente a los costos de ubicación y se resuelve utilizando un problema especial de la mochila (KP), y el segundo a los costos de enrutamiento y se resuelve utilizando el problema del viajante de comercio asimétrico ad hoc (ATSP).

Albareda-Sambola, Fernández, y Laporte (2007), presentan un trabajo que considera un LRP estocástico con limitación de capacidad en los depósitos, y un vehículo para cada depósito abierto. Se propone un modelo de dos etapas, en la primera se determina un conjunto de depósitos abiertos y unas rutas a priori que

[28] ALVAREZ, Cesar. Optimización de enjambre de partículas (PSO) aplicada al problema de la P-mediana. Bucaramanga. Universidad Industrial de Santander. 2013.p.45

visitan cada cliente exactamente una vez. Y en la segunda etapa, se diseñan las rutas reales a posteriori, en el momento en que se conoce el conjunto de clientes que requieren el servicio. Se propone un modelo matemático, un límite inferior y una heurística. El límite inferior se calcula por separado limitando los costos asociados con los depósitos y el costo de las rutas reales. La heurística determina un subconjunto mínimo de depósitos abiertos por la solución de un problema de "knapsack", se asignan los clientes a estos depósitos por la solución de un problema estocástico de asignación generalizada, y se construyen las rutas. Finalmente se aplica a la solución resultante un procedimiento de búsqueda local.

Barreto, López Borges, Ferreira, Sousa santos (2008), presentan una herramienta de apoyo a las decisiones que implementa un problema conjunto de localización de instalaciones y enrutamiento de vehículos con limitaciones de capacidad (CLRP), con dos niveles (depósitos y clientes), y una flota de vehículos homogénea y con limitaciones de capacidad. Se utiliza un método secuencial, más específicamente una heurística secuencial de primero- distribución segundo-localización, utilizando varias técnicas de agrupamiento jerárquico y no jerárquico, y consiste en cuatro pasos principales: construir grupos de clientes con una capacidad limitada; determinar la distribución en cada grupo de clientes; mejorar las rutas y localizar los depósitos y asignar las rutas de estos.

Barreto, Ferreira, Paixao y Sousa Santos (2007), analizan el LRP con límites de capacidad en vehículos empleando una heurística secuencial basada en el análisis de clúster usando técnicas de agrupamiento y proximidad, para obtener varias versiones de la heurística. Donde, inicialmente se construyen grupos de clientes, se determina la ruta en cada grupo, se mejora y finalmente se localizan los centros de distribución y se asignan las rutas. Los mejores resultados se observan cuando se utiliza método jerárquico de una sola fase y 'average measure' de las técnicas de proximidad.

Caballero, González, Guerrero, Molina y Paralera (2007), presentan un modelo aplicado a un caso real para la ubicación de dos plantas de incineración de desechos sólidos de animales y encontrar al mismo tiempo las mejores rutas para el transporte de los residuos desde 93 plantas de sacrificio (mataderos). Se incluyen objetivos económicos y sociales como el rechazo social de habitantes por cuyos pueblos pasan las rutas o de comunidades aledañas a la planta de incineración, emplean MOAMP (Metaheurístico multiobjetivo mediante un procedimiento de memoria adaptativa) que por medio de búsqueda tabú, genera un conjunto inicial de puntos factibles.

Aksen y Altinkemer (2008), emplean relajación Lagrangeana para descomponer LRP en subproblemas independientes. En éste, los depósitos cuentan con capacidad ilimitada mientras los vehículos tienen restricción de capacidad. La demanda que es determinística, está dada por clientes que se dirigen a las tiendas minoristas y clientes on-line para quienes el tiempo de llegada de su pedido se maneja de forma estática, es decir se omite la naturaleza probabilística en los tiempos de entrega.

El trabajo presentado por Ambrosino, Sciomachen, y Scutellà (2009), propone un LRP que se caracteriza por un depósito central, un conjunto de clientes divido en regiones, y una flota heterogénea de vehículos. El problema consiste en localizar un depósito para cada región en correspondencia con algunos sitios de los clientes, la asignación de los vehículos a las regiones y el diseño de las rutas de los vehículos de tal manera que la demanda total de cada ruta no exceda la capacidad del vehículo. Se propone una heurística de dos fases para este problema, la primera determina una solución factible inicial y, a continuación, se mejora mediante el uso de grandes técnicas de búsqueda local de vecindad (local search neighborhood).

Yu, Lin, Lee y Ting (2010), se centran en la solución de LRP con limitaciones de capacidad en depósitos y rutas. Se utiliza un esquema de representación para la solución del LRP, en el cual las rutas pueden ser terminadas al azar por los ceros simulados, o por las limitaciones de la capacidad de la ruta. Este problema se denomina SALRP, debido a que el algoritmo de solución es una heurística de recocido simulado (SA).

Tavakkoli-Moghaddama, Makuib y Mazloomic en 2010, plantean un problema LRP biobjetivo: reducción al mínimo del costo total y maximizar el total de la demanda servida simultáneamente. Los autores proponen una nueva formulación matemática con una búsqueda dispersa multi-objetivo (MOSS) y un algoritmo de búsqueda tabú elite (ETS) para obtener una frontera de Pareto-óptima y validar tanto su calidad de la solución y el nivel de la diversidad en los problemas de prueba con 2-10 depósitos, 3-8 vehículos, y 5-40 clientes.

Pirkwieser y Raid (2010), presentan una aplicación de búsqueda en entorno variable (VNS) para solucionar LRP periódico (PLRP) con restricciones en la capacidad de depósitos y vehículos, donde los clientes deben ser atendidos en varias ocasiones durante el periodo de planificación. En la solución, se introducen procedimientos de búsqueda en entorno variable muy grande (VLNS) basados en programación lineal entera (ILP) que se aplica tanto a LRP como a PLRP.

Jabal-Ameli, Aryanezhad, y Ghaffari-Nasab (2011), presentan un CLRP (Problema con limitaciones de capacidad de localización de instalaciones y enrutamiento de vehículos). En este se describe un algoritmo de Búsqueda Descendente de Entorno Variable (VND), que es la versión determinista de la Búsqueda de Entorno Variable (VNS), para resolver el CLRP. Se explora sucesivamente un conjunto de estructuras de entornos predefinidos para proporcionar una mejor solución. Se asume que los clientes adquieren su demanda necesaria de un único proveedor. La solución inicial se calcula mediante

la resolución de un problema de primera ubicación y asignación y luego la construcción de las rutas para cada almacén con ayuda de algoritmo de ahorro de Clarke y Wright para el CVRP (Clarke & Wright, 1964).

Rath y Gutjahr (2011), desarrollan una técnica de solución basada en el 'algoritmo Epsilon-restricción adaptativo (AECA)' para dar solución al CLRP en la ubicación de almacenes y transporte de elementos de ayuda humanitaria para atender emergencias provocadas por desastres naturales. Aunque se trata de LRP con restricciones de capacidad, los vehículos pueden regresar al depósito para cubrir otras rutas, cumpliendo con un tiempo límite. En el modelo se deben cumplir tres objetivos: minimizar costos fijos por ubicación de depósitos, minimizar costos operativos y maximizar la cobertura de la demanda. Para analizar el rendimiento del algoritmo a gran escala, se generan datos artificiales donde las coordenadas de clientes y depósitos se originan aleatoriamente de acuerdo con una distribución uniforme y se encuentra que este enfoque resuelve casos más grandes que los que presentan Rath, Doerner y Gutjahr (2009).

Manzour-al-Ajdad, Torabi y Salhi (2011), estudian un LRP plana con instalación única (PSFLRP) que consiste en encontrar la ubicación de un solo depósito con capacidad ilimitada en un espacio continuo y determinar las rutas de los vehículos procedentes de una instalación elegida. Para esto, proponen un algoritmo jerárquico basado en esquemas de búsqueda local que exploran el vecindario de la ubicación integrando intensificación y diversificación. A pesar de superar las heurísticas hasta ese momento propuestas para PSFLRP, requiere mayor esfuerzo computacional.

Carnes y Shomoys (2011), desarrollan un esquema primal-dual usando relajación de Lagrange para proporcionar un algoritmo de 2- aproximación y dar solución al k-LRP, donde k es la cota superior del número de depósitos que pueden ser

abiertos, este trabajo complementa el presentado por Goemans y Williamson (1995).

Karaoglan, Altiparmak, Kara y Dengiz (2011), consideran una variante del LRP que incluye entrega y recogida simultánea (LRPSPD) que surge del contexto de logística inversa, para el cual desarrollan un enfoque heurístico en dos fases (localización y enrutamiento de vehículos) basado en la metaheurística recocido simulado, tp_SA, para instancias de mediano y gran tamaño, dada una solución inicial generada gracias al uso dos enfoques heurísticos secuenciales de inicialización, el primero llamado 'Algoritmo Clarke y Wright extendido' (ECWA) que implementa penalidades de costos para resolver los problemas de localización y asignación de rutas heurísticamente; y el segundo es una 'heurística para el problema de localización de instalaciones' (FLPH) que resuelve el problema de ubicación optimo y el de ruteo de forma heurística. Las dos fases mencionadas se coordinan de tal forma que el espacio de solución se explora de manera eficiente.

Albareda-Sambola, Fernández y Níquel (2012), consideran el problema multiperiodo de ubicación de instalaciones y enrutamiento de vehículos con escalas de tiempo desconectadas (MLRPDS). El MLRPDS es un problema de ubicación de la instalación discreta definida en un horizonte de tiempo finito, en el que las decisiones de ubicación y enrutamiento se toman en diferentes escalas de tiempo. Debido a la alta complejidad de este problema, se propone una aproximación basada en la sustitución de rutas de vehículos por árboles de expansión, y su capacidad para proporcionar soluciones de buena calidad se evalúa en una serie de experimentos computacionales.

Hua-Li, Xun-Qing, & Yao-Feng (2012), tratan una aplicación para el LRP: la programación de un sistema de emergencias urbano, en el cual se debe planificar el suministro eficiente de las operaciones de socorro y establecer un sistema de gestión de desastres, que incluye dos componentes clave de un sistema de

logística, en especial ubicación de la instalación y el ruteo de vehículos. Proponen un modelo de programación de dos niveles: maximizar la satisfacción del tiempo total servido y minimizar costos totales. Es utilizado un algoritmo genético para la solución del problema.

Doulabi y Seifi (2013), desarrollan una heurística de inserción para combinar las decisiones de localización al problema de ruteo con arco capacitado (LARP) junto con una heurística de ubicación y asignación dentro de un marco de recocido simulado demostrando un significativo potencial de ahorro en costos.

Jarboui, Derbel, Hanafi, y Mladenovic. (2013), proponen diversas heurísticas de búsqueda local (VNS) para resolver el problema de localización de instalaciones y enrutamiento de vehículos; con múltiples depósitos con limitaciones de capacidad y un vehículo por depósito. Se integra un descenso de vecindad variable (VND) como la búsqueda local, dentro de la heurística general de vecindad variable (VNS) para resolver este problema. Se compone de cinco estructuras de vecindad, que son algunas de las variantes de los operadores de inserción y de intercambio definidos sobre las decisiones de enrutamiento y la ubicación.

Ehrenberg y Zimmerman (2014), consideran una variante del LRP de muchos a muchos que combina recolección y entrega, el problema implica tres decisiones: ubicación de centros, asignación de los clientes a los centros y la construcción de rutas de partida y llegada a los centros establecidos; donde proponen un algoritmo genético que intenta simular la teoría de Darwin de la selección natural. El método se aplica iterativamente a una población de soluciones de modo que en promedio cada nueva generación tiende a ser mejor que la anterior según los criterios de aptitud predeterminados. El algoritmo pretende maximizar y el criterio de terminación es generalmente un número específico de generaciones o un tiempo límite de ejecución para seleccionar la mejor solución encontrada.

2.2.4 Algoritmos híbridos en la solución de LRP. Prins y Prodhon (2008), desarrollan un Algoritmo memético con gestión de la población (MA/PM), este método genético hibrido combina técnicas de búsqueda local y gestión de la población que filtra la entrada de descendencia en la población gracias a un umbral de distancia dado, para resolver LRP con vehículos y depósitos capacitados (CLRP) y una combinación de días de servicio para cada cliente.

Dentro de la literatura existen pocas publicaciones disponibles referentes a un problema un poco más realista, LRP periódico o PLRP. En 2011, Prodhon abordó este problema combinando VRP periódico y LRP. Consiste en la adición de un horizonte multi-periodo al LRP y consta de tres tipos de decisiones: determinar el conjunto de depósitos a ser abiertos, la combinación de días de servicio que se debe asignar a los clientes y las rutas provenientes de cada depósito para cada uno de los períodos, con el fin de minimizar el costo total. Se propone un algoritmo híbrido evolutivo para resolver los casos de gran tamaño de la PLRP con dos depósitos y vehículos capacitados. El método combina una búsqueda local evolutiva (ELS) con una heurística basada en el *Randomized Extended Clarke and Wright Algorithm* (RECWA) para crear soluciones factibles. El método ha sido probado en tres tipos de casos a gran escala, con casos especiales: como horizonte temporal de un día (LRP) o un depósito (PVRP).

Duhamel, Lacomme, Prins y Prodhon (2010), para resolver el LRP con restricciones de capacidad (CLRP), proponen el método de solución 'procedimiento codicioso de búsqueda adaptativa aleatorizado (GRASP)' combinado con 'búsqueda local evolutiva (ELS). Este híbrido, toma las ventajas de ambos métodos, a saber, un enfoque multi-inicio del GRASP que proporciona soluciones iniciales mejorándolas con una búsqueda local, y la eficiencia del ELS para generar nuevas soluciones en cada iteración, de las cuales selecciona la mejor. El punto clave para la eficiencia del GRASP-ELS es alternar giras TSP (problema del agente viajero) con n clientes y soluciones CLRP completas.

Derbel, Jarboui, Hanafi y Cabchoub (2012), desarrollan un algoritmo híbrido combinando un algoritmo genético (GA) con búsqueda local iterativa (ILS) con cuatro estructuras de barrio, tratando un LRP con múltiples almacenes capacitados y vehículos con capacidad ilimitada. Alternando GA e ILS en el enfoque híbrido se refleja la intersección entre la característica de búsqueda global que se obtiene mediante el uso de GA (diversificación) y la capacidad de ILS para encontrar óptimos (intensificación). El algoritmo propuesto supera, en términos de costo total, los resultados obtenidos por Albreda-Sambola, Díaz y Fernández (2005) que emplean búsqueda tabú.

Escobar, Linfati y Toth (2012) proponen un algoritmo híbrido de dos fases para solucionar CLRP con un conjunto de demandas deterministas. El algoritmo, consta de una fase de construcción cuyo objetivo es generar una solución factible inicial S_0 que generalmente emplea tiempos computacionales muy cortos, empleando un procedimiento híbrido que combina técnicas exactas con la heurística de Lin y Kemighan (LKH, por sus siglas en inglés) para reconocer grupos de clientes que se pueden visitar con la misma ruta; seguido de un procedimiento de división para minimizar el costo de ruteo. La segunda fase, llamada de mejora, consiste en mejorar la solución S_0 dada en la primera fase aplicando búsqueda tabú granular modificado con diferentes estrategias de diversificación para probar la calidad de la solución actual. Además, considera un procedimiento de perturbación aleatoria para evitar que el algoritmo se estanque en un óptimo local para un número dado de iteraciones.

2.3 INTELIGENCIA DE ENJAMBRES

La inteligencia de enjambres es una disciplina de la inteligencia artificial, que tiene que ver con el diseño de los sistemas multi-agentes inteligentes y hace referencia al comportamiento colectivo de los insectos sociales tales como hormigas,

termitas, abejas y avispas, y a las propiedades para mantener un enjambre o una colonia. Muchos aspectos de las actividades colectivas de los insectos sociales son auto-organizados y trabajan sin un control central, por ejemplo, las hormigas cortadoras de hojas, cortan piezas en forma de hojas[29].

Como se menciona anteriormente, una característica de la inteligencia de enjambres, es que los agentes computacionales de una población son capaces de percibir y modificar su ambiente de manera local, lo cual hace posible la comunicación entre los individuos que detectan los cambios en el ambiente generado por el comportamiento de toda la población. Para esta disciplina existen diferentes técnicas de optimización, entre estas se encuentran la optimización por enjambre de partículas y la optimización por colonia de hormigas, como las principales técnicas; y por otro lado están algoritmos de optimización por enjambre de bacterias, la búsqueda por difusión estocástica y el algoritmo de colmena de abejas[30].

2.4 METAHEURÍSTICA DE OPTIMIZACIÓN COLONIA DE HORMIGAS (ACO)

Optimización colonia de hormigas (ACO) es una metaheurística basada en el comportamiento natural de las hormigas en su proceso de búsqueda de alimento. La tendencia lógica de cada individuo es reducir el esfuerzo y el tiempo necesario para recolectar el alimento, lo cual logra al disminuir la distancia entre el hormiguero y la fuente de alimento. Una hormiga es un individuo relativamente simple, pero llevar a cabo su labor resulta altamente complejo. El éxito radica en la interacción de muchos individuos con el ambiente y la comunicación indirecta entre ellos por medio de sustancias químicas conocidas como feromonas. Este comportamiento es usado para encontrar soluciones de buena calidad a problemas de optimización caracterizados por un espacio de solución bastante

[29] BLUM Christian y LI. SWARM Xiaodong. Intelligence in Optimization. Natural Computing Series 2008, pp 43-85

[30] MUÑOZ Mario A., LÓPEZ Jesús A. y CAICEDO Eduardo F.. Ob. Cit.

amplio y complejo. Para esto se debe hacer una equivalencia entre la colonia de hormigas naturales y un sistema artificial que se mueve dentro de un ambiente computacional.[31]

La metaheurística ACO se ha clasificado como parte de la computación evolutiva, más específicamente dentro de la inteligencia de enjambres. La computación evolutiva, es una aplicación que reúne modelos y experimentos inspirados, por lo general en el paradigma de la selección natural de Darwin y modelan la evolución como un proceso particular de optimización. Los algoritmos de computación evolutiva (EC) comprenden un conjunto de técnicas iterativas que manejan una población de individuos que son evolucionados o modificados mediante una serie de reglas que han sido claramente especificadas. En cada iteración hay periodos de auto adaptación, los cuales implican cambios en el individuo; son alternados con periodos de cooperación, lo que implica el intercambio de información entre individuos[32].

De acuerdo a ciertas características principales como: existencia de individuos que pueden representar soluciones a un problema, proceso evolutivo para definir los cambios en la población, medida de factibilidad de la solución obtenida, mecanismo de intensificación y de diversificación y un mecanismo que permita identificar la información de un individuo; la computación evolutiva es dividida en dos grupos: la inteligencia de enjambres y algoritmos evolutivos. Las diferencias entre estos dos grupos radica en que la característica principal en el primero, es que los agentes pueden modificar su ambiente y en el segundo grupo no puede ser modificado. Mientras que para la inteligencia de enjambres, la interacción con

[31] TORO, Eliana, SANTA, Jhon y GRANADA, Mauricio. Solución del problema de ruteamiento de vehículos en la distribución de papa en Colombia. Tesis Ingeniería Industrial. Universidad Tecnológica ce Pereira. Pereira. 2013. p. 5.

[32] MUÑOZ Mario A., LÓPEZ Jesús A. y CAICEDO Eduardo F. Inteligencia de enjambres: sociedades para la solución de problemas (una revisión). Revista Ingeniería e Investigación Vol. 28 No. 2, Agosto De 2008 (119-130).

la función objetivo corresponde al ambiente al ser explorado, para algoritmos evolutivos corresponde al nivel de desempeño[33].

La optimización por colonia de hormigas (ACO), fue propuesto inicialmente por Colorni, Dorigo y Maniezzo (1991). Es una de las técnicas más exitosas en el campo de la inteligencia de enjambres y está inspirada a partir del comportamiento de algunas especies de hormigas. Las hormigas reales encuentran rutas más cortas entre sus colonias y la fuente de los alimentos mediante la explotación de información de la feromona. El comportamiento anterior de las hormigas reales ha inspirado el sistema de hormigas, el cual es un algoritmo en el que un conjunto de hormigas artificiales cooperan para la solución de un problema mediante el intercambio de información a través de la feromona depositada en los bordes del grafo[34].

La metaheurística ACO presenta tres fases para las iteraciones, después de que se ha iniciado el algoritmo. Estas fases son: en cada iteración, las hormigas construyen un número de soluciones; estas soluciones se mejoran a través de una búsqueda local, y, finalmente, se actualiza la feromona[35].

Los problemas que se pueden resolver por la optimización de colonia de hormigas son problemas de optimización combinatoria y se pueden caracterizar por los siguientes aspectos[36]:

- Un conjunto finito de componentes $C = \{ c1, c2, \dots c_l\}$, es dado.
- Existe un conjunto finito de restricciones definido por el problema a solucionar.

[33] Ibíd., p. 120.
[34] DORIGO Marco y GAMBARDELLA Luca Maria. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. Ieee transactions on evolutionary computation, vol. 1, no. 1, april 1997
[35] DORIGO Marco, BIRATTARI Mauro y T ST¨UTZLE homas. Ant Colony Optimization. Artificial Ants as a Computational Intelligence Technique. IEEE Computational Intelligence Magazine November 2006
[36] DORIGO y DI Caro. The Ant optimization Meta-Heuristic. Iridia. 1999

- El problema presenta diferentes estados que son definidos en términos de secuencias de componentes š $= < c1, c2, \ldots c_r \ldots >$ sobre los elementos de C. Si α es el conjunto de todas las secuencias posibles y el conjunto q̃, son todas las sub secuencias y es un sub conjunto de α. Los elementos en q̃ definen los estados posibles del problema. La longitud de la secuencia š, es decir, el número de componentes en esta secuencia, se expresa por | š|.
- Una solución β es un elemento de q̃, que verifica todos los requisitos del problema.
- Se asigna una estructura de vecindario definida por: $š_2$ es un vecino de $š_1$ si: (i) tanto $š_1$ como $š_2$ pertenecen a α, (ii) el estado de $š_2$ puede alcanzarse desde $š_1$ en un paso lógico, es decir, si r es el último componente de la secuencia $š_1$, debe existir una componente $s \in$ ¥ tal que $š_2 =< š_1, s >$.
- Existe un costo $C(\beta)$ asociado a cada solución β.

En ACO, una hormiga artificial construye una solución por la que atraviesa el grafo de construcción totalmente conectado $G_c(V, E)$, donde V es un conjunto de vértices y E es un conjunto de aristas. Este grafo se puede obtener a partir del conjunto de componentes de la solución C en dos formas: los componentes pueden ser representados ya sea por los vértices o por los bordes. Las hormigas artificiales se mueven de vértice a vértice a lo largo de los bordes del grafo, incrementalmente en construcción de una solución parcial. Además, las hormigas depositan una cierta cantidad de feromona en los componentes; es decir, ya sea en los vértices o en los bordes que atraviesan. La cantidad de feromona depositada puede depender de la calidad de la solución encontrada. Las siguientes hormigas utilizan la información de la feromona como una guía hacia las regiones prometedoras del

espacio de búsqueda[37]. El grafo G_c se llama grafo de construcción y los elementos de E se denominan conexiones[38]. Por lo tanto, tenemos que[39]:

- Las secuencias de nodos o aristas, son los estados $\check{s}$ que son correspondidos con caminos en el grafo,
- las componentes nr son los nodos del grafo,
- las aristas del grafo, a_{rs} son conexiones que definen la estructura del vecindario. $\check{s}_2 =< \check{s}_1, s >$ será vecino de $\check{s}_1$ si el nodo r es la última componente de $\check{s}_1$ y la arista a_{rs} existe en el grafo,
- deben existir los costos explícitos c_{rs} asociados con cada arista,
- las componentes o las conexiones deben tener asociados rastros de feromona τ, que representan un tipo de memoria indirecta y a largo plazo del proceso de búsqueda, como valores heurísticos h, que representan la información de la heurística disponible en el problema a resolver.

Un procedimiento de optimización local o la recopilación de información global se pueden utilizar para decidir si es útil o no depositar feromonas adicionales para sesgar el proceso de búsqueda de una perspectiva no local.[40].

El proceso de actualizar las feromonas, consiste en modificar los rastros de la sustancia, dejados por la hormiga. El valor de los senderos puede aumentar, como

[37] DORIGO Marco, BIRATTARI Mauro y STÜTZLE Thomas. Ant Colony Optimization, Artificial Ants as a Computational Intelligence Technique. IEEE Computational Intelligence Magazine. November 2006

[38] DORIGO Y STÜTZLE. Ant Colony Optimization. Bradford Books. MIT Press, Cambridge, MA. (2004)

[39]ALONSO, Sergio. CORDÓN, Oscar. FERNÁNDEZ DE VIANA, Iñaki. HERRERA, Francisco. La Metaheurística de Optimización Basada en Colonias de Hormigas: Modelos y Nuevos Enfoques. Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S. Ingeniería Informática, C/ Periodista Daniel Saucedo Aranda s/n, 18071 Granada (España)

[40] DORIGO Y STÜTZLE. Ant Colony Optimization. Bradford Books. MIT Press,Cambridge, MA. 2004

las hormigas depositan la feromona en los componentes o conexiones que utilizan; o puede disminuir debido a la evaporación de feromonas[41].

Tabla 3. Algoritmo: Metaheurísitica ACO

Algoritmo: Metaheurística ACO

Establecer parámetros, inicializar los rastros de feromona.
while la condición de terminación no se cumpla do
ConstructAntSolutions- construir las soluciones hormiga
ApplyLocalSearch (opcional)- *aplicar búsqueda local*
UpdatePheromones- actualizar feromona.
End while

Fuente: B*IRATTARI M., DORIGO D, & STÜTZLE T*. Ant Colony Optimization. *Artificial Ants as a Computational Intelligence Technique*. IEEE Computational Intelligence Magazine. 2006. p. 31.

2.4.1 Principales Algoritmos de Optimización Basados en Colonia de Hormigas. En la literatura se han propuesto varios algoritmos de ACO. Dentro de los principales algoritmos disponibles para problemas de optimización combinatoria *NP-hard,* se encuentran el *Sistema de hormigas (SH),* el *Sistema de Colonia de Hormigas (SCH),* el *Sistema de Hormigas Max-Min (SHMM), el SH con ordenación* y el *Sistema Mejor-Peor Hormiga (SMPH).* A continuación se presentan algunos de estos algoritmos ACO.

2.4.1.1 El Sistema de Hormigas: El Sistema de Hormigas fue el primer algoritmo ACO propuesto en la literatura, y fue desarrollado por Dorigo, Maniezzo y Colorni (1991). Su característica principal es que, en cada iteración, los valores de feromonas son actualizados por todas las *m* hormigas que han construido una solución en la propia iteración. La feromona τ_{ij}, asociado a la arista que une las ciudades i y j, se actualiza como sigue[42]: primero todos los rastros de feromona

[41] Ibíd.
[42] BIRATTARI M., DORIGO D, & STÜTZLE T.. Ant Colony Optimization. *Artificial Ants as a Computational Intelligence Technique*. IEEE Computational Intelligence Magazine. 2006 Pag. 3.

se reducen en un factor constante, implementándose de esta manera la evaporación de la feromona. En seguida, cada hormiga de la colonia deposita una cantidad de feromona que es en función de la calidad de la solución[43]:

$$\tau_{ij} \leftarrow (1-\rho)\tau_{ij} + \Delta\tau_{ij}^{k} \qquad (23)$$

Donde ρ es la tasa de evaporación, m es el número de hormigas, y $\Delta\tau_{ij}^{k} = f\big(C(s_k)\big)$ es la cantidad de feromona establecido en el borde $(i\,,j)$ por la hormiga k.

En la construcción de una solución, las hormigas selecciona la siguiente ciudad para ser visitado a través de un mecanismo estocástico. Cuando k hormiga está en la ciudad i y ha construido hasta ahora la solución parcial s^p, la probabilidad de ir a la ciudad j viene dada por[44]:

$$P_j^h \begin{cases} \dfrac{\tau_j^s(\eta_j)^{\alpha}}{\sum_{j\epsilon O_s^h}\tau_j^s(\eta_j)^{\alpha}} \;\; si\; c\; i\; j \;\in\; N(s^p), \\ 0,\; de\; lo\; contrario \end{cases} \qquad (24)$$

Donde $\mathrm{N}(s^p)$ es el conjunto de los componentes viables; es decir, los bordes $(i\,,l)$, donde l es una ciudad aún no visitado por la hormiga k. Los parámetros α y β controlan la importancia relativa de la feromona en comparación con la información heurística η_{ij}.

2.4.1.2 Sistema Colonia de Hormigas: Es uno de los primeros sucesores del SH que introduce modificaciones importantes con respecto al ACO. Se hace la introducción de una actualización de feromona local además de la actualización de feromona realizada al final del proceso de construcción (llamado actualización de

[43] Alonso, op. Cit pag 14.
[44] BIRATTARI. *Op. cit. P*ag. 31.

feromona fuera de línea). La actualización de feromona local se lleva a cabo por todas las hormigas después de cada paso de la construcción. Cada hormiga la aplica a la última arista recorrida[45]:

$$\tau_{ij} \leftarrow (1-\varphi)\tau_{ij} + \varphi\tau_0 \qquad (25)$$

Donde $\varphi \in (0,1]$ es el coeficiente de evaporación de feromona, y τ_0 es el valor inicial de la feromona.

La aplicación de la regla de actualización local hace que los rastros de feromona entre las conexiones recorridas por las hormigas disminuyan. Esto lleva una técnica de exploración adicional del SCH ya que las conexiones atravesadas por un gran número de hormigas son cada vez menos atractivas para el resto de hormigas, que las recorridas en la iteración actual. Esto ayuda a diversificar la búsqueda para que no todas las hormigas sigan el mismo camino y así, producir nuevas soluciones. La actualización de feromona fuera de línea, se aplica al final de cada iteración por sólo una hormiga, que puede ser la mejor solución de la iteración o la mejor solución global[46].

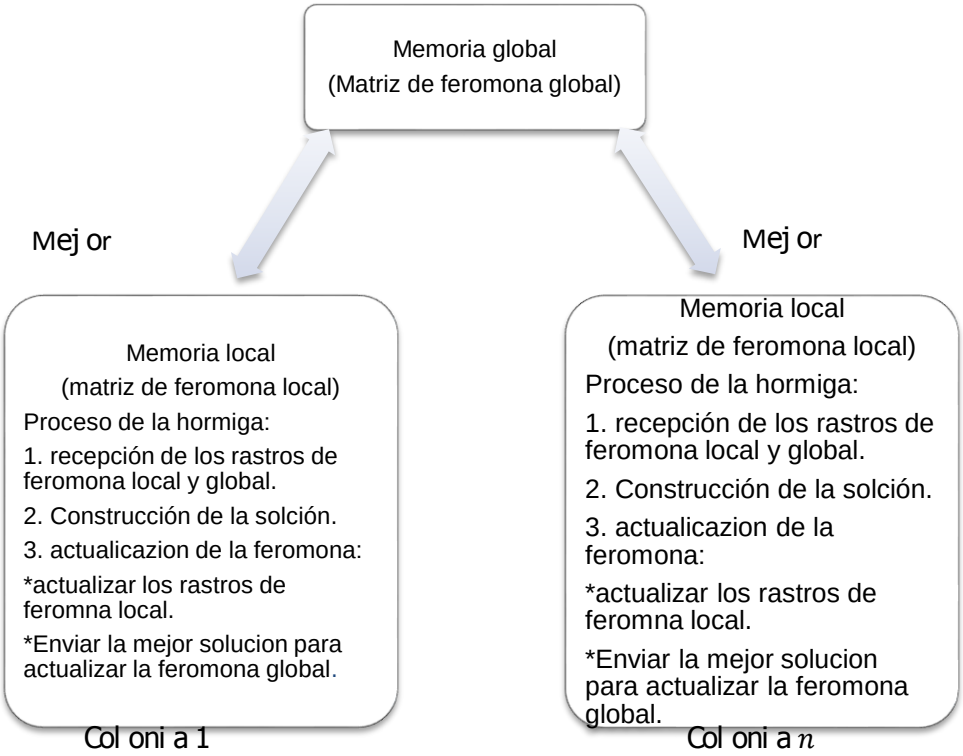
El SCH usa una regla de transición diferente a la del SH, y se denomina regla proporcional pseudo- aleatoria: la probabilidad de que una hormiga pase de la ciudad i a la ciudad j depende de una variable aleatoria q distribuida uniformemente sobre [0, 1], y un parámetro q_0; si $q \leq q_0$, entonces $j = \arg max \{\tau_j^s(\eta_j)^\alpha\}$, de lo contrario se usa la ecuación para el SH (Ec. 24)[47].

[45] Ibid. pag 32.
[46] Alonso. *Op. cit*. Pag 18
[47] B*IRATTARI. Op. cit. Pag 32*

2.4.1.3 Múltiples Colonias de hormigas. El MCH es un algoritmo basado en SCH, y fue propuesto por Gambardella en 1999, para solucionar un problema multi-objetivo de ruteo de vehículos con múltiples ventanas de tiempo.

En los algoritmos de múltiples colonias de hormigas, varias colonias cooperan para encontrar buenas soluciones y resolver un problema. La cooperación se lleva a cabo mediante el intercambio de información sobre la mejor solución[48]. La exploración del espacio de búsqueda en cada colonia se guía por diferente información heurística. Varias características específicas se introducen en el algoritmo con el fin de mejorar la eficiencia de la búsqueda[49].

Figura 2. Intercambio de información en el algoritmo múltiple colonia de hormigas



[48] MELO. L, PEREIRA. F & COSTA. E. MC-ANT: A Multi-Colony Ant Algorithm. Artificial Evolution. Springer. 2009.
[49] UDOMSAKDIGOOL. A. & KACHITVICHYANUKUL. V. 2008. Multiple colony ant algorithm for job-shop scheduling problem. International Journal of Production Research.

Fuente: Udomsakdigool. A, & Kachitvichyanukul. V. Multiple colony ant algorithm for job-shop scheduling problem. International Journal of Production Research. 2008.

2.4.1.4 Sistema de Hormigas Max- Min. El SHMM es una de las extensiones del SH que mejor rendimiento muestran. Fue desarrollado por Stützle y Hoos en 1996[50]. La mayor diferencia de SHMM con los demás algoritmos de ACO es la forma en que se realiza la actualización de la feromona. En la formulación original del algoritmo Colonia de Hormigas cada hormiga coloca cierta cantidad de feromona en cada paso. Mientras que, para el SHMM la actualización de feromona se implementa fuera de línea, de manera similar como se realiza en SCH. La mejor hormiga a la que se le permite añadir feromona puede ser la que tiene la mejor solución de la iteración o la mejor solución global[51].

Otros aspectos que caracterizan este algoritmo son los siguientes[52]:

Los valores posibles para los rastros de feromona están limitados al rango $[\tau_{min,}\, \tau_{max}]$. Por lo tanto, la probabilidad de un estancamiento del algoritmo disminuye al darle a cada conexión existente una probabilidad aunque bastante pequeña, de ser escogida.

El SHMM inicializa los rastros de feromona a una estimación del máximo permitido para un rastro. Esto lleva a una componente adicional de diversificación del algoritmo, debido a que al inicio las diferencias relativas entre los rastros de feromona no serán muy cargadas.

[50] Ibid. Pag 32.
[51] Alonso. Op. cit. Pag 20
[52] Ibid. Pag. 21.

2.4.2 Aplicaciones de la Metaheurística ACO. La metaheurística de optimización basada en colonia de hormigas (ACO), se ha utilizado para resolver diferentes tipos de problemas como el enrutamiento, asignación, problema de secuenciación de tareas y enrutamiento de red, entre otros. A continuación se presenta la aplicación de ACO en algunos problemas:

Dorigo y Gambardella (1997), presentan el problema del agente viajero (TSP), el cual consiste en encontrar un recorrido cerrado más corto que visita a todas las ciudades en un conjunto dado. En este estudio, el grafo de TSP está completamente conectado, las ciudades están en un plano y existe un camino (borde) entre cada par de ciudades. Para obtener la solución a este problema se utiliza una colonia de hormigas artifíciales capaces de generar recorridos factibles sucesivamente más cortos mediante el uso de información acumulada en forma de un rastro de feromonas depositado en los bordes del grafo de TSP.

Favaretto, Moretti *y* Pellegrini (2007), proponen un algoritmo basado en el sistema de colonia de hormigas (ACS), para la solución del problema de ruteo de vehículos con múltiples ventanas de tiempo (VRPMTW), teniendo en cuenta las estricciones periódicas, el tiempo total ponderado, viajes y el tiempo de espera, tiene que ser minimizado, satisfaciendo las limitaciones relacionadas con múltiples ventanas de tiempo y periodicidad. El ACS fue el primer algoritmo de ACO.

Rizzoli, Montemanni, Lucibello y Gambardella (2007), describen cómo la optimización de colonia de hormigas puede ser utilizado con éxito para resolver un número de variantes de la VRP, tanto para algunos de los casos de problemas básicos (VRP con limitaciones de capacidad, la VRP con ventanas de tiempo, la VRP con recogida y entrega) y para algunos de las extensiones dinámicas (el problema de ruteo de vehículos en función del tiempo y de la on-line vehículo problema de enrutamiento), donde se ha aplicado recientemente ACO y su capacidad de encontrar eficientemente soluciones en un corto período de tiempo.

Bulent Catay (2009), aplican ACO al problema de enrutamiento de vehículos con retiros y entregas (VRPPD). VRPPD determina un conjunto de rutas de vehículos originarios y que terminan en un solo depósito y visitan todos los clientes exactamente una vez. Los vehículos no sólo están obligados a entregar los bienes, sino también para recoger algunas mercancías de los clientes. Se abordan dos tipos de VRPPD: El problema mixto de ruteo de vehículos con los viajes de regreso (MVRPB) y el problema de ruteo de vehículos con recogida y entrega simultánea (VRPSPD).

Lee, ZJ Lee, Lin y Ying (2010), presentan una optimización por colonia de hormigas mejorada (EACO) para el problema de enrutamiento de vehículos capacitados. Para la solución del problema se utiliza el recocido simulado (SA) y la optimización basada en colonias de hormigas. En el algoritmo propuesto, el recocido simulado proporciona una buena solución inicial para la optimización basada en colonias de hormigas. Por otra parte, se utiliza la optimización por colonias de hormigas basada en la obtención de información para mejorar el rendimiento de la búsqueda.

Kazharov y Kureichik (2010), presentan un trabajo dedicado a la solución de los problemas de transporte con algoritmos de colonias de hormigas. Se aplican los algoritmos de colonias de hormigas genéticos y estándares para resolver este problema. Los resultados permiten juzgar la elección de los parámetros óptimos para la solución de los problemas de transporte con diversos datos iniciales. Se realiza una comparación experimental con las siguientes opciones de búsqueda: el algoritmo branch-and-bounds, el algoritmo vecino más cercano, el algoritmo genético modificado, y el algoritmo estándar de colonia de hormigas.

Hlaing y Khine (2011), proponen un enfoque para la solución del problema del viajante de comercio basado en un algoritmo de colonia de hormigas mejorado

con una heurística de optimización local. Se proponen dos modificaciones principales propuestas por ACO: las hormigas se colocan inicialmente en diferentes ciudades para evitar el estancamiento y para que la búsqueda de ACO sea más efectiva; y la entropía de la información que se introduce se ajusta a los parámetros del algoritmo.

Ting y Chen (2013), presentan un algoritmo de optimización basada en colonias de hormigas múltiple (MACO), para resolver la LRP con limitaciones de capacidad (CLRP) en depósitos y vehículos. Se propone una estructura jerárquica, con la ubicación de instalaciones (FLP), como el problema principal y enrutamiento de vehículos con múltiples depósitos (MDVRP), como un subproblema. Este estudio resuelve el CLRP utilizando métodos anidados basados en el algoritmo de optimización de colonia de hormigas. El MDVRP se resuelve dentro del problema de localización de instalaciones. En cada iteración, dos colonias de hormigas (selección de ubicación y asignación de clientes) se comunican entre sí a través de la regla global de actualización de feromona.

2.5 BÚSQUEDA LOCAL ITERATIVA (ILS)

El ILS es una metaheurística que consiste en la aplicación iterativa de una heurística de búsqueda local (LS) y el uso de un criterio de perturbación como mecanismo que favorece la exploración, permitiendo así encontrar mejores soluciones. En cada iteración, se genera una nueva solución inicial empleando la heurística LS, y ésta se convierte en el nuevo punto de partida para la búsqueda. En cada iteración, se aplica perturbación a la solución encontrada hasta el momento, y su ejecución se detiene cuando alcanza un número máximo de iteraciones o cuando cumple un criterio de parada específico. A pesar de su

simplicidad, los algoritmos basados en ILS han demostrado tener un enfoque muy eficaz para resolver problemas de optimización combinatoria[53].

El marco general del ILS es el siguiente:

- x_0 es la solución inicial para un proceso de ILS.
- Se aplica búsqueda local a x_0 generando una mejor solución $\hat{x}$.
- Se hace perturbación a $\hat{x}$ para obtener otra solución x'.
- Después de esto, se aplica de nuevo búsqueda local a x'para obtener un nuevo óptimo local $\tilde{x}$.

La solución $\tilde{x}$ reemplaza a $\hat{x}$ solo si se cumple el criterio de aceptación. Para la implementación de ILS en LRP, la perturbación se hace de la siguiente manera: se selecciona un depósito al azar y se trasladan los clientes ya asignados a otro depósito, esto genera una nueva solución que da la oportunidad de abrir un depósito que no ha sido explorado[54].

Tabla 4. Algoritmo: estructura general ILS:

1 Sea $\boldsymbol{x_0}$ la solución inicial
2 $\hat{\boldsymbol{x}} \leftarrow$ Búsqueda local ($\boldsymbol{x_0}$);
3 Repeat
4 $\boldsymbol{x} \leftarrow$ Perturbación ($\hat{\boldsymbol{x}}$);
5 $\tilde{\boldsymbol{x}} \leftarrow$ Búsqueda local ($\boldsymbol{x}$);
6 if FEVAL($\hat{\boldsymbol{x}}$) <FEVAL($\tilde{\boldsymbol{x}}$) then $\hat{\boldsymbol{x}} \leftarrow \tilde{\boldsymbol{x}}$;
7 Until Se cumple el criterio de parada

Fuente: Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. Genetic algorithm with iterated local search for solving a location-routing problem. Expert systems with applications. 2012. p.5.

[53] CUERVO, D., GOOS, P., SÖRENSEN, K. & ARRÁIZ, E. An iterated local search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research. 2014 p. 3.
[54] Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. (2010). An iterated local search for solving a location-routing problem. Electronic Notes in Discrete Mathematics, 36, 875–882.

2.5.1 Búsqueda Local. Una búsqueda local es un método de optimización clásica que consiste en generar un óptimo local, mediante la exploración en los alrededores de una solución dada. Esta potente metaheurística es aplicada a una gran cantidad de problemas de optimización combinatoria, Thierens (2004).

Uno de los componentes principales en el diseño de una búsqueda local es la elección de estructuras de vecindario éstas se construyen mediante la modificación de algunos de los componentes de una solución dada para generar una nueva, es decir, el vecindario define la forma de llegar a una nueva solución con pocas modificaciones, el uso de varias estructuras ayuda a guiar la búsqueda y a evitar el estancamiento en óptimos locales[55]. Derbel, Jarboui, Hanafi y Chabchoub (2012), emplean cuatro estructuras de vecindario denotadas N_1, N_2, N_3 y N_4, que tienen en cuenta únicamente los depósitos abiertos, de la siguiente manera:

N_1 y N_2 se llevan a cabo entre dos rutas diferentes con un intento de mejorar la solución. En el vecindario N_1, se realiza un movimiento de intercambio *(swap move)*, se seleccionan al azar dos clientes asignados a dos depósitos diferentes y se intercambian, se modifican las rutas, pero no el número ni el orden.

En el vecindario N_2 *(insertion move),* se selecciona un cliente de una ruta y se inserta en otra ruta, se debe verificar que la capacidad del depósito donde estará el cliente pueda satisfacer su demanda.

N_3 y N_4 se llevan a cabo en las mismas rutas. Para el vecindario N_3 *(swap move),* se intercambian las posiciones de dos clientes y para N_4 *(insertion move)* se inserta un cliente entre otros dos de estos.

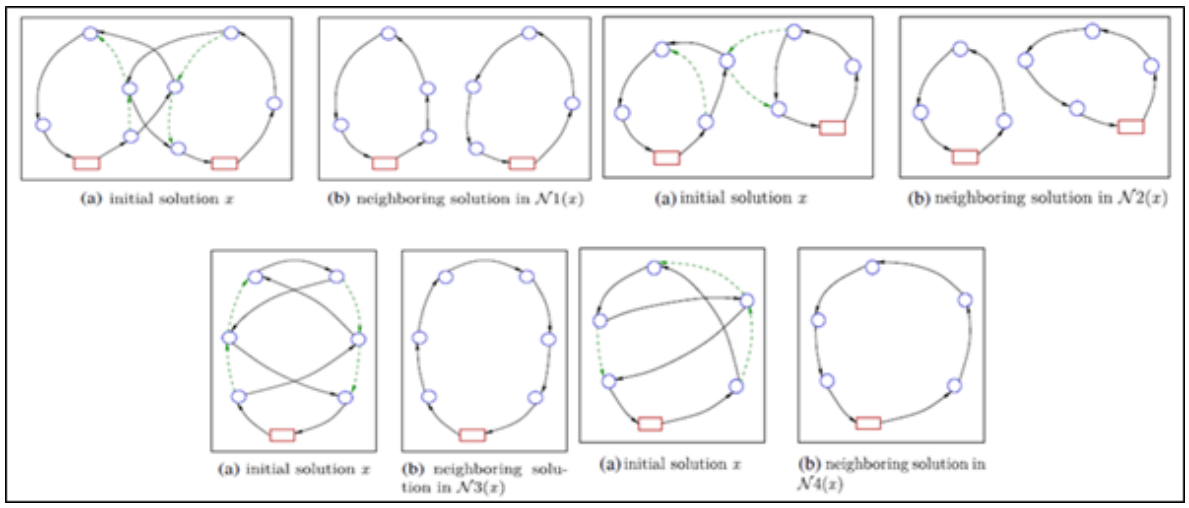
[55] Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. (2012). Genetic algorithm with iterated local search for solving a location-routing problem. Expert systems with applications. p. 5-6.

Tabla 5. Algoritmo: Búsqueda local

Input: $\mathbf{x}$: solución inicial
1 $\mathbf{x_1}\leftarrow$ Primera mejora de $\mathbf{x}$ usando la estructura de barrio $\mathbf{N_1}$
2 $\mathbf{x_2}\leftarrow$ Primera mejora de $\mathbf{x_1}$ usando la estructura de barrio $\mathbf{N_2}$
3 $\mathbf{x_3} \leftarrow$ Primera mejora de $\mathbf{x_2}$ usando la estructura de barrio $\mathbf{N_3}$
4 $\mathbf{x_4} \leftarrow$ Primera mejora de $\mathbf{x_3}$ usando la estructura de barrio $\mathbf{N_4}$
5 if FEVAL ($\mathbf{x_4}$) < FEVAL ($\mathbf{x_1}$) then
6 $\quad \mathbf{x} \leftarrow \mathbf{x_4}$;
7 $\quad$ Go to Línea 1

Fuente: Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. (2012). Genetic algorithm with iterated local search for solving a location-routing problem. Expert systems with applications. p. 6.

Figura 3. Estructuras de vecindad



Fuente: Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. (2012). Genetic algorithm with iterated local search for solving a location-routing problem. Expert systems with applications. p. 6.

2.5.2 Criterio de perturbación. La perturbación es un factor clave del algoritmo ILS porque determina la porción de la solución localmente óptima que se modifica. Si la perturbación es demasiado pequeña, el algoritmo de ILS tiende a quedarse atrapado en una solución localmente óptima. Por otra parte, si la

perturbación es demasiado grande, la información contenida dentro de la solución localmente óptima se pierde y el algoritmo ILS se comporta como si una solución inicial al azar se utilizara en cada iteración[56].

Derbel et al. (2012) emplean la perturbación con el interés de explorar depósitos cerrados y tratar un tipo diferente de soluciones, así, cada vez que una perturbación se realiza durante todo el procedimiento de ILS, se crea una nueva solución. Esta perturbación se da de la siguiente manera: se selecciona un depósito abierto al azar, los clientes asignados a este depósito son trasladados de manera arbitraria a otro, ya sea abierto o cerrado. Este tipo de movimiento genera un nuevo tipo de solución mediante el cierre/apertura de algunos depósitos.

2.5.3 Implementación ILS. La metaheurística de búsqueda local iterativa (ILS), ha sido aplicada para resolver diversos problemas logísticos, algunos de los cuales se mencionan a continuación:

Reichelt, Gmilkowsky y Linser (2005) implementaron ILS en el problema de diseño de redes de comunicación confiables, para éste, definen un método de inicialización, un operador de búsqueda local, un operador de mutación, un criterio de aceptación y un criterio de parada.

Para el problema de enrutamiento de vehículos con ventanas de tiempo (VRPTW por sus siglas en inglés), Ibaraki, Imahori, Nonobe, Sobue, Uno y Yagiura (2007) emplean un ILS cuyo método asigna los clientes a los vehículos y determina el orden de visita de cada uno.

Cordeau, Laporte y Pasin (2008), aplica ILS para resolver el problema de diseño de una red logística con asignación única (LNDP por sus siglas en inglés), donde

[56] CUERVO, D., GOOS, P., SÖRENSEN, K. & ARRÁIZ, E.(2014) An iterated local search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research. p. 5.

se toman decisiones respecto a las selección de proveedores, ubicación de plantas y almacenes, asignación de actividades a estas instalaciones y flujos de materias primas y producto terminado en la red.

Laurent y Hao (2009) desarrollan un algoritmo basado en ILS para resolver el problema de programación de vehículos con múltiples almacenes (MDVSP, por sus siglas en inglés), que tiene por objetivo la programación de vehículos de transporte público alojados en depósitos con capacidad limitada, para cubrir una secuencia de viajes consecutivos, satisfaciendo un conjunto de restricciones y minimizando una función de costo.

Derbel, Jarboui, Hanafi y Cabchoub (2010) desarrollan un enfoque heurístico basado en búsqueda local iterativa (ILS) para resolver LRP estático con depósitos capacitados y un solo vehículo sin límite de capacidad, donde se diversifica el espacio de soluciones a nivel de la ubicación mediante la apertura de nuevos subconjuntos de las instalaciones y luego se aplica intensificación en la fase de ruteo. Se proporciona un estudio computacional para comparar este enfoque con la heurística de búsqueda tabú, TS, propuesta por Albreda-Sambola, Díaz y Fernández (2005), donde los resultados obtenidos demuestran que el tiempo medio necesario para el cálculo de la solución es mayor para TS especialmente en casos pequeños.

Geiger (2011) propone y aplica una combinación de ILS con búsqueda en entorno variable (VNS) para la resolución del problema de optimización multi-objetivo: ‘flow shop scheduling’, esta combinación da lugar a dos principios de búsqueda: intensificación, a través de VNS y diversificación a través de perturbaciones e iteraciones en un espacio de búsqueda (ILS).

Un problema de ruteo de vehículos periódico con ventanas de tiempo (PVRPTS, por sus siglas en inglés), es abordado por Michallet, Prins, Amodeo, Yalaoui y

Vitry (2013), quienes proponen una metaheurística ILS multi-inicio (MS-ILS), que se basa en un procedimiento de búsqueda local, elaborado sobre la base de funciones de penalización por tramos lineales.

Cuervo, Goos, Sorensen y Arráiz (2014) emplean ILS para dar solución al VRP con Backhauls (VRPB, por sus siglas en ingles), que es una extensión de VRP que se ocupa de dos tipos de clientes: *linehaul y backhaul*, los primeros son los que requieren los bienes del depósito, y los segundos quienes envían mercancías al depósito, es decir los proveedores, utilizado generalmente para modelar el proceso de distribución de empresas que en un solo centro reciben materias primas y despachan producto terminado.

3 DESARROLLO DEL FRAMEWORK BASADO EN ACO E ILS PARA LA SOLUCIÓN DEL LRP

Para la solución del LRP se construye un algoritmo híbrido, que consta de dos etapas, en la primera etapa se encuentra la solución por medio de la heurística sistema colonia de hormigas (ACS, por sus siglas en inglés), y en la segunda etapa la solución hallada se mejora con la heurística búsqueda local iterativa (ILS, por sus siglas en inglés). El presente framework, para el algoritmo ACO, se basa en los planteamientos de Thing y Chen (2013), de cuyo trabajo se extraen las fórmulas de explotación y exploración para cada uno de los procesos. Thing y Chen (2013) proponen el uso de tres colonias, cada una con una matriz de feromona y sus respectivas normas de actualización. En la aplicación de ILS se tiene en cuenta lo planteado por Derbel, Jarboui, Hanafi y Chabchoub (2010).

A continuación se describe el comportamiento de cada una de las colonias:

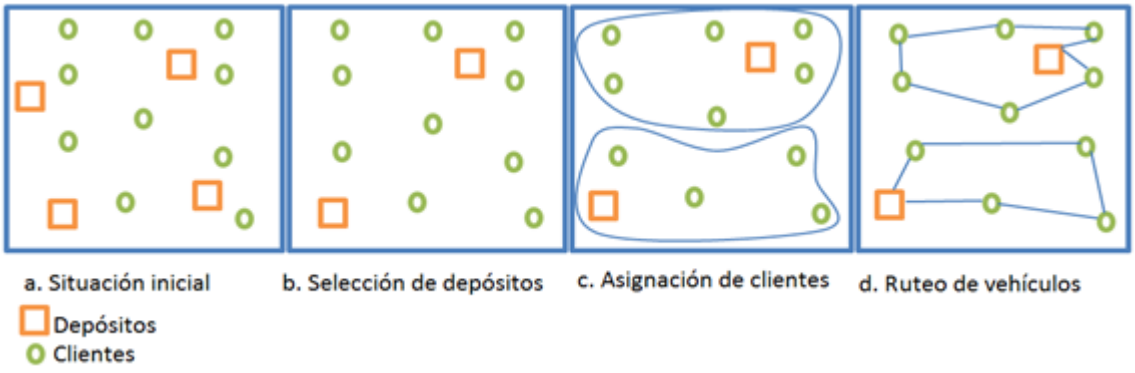
Primera colonia: Selección de depósitos. Para los problemas determinísticos, existen dos tipos de instalaciones para ser localizadas: *primarias,* si estas son los orígenes y destinos de los recorridos de los vehículos, y *secundarias* cuando sólo pueden ser depósitos intermedios[57]. Para este trabajo se consideran instalaciones primarias con una capacidad limitada.

Segunda colonia: asignación de clientes. En esta colonia

Tercera colonia: ruteo de vehículos.

[57] Albareda-Sambola M., Díaz J. & Fernandez E. opt. Cit. Pag. 408

Figura 4. Representación de un ejemplo solución para LRP



Las colonias trabajan de forma iterativa hasta cumplir un criterio de parada. En el presente trabajo, el criterio de parada será un número de iteraciones previamente establecido. En el algoritmo, se actualiza la feromona para el camino seguido (solución) por cada hormiga, haciendo además una actualización adicional para la mejor solución encontrada en todo el proceso. Esto permite un mejor equilibrio entre la exploración y la explotación.

La tabla 6, representa el algoritmo general de ACO+ILS que se implementa para dar solución al LRP, el proceso es descrito en los subcapítulos posteriores.

Tabla 6. Algoritmo: ACO+ILS para LRP

Algoritmo: ACO+ILS para LRP

Iniciar algoritmo

Para cada iteración, $s = 1 \dots S$

Para cada hormiga, $h = 1 \dots b$

Iniciar parámetros $(\alpha, \beta, \gamma, \rho, \rho', \rho'', q_0, q_0', q_0'', m, n, S', b')$

Exportar desde Excel datos de demanda, capacidad y costos

Construir matriz inicial de feromona para los procesos de selección y asignación.

Hallar cantidad de depósitos a seleccionar $\boldsymbol{P_s^h}$

Para cada $\boldsymbol{P_s^h}$

Seleccionar depósitos de los m *candidatos*

Contrastar capacidad de depósitos seleccionados vs demanda total

Actualización local de feromona para selección

Para cada cliente, $i = 1 \dots n$

Asignar clientes uno a uno al conjunto de depósitos seleccionados

Comprobar capacidad paso a paso

Actualización local de feromona para asignación

Fin n clientes

Ruteo de vehículos */*Para cada conjunto de depósitos y clientes asignados*/*

Fin $\boldsymbol{P_s^h}$

Encontrar la mejor solución de las $\boldsymbol{b}$ *construidas* $'\boldsymbol{Th}'$ */*Mejor de la iteración* $\boldsymbol{s}$**/*

Fin b hormigas

Encontrar mejor solución global $(\boldsymbol{Ts})$ */*Mejor encontrada hasta cada iteración* $\boldsymbol{s}$**/*

Actualización global de feromona

Aplicar búsqueda local iterativa (ILS) a $(\boldsymbol{Ts})$

Fin de las S iteraciones

Terminar algoritmo

3.1 SELECCIÓN DE DEPÓSITOS

Para la selección de depósitos, se tiene una colonia con una cantidad b de hormigas, que debe ser la misma para la asignación de clientes y un número de iteraciones S.

En cada iteración s, cada hormiga h, elige cierta cantidad de depósitos P_s^h, éste es el punto de partida para iniciar el proceso de selección, la ecuación (27) ilustra la forma de hallar este valor:

$$P_s^h = \left[\frac{\sum_{i=1}^{n} d_i}{\sum_{j=1}^{n} \frac{R_j}{m}}\right] + U(1, r) \qquad (26)$$

Se debe cumplir que $P_s^h \leq m$. Si esto no se cumple, P_s^h se hace igual a m.

La descripción de los términos en (27) es la siguiente:

$\boldsymbol{d_i}$ es la demanda del cliente $\boldsymbol{i}$

$\boldsymbol{R_j}$ es la capacidad del depósito $\boldsymbol{j}$

$\boldsymbol{m}$ es el número de depósitos candidatos

$[\boldsymbol{x}]$ es el menor entero mayor o igual a $\boldsymbol{x}$

$\boldsymbol{U(1, r)}$ es un número aleatorio (entero) que sigue una distribución uniforme en el intervalo $[\boldsymbol{1, r}]$

$\boldsymbol{r}$ es un número predeterminado, sirve para definir la cantidad de depósitos que se desea abrir, la cantidad $\boldsymbol{P_s^h}$ puede ser mayor (probablemente), cuanto mayor sea el valor de r que se elige.

Los $\boldsymbol{P_s^h}$ depósitos son escogidos por cada hormiga $\boldsymbol{h}$ sucesivamente de los $\boldsymbol{m}$ depósitos candidatos, de acuerdo a las reglas de selección que se muestran en las ecuaciones (28) y (29).

$$j = \begin{cases} \underset{j \epsilon O_s^h}{\text{argmax}} \left[\tau_j^s \left(\eta_j\right)^{\alpha}\right], & Si\ q\ \leq\ q_0 \\ J, & Caso\ contrario \end{cases} \qquad (27)$$

Para cada cliente i, dentro de O_s^h se elige el depósito j que de mayor valor a la función $\tau_j^s \left(\eta_j\right)^{\alpha}$ si $q \leq q_0$. En caso contrario, los depósitos se eligen del conjunto O_s^h usando la distribución de probabilidad dada por la ecuación (29):

$$J: \begin{cases} P_j^h = \frac{\tau_j^s (\eta_j)^{\alpha}}{\sum_{j \epsilon O_s^h} \tau_j^s (\eta_j)^{\alpha}}, & Si\ j \quad \epsilon\ O_s^h \\ P_j^h = 0, & Caso\ contrario \end{cases} \qquad (28)$$

Para esta ecuación, la descripción de los términos usados es:

O_s^h representa el conjunto de depósitos candidatos que no han sido aún seleccionado por la hormiga h en la iteración s.

$\boldsymbol{\tau_j^s}$ es la cantidad de feromona en el depósito j en la iteración s.

$\boldsymbol{\eta_j}$ es la relación entre capacidad (R_j) y el costo fijo por instalación (F_j) del depósito j:

$$\boldsymbol{\eta_j} = \frac{R_j}{F_j} \qquad (29)$$

$\boldsymbol{\alpha}$ es el parámetro que determina la influencia relativa de τ_j^s versus η_j $(\alpha > 0)$

$\boldsymbol{q}$ es una variable aleatoria con distribución uniforme entre $[0,1]$ con una frecuencia q_0 previamente definida.

La solución generada por cada hormiga, en cada iteración se representa en un conjunto llamado $\boldsymbol{W_s^h}$.

3.1.1 Regla de actualización local de feromona. Las hormigas en su recorrido van dejando rastros de feromona. Para el presente trabajo una hormiga artificial actualiza la feromona para la selección de depósitos mediante la siguiente regla:

$$\tau_j^s \leftarrow (1-\rho)\tau_j^s + \rho\tau_j^0 \quad si\ j \in T_h \qquad (30)$$

Donde

τ_j^0 es el valor inicial de la feromona, dado por la ecuación (32).

$$\tau_j^0 = \frac{1}{F_j} \qquad (31)$$

T_h es la solución construida por la hormiga h.

ρ es el parámetro de actualización de la feromona ($\rho \geq 0$)

Así, la matriz inicial de feromona para cada depósito *j* está dada de la siguiente manera:

$$\tau_j^0 = \left[\tau_1^0 = \frac{1}{F_1} \quad \tau_2^0 = \frac{1}{F_2} \quad \tau_3^0 = \frac{1}{F_3} \quad \ldots \quad \tau_m^0 = \frac{1}{F_m}\right]$$

(Ver ejemplo en el Anexo B)

3.2 ASIGNACIÓN DE CLIENTES

La asignación de clientes a depósitos se realiza mediante otra colonia de hormigas, el número de hormigas que se eligen es igual que para la selección de las instalaciones, esto es b . La colonia se encarga de asignar a cada cliente i un depósito k de los ya seleccionados $(i \neq k)$. Para cada conjunto W_s^h habrá una hormiga que haga la respectiva asignación.

El proceso de asignación define una relación uno a uno, es decir, cada cliente i se asigna a un único depósito k (se denotan con k los depósitos que hacen parte de W_s^h). La relación se define de acuerdo a la ecuación (33):

$$\mathrm{k} = \begin{cases} \max\limits_{\mathrm{k}\epsilon \mathrm{W}_\mathrm{s}^\mathrm{h}} \left[\xi_{\mathrm{ik}}^{\mathrm{s}} (\Psi_{\mathrm{ik}}^{\mathrm{s}})^{\beta}\right], & Si\ q' \leq q_0' \\ K, & Caso\ contrario \end{cases} \qquad (32)$$

Si $q' \leq q'_0$, el cliente i se asigna al depósito k que genere el mayor valor de la función $\xi_{ik}^s (\Psi_{ik}^s)^{\beta}$, en caso contrario se elige el depósito k dentro del conjunto W_s^h usando la distribución de probabilidad inducida por la ecuación (34):

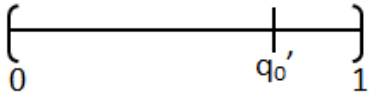
$$K: P_{ik}^h = \frac{\xi_{ik}^s (\Psi_{ik}^s)^{\beta}}{\sum_{k\epsilon W_s^h} \xi_{ik}^s (\Psi_{ik}^s)^{\beta}} \qquad (33)$$

La descripción de los términos en (32) y (33) es la siguiente:

$\boldsymbol{W_s^h}$ es el conjunto de depósitos seleccionados por la hormiga h en la iteración s.
$\boldsymbol{\xi_{ik}^s}$ es la cantidad de feromona entre el cliente i y el depósito k en la iteración s.
$\boldsymbol{\beta}$ es el parámetro que determina el efecto relativo de ξ versus Ψ.
$\boldsymbol{q}'$ es una variable aleatoria entre $[0,1]$
$\boldsymbol{q_0'}$ es el parámetro que determina la importancia relativa de la ecuación de explotación (33) versus la ecuación de exploración (34). Esto significa que cuanto más alejado sea q_0' de 0, menor será la importancia que se le dará a la ecuación de exploración, como se muestra en la figura.

En el ejemplo de la ilustración 8, se está dando mayor importancia a la ecuación de explotación.

Figura 5. Ubicación del parámetro q_0'



$\boldsymbol{\Psi_{ik}^{s}}$ es el reciproco deD_{ik}, donde:

$$\boldsymbol{D_{ik}} = \min_{l \epsilon A_k^{sh}} C_{il} \qquad (34)$$

$\boldsymbol{A_k^{sh}}$ es el conjunto de nodos (incluyendo el depósito k) que han sido asignados al sitio k por la hormiga h en la iteración s, (inicialmente sólo el depósito k).

$\boldsymbol{C_{il}}$ es el costo asociado a la distancia entre los nodos i y l.

Si el límite de la capacidad del depósito k se viola, la asignación de clientes se revisa a través de los siguientes pasos:

1. *Para cada regla de decisión se realiza la comparación de la capacidad de los depósitos seleccionados con la demanda de cada cliente.*
2. *Para la regla de decisión en donde* $q' \leq q_0'$*, si la capacidad del deposito* j *no cumple con la demanda, se asigna un valor muy grande a su costo asociado, por lo tanto, este depósito no es seleccionado por la función de* $argmax$*. Luego de asignar el cliente al depósito se actualiza la capacidad de este restando su demanda a la capacidad total de la instalación.*
3. *Para la regla de decisión en donde* $q' \geq q_0'$*, si la capacidad del depósito* j *no cumple con la demanda, se asigna un valor de 0 para la información heurística de este depósito, así se anula* la probabilidad de ser seleccionado. *Luego de asignar el cliente al depósito se actualiza la capacidad del mismo de la misma forma que el en numeral 3.*

3.2.1 Regla de actualización local de feromona. La regla de actualización de feromona para la asignación de clientes, se realiza mediante la ecuación (36):

$$\xi_{ij}^{s} \leftarrow (1-\rho')\xi_{ij}^{s} + \rho'\,\xi_0 \qquad si\ la\ arista\ (i,j) \in T_h \qquad (35)$$

Donde T_h es la solución construida por la hormiga h.

ρ' es el parámetro de evaporación de la feromona.

ξ_0 es el nivel inicial de feromona. Se puede tomar un número muy pequeño.

La matriz de feromona se representa así:

$$\xi_0 = \begin{bmatrix} \xi_{11} & \xi_{12} & \cdots & \xi_{1m} \\ \xi_{21} & \xi_{22} & \cdots & \xi_{2m} \\ \vdots & \vdots & & \vdots \\ \xi_{n1} & \xi_{n2} & \cdots & \xi_{nm} \end{bmatrix}$$

(ver el ejemplo en el anexo C)

3.3 RUTEO DE VEHÍCULOS

El siguiente paso que se lleva a cabo en el framework es la construcción de las rutas que realiza cada vehículo partiendo del depósito. El problema de ruteo de vehículos se considera como un problema independiente para cada uno de los depósitos seleccionados y sus respectivos clientes asignados.

La tabla 7 representa el algoritmo ACO para solucionar el problema de ruteo de vehículos, éste se describe en el desarrollo del presente subcapítulo.

Tabla 7. Algoritmo ACO para Ruteo de Vehículos

Algoritmo ACO para Ruteo de vehículos
Iniciar algoritmo
Para cada depósito seleccionado y respectivos clientes asignados
Para cada iteración, $s' = 1 \ldots S'$
Para cada hormiga, $h' = 1 \ldots b'$
Construir matriz inicial de feromona
Construir soluciones VRP

Actualización local de feromona

Encontrar la mejor ruta construida entre las b' hormigas$(T_{s'})$

Fin b' hormigas

Encontrar mejor ruta global entre las S' iteraciones$(T_{b'})$

Encontrar costo del peor recorrido entre las S' iteraciones (L'_w)

Actualización global de feromona

Fin de las S' iteraciones

Fin depósito seleccionado

Terminar algoritmo

Así, para cada depósito k se elige un conjunto S' de iteraciones y un número b' de hormigas que realizan la tarea de ruteo. La hormiga h', parte desde el depósito a un nodo de manera aleatoria, se mueve desde este nodo i a un nodo v siguiendo las reglas de construcción dadas por las ecuaciones 37 y 38.

Antes que una hormiga escoja el siguiente nodo a visitar, se genera un número aleatorio q", que determina la ecuación a seguir para la construcción de la ruta.

$$v = \begin{cases} \underset{v \epsilon Z_{s'}^{h'}}{\operatorname{argmax}} \left[\zeta_{iv}^{s'} \left(\varphi_{iv}^{s'}\right)^{\gamma}\right], & Si\ q" \leq\ q_0'' \\ V, & Caso\ contrario \end{cases} \qquad (36)$$

Si $q'' \leq q_0''$ la hormiga h' se mueve del nodo i al nodo v que genere el mayor valor a la función $\zeta_{iv}^{s'} \left(\varphi_{iv}^{s'}\right)^{\gamma}$, en caso contrario se elige el nodo dentro del conjunto $Z_{s'}^{h'}$ usando la distribución de probabilidad dada por la ecuación (38):

$$V: \begin{cases} P_{iv}^{h'} = \dfrac{\zeta_{iv}^{s'} \left(\varphi_{iv}^{s'}\right)^{\gamma}}{\sum_{v \epsilon Z_{s'}^{h'}} \zeta_{iv}^{s'} \left(\varphi_{iv}^{s'}\right)^{\gamma}}, & Si\ v \in\ Z_{s'}^{h'} \\ P_{iv}^{h'} = 0, & Caso\ contrario \end{cases} \qquad (37)$$

Una vez visitados todos los nodos, se retorna al depósito de inicio correspondiente.

La descripción de cada uno de los términos es la siguiente:

$\boldsymbol{Z_{is'}^{h'}}$ es el conjunto de nodos aún no visitados aún por la hormiga h' en la iteración s'

$\boldsymbol{\zeta_{iv}^{s'}}$ es la cantidad de feromona en la arista (i, v) en la iteración s'

$\boldsymbol{\gamma}$ es el parámetro que denota la influencia relativa de la información de la feromona versus la información heurística $(\gamma > 0)$

$\boldsymbol{q''}$ es una variable aleatoria dentro de una distribución $[0,1]$

$\boldsymbol{q_0''}$ es el parámetro que determina la importancia relativa entre la ecuación de explotación y la ecuación de exploración.

$\boldsymbol{\varphi_{iv}^{s'}}$ es el ahorro en la combinación de dos nodos i, v en una excursión en lugar de servirlos en dos excursiones diferentes y es calculado con la ecuación (39):

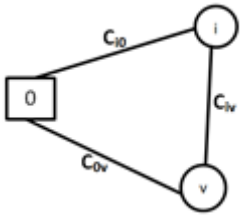
$$\varphi_{iv}^{s'} = C_{io} + C_{0v} - C_{iv} \qquad (38)$$

Aquí:

$\boldsymbol{C_{iv}}$ es la distancia entre los nodos i, v

Nodo 0 denota el depósito seleccionado.

Figura 6. Grafo representativo de $\boldsymbol{C_{iv}}$



3.3.1 Regla de actualización local de feromona

Para el VRP, la regla de actualización local, está representada por la ecuación (40), esta se aplica inmediatamente después de que cada hormiga cruza el nodo (i, v) durante la construcción de la ruta:

$$\zeta_{iv}^{s'} \leftarrow (1-\rho'')\zeta_{iv}^{s'} + \rho''\zeta_0 \qquad si\ la\ arista\ (i,j) \in\ T_{h'} \qquad (39)$$

Donde

$\boldsymbol{T_{h'}}$ es la ruta construida por la hormiga h'

$\boldsymbol{\rho''}$ es el parámetro de evaporación de la feromona en un rango de $[0,1]$

$\boldsymbol{\zeta_0}$ es el nivel inicial en la matriz de feromona.

3.3.2 Regla de actualización global para VRP. En esta sección se realiza la actualización global para el VRP, en la cual se modifican los valores de la feromona de los arcos de las rutas. Para esta actualización solo se considera la mejor solución global y la mejor solución de la iteración, esto se realiza al final de cada iteración. Con este proceso se puede guiar al algoritmo a encontrar mejores soluciones, y desvía el proceso a un espacio de búsqueda no local, dependiendo de la cantidad de feromona que se aplica en los arcos. Para las mejores rutas, la mejor ruta global T_b' y la mejor ruta de la iteración $T_{s'}$ del VRP, se les permite poner feromona en los bordes que les pertenecen. La regla de actualización global se define:

$$\zeta_{iv}^{s'+1} = (1-\rho'')\zeta_{iv}^{s'} + \rho''\Delta\zeta_{iv}^{s'} \qquad (40)$$

Aquí,

$$\Delta\zeta_{iv}^{s'} = \begin{cases} [(L_w' - L_b') + (L_w' - L_{s'}')]/\ L_w' & si\ \{(i,v)\ \in\ T_b'\ o\ T_{s'}'\} \\ 0 & Caso\ contrario \end{cases} \qquad (41)$$

Para aquellos bordes donde $\Delta\zeta_{iv}^{s'}$ se hace igual a 0, lo que se da es un proceso de evaporación de feromona en aquellos bordes que no pertenecen a la mejor solución (ya sea global o de la iteración correspondiente).

Donde L_b' y $L_{s'}'$ definen el costo de la ruta de la mejor solución global y la mejor solución de la iteración del VRP, respectivamente, el valor de L_w' representa el costo del recorrido de la peor solución de la iteración actual. (ver ejemplo en el Anexo D)

3.4 ACTUALIZACIÓN GLOBAL DE FEROMONA

3.4.1 Regla de actualización global para la selección. La actualización global de feromona tanto para la selección como para la asignación de clientes, se usa teniendo en cuenta la mejor solución global T_b y la mejor solución de la iteración T_s .

La regla de actualización está dada por la siguiente ecuación:

$$\tau_j^{s+1} = (1-\rho)\tau_j^s + \rho\Delta\tau_j^s \qquad (42)$$

Donde,

$$\Delta\tau_{j=}^{s}\begin{cases}[(l_w - l_b) + (l_w - l_s)] * \frac{n_j}{l_w} & si\ \{j \in T_b\ o\ T_s\} \\ 0\ en\ caso\ contrario\end{cases} \qquad (43)$$

Donde, $l_b\ y\ l_s$ definen el costo total de la mejor solución global y la mejor solución de la iteración del LRP, y l_w costo total de la peor solución en la iteración actual, n_j es el número de clientes asignados a la ubicación j.

3.4.2 Regla de actualización global para la asignación de clientes. La regla de actualización está dada por la ecuación:

$$\xi_{ij}^{s+1} = (1 - \rho')\xi_{ij}^{s} + \rho' \, \Delta\xi_{ij}^{s} \qquad (44)$$

Donde,

$$\Delta\xi_{ij}^{s} = \begin{cases} [(l_w - l_b) + (l_w - l_s)]/\, l_w & si \ \{j \in T_b \ o \ T_s\} \\ 0 & En\ caso\ contrario \end{cases} \qquad (45)$$

Donde, $l_b \ y \ l_s$ definen el costo total de la mejor solución global y la mejor solución de la iteración del LRP, y l_w costo total de la peor solución en la iteración actual. Se definen de la misma manera que para la regla de actualización global para la selección de instalaciones.

3.5 APLICACIÓN DE BÚSQUEDA LOCAL ITERATIVA - ILS

Cuando se hace el VRP para todos los depósitos se genera la solución conjunta, es decir aquella que muestre selección, asignación y ruteo con su respectivo costo, de esta forma se elige la mejor solución de la iteración para una posterior actualización global de feromona (se sigue la misma metodología que en VRP), finalmente, esta solución es mejorada aplicando las técnicas de búsqueda local y perturbación dadas por el ILS.

En Tabla 8, se observa el procedimiento del algoritmo ILS para mejorar la solución encontrada con colonia de hormigas. N_1, N_2, N_3 y N_4, corresponden a las estructuras de barrio mencionadas en el subcapítulo 2.3.

Tabla 8. Algoritmo ILS para mejorar la solución encontrada con ACO

Algoritmo ILS para mejorar la solución encontrada con ACO
Iniciar algoritmo
Solución Inicial Ts
Para cada iteración $\boldsymbol{t = 1 \ldots T}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts1$*: Mejorar solución* Ts *empelando* N_1
Si $Costo\ Ts1 < Costo\ Ts \Rightarrow Ts \leftarrow Ts1$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts2$: *Mejorar solución* Ts *empelando* N_2
Si $Costo\ Ts2 < Costo\ Ts \Rightarrow Ts \leftarrow Ts2$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts3$: *Mejorar solución* Ts *empelando* N_3
Si $Costo\ Ts3 < Costo\ Ts \Rightarrow Ts \leftarrow Ts3$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts4$: *Mejorar solución* Ts *empelando* N_4
Si $Costo\ Ts4 < Costo\ Ts \Rightarrow Ts \leftarrow Ts4$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts5$: *Mejorar solución* Ts *empelando criterio de perturbación*
Si $Costo\ Ts5 < Costo\ Ts \Rightarrow Ts \leftarrow Ts5$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts6$: *Mejorar solución* Ts *empelando* N_1
Si el $Costo\ Ts6 < Costo\ Ts \Rightarrow Ts \leftarrow Ts6$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts7$: *Mejorar solución* Ts *empelando* N_2
Si $Costo\ Ts7 < Costo\ Ts \Rightarrow Ts \leftarrow Ts7$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$
$Ts8$: *Mejorar solución* Ts *empelando* N_3
Si $Costo\ Ts8 < Costo\ Ts \Rightarrow Ts \leftarrow Ts8$
Fin $\boldsymbol{T'}$
Para cada $\boldsymbol{t' = 1 \ldots T'}$

$Ts9$: *Mejorar solución* Ts *empelando* N_4
Si $Costo\ Ts9 < Costo\ Ts \Rightarrow Ts \leftarrow Ts9$
Fin T′
Fin T iteraciones
Terminar algoritmo

Los algoritmos ACO e ILS, se conectan a través de la actualización global de feromona por medio de la mejor solución global.

La aplicación de ILS se hace siguiendo el marco general planteado por Derbel, Jarboui, Hanafi y Chabchoub (2010):

- T_s es la solución inicial para el proceso de ILS.
- Se aplica **búsqueda local** a T_s generando una **mejor solución** $\boldsymbol{T'_s}$. En la búsqueda local se emplean las estructuras de barrio planteadas por Derbel, Jarboui, Hanafi y Chabchoub (2012).
- Se hace perturbación a T'_s para obtener otra solución T''_s.
- Después de esto, se aplica de nuevo búsqueda local a T''_s para obtener un nuevo óptimo local $T_s^{\sim}$.
- Se hace $T_s \leftarrow T_s^{\sim}$ y se continúa el proceso iterativo.

3.5.1 Ejemplo prototipo III – ILS. Para mayor comprensión, se plantea un ejemplo con una solución inicial representada por un vector V que representa los depósitos seleccionados y una matriz A que indica el orden de cada ruta:

$$T_s \quad \begin{cases} V = \{1\ 2\ 5\} \\ A = \begin{Bmatrix} 10 & 4 & 3 \\ 1 & 9 & 6 \\ 7 & 8 & 5 \\ 2 & 0 & 0 \end{Bmatrix} \end{cases}$$

Teniendo en cuenta que los depósitos candidatos son 1, 2, 3, 4 y 5, en esta solución, los depósitos cerrados son: 3 y 4.

1. Para la búsqueda local inicial, se emplean cuatro estructuras de barrio llamadas N_1, N_2, N_3 y N_4, como se describe en el marco teórico.

Estructura de barrio N_1: se seleccionan 2 clientes al azar, asignados a depósitos diferentes, para el ejemplo, se selecciona el cliente $i = 1$ e $i = 6$, el cliente $i = 1$ que inicialmente está asignado al depósito $j = 1$, se traslada al depósito $j = 5$, y el cliente $i = 6$, asignado al depósito $j = 5$, se traslada al depósito $j = 1$. Así, se genera una nueva solución T_{s1}:

$$T_{s1} \begin{cases} V = \{1\ 2\ 5\} \\ A = \begin{Bmatrix} 10 & 4 & 3 \\ 6 & 9 & 1 \\ 7 & 8 & 5 \\ 2 & 0 & 0 \end{Bmatrix} \end{cases}$$

Este proceso se hace hasta encontrar una solución T_{s1} cuyo costo total sea inferior al de la solución T_s , de tal manera que T_{s1} sea la primer mejora deT_s. En este caso se supone que la solución dada anteriormente cumple con esta condición. Con esta estructura de barrio, la dirección de la ruta no cambia.

Estructura de barrio N_2: de la solución T_{s1} se selecciona un cliente al azar y se traslada a un depósito abierto seleccionado aleatoriamente. Para el ejemplo, se selecciona el cliente $i = 3$ y se inserta a la ruta del depósito $j = 2$.

Así se genera una nueva solución T_{s2}:

$$T_{s2} \begin{cases} V = \{1\ 2\ 5\} \\ A = \begin{Bmatrix} 10 & 3 & 0 \\ 6 & 4 & 1 \\ 7 & 9 & 5 \\ 2 & 8 & 0 \end{Bmatrix} \end{cases}$$

Este proceso se realiza hasta encontrar una solución T_{s2} que genere un costo total inferior al generado por T_{s1}. La dirección de la ruta no cambia. De nuevo se supone que la solución anterior cumple la condición.

Estructura de barrio N_3: de la solución T_{s2} se intercambia la posición de dos clientes de la misma ruta. Para el ejemplo, se selecciona el depósito $j = 2$ y de ésta los clientes $i = 3$ e $i = 9$, se intercambian sus posiciones en la ruta, la nueva solución T_{s3}, es:

$$T_{s3}\begin{cases} V = \{1\ 2\ 5\} \\ A = \begin{Bmatrix} 10 & 9 & 0 \\ 6 & 4 & 1 \\ 7 & 3 & 5 \\ 2 & 8 & 0 \end{Bmatrix} \end{cases}$$

Este proceso se repite hasta encontrar una solución T_{s3} que genere un costo inferior al generado por T_{s2}. Para continuar, se supone que la solución anterior cumple la condición.

Estructura de barrio N_4: de la solución T_{s3}, se selecciona el depósito abierto $j = 1$, se selecciona el cliente $i = 2$ y se inserta entre los clientes $i = 10$ e $i = 6$, la nueva solución T_{s4} es:

$$T_{s4}\begin{cases} V = \{1\ 2\ 5\} \\ A = \begin{Bmatrix} 10 & 9 & 0 \\ 2 & 4 & 1 \\ 6 & 3 & 5 \\ 7 & 8 & 0 \end{Bmatrix} \end{cases}$$

Como el proceso es una mejora de la solución inicial (T_s), se entiende que T_{s4} genera un menor costo que T_s, por tanto $T_s \leftarrow T_{s4}$. Se repite el proceso una cantidad t_1 de iteraciones.

Para cada estructura de vecindad se define un número máximo de iteraciones, para evitar estancamiento al encontrar un óptimo local.

$$T_s' \leftarrow T_s \; Mejorado \; con \; búsqueda \; local$$

2. Se aplica el criterio de perturbación a la solución mejorada encontrada anteriormente, anterior llamada T_s', de ésta, se selecciona un depósito abierto y sus clientes se re-asignan a otro depósito ya sea abierto o cerrado. Para el ejemplo, se selecciona el depósito $j = 5$, y se trasladan sus clientes al depósito $j = 3$, lo que genera una solución T_s'':

$$T_s'' \quad \begin{cases} V = \{1 \;\; 2 \;\; 3\} \\ A = \begin{Bmatrix} 10 & 9 & 0 \\ 2 & 4 & 1 \\ 6 & 3 & 5 \\ 7 & 8 & 0 \end{Bmatrix} \end{cases}$$

3. Se realiza búsqueda local a la solución T_s'', siguiendo las estructuras de barrio usadas en el numeral 1, con un número t_1 de iteraciones. Así, se genera una nueva solución $T_s^{\sim}$. Se evalúa el criterio de aceptación (costo) de las soluciones T_s' y $T_s^{\sim}$. Si el costo de la solución $T_s^{\sim}$ es menor que el de T_s', la solución T_s', es reemplazado por $T_s^{\sim}$.

$$T_s' \leftarrow T_s^{\sim}$$

Los pasos 2 y 3 se repiten un número t_1 de iteraciones.

Una vez culminado el proceso que emplea ILS, el algoritmo híbrido continúa con la solución T_s, de tal manera que:

$$T_s \leftarrow T_s'$$

Donde, T_s' es la mejor solución encontrada al implementar ILS.

4 RESULTADOS

4.1 BANCO DE PRUEBAS

A través del banco de pruebas se crea el conjunto de instancias, donde, se dan valores específicos a los parámetros del problema, permitiendo así evaluar los resultados obtenidos.

Los parámetros para el problema conjunto de localización de instalaciones y ruteo de vehículos son los siguientes:

- n: número de clientes
- m: número de depósitos candidatos
- F: costo fijo por apertura de depósitos
- R: capacidad de cada depósito
- d: demanda de cada cliente
- C: costo asociado a la distancia entre clientes-clientes y clientes-depósitos

Prodhon y Prins (2014) en su estudio reciente de LRP, señalan cuatro conjuntos de referencia estándar aplicados en métodos heurísticos, de acuerdo a las diferentes variantes del problema, también señalan cantidad de clientes y depósitos para cada uno.

Tabla 9 Conjunto de instancias para LRP

Instancias	Capacidad Vehículos	Capacidad depósitos	Número de depósitos	Número de clientes	Cantidad de instancias
'Conjunto AS' **(Albareda-Sambola et al., 2005)**	I limitada	Limitada	5-10	10-30	450
'Conjunto Tuzun' **(Tuzun & Burke, 1999)**	Limitada	Ilimitada	10 ó 20	{100,150,200}	36
'Conjunto Barreto' **(Barreto,2004)**	Limitada	Ilimitada	2-5	21-318	19
'Conjunto Prodhon' **(Prodhon, 2006)**	Limitada	Limitada	5-20	20-200	30

Fuente: PRINS, Christian & PRODHON, Caroline. A survey of recent research on location-routing problems. European journal of operational research. [Online], Junio 2014.

Drexl y Schneider (2014), presentan un resumen del LRP, en el cual se incluye el problema estándar y todas sus variantes. Adicionalmente, se hace referencia a varios conjuntos de instancias utilizados en la literatura para medir la calidad de los diferentes métodos de solución. Estas instancias se publican de acuerdo a las variantes del problema, orden cronológico, y en su mayoría, cuentan con un enlace del sitio donde se puede acceder a estos conjuntos.

Para el caso del LRP estándar, en la siguiente tabla se presentan algunos conjuntos de instancias:

Tabla 10. Conjunto de instancias para LRP

Acrónimo	Primer de referencia; observaciones	Link
Perl	Introducido en la tesis inédita Ph. D. de Perl (1983).	sweet.ua.pt/sbarreto/private/-SergioBarretoHomePage.htm
TB	Tuzun y Burke (1999)	prodhonc.free.fr/homepage
B	Introducido en la tesis inédita Ph. D. de Barreto (2003).	sweet.ua.pt/sbarreto/private/-SergioBarretoHomePage.htm
ADF	Albareda-Sambola, Díaz, y Fernández (2005)	No disponible en Internet
PPW	Prins, Prodhon, y Wolfler Calvo (2006a)	prodhonc.free.fr/homepage
ABR	Akca, Berger, y Ralphs (2009)	claudio.contardo.org/instances

Fuente: DREXL, Michael & SCHNEIDER,Michael. A survey of location-routing problems. Technical Report. [Online], Noviembre 2013.

Como se observa en la tabla 10, no es posible acceder al conjunto de instancias **ADF** o **'Conjunto AS'** con los cuales se identifica el problema específico descrito, por lo tanto, se crea el banco de pruebas para la presente investigación.

Dadas las características del problema, en el banco de pruebas creado para la evaluación de resultados, se toman algunas características del '**Conjunto AS'** [58]como cantidades y ubicación de clientes y depósitos, incluyendo instancias hasta con 50 clientes. Para todos los casos se toma una relación demanda/capacidad=0,5, para esto, se generan valores de demanda aleatorios dentro de una distribución uniforme [500,1000], y con esto se obtiene la demanda total que junto con el valor correspondiente de D/C generan el valor de capacidad total, dependiendo de la cantidad de depósitos la capacidad correspondiente está

[58] ALBAREDA-SAMBOLA, Maria. DIAZ, Juan. FERNANDEZ, Elena. A compact model and tight bounds for a combined location routing problem. Computers and Operations Research 32

dada por las siguientes proporciones: para 5 depósitos: 20%, 15%, 18%, 30% y 17%; para 10 depósitos: 5%, 15%, 7%, 8%, 12%, 6%, 16%, 14%, 4% y 13%.

El costo de instalación de depósitos está dado por la siguiente formula:

$$F_j = \left(1 + \frac{D}{R_j}\right) * \tilde{c} \qquad (46)$$

El valor de $\tilde{c}$ corresponde al promedio de los costos de las aristas, en la relación

(depósito-clientes), después de eliminar los $(|I| + |J|)/2$ arcos de mayor valor incidentes a cada cliente.

En la siguiente tabla se especifica la variación que tendrá cada uno de los parámetros para la creación de instancias:

Tabla 11. Parámetros banco de pruebas

Parámetro	Nivel 1	Nivel 2
M	$\boldsymbol{m1} = 5$	$\boldsymbol{m2} = 10$
N	$\boldsymbol{n1} = 30$	$\boldsymbol{n2} = 50$
C	$\boldsymbol{p}$**:**la ubicación de clientes y depósitos potenciales son generados aleatoriamente en un cuadro de 1000x1000. Así, las distancias euclidianas corresponden a los costos de los arcos respectivos	$\boldsymbol{g}$**:** Se crean 4 clusters, los cuales están distribuidos de manera uniforme en los siguientes cuadros: 1. x(0-500);y(0-500) 2. x(0-500); y(500-1000) 3. x(500-1000); y(0-500) 4. x(500-1000); y(500-1000) La ubicación de depósitos se da en una distribución uniforme en las coordenadas: x(250-750); y(250-750). Las distancias euclidianas corresponden a los costos de los arcos respectivos.

Con esta estructura, las instancias a trabajar son las siguientes:

Tabla 12. Instancias banco de pruebas

Instancia	depósitos	clientes	C
$\boldsymbol{m1n1r1ap\_1}$	5	30	1000x1000
$\boldsymbol{m2n1r1ap\_2}$	10	30	1000X1000
$\boldsymbol{m1n2r1ap\_3}$	5	50	1000x1000
$\boldsymbol{m2n2r1ap\_4}$	10	50	1000X1000
$\boldsymbol{m1n1r1ag\_5}$	5	30	Clusters
$\boldsymbol{m2n1r1ag\_6}$	10	30	Clusters
$\boldsymbol{m1n2r1ag\_7}$	5	50	Clusters
$\boldsymbol{m2n2r1ag\_8}$	10	50	Clusters

4.2 DISEÑO DE EXPERIMENTOS

Con el fin de estudiar el efecto de la variación de los parámetros del algoritmo híbrido ACO+ILS, sobre el valor de la función objetivo y establecer aquella configuración que lleve a mejores resultados del LRP, se emplea un diseño factorial fraccionado 2^{k-2}, que permite variar cada uno de los parámetros en dos niveles.

4.2.1 Análisis de parámetros del método ACO+ILS. Los siguientes parámetros son determinantes en el desempeño del método ACO:

- α, β y γ que representan la relación entre información heurística y de feromona para las colonias de selección, asignación y VRP respectivamente.

- q_0, q_0' y q_0'' que representan la relación entre las ecuaciones de exploración y explotación para las colonias de selección, asignación y VRP respectivamente.
- ρ, ρ' y ρ'' son los parámetros de evaporación de feromona para los procesos de selección, asignación y VRP respectivamente.

- S y S' indican el número de iteraciones para el algoritmo ACO general y el ACO para el ruteo de vehículos.

- b y b' corresponde al número de hormigas para el algoritmo ACO en general y el ACO para ruteo de vehículos.

Los siguientes parámetros son representativos del algoritmo ILS:

- T representa el número de iteraciones para el proceso de búsqueda local y la cantidad máxima de iteraciones de cada una de las estructuras de vecindad.

De los parámetros descritos, los valores de , α, β , γ, se toma el mismo valor para todos los procesos y se denota α. ρ, ρ', ρ'', se denota en general ρ. Sucede igual con S' y S , que se definen como $S.q_0$ representa el mismo valor para los parámetros: q_0, q_0', y q_0''. b indica la cantidad de hormigas tanto para VRP como para el proceso general. El valor de $r = 3$, como en los experimentos preliminares de Ting y Hochen (2012) de donde también se obtienen los niveles alto y bajo. Con esto, se tienen en cuenta 6 factores, y se aplica un diseño factorial fraccionado 2^{6-2} .

Tabla 13. Descripción factores del diseño experimental

Factor	Nivel alto(+)	Nivel bajo (-)
q_0: Parámetro explotación vs exploración – Selección, asignación y VRP	0,9	0,1
α: Parámetro que relaciona la importancia entre información heurística y valor de feromona – Selección, asignación y VRP	3	1
S : Número de iteraciones para el algoritmo ACO+ILS y VRP	5	2
b: Número de hormigas de ACO general y VRP	5	2

Factor	Nivel alto(+)	Nivel bajo (-)
T: Número de iteraciones para ILS	5	2
ρ: parámetro de evaporación de feromona – Selección, asignación y VRP respectivamente	0,9	0,1

4.3 RESULTADOS DEL DISEÑO EXPERIMENTAL

El algoritmo es implementado en MatLab® versión R2012a, en un equipo con procesador Intel Core i5 con 6 GB de memoria RAM instalada. Los datos de demanda, capacidad y costos son exportados desde Excel.

Tabla 14. Factores del diseño experimental

Factor	(+)	(-)
A:q_0	0,9	0,1
B:α	3	1
C:S	5	2
D:b	5	2
E:T	5	2
F: ρ	0,9	0,1

Tabla 15. Resultado del diseño experimental

Factores						Valor promedio de la función objetivo							
A	B	C	D	E	F	$m1n1$ $r1ap\_1$	$m2n1$ $r1ap\_2$	$m1n2$ $r1ap\_3$	$m2n2$ $r1ap\_4$	$m1n1$ $r1ag\_5$	$m2n1$ $r1ag\_6$	$m1n2$ $r1ag\_7$	$m2n2$ $r1ag\_8$
-	-	-	-	-	-	14521	18611	18755	25216	12497	16771	20587	20952
+	-	-	-	+	-	11329	15202	13077	18531	9191	12058	12858	13867
-	+	-	-	+	+	11982	16429	13429	19675	10311	13025	15652	14218
+	+	-	-	-	+	11149	16239	13615	18921	9224	12009	13272	12846
-	-	+	-	+	+	13770	18975	16416	22590	12328	12492	18774	19119
+	-	+	-	-	+	10990	15248	13239	19460	9146	11711	11531	12905
-	+	+	-	-	-	10732	16429	13017	19784	10803	12507	13915	13380
+	+	+	-	+	-	10630	16251	12582	19201	8630	11562	12050	11924

Factores						Valor promedio de la función objetivo							
A	B	C	D	E	F	$m1n1$ $r1ap\_1$	$m2n1$ $r1ap\_2$	$m1n2$ $r1ap\_3$	$m2n2$ $r1ap\_4$	$m1n1$ $r1ag\_5$	$m2n1$ $r1ag\_6$	$m1n2$ $r1ag\_7$	$m2n2$ $r1ag\_8$
-	-	-	+	-	+	12634	19395	18181	23390	12712	15538	18973	19193
+	-	-	+	+	+	11168	14488	13277	19263	8722	12004	11339	12605
-	+	-	+	+	-	12306	15742	13794	19489	9738	12770	14346	14508
+	+	-	+	-	-	11220	15866	12562	19974	8693	11581	12643	11807
-	-	+	+	+	-	13347	17207	16485	19093	10848	14016	18486	16922
+	-	+	+	-	-	11123	16397	12834	22023	9210	12431	11688	12145
-	+	+	+	-	+	12183	15799	12909	17916	9029	12203	13063	13214
+	+	+	+	+	+	9832	14334	12242	16913	8474	11447	11208	11673

4.4 ANALISIS DEL DISEÑO EXPERIMENTAL

En la tabla 16 se observan los efectos estimados para cada una de las instancias, una vez aplicado el diseño experimental propuesto, el cual consta de 8 tratamientos y 4 réplicas.

Tabla 16. Efectos estimados para la función objetivo

Factores	Efecto estimado							
	$m1n1$ $r1ap\_1$	$m2n1$ $r1ap\_2$	$m1n2$ $r1ap\_3$	$m2n2$ $r1ap\_4$	$m1n1$ $r1ag\_5$	$m2n1$ $r1ag\_6$	$m1n2$ $r1ag\_7$	$m2n2$ $r1ag\_8$
A: q_0	-1754	-1821	-2445	-1608	-2122	-1814	-4651	-3967
B: alpha	-1106	-1016	-2264	-2212	-1219	-1240	-2261	-3017
C: S	-463	-128	-871	-935	-328	-923	-1119	-1089
D: b	-161	-520	-231	-665	-588	-18	-862	-893
E: t	-23	-669	-476	-1491	-384	-672	-120	-201
F: rho	-187	-100	25	-648	42	-408	-345	33

El efecto de los factores A, B, C, D y E tiene un valor negativo, es decir que al pasar de un nivel bajo al nivel alto, el valor de la función objetivo disminuye, para el factor F, se observa este comportamiento en la mayoría de los casos. Dado que

el objetivo es minimizar el costo asociado a la instalación de depósitos y ruteo de vehículos, todos los factores deben asumir un nivel alto.

Con el fin de observar el conjunto de factores que tiene mayor influencia sobre la función objetivo, se ilustran los diagramas de Pareto para cada instancia, usando el software estadístico Minitab15.

Figura 7. Diagrama Pareto instancia 1

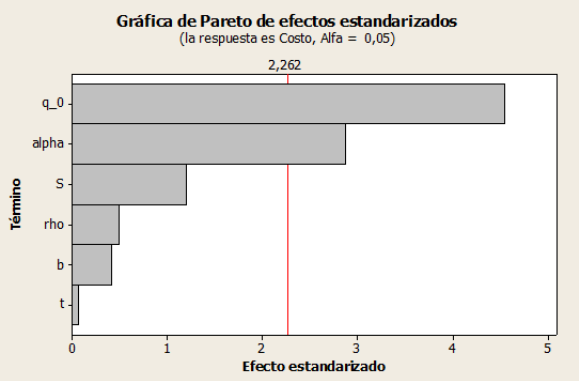


Figura 8. Diagrama Pareto instancia 2

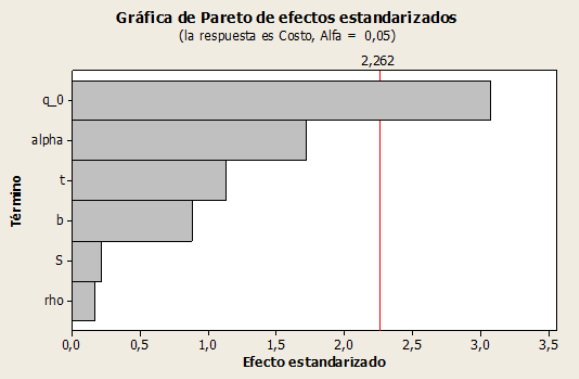


Figura 9. Diagrama Pareto instancia 3

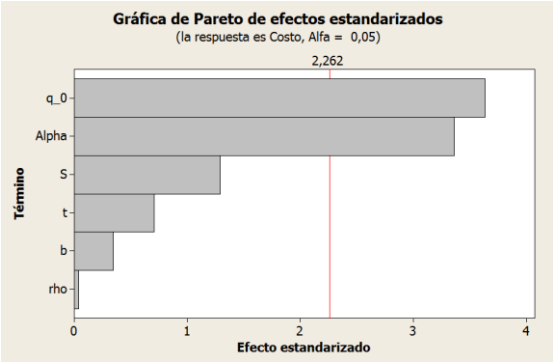


Figura 10. Diagrama Pareto instancia 4

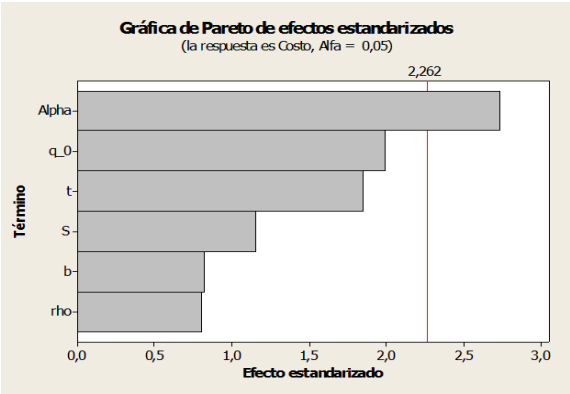


Figura 11. Diagrama Pareto instancia 5

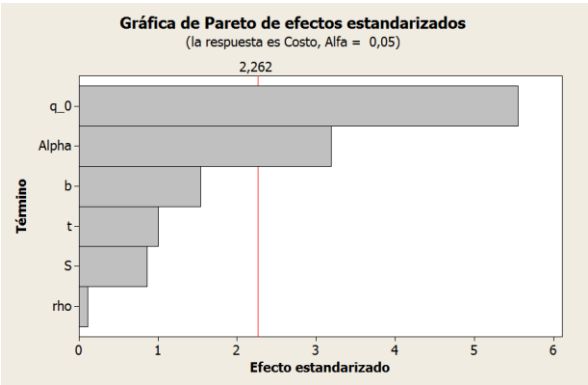


Figura 12. Diagrama Pareto instancia 6

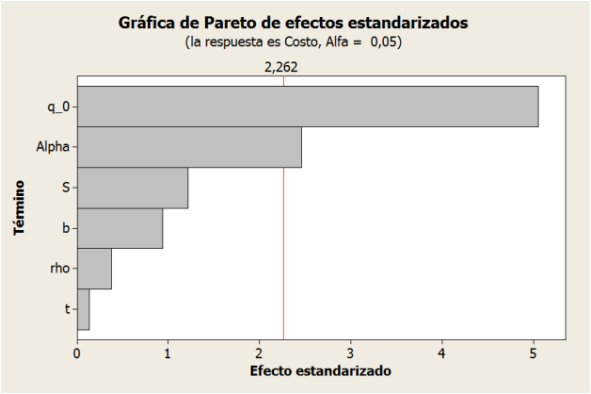


Figura 13. Diagrama Pareto instancia 7

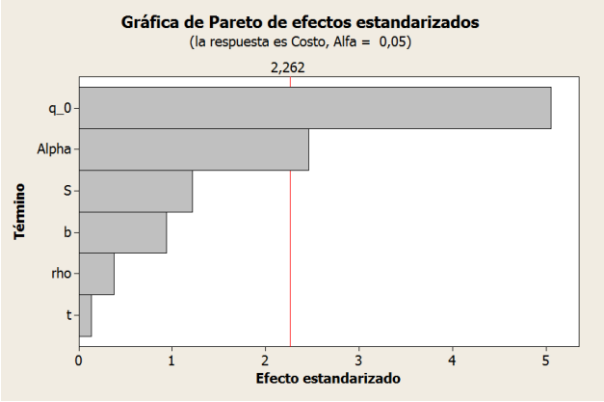
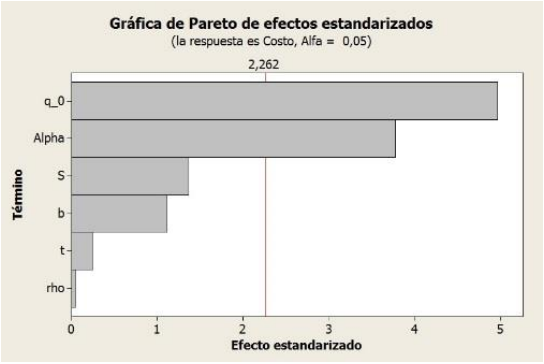


Figura 14. Diagrama Pareto instancia 8



En los diagramas de Pareto, se pueden observar en orden de importancia el efecto de cada uno de los factores sobre la función objetivo. En la mayoría de los casos, los factores con mayor efecto son q_0: parámetro que representa la importancia entre las ecuaciones de exploración vs explotación, α: parámetro que relaciona información de feromona vs heurística y S: número iteraciones ACO. Los factores b, t y ρ no representan efectos significativos en las 8 instancias evaluadas, con esto su valor puede tomarse en nivel alto o bajo.

De acuerdo a los efectos estimados y los resultados del diseño experimental, se establecen los siguientes valores para los parámetros ACO+ILS. $q_0 = 0{,}9, q_0' = 0{,}9, q_0'' = 0{,}9, \alpha = 3, \beta = 3, \gamma = 3, S = 5, S' = 5, b = 5, b' = 5, t = 5, \rho = 0{,}9,$ $\rho' = 0{,}9\ y\ \rho'' = 0{,}9$. Para los parámetros que no tienen un efecto significativo, se toma sus valores en el nivel alto dado que en estos casos se observan mejores resultados.

Se evidencian mejores resultados cuando se incrementa la probabilidad de trabajar con las ecuaciones de explotación, esto se debe a la baja cantidad de iteraciones, pues la retroalimentación en la información de feromona dada por la actualización global, se hace como máximo 5 veces en el proceso general, de acuerdo al número máximo de iteraciones S utilizadas dentro del diseño factorial. Sin embargo, el hecho de usar un número mayor de iteraciones aumenta considerablemente el tiempo computacional y por tanto el análisis, es por esta razón que dicha opción no se considera.

4.5 COMPARACIÓN ACO vs ACO+ILS

Teniendo en cuenta la mejor combinación de factores encontrada con el diseño experimental, se evalúan las 8 instancias planeadas para ACO y ACO+ILS por separado, con el fin de analizar el comportamiento en cuanto a costo y tiempo

computacional cuando se emplean las técnicas de búsqueda local y perturbación dadas por ILS al ACO.

Tabla 17. Comparación ACO vs ACO+ILS

INSTANCIAS	ACO		ACO+ILS		COMPARACIÓN	
	Costo	Tiempo (s)	Costo	Tiempo (s)	Δ Tiempo (s)	Δ Costo
$m1n1r1ap\_1$	10985	1204	9764	1654	450	1221
$m2n1r1ap\_2$	15814	994	14118	1374	380	1696
$m1n2r1ap\_3$	13289	3459	11293	4003	544	1996
$m2n2r1ap\_4$	23287	2463	17919	2884	421	5368
$m1n1r1ag\_5$	9597	1409	8577	1708	299	1020
$m2n1r1ag\_6$	12942	1086	10810	1198	112	2132
$m1n2r1ag\_7$	10033	3070	9306	3980	910	727
$m2n2r1ag\_8$	16570	2490	12276	2701	211	4294

Figura 15. Costo LRP con ACO vs ACO+ILS.

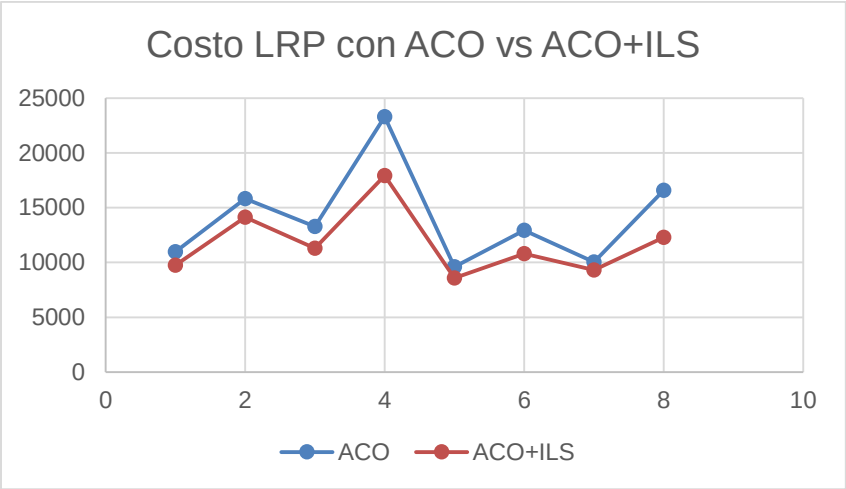


Tabla 18. Disminución costo vs aumento en tiempo ACO+ILS

Instancia	% Disminución costo	% Aumento tiempo
$m1n1r1ap\_1$	11,12	37,38
$m2n1r1ap\_2$	10,72	38,23
$m1n2r1ap\_3$	15,02	15,73
$m2n2r1ap\_4$	23,05	17,09
$m1n1r1ag\_5$	10,63	21,22
$m2n1r1ag\_6$	16,47	10,31
$m1n2r1ag\_7$	7,25	29,64
$m2n2r1ag\_8$	25,91	8,47

Como se observa en la Figura 15 y en las tablas 17 y 18, en todos los casos, se evidencian disminuciones en el costo cuando se emplean técnicas de ILS, sin embargo, el aumento en el tiempo computacional no supera el 40%, con esto, el algoritmo híbrido propuesto asegura una mejora significativa en la disminución de costos de instalación y ruteo, sin afectar considerablemente el tiempo requerido, se puede decir entonces que se tiene un algoritmo que además de ser eficaz es eficiente. El buen desempeño se logra gracias al uso de las estructuras de vecindad y perturbación, empleadas por cada iteración, las cuales permiten

explorar óptimos locales generados con el algoritmo ACO que a su vez emplea las mejores soluciones encontradas para hacer actualización global de feromona y retroalimentar el proceso.

5 CONCLUSIONES

Se cumplieron en su totalidad los objetivos propuestos en este trabajo de investigación. El algoritmo híbrido para resolver el problema conjunto de localización de instalaciones y ruteo de vehículos, basado en las metaheurísticas ACO e ILS implementado en el software Matlab®, arroja resultados factibles. Por lo tanto, esta herramienta computacional generada es una nueva alternativa de solución para problemas con un número reducido de iteraciones, que permite obtener resultados en tiempos de cómputo considerables, lo cual es un aspecto importante dentro de las organizaciones que consideren su implementación.

Dentro de la literatura se evidencia la dificultad para obtener las instancias de los investigadores que han abordado el presente problema en específico, por lo tanto, no se consideró comparar el algoritmo con otro propuesto en la literatura, en términos de costo y tiempos de computo.

De acuerdo con los resultados del diseño experimental, el factor con mayor influencia sobre la función objetivo, es el parámetro q_o, que representa la importancia de las ecuaciones de exploración vs las de explotación, donde su valor más alto genera mejores resultados, es decir cuando aumenta la probabilidad de emplear las ecuaciones de explotación. Los parámetros α (relación de importancia entre información heurística y de feromona) y S (número de iteraciones) también influyen significativamente sobre la función objetivo. En general, se obtienen mejores resultados cuando se trabaja con los niveles altos de los factores analizados.

Los resultados evidencian mayor rendimiento del algoritmo híbrido propuesto frente al algoritmo ACO tradicional con respuestas que disminuyen el costo

promedio hasta en un 25,91%, sin aumentar considerablemente el tiempo computacional.

6 RECOMENDACIONES

Con base en el algoritmo híbrido propuesto, hacer las modificaciones pertinentes para emplearlo en las diferentes variaciones del LRP, esto con el fin de ajustarlo más a escenarios reales.

Integrar las estructuras y perturbación basadas en la metaheurística ILS para cada ciclo u hormiga ‘b’ y comparar su desempeño con el algoritmo propuesto.

Realizar futuras investigaciones basadas en el presente trabajo, empleando el banco de pruebas propuesto y a su vez crear más bancos de pruebas que permitan aumentar las instancias para el benchmark.

BIBLIOGRAFÍA

AKSEN, Deniz & ALTINKEMER, Kemal. A location-routing problem for the conversion to the click-and-mortar retailing: the static case. European Journal of Operational Research. Marzo 2007.

ALBAREDA-SAMBOLA, Maria. DIAZ, Juan. FERNANDEZ, Elena. A compact model and tight bounds for a combined location routing problem. Computers and Operations Research 32. 2005.

ALBAREDA-SAMBOLA, Maria. FERNÁNDEZ, Elena, & Nickel, S. Multiperiod locationrouting with decoupled time scales. European Journal of Operational Research. 2012.

ALBAREDA-SAMBOLA, Maria. FERNÁNDEZ, Elena. & LAPORTE, Gilbert. Heuristic and lower bound for a stochastic location-routing problem. European Journal of Operational Research. 2007.

ALONSO, Sergio. CORDÓN, Oscar. FERNÁNDEZ DE VIANA, Iñaki. HERRERA, Francisco. La Metaheurística de Optimización Basada en Colonias de Hormigas: Modelos y Nuevos Enfoques. Departamento de Ciencias de la Computación e Inteligencia Artificial, E.T.S. Ingeniería Informática, C/ Periodista Daniel Saucedo Aranda s/n, 18071 Granada (España).

ALTIPARMAKB, Fulya. DENGIZC, Berna. KARAOGLANA, Ismail. KARAC, Imdat. A branch and cut algorithm for the location-routing problem with simultaneous pickup and delivery. European Journal of Operational Research. 2011.

ALVAREZ, Cesar. Optimización de enjambre de partículas (PSO) aplicada al problema de la P-mediana. Bucaramanga. Universidad Industrial de Santander. 2013.

AMAYA, C., GUERRERO, W. J., PRODHON, C., & VELASCO, N. Hybrid heuristic for the inventory location-routing problem with deterministic demand. International Journal of Production Economics

AMBROSINO, Daniela, Sciomachen, Anna, & Scutellà, Maria. A heuristic based on multiexchange techniques for a regional fleet assignment location-routing problem. Computers and Operations Research.

AMODEO, Lionel, MACHALLET, Julien, PRINS, Christian, YALAOUI, Farouk & VITRY, Grégoire. Multi-start iterated local search for the periodic vehicle routing problema with time Windows and time spread constraints on services. Computers & operations research.

ARPIN, D., LAPORTE, G. & NOBERT. An exact algorithm for solving a capacitated location routing problem. Annids of Operations Research.

ARRÁIZ, Emely, CUERVO, Daniel, GOOS, Peter & SÖRENSEN, Kenneth. An iterated local search algorithm for the vehicle routing problem with backhauls. European journal of operational research

BARRETO, Sérgio, FERREIRA, Carlos, PAIXÃO, José & SANTOS, Beatriz. Using clustering analysis in a capacitated location-routing problem. European journal of operational research.

BARRETO, Sérgio. BORGES LOPES, Rui. FERREIRA, Carlos. SOUSA SANTOS, Beatriz. A decision-support tool for a capacitated location-routing problem. Decision Support Systems 46.

BELENGUER, José-Manuel, BENAVENT, Enrique, PRINS, Christian, PRODHON, Caroline, & WOLFLER-CALVO, Roberto. A branch-and-cut method for the capacitated location-routing problem. Computers and Operations Research. 2011.

BIRATTARI, Mauro. DORIGO, Marco. STUTZLE, Thomas. Ant Colony Optimization. Artificial Ants as a Computational Intelligence Technique. Ieee computational intelligence magazine Noviembre 2006.

BLUM, Christian. LI, Xiaodong. Swarm Intelligence in Optimization. Natural Computing Series. 2008.

BRUNS, A.D., Zweistufige Standortplanung unter Beru¨ cksichtigung von Tourenplanungsaspekten – Primale Heuristiken und Lokale Suchverfahren, PhD Dissertation,Sankt Gallen University. 1998.

BRUNS, Arno, KLOSE, Andreas, STAHLY, Paul,. Restructuring of Swiss parcel delivery services. OR Spektrum, 2000.

CABALLERO, Rafael, GONZÁLEZ, Mercedez, GUERRERO, María, MOLINA, Julián, & PARALERA, Concepción. Solving a multiobjective location routing problem with a metaheuristic based on tabu search. Application to a real case in Andalusia. European journal of operational research.

CAICEDO, Eduardo F. LÓPEZ, Jesús A. MUÑOZ, Mario A. Inteligencia de enjambres: sociedades para la solución de problemas (una revisión). Revista ingeniería e investigación vol. 28 no. 2.

CALVO, Roberto, PRINS, Christian & PRODHON, Caroline. A memetic algorithm with population management (MA/PM)) for the periodic location-routing problem. Springer.

CATAY, Bulent. Ant Colony Optimization and Its Application to the Vehicle Routing Problem with Pickups and Deliveries. Natural intelligence for scheduling, planning and packing problems. Studies in computational intelligence.

CHABCHOUB, Habib, DERBEL, Houda, JARBOUI, Bassem & HANAFI, Said. An iterated local search for solving a location-routing problem. Electronic notes in discrete mathematics, 2011.

CHEN, Chia-Ho. TING, Ching-Jung. A multiple ant colony optimization algorithm for the capacitated location routing problem. Int. J. Production Economics. 2013.

CORDEAU, Jean-FranÇois, LAPORTE, Gilbert & PASIN, Federico. An iterated local search heuristic for the logistics network design problem with single assignment. International journal of production economics. Febrero 2008.

DI CARO, Gianni. DORIGO, Marco. The Ant optimization Meta-Heuristic. Iridia. 1999.

DORIGO, Marco. STÜTZLE. Thomas. Ant Colony Optimization. Bradford Books. MIT Press, Cambridge, MA. 2004.

DOULABI, Hossein. & SEIFI, Abbas. Lower and upper bounds for location-arc routing problems with vehicle capacity constraints. European journal of operational research, Junio 2012

DUHAMEL, Chistophe, LACOMME, Philippe, PRINS, Christian, & PRODHON, Caroline. A GRASPxELS approach for the capacitated location-routing problem. Computers and operations research. Junio 2009

ESCOBAR, John, LINFATI, Rodrigo, & TOTH, Paolo. A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. Computers and operations research. Junio

FAVARETTO, Daniela. MORETTI, Elena. PELLEGRINI, Paola. Ant colony system for a VRP with multiple time windows and multiple visits. Journal of Interdisciplinary Mathematics. 2007.

GAMBARDELLA L.M. RIZZOLI, A.E. LUCIBELLO, E. . MONTEMANNI, R. Ant colony optimization for real-world vehicle routing problems. From theory to applications. Swarm Intell. 2007.

GAMBARDELLA, Luca Maria. DORIGO, Marco. Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. Ieee transactions on evolutionary computation, vol. 1, no. 1, Abril 1997.

GEIGER, Martin Josef. Decision support for multi-objetive flow shop scheduling by the Pareto Iterated Local Search methodology. Computers & industrial engineering, Mayo 2011.

GELDERS, L.F, NAMBIAR, Jay, VAN WASSENHOVE, L.N. A large scale location-allocation problem in the natural rubber industry. European Journal of Operational Research. Febrero de 1981.

GMILKOWSKY Peter, LINSER Sebastián & REICHELT, Dirk,. A Study of an Iterated Local Search on the Reliable Communication Networks Design Problem. Springer, 2004

GUTJAHR, Walter & RATH, Stefan. A math-heuristic for the warehouse location routing problem in disaster relief. Computers and operations research. Julio 2011 [en linea], Disponible en: http://dx.doi.org/10.1016/j.cor.2011.07.016.

HAO, Jin-Kao & LAURENT, Benoît. Iterated local search for the multiple depot vehicle scheduling problem. Computers & industrial engineering, Diciembre 2008.

HLAING, Zar Chi Su Su. Khine May Aye. An Ant Colony Optimization Algorithm for Solving Traveling Salesman Problem. International Conference on Information Communication and Management. 2011.

HUA-LI, Sun. XUN-QING, Wang, & YAO-FENG, Xue. A bi-level programming model for a multi-facility location-routing problem in urban emergency system. In L. Zhang & C. Zhang (Eds.), Engineering education and management. 2012.

IBARAKI, Toshihide, IMAHORI, Shinji, NONOBE, Koji, SOBUE, Kensuke, UNO Takeaki & YAGIURA, Mutsunori. An iterated local search algoritm for the vehicle routing problem with convex time penalty functions. Discrete applied mathematics. Mayo 2007 2004

JABAL-AMELI, M., ARYANEZHAD, M., & GHAFFARI-NASAB, N. A variable neighborhood descent based heuristic to solve the capacitated location routing problem. International Journal of Industrial Engineering Computations. 2011.

JARBOUI, Bassem. DERBEL, Houda. HANAFI, Said. & MLADENOVIC, Nenad. Variable neighborhood search for location routing. Computers and Operations Research. 2013.

KAZHAROV, A. A. KUREICHIK V. M. Ant Colony Optimization Algorithms for Solving Transportation Problems. Journal of Computer and Systems Sciences International. 2010.

LAPORTE, Gilbert, NOBERT, Y. An exact algorithm for minimizing routing and operating costs in depot location. European Journal of Operational Research. Febrero de 1981.

LAPORTE, Gilbert, NOBERT, Y., ARPIN, D., 1986. An exact algorithm for solving a capacitated location-routing problem. Annals of Operations Research 6. 1986.

LEE, Chou-Yuan. LEE, ZNE-Jung. LIN, Shih-Wei. YING, Kuo-Ching. An enhanced ant colony optimization (EACO) applied to capacitated vehicle routing problem. Applied Intelligence. Febrero 2010.

Madsen, O.B.G., Methods for solving combined two level location-routing problems of realistic dimensions. European Journal of Operational Research. 1983.

MANZOUR-AL-AJDAD, S., TORABi, S., & SALHI, S. A hierarchical algorithm for the planar single-facility location routing problem. Computers and operations research. Mayo 2011.

NAGY, Gábor, & SALHI, Salhi. Location-routing: Issues, models and methods. European journal of operational research. Junio 2006.

PIRKWIESER, Sandro, & RAIDL, Günther. Variable neighborhood search coupled with ILPbased very large neighborhood searches for the (periodic) location-routing problem. Springer, 2010.

PRINS, Christian & PRODHON, Caroline. A memetic algorithm with population management (MAjPM)) for the periodic location-routing problem. In M. J. Blesa, C. Blum, G. Raidl, A. Roli, & M. Sampels (Eds.), Hybrid metaheuristics. Lecture notes in computer science. 2008.

PRINS, Christian & PRODHON, Caroline. A survey of recent research on location-routing problems. European journal of operational research., Junio 2014

PRODHON, Caroline. A hybrid evolutionary algorithm for the periodic location routing problem. European Journal of Operational Research. 2011.

RAND, Graham K. SALHI Said. The effect of ignoring routes when locating depots. European Journal of Operational Research, Marzo 1989.

SAMARA, Saadedeen. Zografos, Konstantinos. A combined location-routing model for hazardous waste transportation and disposal. Transportation Research Record. 1989.

Tavakkoli-Moghaddam, R., Makui, A., & Mazloomi, Z. A new integrated mathematical model for a bi-objective multi-depot location-routing problem a solved by a multi-objective scatter search algorithm. Journal of Manufacturing Systems. 2010.

THIERENS, Dirk. Population-Based Iterated Local Search: Restricting Neighborhood Search by Crossover. Springer. 2004

Watson-Gandy, C.D.T. Depot location with van salesmen – a practical approach. Omega. Junio 1973.

WU, Tai-His. LOW, Chinyao. BAI Jiunn-Wei. Heuristic solutions to multi-depot location-routing problems. Computers and Operations. 2002. [

YU, Vincent F . LIN Shih-Wei, LEEC, Wenyih. TINGD, Ching-Jung. A simulated annealing heuristic for the capacitated location routing problem. Computers and Industrial Engineering., Marzo de 2010.

ANEXO A. DIAGRAMA DE FLUJO DEL ALGORITMO HÍBRIDO

El diagrama de flujo inicia con los parámetros de evaporación de feromona (ρ), los que indican la importancia relativa entre exploración y explotación (q) y la relevancia entre información heurística y de feromona (α, β, γ).

Las instancias: S, b, S', b' y T indican el número de iteraciones y de hormigas para el algoritmo en general, número de iteraciones y hormigas para el VRP y la cantidad de iteraciones para el ILS respectivamente. Se definen adicionalmente, la cantidad de clientes (n), la demanda de los mismos (d_j), cantidad de depósitos candidatos (m), con sus costos de instalación (F_j) y sus capacidades (R_j) y la matriz C_{il} que indica el costo asociado a la distancia recorrida entre el cliente i y $l, (l \in IUJ)$.

Con la información anterior, se inicia el primer proceso que consiste en la construcción de las matrices iniciales de feromona para los procesos de selección y asignación.

Las variables L_b y L_s hacen referencia a los mejores costos globales y en la iteración s respectivamente, usados para la actualización de feromona, éstos inician como costos muy altos reemplazados posteriormente por los mejores.

Para una cantidad S de iteraciones, se abre un ciclo 'For' con la cantidad b de hormigas, que inicia con el proceso de selección de depósitos y su respectiva actualización local de feromona, para continuar con la asignación de clientes a depósitos que también incluye actualización local de feromona.

De manera independiente, para cada uno de los depósitos seleccionados se realiza el proceso de VRP con una cantidad de S' iteraciones, con b' hormigas, éste proceso incluye actualización local y global de feromona donde se tienen en cuenta T'_h como la mejor solución de cada iteración y T'_s como la mejor solución global y sus costos asociados (ver ilustraciones 16 y 17). El resultado de este

proceso es una ruta por cada depósito seleccionado, para la atención de sus clientes asignados.

A la mejor solución de cada iteración (aquella que genere el menor valor dado por la suma del costo de las rutas y el de apertura de depósitos), se aplica la metaheurística ILS con el objetivo de mejorar dicha solución, esto incluye aplicar en primer lugar, Búsqueda Local donde se establecen 4 estructuras de vecindad denotadas como N1, N2, N3 y N4. Las dos primeras se realizan entre dos rutas seleccionadas al azar y las dos últimas implican una posible modificación en la ruta de un solo depósito. Se usa una perturbación para mejorar la solución encontrada en el paso anterior. Para mejor efectividad del algoritmo se tienen dos opciones para esta perturbación: la primera es asignar todos los clientes de un depósito a uno nuevo que no haya sido abierto con el ACO y la segunda opción es trasladar los clientes de un depósito a otro. Y finalmente se realizan las cuatro estructuras de vecindad nuevamente. El proceso de Búsqueda Local se repite para encontrar una mejor solución con su respectivo costo (ver ilustración 18), los cuales se usan para la actualización global de feromona (ver ilustración 19). La solución al problema conjunto de localización de instalaciones y ruteo de vehículos es la mejor de las encontradas en todo el proceso.

Figura 16. Diagrama de flujo del algoritmo hibrido de colonia de hormigas e ILS

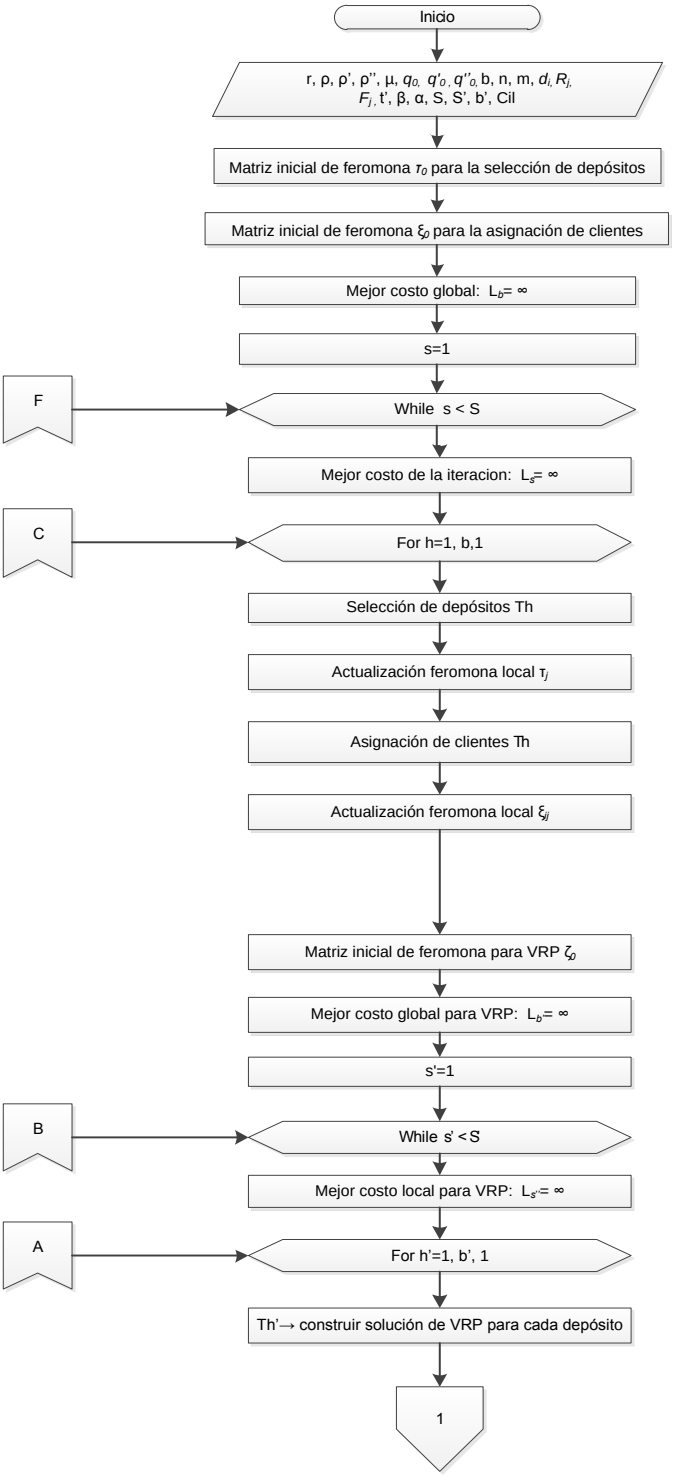


Figura 17. Diagrama de flujo del algoritmo hibrido de colonia de hormigas e ILS (continuación)

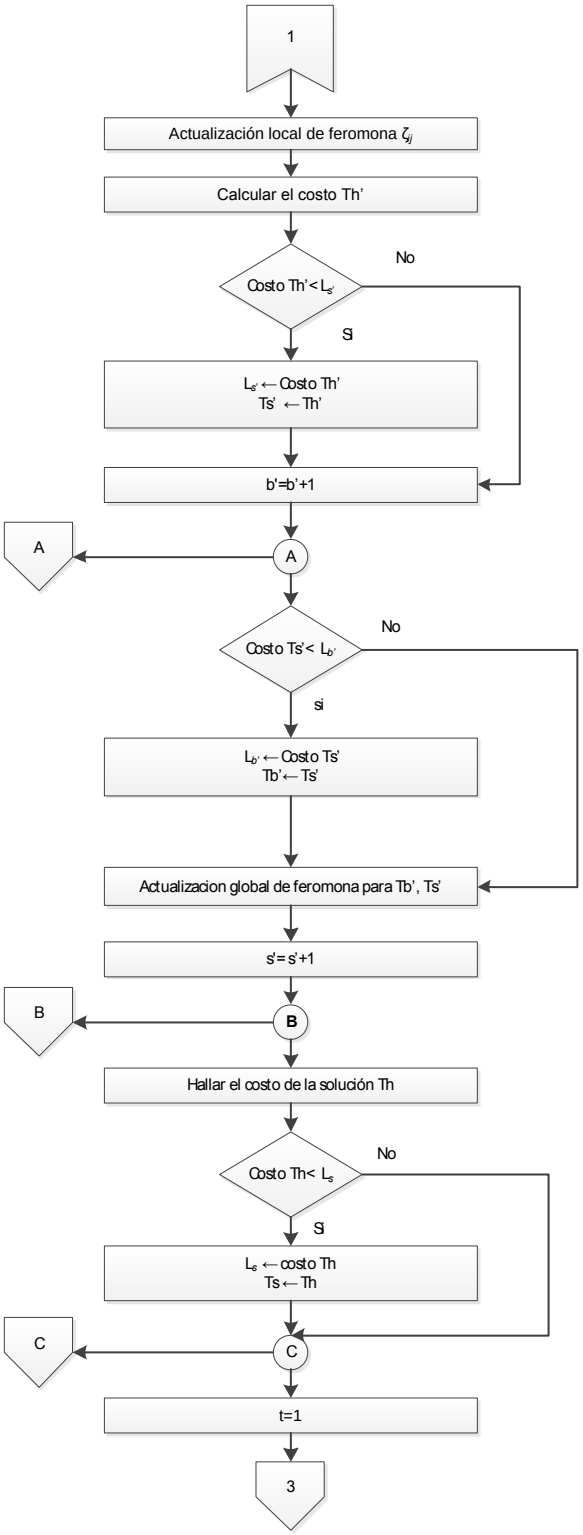


Figura 18. Diagrama de flujo del algoritmo hibrido de colonia de hormigas e ILS (continuación)

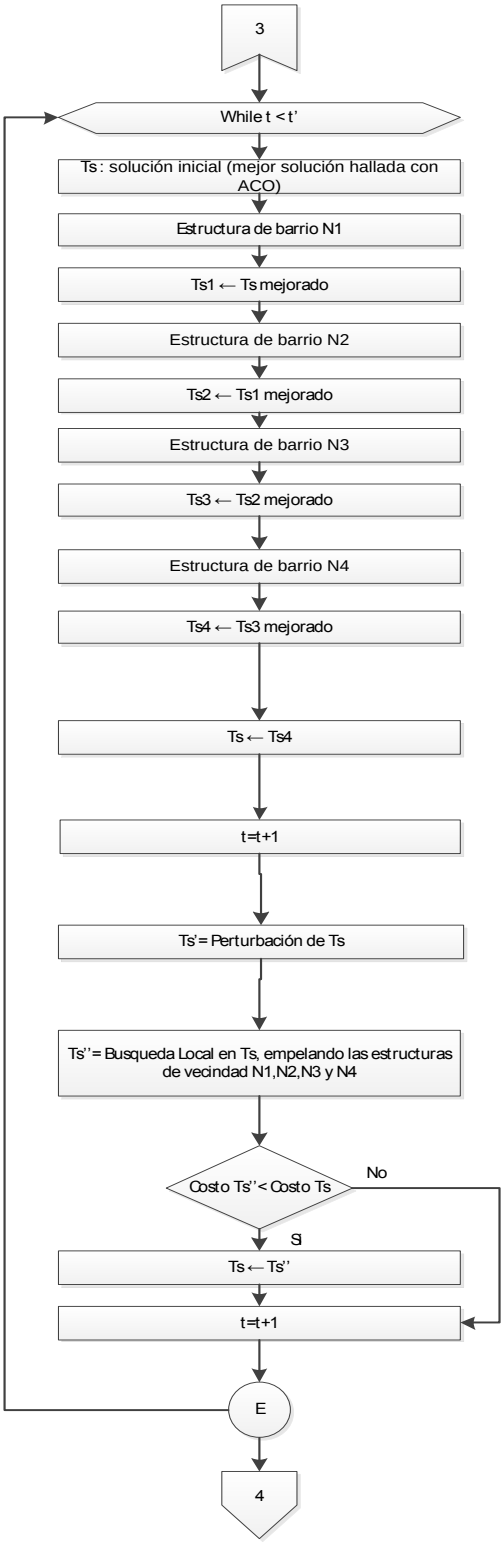
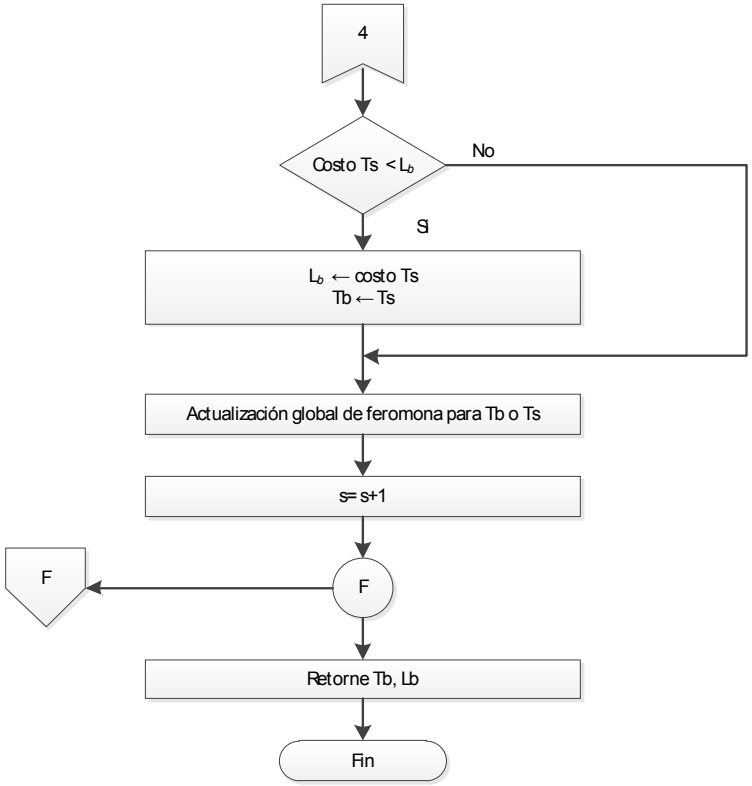


Figura 19. Diagrama de flujo del algoritmo hibrido de colonia de hormigas e ILS (continuación)


4
Costo Ts < L_b
No
Si
L_b ← costo Ts
Tb ← Ts
Actualización global de feromona para Tb o Ts
s= s+1
F
F
Retorne Tb, Lb
Fin

ANEXO B. PROCESO DE SELECCIÓN DEL EJEMPLO PROTOTIPO I

A continuación se presenta un ejemplo que permite entender los conceptos anteriormente expuestos y el desarrollo del toolbox en matlab. El ejemplo es propio de los autores y se basa el artículo de Thing y Chen (2013), explicado a lo largo del framework desarrollado.

Los datos de entrada son los siguientes:

Tabla 19. Datos iniciales ejemplo prototipo I

m	n	R_1	R_2	R_3	R_4	R_5	d_1	d_2	d_3	d_4	d_5	d_6	d_7
5	7	40	55	35	50	60	20	15	25	10	20	10	15
F_1	F_2	F_3	F_4	F_5									
100	200	120	150	150									

Los valores de demanda y capacidad están dados en toneladas y los costos asociados a la instalación de depósitos en millones COP.

Con los datos anteriores, se tiene la matriz inicial de feromona:

$$\tau_j^0 = \left[\frac{1}{100} \quad \frac{1}{200} \quad \frac{1}{120} \quad \frac{1}{150} \quad \frac{1}{150}\right]$$

Información heurística:

$\eta_1 = \frac{40}{w100}$; $\eta_2 = \frac{55}{200}$; $\eta_3 = \frac{35}{120}$; $\eta_4 = \frac{50}{150}$; $\eta_5 = \frac{60}{150}$

Los siguientes parámetros se toman de experimentos preliminares de Thing y Chen (2013):

Tabla 20 Parámetros para ejemplo Prototipo I

b	α	β	γ	ρ	ρ'	ρ''	q_0	q_0'	q_0''	S
4	1	1	4	0, 1	0, 1	0, 1	0, 5	0, 1	0, 5	10

Para el proceso de selección no se tienen en cuenta todos los parámetros mencionados pues algunos son exclusivos de colonias posteriores.

- **Iteración $s = 1$:**

  $\boldsymbol{h = 1;\ U(1,3) = 1:}$

  $P_1^1 = \frac{20+15+25+10+20+10+15}{\frac{40+55+35+50+60}{5}} + 1 = 2{,}4 + 1 = 3 + 1 = 4\ depósitos$

  $\boldsymbol{h = 2;\ U(1,3) = 2:}$

  $P_1^2 = \frac{20+15+25+10+20+10+15}{\frac{40+55+35+50+60}{5}} + 2 = 2{,}4 + 2 = 3 + 2 = 5\ depósitos$

  $\boldsymbol{h = 3;\ U(1,3) = 3:}$

  $P_1^3 = \frac{20 + 15 + 25 + 10 + 20 + 10 + 15}{\frac{40 + 55 + 35 + 50 + 60}{5}} + 3 = 2{,}4 + 3 = 3 + 3 = 6$

  $= 5\ depósitos$

  $\boldsymbol{h = 4;\ U(1,3) = 1:}$

  $= \frac{20 + 15 + 25 + 10 + 20 + 10 + 15}{\frac{40 + 55 + 35 + 50 + 60}{5}} + 1 = 2{,}4 + 1 = 3 + 1 = 4$

Este proceso se repite hasta cumplir un límite de 25 iteraciones.

Para representar el proceso de selección, sólo se tiene en cuenta la iteración $s = 1$ y la solución dada por la hormiga $h = 1$.

Se tiene $P_1^1 = 4$ por tanto la hormiga $h = 1$ selecciona 4 depósitos de los $m = 5$ que tiene disponibles.

Antes de seleccionar un depósito, se genera un número aleatorio q que guía cada elección.

- $q = 0{,}3$

  Como $q \leq q_0$ la selección del primer depósito se da por la ecuación (28):

  $\underset{j \in O_1^1}{\operatorname{argmax}} \left[\left(\frac{1}{100}\right)(0{,}4)^1\right], \left[\left(\frac{1}{200}\right)(0{,}275)^1\right], \left[\left(\frac{1}{120}\right)\left(\frac{11}{40}\right)^1\right], \left[\left(\frac{1}{150}\right)\left(\frac{1}{3}\right)^1\right], \left[\left(\frac{1}{150}\right)(0{,}4)^1\right]$

$\underset{j\epsilon O_1^1}{\text{argmax}}[0{,}004], [0{,}0014], [0{,}0023]\ , [0{,}0022], [0{,}0027]$

El valor máximo de la función es 0,004 que corresponde al depósito $j = 1$.

Por consiguiente, se selecciona este depósito.

El conjunto de depósitos aún no seleccionados queda así:

$O_1^1 = \{2{,}3{,}4{,}5\}$

- $q = 0{,}1$

  $\underset{j\epsilon O_1^1}{\text{argmax}}[0{,}0014], [0{,}0023]\ , [0{,}0022], [0{,}0027]$

  El siguiente depósito elegido es el número 5 que corresponde al valor máximo (0,0027).

  $O_1^1 = \{2{,}3{,}4\}$

- $q = 0{,}4$

  $\underset{j\epsilon O_1^1}{\text{argmax}}[0{,}0014], [0{,}0023]\ , [0{,}0022]$

  El siguiente depósito elegido es el número 3, que corresponde al valor máximo (0,0023).

  $O_1^1 = \{2{,}4\}$

- $q = 0{,}8$

  Como $q > q_0$, se usa la distribución de probabilidad dada en (29):

$$J: P_2^1 = \frac{\tau_2^1(\eta_2)^1}{\sum_{j\epsilon O_1^1} \tau_j^1(\eta_j)^1} = \frac{\left(\frac{1}{200}\right)(0{,}275)^1}{\left(\frac{1}{200} * 0{,}275\right) + \left(\frac{1}{150} * \frac{1}{3}\right)} = \frac{99}{259}$$

$$J: P_4^1 = \frac{\tau_4^1(\eta_4)^1}{\sum_{j\epsilon O_1^1} \tau_j^1(\eta_j)^1} = \frac{\left(\frac{1}{150}\right)\left(\frac{1}{3}\right)^1}{\left(\frac{1}{200} * 0{,}275\right) + \left(\frac{1}{150} * \frac{1}{3}\right)} = \frac{160}{259}$$

Teniendo en cuenta las estas probabilidades se usa "la ruleta" y se selecciona el depósito$j = 4$. El proceso se detiene porque se ha alcanzado un valor $P_1^1 = 4$.

El conjunto de depósitos seleccionados por la hormiga $h = 1$, en la iteración $s = 1$ es:

$W_1^1 = \{1,5,3,4\}$

ANEXO C. ASIGNACIÓN DE CLIENTES A LOS DEPÓSITOS DEL EJEMPLO PROTOTIPO I

El valor de feromona inicial se hace igual a 0, $\xi_0 = 0{,}01$ para todas las aristas.

C_{il} es el costo asociado a la distancia entre los nodos i y l. Donde $l \in I \cup J$. Se tiene la matriz C_{il} (filas: clientes, columnas: clientes y depósitos):

Tabla 21. Matriz de costos asociados a la distancia entre clientes-clientes y clientes-depósitos, del ejemplo prototipo I

	1	2	3	4	5	6	7	d1	d2	d3	d4	d5
1	0	50	80	120	180	190	210	50	100	200	220	180
2	50	0	30	70	130	140	160	50	80	140	190	140
3	80	30	0	40	100	110	130	80	30	30	100	100
4	120	70	40	0	60	70	90	110	20	100	80	70
5	180	130	100	60	0	20	30	120	90	60	70	30
6	190	140	110	70	20	0	20	140	110	50	10	60
7	210	160	130	90	30	20	0	200	120	90	20	100

En la iteración $s = 1$ la hormiga $h = 1$ elige la solución ($W_1^1 = \{1,5,3,4\}$) dada por la $h = 1$ de la primera colonia (selección):

Inicialmente, se tiene:

$A_1^{11} = \{d1\}$; $A_3^{11} = \{d3\}$; $A_4^{11} = \{d4\}$; $A_5^{11} = \{d5\}$

Iteración $s = 1$:

- La hormiga $h = 1$ parte del cliente $i = 1$, para iniciar su proceso de asignación.

  Se halla la información heurística que relaciona el cliente $i = 1$ con cada depósito k:

$$D_{11} = \min_{l \epsilon A_1^{11}}(C_{1d1}) = \min_{l \epsilon A_1^{11}}(50) = 50 \qquad \Rightarrow \qquad \Psi_{11}^1 = \frac{1}{50}$$

$$D_{13} = \min_{l \epsilon A_3^{11}}(C_{1d3}) = \min_{l \epsilon A_3^{11}}(200) = 200 \qquad \Rightarrow \qquad \Psi_{13}^1 = \frac{1}{200}$$

$$D_{14} = \min_{l \epsilon A_4^{11}}(C_{1d4}) = \min_{l \epsilon A_4^{11}}(220) = 220 \qquad \Rightarrow \qquad \Psi_{14}^1 = \frac{1}{220}$$

$$D_{15} = \min_{l \epsilon A_5^{11}}(C_{1d5}) = \min_{l \epsilon A_5^{11}}(180) = 180 \qquad \Rightarrow \qquad \Psi_{15}^1 = \frac{1}{180}$$

Se genera un aleatorio q'.

$q' = 0{,}07$

Como $q' \leq q_0'$, se sigue la ecuación (33):

$$\underset{k \in W_1^1}{\operatorname{argmax}} \left[(0{,}01)\left(\frac{1}{50}\right)^1\right], \left[(0{,}01)\left(\frac{1}{200}\right)^1\right], \left[(0{,}01)\left(\frac{1}{220}\right)^1\right], \left[(0{,}01)\left(\frac{1}{180}\right)^1\right]$$

$$\underset{k \in W_1^1}{\operatorname{argmax}} [0{,}0002], [0{,}00005], [0{,}000045], [0{,}000056]$$

Como el valor máximo para la función se alcanza cuando $k = 1$, el cliente $i = 1$ se asigna a este depósito.

El conjunto A_1^{11} se modifica agregando el cliente asignado. $A_1^{11} = \{d1{,}1\}$

- La hormiga continúa con el cliente $i = 2$

  Se halla la información heurística que relaciona el cliente $\boldsymbol{i = 2}$ con cada depósito $\boldsymbol{k}$:

$$D_{21} = \min_{l \epsilon A_1^{11}}(C_{2d1}), (C_{21}) = \min_{l \epsilon A_1^{11}}(50), (50) = 50 \qquad \Rightarrow \qquad \Psi_{21}^1 = \frac{1}{50}$$

$$D_{23} = \min_{l \epsilon A_3^{11}}(C_{23}) = \min_{l \epsilon A_3^{11}}(140) = 140 \qquad \Rightarrow \qquad \Psi_{23}^1 = \frac{1}{140}$$

$$D_{24} = \min_{l \epsilon A_4^{11}}(C_{24}) = \min_{l \epsilon A_4^{11}}(190) = 190 \qquad \Rightarrow \qquad \Psi_{24}^1 = \frac{1}{190}$$

$$D_{25} = \min_{l \epsilon A_5^{11}}(C_{25}) = \min_{l \epsilon A_5^{11}}(140) = 140 \qquad \Rightarrow \qquad \Psi_{25}^1 = \frac{1}{140}$$

Se genera un aleatorio q'.

$q' = 0{,}09$

Como $q' \leq q_0'$, se sigue la ecuación (33):

$$\underset{k \epsilon W_1^1}{\operatorname{argmax}} \left[(0{,}01)\left(\frac{1}{50}\right)^1\right], \left[(0{,}01)\left(\frac{1}{140}\right)^1\right], \left[(0{,}01)\left(\frac{1}{190}\right)^1\right], \left[(0{,}01)\left(\frac{1}{140}\right)^1\right]$$

$$\underset{k \epsilon W_1^1}{\operatorname{argmax}} [0{,}0002], [0{,}00007], [0{,}00005], [0{,}00007]$$

Como el valor máximo para la función se alcanza cuando $k = 1$, el cliente $i = 2$ se asigna a este depósito.

El conjunto A_1^{11} se modifica agregando el cliente asignado. $A_1^{11} = \{d1{,}1{,}2\}$

- La hormiga continúa con el cliente $i = 3$

  Se halla la información heurística que relaciona el cliente $\boldsymbol{i = 3}$ con cada depósito $\boldsymbol{k}$:

$$D_{31} = \min_{l \epsilon A_1^{11}}(C_{3d1}), (C_{31}), (C_{32}) = \min_{l \epsilon A_1^{11}}(80), (80), (30) = 30 \quad \Rightarrow \quad \Psi_{31}^1 = \frac{1}{30}$$

$$D_{33} = \min_{l \epsilon A_3^{11}}(C_{33}) = \min_{l \epsilon A_3^{11}}(20) = 20 \quad \Rightarrow \quad \Psi_{33}^1 = \frac{1}{20}$$

$$D_{34} = \min_{l \epsilon A_4^{11}}(C_{34}) = \min_{l \epsilon A_4^{11}}(100) = 100 \quad \Rightarrow \quad \Psi_{34}^1 = \frac{1}{100}$$

$$D_{35} = \min_{l \epsilon A_5^{11}}(C_{35}) = \min_{l \epsilon A_5^{11}}(100) = 100 \quad \Rightarrow \quad \Psi_{35}^1 = \frac{1}{100}$$

  Se genera un aleatorio $\boldsymbol{q}'$.

  $q' = 0{,}03$

  Como $q' \leq q_0'$, se sigue la ecuación (33):

$$\underset{k \epsilon W_1^1}{\operatorname{argmax}} \left[(0{,}01)\left(\frac{1}{30}\right)^1\right], \left[(0{,}01)\left(\frac{1}{20}\right)^1\right], \left[(0{,}01)\left(\frac{1}{100}\right)^1\right], \left[(0{,}01)\left(\frac{1}{100}\right)^1\right]$$

$$\underset{k \epsilon W_1^1}{\operatorname{argmax}} [0{,}000333], [0{,}0005], [0{,}0001], [0{,}0001]$$

  El valor máximo para la función se da cuando $k = 3$, el cliente $i = 3$ se asigna a este depósito. El conjunto A_3^{11} se modifica agregando el cliente asignado. $A_3^{11} = \{d3{,}3\}$

- La hormiga continúa con el cliente $i = 4$

Se halla la información heurística que relaciona el cliente $i = 4$ con cada depósito k:

$$D_{41} = \min_{l \epsilon A_1^{11}}(C_{4d1}), (C_{41}), (C_{42}) = \min_{l \epsilon A_1^{11}}(110), (120), (70) = 70 \quad \Rightarrow \Psi_{41}^1 = \frac{1}{70}$$

$$D_{43} = \min_{l \epsilon A_3^{11}}(C_{4d3}), (C_{43}) = \min_{l \epsilon A_3^{11}}(200), (40) = 40 \quad \Rightarrow \Psi_{43}^1 = \frac{1}{40}$$

$$D_{44} = \min_{l \epsilon A_4^{11}}(C_{44}) = \min_{l \epsilon A_4^{11}}(80) = 80 \quad \Rightarrow \Psi_{44}^1 = \frac{1}{80}$$

$$D_{45} = \min_{l \epsilon A_5^{11}}(C_{45}) = \min_{l \epsilon A_5^{11}}(70) = 70 \quad \Rightarrow \Psi_{45}^1 = \frac{1}{70}$$

Se genera un aleatorio q'.

$q' = 0{,}03$

Como $q' \leq q_0'$, se sigue la ecuación (33):

$$\underset{k \in W_1^1}{\operatorname{argmax}} \left[(0{,}01)\left(\frac{1}{70}\right)^1\right], \left[(0{,}01)\left(\frac{1}{40}\right)^1\right], \left[(0{,}01)\left(\frac{1}{80}\right)^1\right], \left[(0{,}01)\left(\frac{1}{70}\right)^1\right]$$

$$\underset{k \in W_1^1}{\operatorname{argmax}} [0{,}0001429], [0{,}00025], [0{,}000125], [0{,}0001429]$$

El valor máximo para la función se alcanza cuando $k = 3$, el cliente *i*= 4 se asigna a este depósito. El conjunto A_3^{11} se modifica agregando el cliente asignado. $A_3^{11} = \{d3{,}3{,}4\}$

- La hormiga continúa con el cliente $i = 5$

Se halla la información heurística que relaciona el cliente $i = 5$ con cada depósito k:

$$D_{51} = \min_{l \epsilon A_1^{11}}(C_{5d1}), (C_{51})(C_{52}) = \min_{l \epsilon A_1^{11}}(120), (180), (130) = 120 \quad \Rightarrow \Psi_{51}^1 = \frac{1}{120}$$

$$D_{53} = \min_{l \epsilon A_3^{11}}(C_{5d3}), (C_{53}), (C_{54}) = \min_{l \epsilon A_3^{11}}(60), (100), (60) = 60 \quad \Rightarrow \Psi_{53}^1 = \frac{1}{60}$$

$$D_{54} = \min_{l \epsilon A_4^{11}}(C_{54}) = \min_{l \epsilon A_4^{11}}(70) = 70 \quad \Rightarrow \Psi_{54}^1 = \frac{1}{70}$$

$$D_{55} = \min_{l \epsilon A_5^{11}}(C_{55}) = \min_{l \epsilon A_5^{11}}(30) = 30 \quad \Rightarrow \Psi_{55}^1 = \frac{1}{30}$$

Se genera un aleatorio q'.

$q' = 0{,}02$

Como $q' \leq q_0'$, se sigue la ecuación (33):

$$\underset{k\epsilon W_1^1}{\text{argmax}}\left[(0{,}01)\left(\frac{1}{120}\right)^1\right],\left[(0{,}01)\left(\frac{1}{60}\right)^1\right],\left[(0{,}01)\left(\frac{1}{70}\right)^1\right],\left[(0{,}01)\left(\frac{1}{30}\right)^1\right]$$

$$\underset{k\epsilon W_1^1}{\text{argmax}}[0{,}0000833],[0{,}000167],[0{,}0001429],[0{,}00033]$$

Como el valor máximo para la función es generado cuando $k = 5,$ el cliente $i = 5$ será asignado a este depósito. El conjunto A_5^{11} se modifica agregando el cliente asignado $A_5^{11} = \{d5{,}5\}$.

- La hormiga continúa con el cliente $i = 6$

  Se halla la información heurística que relaciona el cliente $\boldsymbol{i = 6}$ con cada depósito $\boldsymbol{k}$:

$$D_{61} = \min_{l\epsilon A_1^{11}}(C_{6d1}),(C_{61})(C_{62}) = \min_{l\epsilon A_1^{11}}(140),(190),(140) = 140 \quad \Rightarrow \Psi_{61}^1 = \frac{1}{140}$$

$$D_{63} = \min_{l\epsilon A_3^{11}}(C_{6d3}),(C_{63}),(C_{64}) = \min_{l\epsilon A_3^{11}}(50),(110),(70) = 50 \quad \Rightarrow \Psi_{63}^1 = \frac{1}{50}$$

$$D_{64} = \min_{l\epsilon A_4^{11}}(C_{64}) = \min_{l\epsilon A_4^{11}}(10) = 10 \quad \Rightarrow \Psi_{64}^1 = \frac{1}{10}$$

$$D_{65} = \min_{l\epsilon A_5^{11}}(C_{6d5}),(C_{65}) = \min_{l\epsilon A_5^{11}}(60),(20) = 60 \quad \Rightarrow \Psi_{65}^1 = \frac{1}{20}$$

  Se genera un aleatorio $\boldsymbol{q}'$.

  $q' = 0{,}5$

  Como $q' > q_0'$, se hallan las probabilidades para cada uno de los depósitos, siguiendo la ecuación (34)

$$K\colon P_{ik}^h = \frac{\xi_{ik}^s(\Psi_{ik}^s)^\beta}{\sum_{k\epsilon W_s^h}\xi_{ik}^s(\Psi_{ik}^s)^\beta}$$

$$K\colon P_{61}^1 = \frac{\xi_{61}^1(\Psi_{61}^1)^1}{\sum_{k\epsilon W_1^1}\xi_{6k}^1(\Psi_{6k}^1)^1}$$

$$= \frac{0{,}01 * (\frac{1}{140})^1}{0{,}01 * \left(\frac{1}{140}\right)^1 + 0{,}01 * \left(\frac{1}{50}\right)^1 + 0{,}01 * \left(\frac{1}{10}\right)^1 + 0{,}01 * (\frac{1}{20})^1}$$

$$= \frac{5}{124}$$

$$K: P_{63}^1 = \frac{\xi_{63}^1 (\Psi_{63}^1)^1}{\sum_{k \epsilon W_1^1} \xi_{6k}^1 (\Psi_{6k}^1)^1}$$

$$= \frac{0{,}01 * (\frac{1}{50})^1}{0{,}01 * \left(\frac{1}{140}\right)^1 + 0{,}01 * \left(\frac{1}{50}\right)^1 + 0{,}01 * \left(\frac{1}{10}\right)^1 + 0{,}01 * (\frac{1}{20})^1}$$

$$= \frac{7}{62}$$

$$K: P_{64}^1 = \frac{\xi_{64}^1 (\Psi_{64}^1)^1}{\sum_{k \epsilon W_1^1} \xi_{6k}^1 (\Psi_{6k}^1)^1}$$

$$= \frac{0{,}01 * (\frac{1}{10})^1}{0{,}01 * \left(\frac{1}{140}\right)^1 + 0{,}01 * \left(\frac{1}{50}\right)^1 + 0{,}01 * \left(\frac{1}{10}\right)^1 + 0{,}01 * (\frac{1}{20})^1}$$

$$= \frac{35}{62}$$

$$K: P_{65}^1 = \frac{\xi_{65}^1 (\Psi_{65}^1)^1}{\sum_{k \epsilon W_1^1} \xi_{6k}^1 (\Psi_{6k}^1)^1}$$

$$= \frac{0{,}01 * (\frac{1}{20})^1}{0{,}01 * \left(\frac{1}{140}\right)^1 + 0{,}01 * \left(\frac{1}{50}\right)^1 + 0{,}01 * \left(\frac{1}{10}\right)^1 + 0{,}01 * (\frac{1}{20})^1}$$

$$= \frac{35}{124}$$

Usando la 'ruleta', la hormiga $h = 1$ asigna al cliente $i = 6$ al depósito $k = 4$.

El conjunto A_4^{11} se modifica agregando el cliente asignado $A_4^{11} = \{d4{,}6\}$.

- La hormiga continúa con el cliente $i = 7$

Se halla la información heurística que relaciona el cliente $i = 7$ con cada depósito k:

$$D_{71} = \min_{l \epsilon A_1^{11}} (C_{7d1}), (C_{71})(C_{72}) = \min_{l \epsilon A_1^{11}} (200), (210), (50) = 50 \quad \Rightarrow \Psi_{71}^1 = \frac{1}{50}$$

$$D_{73} = \min_{l \epsilon A_3^{11}} (C_{7d3}), (C_{73}), (C_{74}) = \min_{l \epsilon A_3^{11}} (90), (130), (90) = 90 \quad \Rightarrow \Psi_{73}^1 = \frac{1}{90}$$

$$D_{74} = \min_{l \epsilon A_4^{11}} (C_{7d4}), (C_{76}) = \min_{l \epsilon A_4^{11}} (20), (20) = 20 \quad \Rightarrow \Psi_{74}^1 = \frac{1}{20}$$

$$D_{75} = \min_{l \epsilon A_5^{11}} (C_{6d5}), (C_{75}) = \min_{l \epsilon A_5^{11}} (60), (30) = 30 \quad \Rightarrow \Psi_{75}^1 = \frac{1}{30}$$

Se genera un aleatorio q'.

$q' = 0{,}034$

Como $q' \leq q_0'$, se sigue la ecuación (33):

$$\underset{k \in W_1^1}{\operatorname{argmax}} \left[(0{,}01)\left(\frac{1}{50}\right)^1\right], \left[(0{,}01)\left(\frac{1}{90}\right)^1\right], \left[(0{,}01)\left(\frac{1}{20}\right)^1\right], \left[(0{,}01)\left(\frac{1}{30}\right)^1\right]$$

$$\underset{k \in W_1^1}{\operatorname{argmax}} [0{,}0002], [0{,}000111], [0{,}0005], [0{,}00033]$$

El valor máximo para la función es generado cuando $k = 4$, el cliente $i = 7$ es asignado a este depósito.

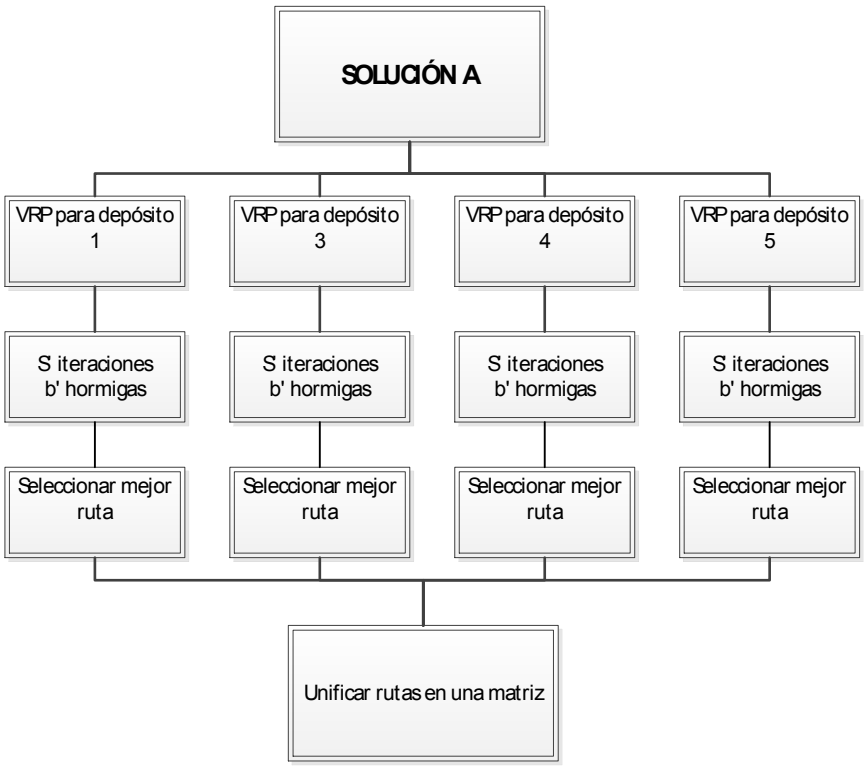
La solución dada por cada hormiga, puede ser representada por una matriz donde cada columna representa el depósito seleccionado y los valores en éstas, los clientes asignados en el orden en que fueron seleccionados.

Así, la solución de $h = 1$ en $s = 1$, se representa:

$$A = \begin{vmatrix} 1 & 5 & 3 & 6 \\ 2 & 0 & 4 & 7 \end{vmatrix}$$

ANEXO D. SOLUCIÓN DEL PROBLEMA DE RUTEO PARA UN EJEMPLO PROTOTIPO II

Del ejemplo prototipo I, se tiene la respuesta $A = \begin{vmatrix} 1 & 5 & 3 & 6 \\ 2 & 0 & 4 & 7 \end{vmatrix}$, para la cual, en términos generales, el VRP se realiza como se muestra en la ilustración:



Para ilustrar la manera como trabaja el algoritmo para en el proceso de VRP, se describe un ejemplo llamado 'Prototipo II', que parte de una solución independiente al ejemplo anterior:

$W_1^1 = \{2,4\}$

$$E = \begin{vmatrix} 1 & 2 \\ 4 & 3 \\ 5 & 0 \\ 6 & 0 \\ 7 & 0 \end{vmatrix}$$

En esta solución, se tiene que los depósitos seleccionados son $k = 2$ y $k = 4$, donde los clientes $1{,}4{,}5{,}6\ y\ 7$ son asignados a $k = 2$ y los clientes $2\ y\ 3$ a $k = 4$.

Para el VRP de cada uno de los depósitos seleccionados, se tiene $b' = 2$ y $S' = 3$. Como se trata de un proceso extenso, en el ejemplo Prototipo II sólo se considera el depósito $k = 2$. Los costos C_{il} para los clientes asignados a este depósito son:

Tabla 22. Matriz de costos asociados a la distancia entre clientes-clientes, clientes-depósitos para ejemplo prototipo II

	1	**4**	**5**	**6**	**7**	**d2**
1	0	23	34	21	13	20
4	23	0	23	32	24	30
5	34	23	0	50	12	24
6	21	32	50	0	20	32
7	13	24	12	20	0	15

Para el valor inicial de feromona que relacionan los clientes asignados a los depósitos, se toma un valor $\zeta_0 = 0{,}01$ al igual que para el proceso de asignación.

Para los clientes asignados a $k = 2$, la matriz inicial de feromona es:

Tabla 23. Matriz de feromona para el depósito $K = 2$ y sus clientes asignados, del ejemplo Prototipo II

	1	**4**	**5**	**6**	**7**
1		0,01	0,01	0,01	0,01
4	0,01		0,01	0,01	0,01
5	0,01	0,01		0,01	0,01
6	0,01	0,01	0,01		0,01
7	0,01	0,01	0,01	0,01	

Iteración $s' = 1$:

- Hormiga $h' = 1$, va del depósito al cliente $i = 5$ (escogido aleatoriamente). El costo de la ruta hasta el momento es: $Costo = 24$

  El conjunto de nodos no seleccionados aún en la iteración $s' = 1$ y la $h' = 1$, está conformado así:

  $Z_1^1 = \{1\ 4\ 6\ 7\}$

  Se halla el ahorro (φ) entre $i = 5$ y los demás clientes:

  $\varphi_{51}^1 = C_{50} + C_{01} - C_{51} = 24 + 20 - 34 = 10$

  $\varphi_{54}^1 = C_{50} + C_{04} - C_{54} = 24 + 30 - 23 = 31$

  $\varphi_{56}^1 = C_{50} + C_{06} - C_{56} = 24 + 32 - 50 = 6$

  $\varphi_{57}^1 = C_{50} + C_{07} - C_{57} = 24 + 15 - 12 = 27$

  Se genera un aleatorio $q" = 0{,}3$, como $q" \leq q_0''$, se sigue la ecuación (11):

  $$v = \underset{v \epsilon Z_1^1}{\arg max}\left[\zeta_{5v}^1 (\varphi_{5v}^1)^4\right] = \underset{v \epsilon Z_1^1}{\arg max}[0{,}01*(10)^4][0{,}01*(31)^4][0{,}01*(6)^4][0{,}01*(27)^4]$$

  $$v = \underset{v \epsilon Z_1^1}{\arg max}[100][9235{,}211][19{,}96][5314{,}41]$$

  El valor máximo para la función es generado cuando $v = 4$, por tanto, la hormiga que está en el nodo $i = 5$, se dirige al nodo $i = 4$.

  $Costo = 24 + 23 = 47$

  Se hace actualización local para el arco (i, v):

  $\zeta_{54}^1 \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

  Como se observa, el valor de feromona no cambia porque para la primera iteración la cantidad de feromona es ζ_0.

  Se actualiza el conjunto $Z_1^1 = \{1\ 6\ 7\}$

  Una vez ubicada la hormiga en $i = 4$, elige el siguiente nodo.

  Se halla la relación de φ entre $i = 4$ y los demás clientes que pertenecen a Z_1^1:

  $\varphi_{41}^1 = C_{40} + C_{01} - C_{41} = 30 + 20 - 23 = 27$

  $\varphi_{46}^1 = C_{40} + C_{06} - C_{46} = 30 + 32 - 32 = 30$

  $\varphi_{47}^1 = C_{40} + C_{07} - C_{47} = 30 + 15 - 24 = 21$

  Se genera un aleatorio $q" = 0{,}6$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_1^1}{\arg max} \left[\zeta_{4v}^1 (\varphi_{4v}^1)^4\right] = \underset{v \epsilon Z_1^1}{\arg max}^{[0,01*(27)^4][0,01*(30)^4][0,01*(21)^4]}$$

$$v = \underset{v \epsilon Z_1^1}{\arg max}^{[5314,41][8100][1944,81]}$$

El valor máximo para la función es generado cuando $v = 6$, por tanto, la hormiga que está en el nodo $i = 4$, se dirige al nodo $i = 6$.

$Costo = 24 + 23 + 32 = 79$

Se hace actualización local para el arco (i, v):

$\zeta_{46}^1 \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

Como se observa, el valor de feromona no cambia porque para la primera iteración la cantidad de feromona es ζ_0.

Se actualiza el conjunto $Z_1^1 = \{1\ \ 7\}$

Una vez ubicada la hormiga en $i = 6$, elige el siguiente cliente.

Se halla la relación de φ entre $i = 6$ y los demás clientes que pertenecen a Z_1^1:

$\varphi_{61}^1 = C_{60} + C_{01} - C_{61} = 32 + 20 - 21 = 31$

$\varphi_{67}^1 = C_{60} + C_{07} - C_{67} = 32 + 15 - 13 = 34$

Se genera un aleatorio $q" = 0{,}6$, como $q'' > q_0''$, se deben hallar las probabilidades y a partir de estas, seleccionar el siguiente cliente:

$$P_{61}^1 = \frac{\zeta_{61}^1 (\varphi_{61}^1)^4}{\sum_{v \epsilon Z_{s'}^{h'}} \zeta_{6v}^1 (\varphi_{6v}^1)^4} = \frac{0{,}01 * (31)^4}{0{,}01 * (31)^4 + 0{,}01 * (34)^4} = 0{,}41$$

$$P_{67}^1 = \frac{\zeta_{67}^1 (\varphi_{67}^1)^4}{\sum_{v \epsilon Z_{s'}^{h'}} \zeta_{6v}^1 (\varphi_{6v}^1)^4} = \frac{0{,}01 * (34)^4}{0{,}01 * (31)^4 + 0{,}01 * (34)^4} = 0{,}59$$

Con la 'ruleta', se elige el cliente $i = 1$

$Costo = 24 + 23 + 32 + 21 = 100$

Se hace actualización local para el arco (i, v):

$\zeta_{61}^1 \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

Como se observa, el valor de feromona no cambia porque para la primera iteración la cantidad de feromona es ζ_0.

Se actualiza el conjunto $Z_1^1 = \{7\}$

Una vez ubicada la hormiga en $i = 1$, elige el siguiente cliente.

Se halla la relación de φ entre $i = 1$ y los demás clientes que pertenecen a Z_1^1:

$\varphi_{17}^1 = C_{10} + C_{07} - C_{17} = 20 + 15 - 13 = 22$

Se genera un aleatorio $q" = 0{,}8$, como $q'' > q_0''$, se deben hallar las probabilidades y a partir de estas, seleccionar el siguiente cliente:

$$P_{17}^1 = \frac{\zeta_{17}^1(\varphi_{17}^1)^4}{\sum_{v \epsilon Z_{s'}^{h'}} \zeta_{6v}^1(\varphi_{6v}^1)^4} = \frac{0{,}01 * (22)^4}{0{,}01 * (22)^4} = 1$$

Con la 'ruleta', se elige el cliente $i = 7$

$Costo = 24 + 23 + 32 + 21 + 13 = 113$

Se hace actualización local para el arco (i, v):

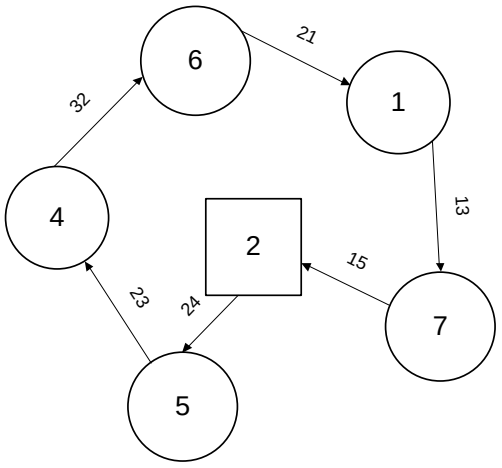
$\zeta_{17}^1 \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

Como se observa, el valor de feromona no cambia porque para la primera iteración la cantidad de feromona es ζ_0.

Ahora el conjunto Z_1^1 no contiene ningún elemento, por tanto la hormiga $h' = 1$ regresa al depósito.

$Costo = 24 + 23 + 32 + 21 + 13 + 15 = 128$

La ruta construida por la hormiga $h' = 1$ es:



Con un $Costo = 128$.

- $h' = 2$

  Hormiga $h' = 2$, va del depósito al cliente $i = 1$ (escogido aleatoriamente). El costo de la ruta hasta el momento es: $Costo = 20$

  El conjunto de nodos no seleccionados aún en la iteración $s' = 1$ y la $h' = 2$, está conformado así:

  $Z_1^2 = \{4\ 5\ 6\ 7\}$

  Se halla el ahorro (φ) entre $i = 1$ y los demás clientes:

  $\varphi_{14}^1 = C_{10} + C_{04} - C_{14} = 20 + 30 - 23 = 27$

  $\varphi_{15}^1 = C_{10} + C_{05} - C_{15} = 20 + 24 - 34 = 10$

  $\varphi_{16}^1 = C_{10} + C_{06} - C_{16} = 20 + 32 - 21 = 31$

  $\varphi_{17}^1 = C_{10} + C_{07} - C_{17} = 20 + 15 - 13 = 22$

  Se genera un aleatorio $q" = 0{,}1$, como $q" \leq q_0''$, se sigue la ecuación (11):

  $$v = \underset{v \epsilon Z_1^2}{\arg max}\left[\zeta_{1v}^1 (\varphi_{1v}^1)^4\right] = \underset{v \epsilon Z_1^2}{\arg max}^{[0{,}01*(27)^4][0{,}01*(10)^4][0{,}01*(31)^4][0{,}01*(22)^4]}$$

  $$v = \underset{v \epsilon Z_1^2}{\arg max}\left[\zeta_{1v}^1 (\varphi_{1v}^1)^4\right] = \underset{v \epsilon Z_1^2}{\arg max}^{[5314{,}41][100][9235{,}21][2342{,}56]}$$

  El valor máximo para la función es generado cuando $v = 6$, por tanto, la hormiga que está en el nodo $i = 1$, se dirige al nodo $i = 6$.

  $Costo = 20 + 21 = 41$

  Se hace actualización local para el arco (i, v):

  $\zeta_{16}^1 \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

Se actualiza el conjunto $Z_1^2 = \{4\ 5\ 7\}$

Una vez ubicada la hormiga en $i = 6$, elige el siguiente nodo.

Se halla la relación de φ entre $i = 6$ y los demás clientes que pertenecen a Z_1^2:

$\varphi_{64}^1 = C_{60} + C_{04} - C_{64} = 32 + 30 - 32 = 30$

$\varphi_{65}^1 = C_{60} + C_{05} - C_{65} = 32 + 24 - 50 = 6$

$\varphi_{67}^1 = C_{60} + C_{07} - C_{67} = 32 + 15 - 20 = 27$

Se genera un aleatorio $q" = 0{,}4$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_1^2}{\arg max}\left[\zeta_{6v}^1 (\varphi_{6v}^1)^4\right] = \underset{v \epsilon Z_1^2}{\arg max}^{[0{,}01*(30)^4][0{,}01*(6)^4][0{,}01*(27)^4]}$$

$$v = \underset{v \epsilon Z_1^2}{\arg max}\left[\zeta_{6v}^1 (\varphi_{6v}^1)^4\right] = \underset{v \epsilon Z_1^2}{\arg max}^{[8100][12{,}96][5314{,}41]}$$

El valor máximo para la función es generado cuando $v = 4$, por tanto, la hormiga que está en el nodo $i = 6$, se dirige al nodo $i = 4$.

$Costo = 20 + 21 + 32 = 73$

Se hace actualización local para el arco (i, v):

$\zeta_{64}^1 \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

Se actualiza el conjunto $Z_1^2 = \{\ 5\ 7\}$

Una vez ubicada la hormiga en $i = 4$, elige el siguiente nodo.

Se halla la relación de φ entre $i = 4$ y los demás clientes que pertenecen a Z_1^2:

$\varphi_{45}^1 = C_{40} + C_{05} - C_{45} = 30 + 24 - 23 = 31$

$\varphi_{47}^1 = C_{40} + C_{07} - C_{47} = 30 + 15 - 24 = 21$

Se genera un aleatorio $q" = 0{,}05$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_1^2}{\arg max}\left[\zeta_{4v}^1 (\varphi_{4v}^1)^4\right] = \underset{v \epsilon Z_1^2}{\arg max}^{[0{,}01*(31)^4][0{,}01*(21)^4]}$$

$$v = \underset{v \epsilon Z_1^2}{\arg max}\left[\zeta_{4v}^1 (\varphi_{4v}^1)^4\right] = \underset{v \epsilon Z_1^2}{\arg max}^{[9235{,}21][1944{,}81]}$$

El valor máximo para la función es generado cuando $v = 5$, por tanto, la hormiga que está en el nodo $i = 4$, se dirige al nodo $i = 5$.

$Costo = 20 + 21 + 32 + 23 = 96$

Se hace actualización local para el arco (i, v):

$\zeta_{45}^{1} \leftarrow (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}01 = 0{,}01$

Se actualiza el conjunto $Z_1^2 = \{7\}$

Una vez ubicada la hormiga en $i = 5$, elige el siguiente nodo.

Se halla la relación de φ entre $i = 5$ y los demás clientes que pertenecen a Z_1^2:

$\varphi_{57}^{1} = C_{50} + C_{07} - C_{57} = 24 + 15 - 12 = 27$

Se genera un aleatorio $q" = 0{,}2$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_1^2}{\arg max} \left[\zeta_{5v}^{1} (\varphi_{5v}^{1})^4\right] = \underset{v \epsilon Z_1^2}{\arg max} [0{,}01 * (27)^4]$$

$$v = \underset{v \epsilon Z_1^2}{\arg max} \left[\zeta_{5v}^{1} (\varphi_{5v}^{1})^4\right] = \underset{v \epsilon Z_1^2}{\arg max} [5314{,}41]$$

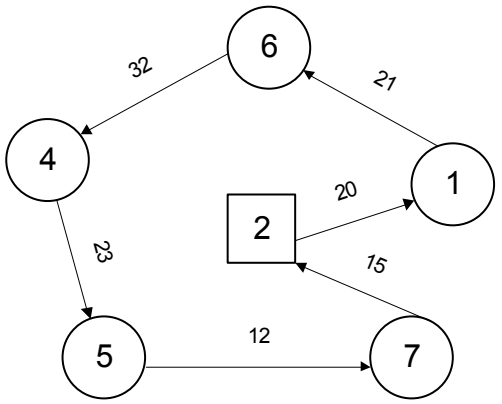
El siguiente nodo es $i = 7$.

$Costo = 20 + 21 + 32 + 23 + 12 = 108$

Del nodo $i = 7$, la hormiga se dirige al depósito pues ya no quedan más clientes por visitar.

$Costo = 20 + 21 + 32 + 23 + 12 + 15 = 123$

La ruta construida por la hormiga $h' = 2$ es:



Elegir mejor hormiga (aquella que genere la mejor solución, es decir con menor costo).

La mejor solución está dada por la hormiga $h' = 2$, con un $Costo = 123$. Ésta es la mejor solución de la iteración.

$$Mejor\ solución\ iteración\ s' = 1: \begin{vmatrix} 1 \\ 6 \\ 4 \\ 5 \\ 7 \end{vmatrix}$$

Como es la primera iteración, la solución de esta hormiga también es la mejor solución global.

$$Mejor\ solución\ global: \begin{vmatrix} 1 \\ 6 \\ 4 \\ 5 \\ 7 \end{vmatrix}\ Costo = 123$$

El costo de la peor solución es $Costo = 128$.

Hacer actualización global de feromona:

$$\Delta\zeta_{16}^{1} = \Delta\zeta_{61}^{1} = \frac{[(128 - 123) + (128 - 123)]}{128} = 0{,}078125$$

$$\Delta\zeta_{64}^{1} = \Delta\zeta_{46}^{1} = \frac{[(128 - 123) + (128 - 123)]}{128} = 0{,}078125$$

$$\Delta\zeta_{45}^{1} = \Delta\zeta_{54}^{1} = \frac{[(128 - 123) + (128 - 123)]}{128} = 0{,}078125$$

$$\Delta\zeta_{57}^{1} = \Delta\zeta_{75}^{1} = \frac{[(128 - 123) + (128 - 123)]}{128} = 0{,}078125$$

Para los demás arcos, $\Delta\zeta_{iv}^{1} = 0$ porque no hacen parte de la mejor solución global, ni de la mejor solución de la iteración.

$$\zeta_{14}^{2} = \zeta_{41}^{2} = (1 - \rho'')\zeta_{14}^{1} + \rho''\Delta\zeta_{14}^{1} = (1 - 0{,}1)0{,}01 + 0{,}1 * 0 = 0{,}009$$

$$\zeta_{15}^{2} = \zeta_{51}^{2} = (1 - \rho'')\zeta_{15}^{1} + \rho''\Delta\zeta_{15}^{1} = (1 - 0{,}1)0{,}01 + 0{,}1 * 0 = 0{,}009$$

$$\zeta_{16}^{2} = \zeta_{61}^{2} = (1 - \rho'')\zeta_{16}^{1} + \rho''\Delta\zeta_{16}^{1} = (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}078125 = 0{,}0098$$

$$\zeta_{17}^{2} = \zeta_{71}^{2} = (1 - \rho'')\zeta_{17}^{1} + \rho''\Delta\zeta_{17}^{1} = (1 - 0{,}1)0{,}01 + 0{,}1 * 0 = 0{,}009$$

$$\zeta_{45}^{2} = \zeta_{54}^{2} = (1 - \rho'')\zeta_{45}^{1} + \rho''\Delta\zeta_{45}^{1} = (1 - 0{,}1)0{,}01 + 0{,}1 * 0{,}078125 = 0{,}0098$$

$$\zeta_{46}^2 = \zeta_{64}^2 = (1-\rho'')\zeta_{46}^1 + \rho''\Delta\zeta_{46}^1 = (1-0{,}1)0{,}01 + 0{,}1 * 0{,}078125 = 0{,}0098$$
$$\zeta_{47}^2 = \zeta_{74}^2 = (1-\rho'')\zeta_{47}^1 + \rho''\Delta\zeta_{47}^1 = (1-0{,}1)0{,}01 + 0{,}1 * 0 = 0{,}009$$
$$\zeta_{56}^2 = \zeta_{65}^2 = (1-\rho'')\zeta_{56}^1 + \rho''\Delta\zeta_{56}^1 = (1-0{,}1)0{,}01 + 0{,}1 * 0 = 0{,}009$$
$$\zeta_{57}^2 = \zeta_{75}^2 = (1-\rho'')\zeta_{57}^1 + \rho''\Delta\zeta_{57}^1 = (1-0{,}1)0{,}01 + 0{,}1 * 0{,}078125 = 0{,}0098$$
$$\zeta_{67}^2 = \zeta_{76}^2 = (1-\rho'')\zeta_{67}^1 + \rho''\Delta\zeta_{67}^1 = (1-0{,}1)0{,}01 + 0{,}1 * 0 = 0{,}009$$

La matriz actualizada es la siguiente:

Tabla 24. Matriz de feromona actualizada globalmente, en s'=1, para k=2

	1	**4**	**5**	**6**	**7**
1		0,009	0,009	0,0098	0,009
4	0,009		0,0098	0,0098	0,009
5	0,009	0,0098		0,009	0,0098
6	0,0098	0,0098	0,009		0,009
7	0,009	0,009	0,0098	0,009	

Iteración $s' = 2$:

- Hormiga $h' = 1$, va del depósito al cliente $i = 4$ (escogido aleatoriamente).
  El costo de la ruta hasta el momento es: $Costo = 30$
  El conjunto de nodos no seleccionados aún en la iteración $s' = 2$ y la $h' = 1$, está conformado así:
  $Z_2^1 = \{1\ 5\ 6\ 7\}$
  Se halla el ahorro (φ) entre $i = 4$ y los demás clientes:
  $\varphi_{41}^1 = C_{40} + C_{01} - C_{41} = 30 + 20 - 23 = 27$
  $\varphi_{45}^1 = C_{40} + C_{05} - C_{45} = 30 + 24 - 23 = 31$
  $\varphi_{46}^1 = C_{40} + C_{06} - C_{46} = 30 + 32 - 32 = 30$
  $\varphi_{47}^1 = C_{40} + C_{07} - C_{47} = 30 + 15 - 24 = 21$
  Se genera un aleatorio $q" = 0{,}3$, como $q" \leq q_0''$, se sigue la ecuación (11):
  $$v = \underset{v \epsilon Z_2^1}{\arg max}\left[\zeta_{4v}^1 (\varphi_{4v}^1)^4\right] = \underset{v \epsilon Z_2^1}{\arg max}\left[0{,}009*(27)^4\right]\left[0{,}0098*(31)^4\right]\left[0{,}0098*(30)^4\right]\left[0{,}009*(21)^4\right]$$

$$v = \underset{v \epsilon Z_2^1}{\arg max}^{[4782,97][9050,51][7938][1750,33]}$$

El valor máximo para la función es generado cuando $v = 5$, por tanto, la hormiga que está en el nodo $i = 4$, se dirige al nodo $i = 5$.

$Costo = 30 + 23 = 53$

Se hace actualización local para el arco (i, v):

$\zeta_{45}^2 \leftarrow (1 - 0{,}1)0{,}0098 + 0{,}1 * 0{,}01 = 0{,}00982$

Se actualiza el conjunto $Z_2^1 = \{1\ 6\ 7\}$

Ubicada la hormiga en $i = 5$, se elige el siguiente nodo:

Se halla el ahorro (φ) entre $i = 5$ y los demás clientes:

$\varphi_{51}^1 = C_{50} + C_{01} - C_{51} = 24 + 20 - 34 = 10$

$\varphi_{56}^1 = C_{50} + C_{06} - C_{56} = 24 + 32 - 50 = 6$

$\varphi_{57}^1 = C_{50} + C_{07} - C_{57} = 24 + 15 - 12 = 27$

Se genera un aleatorio $q" = 0{,}003$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_2^1}{\arg max}\left[\zeta_{5v}^1 \left(\varphi_{5v}^1\right)^4\right] = \underset{v \epsilon Z_2^1}{\arg max}^{[0{,}009*(10)^4][0{,}009*(6)^4][0{,}0098*(27)^4]}$$

$$v = \underset{v \epsilon Z_2^1}{\arg max}^{[90][9050,51][11,66][5208,12]}$$

El valor máximo para la función es generado cuando $v = 7$, por tanto, la hormiga que está en el nodo $i = 5$, se dirige al nodo $i = 7$.

$Costo = 30 + 23 + 12 = 65$

Se hace actualización local para el arco (i, v):

$\zeta_{57}^2 \leftarrow (1 - 0{,}1)0{,}0098 + 0{,}1 * 0{,}01 = 0{,}00982$

Se actualiza el conjunto $Z_2^1 = \{1\ 6\}$

Ubicada la hormiga en $i = 7$, se elige el siguiente nodo:

Se halla el ahorro (φ) entre $i = 7$ y los demás clientes:

$\varphi_{71}^1 = C_{70} + C_{01} - C_{71} = 15 + 20 - 13 = 22$

$\varphi_{76}^1 = C_{70} + C_{06} - C_{76} = 15 + 32 - 20 = 27$

Se genera un aleatorio $q" = 0{,}002$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_2^1}{\arg max}\left[\zeta_{7v}^1 (\varphi_{7v}^1)^4\right] = \underset{v \epsilon Z_2^1}{\arg max}^{[0{,}009*(22)^4][0{,}009*(27)^4]}$$

$$v = \underset{v \epsilon Z_2^1}{\arg max}^{[2108{,}3][4782{,}97]}$$

El valor máximo para la función es generado cuando $v = 6$, por tanto, la hormiga que está en el nodo $i = 7$, se dirige al nodo $i = 6$.

$Costo = 30 + 23 + 12 + 20 = 85$

Se hace actualización local para el arco (i, v):

$\zeta_{76}^2 \leftarrow (1 - 0{,}1)0{,}009 + 0{,}1 * 0{,}01 = 0{,}0091$

Se actualiza el conjunto $Z_2^1 = \{1\}$

Ubicada la hormiga en $i = 6$, se elige el siguiente nodo:

Se halla el ahorro (φ) entre $i = 6$ y los demás clientes:

$\varphi_{61}^1 = C_{60} + C_{01} - C_{61} = 32 + 20 - 21 = 31$

Se genera un aleatorio $q" = 0{,}01$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_2^1}{\arg max}\left[\zeta_{6v}^1 (\varphi_{6v}^1)^4\right] = \underset{v \epsilon Z_2^1}{\arg max}^{[0{,}0098*(31)^4]}$$

$$v = \underset{v \epsilon Z_2^1}{\arg max}^{[9050{,}51]}$$

El valor máximo para la función es generado cuando $v = 1$, por tanto, la hormiga que está en el nodo $i = 6$, se dirige al nodo $i = 1$.

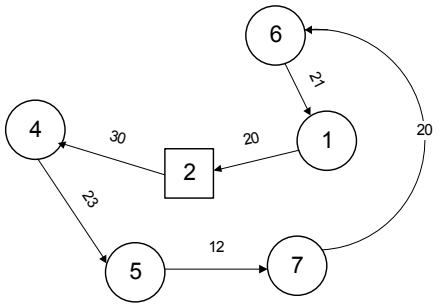
$Costo = 30 + 23 + 12 + 20 + 21 = 106$

Se hace actualización local para el arco (i, v):

$\zeta_{61}^2 \leftarrow (1 - 0{,}1)0{,}0098 + 0{,}1 * 0{,}01 = 0{,}00982$

El conjunto Z_2^1 no contiene más elementos por tanto la hormiga se dirige al depósito.

$Costo = 30 + 23 + 12 + 20 + 21 + 20 = 126$



Matriz actualizada:

Tabla 25. Matriz de feromona actualizada localmente, para $h' = 1$, $s' = 2$, $k = 2$, ejemplo prototipo II

	1	4	5	6	7
1		0,009	0,009	0,00982	0,009
4	0,009		0,00982	0,0098	0,009
5	0,009	0,00982		0,009	0,00982
6	0,00982	0,0098	0,009		0,0091
7	0,009	0,009	0,00982	0,0091	

- Hormiga $h' = 2$, va del depósito al cliente $i = 7$ (escogido aleatoriamente). El costo de la ruta hasta el momento es: $Costo = 15$

  El conjunto de nodos no seleccionados aún en la iteración $s' = 2$ y la $h' = 2$, está conformado así:

  $Z_2^1 = \{1\ 4\ 5\ 6\}$

  Se halla el ahorro (φ) entre $i = 7$ y los demás clientes:

  $\varphi_{71}^1 = C_{70} + C_{01} - C_{71} = 15 + 20 - 23 = 12$

  $\varphi_{74}^1 = C_{70} + C_{04} - C_{74} = 15 + 23 - 24 = 14$

  $\varphi_{75}^1 = C_{70} + C_{05} - C_{75} = 15 + 24 - 23 = 16$

  $\varphi_{76}^1 = C_{70} + C_{06} - C_{76} = 15 + 32 - 20 = 27$

  Se genera un aleatorio $q" = 0{,}4$, como $q" \leq q_0''$, se sigue la ecuación (11):

  $$v = \underset{v \epsilon Z_2^1}{\arg max}\left[\zeta_{7v}^1 (\varphi_{7v}^1)^4\right] = \underset{v \epsilon Z_2^1}{\arg max}\left[0{,}009*(12)^4\right]\left[0{,}009*(14)^4\right]\left[0{,}00982*(16)^4\right]\left[0{,}0091*(27)^4\right]$$

$$v = \underset{v \epsilon Z_2^1}{\arg max}^{[186,62][345,74][643,56][4836,11]}$$

El valor máximo para la función es generado cuando $v = 6$, por tanto, la hormiga que está en el nodo $i = 7$, se dirige al nodo $i = 6$.

$Costo = 15 + 20 = 35$

Se hace actualización local para el arco (i, v):

$\zeta_{76}^2 \leftarrow (1 - 0{,}1)0{,}0091 + 0{,}1 * 0{,}01 = 0{,}00919$

Se actualiza el conjunto $Z_2^1 = \{1\ 4\ 5\}$

Ubicada la hormiga en $i = 6$, se elige el siguiente nodo:

Se halla el ahorro (φ) entre $i = 6$ y los demás clientes:

$\varphi_{61}^1 = C_{60} + C_{01} - C_{61} = 32 + 20 - 21 = 31$

$\varphi_{64}^1 = C_{60} + C_{04} - C_{64} = 32 + 23 - 32 = 23$

$\varphi_{65}^1 = C_{60} + C_{05} - C_{65} = 32 + 24 - 50 = 6$

Se genera un aleatorio $q" = 0{,}62$, como $q" > q_0''$, se hallan las probabilidades:

$$P_{61}^2 = \frac{\zeta_{61}^2(\varphi_{61}^2)^4}{\sum_{v \epsilon Z_2^2} \zeta_{6v}^2(\varphi_{6v}^2)^4} = \frac{0{,}00982 * (31)^4}{0{,}00982 * (31)^4 + 0{,}0098 * (23)^4 + 0{,}009 * (6)^4}$$

$$= 0{,}767$$

$$P_{64}^2 = \frac{\zeta_{64}^2(\varphi_{64}^2)^4}{\sum_{v \epsilon Z_2^2} \zeta_{6v}^2(\varphi_{6v}^2)^4} = \frac{0{,}0098 * (23)^4}{0{,}00982 * (31)^4 + 0{,}0098 * (23)^4 + 0{,}009 * (6)^4}$$

$$= 0{,}232$$

$$P_{65}^2 = \frac{\zeta_{65}^2(\varphi_{65}^2)^4}{\sum_{v \epsilon Z_2^2} \zeta_{6v}^2(\varphi_{6v}^2)^4} = \frac{0{,}009 * (6)^4}{0{,}00982 * (31)^4 + 0{,}0098 * (23)^4 + 0{,}009 * (6)^4}$$

$$= 0{,}001$$

Usando la 'ruleta', se selecciona el cliente $i = 4$.

$Costo = 15 + 20 + 32 = 67$

Se hace actualización local para el arco (i, v):

$\zeta_{64}^2 \leftarrow (1 - 0{,}1)0{,}0098 + 0{,}1 * 0{,}01 = 0{,}00982$

Se actualiza el conjunto $Z_2^1 = \{1\ \ 5\}$

Ubicada la hormiga en $i = 4$, se elige el siguiente nodo.

Se halla el ahorro (φ) entre $i = 4$ y los demás clientes:

$\varphi_{41}^{1} = C_{40} + C_{01} - C_{41} = 30 + 20 - 23 = 27$

$\varphi_{45}^{1} = C_{40} + C_{05} - C_{45} = 30 + 24 - 23 = 31$

Se genera un aleatorio $q" = 0{,}9$, como $q" > q_0''$, se hallan las probabilidades:

$$P_{41}^{2} = \frac{\zeta_{41}^{2}(\varphi_{41}^{2})^4}{\sum_{v \epsilon Z_2^2} \zeta_{4v}^{2}(\varphi_{4v}^{2})^4} = \frac{0{,}009 * (27)^4}{0{,}009 * (27)^4 + 0{,}00982 * (31)^4} = 0{,}345$$

$$P_{45}^{2} = \frac{\zeta_{45}^{2}(\varphi_{45}^{2})^4}{\sum_{v \epsilon Z_2^2} \zeta_{4v}^{2}(\varphi_{4v}^{2})^4} = \frac{0{,}00982 * (31)^4}{0{,}009 * (27)^4 + 0{,}00982 * (31)^4} = 0{,}655$$

Usando la 'ruleta', se selecciona el cliente $i = 5$.

$Costo = 15 + 20 + 32 + 23 = 90$

Se hace actualización local para el arco (i, v):

$\zeta_{45}^{2} \leftarrow (1 - 0{,}1)0{,}00982 + 0{,}1 * 0{,}01 = 0{,}008938$

Se actualiza el conjunto $Z_2^1 = \{1\}$

Ubicada la hormiga en $i = 5$, se elige el siguiente nodo.

Se halla el ahorro (φ) entre $i = 5$ y los demás clientes:

$\varphi_{51}^{1} = C_{50} + C_{01} - C_{51} = 24 + 20 - 34 = 10$

Se genera un aleatorio $q" = 0{,}33$, como $q" \leq q_0''$, se sigue la ecuación (11):

$$v = \underset{v \epsilon Z_2^1}{\arg max}\left[\zeta_{5v}^{1}(\varphi_{5v}^{1})^4\right] = \underset{v \epsilon Z_2^1}{\arg max}^{[0{,}009*(10)^4]}$$

$$v = \underset{v \epsilon Z_2^1}{\arg max}^{[90]}$$

El valor máximo para la función es generado cuando $v = 1$, por tanto, la hormiga que está en el nodo $i = 5$, se dirige al nodo $i = 1$

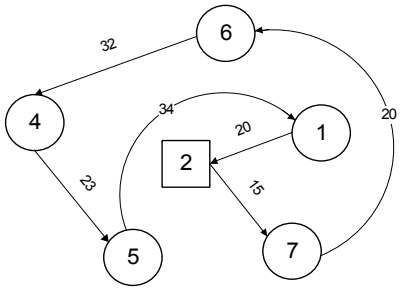
$Costo = 15 + 20 + 32 + 23 + 34 = 124$

Se hace actualización local para el arco (i, v):

$\zeta_{51}^{2} \leftarrow (1 - 0{,}1)0{,}009 + 0{,}1 * 0{,}01 = 0{,}0091$

El conjunto Z_2^1 no contiene más elementos, por tanto la hormiga se dirige al depósito.

$Costo = 15 + 20 + 32 + 23 + 34 + 20 = 144$



Matriz actualizada:

Tabla 26. Matriz de feromona actualizada localmente para h'=2, s'=2, k=2, ejemplo prototipo II

	1	**4**	**5**	**6**	**7**
1		0,009	0,0091	0,00982	0,009
4	0,009		0,008938	0,00982	0,009
5	0,0091	0,008938		0,009	0,00982
6	0,00982	0,00982	0,009		0,00919
7	0,009	0,009	0,00982	0,00919	

Elegir mejor hormiga (aquella que genere la mejor solución, es decir con menor costo).

La mejor solución está dada por la hormiga $h' = 1$, con un $Costo = 126$. Ésta es la mejor solución de la iteración.

$$Mejor\ solución\ iteración\ s' = 2: \begin{vmatrix} 4 \\ 5 \\ 7 \\ 6 \\ 1 \end{vmatrix}$$

$$Mejor\ solución\ global: \begin{vmatrix} 1 \\ 6 \\ 4 \\ 5 \\ 7 \end{vmatrix}\ Costo = 123$$

El costo de la peor solución es $Costo = 144$.

Actualización global de feromona:

$$\Delta\zeta_{16}^{2} = \Delta\zeta_{61}^{2} = \frac{[(144 - 123) + (144 - 126)]}{144} = 0{,}2708$$

$$\Delta\zeta_{46}^{2} = \Delta\zeta_{64}^{2} = \frac{[(144 - 123) + (144 - 126)]}{144} = 0{,}2708$$

$$\Delta\zeta_{45}^{2} = \Delta\zeta_{54}^{2} = \frac{[(144 - 123) + (144 - 126)]}{144} = 0{,}2708$$

$$\Delta\zeta_{57}^{2} = \Delta\zeta_{75}^{2} = \frac{[(144 - 123) + (144 - 126)]}{144} = 0{,}2708$$

$$\Delta\zeta_{76}^{2} = \Delta\zeta_{67}^{2} = \frac{[(144 - 123) + (144 - 126)]}{144} = 0{,}2708$$

Para los demás arcos, $\Delta\zeta_{iv}^{2} = 0$ porque no hacen parte de la mejor solución global, ni de la mejor solución de la iteración.

$$\zeta_{14}^{3} = \zeta_{41}^{3} = (1-\rho'')\zeta_{14}^{2} + \rho''\Delta\zeta_{14}^{2} = (1-0{,}1)0{,}009 + 0{,}1 * 0 = 0{,}0081$$

$$\zeta_{15}^{3} = \zeta_{51}^{3} = (1-\rho'')\zeta_{15}^{2} + \rho''\Delta\zeta_{15}^{2} = (1-0{,}1)0{,}0091 + 0{,}1 * 0 = 0{,}00819$$

$$\zeta_{16}^{3} = \zeta_{61}^{3} = (1-\rho'')\zeta_{16}^{2} + \rho''\Delta\zeta_{16}^{2} = (1-0{,}1)0{,}00982 + 0{,}1 * 0{,}2708 = 0{,}03592$$

$$\zeta_{17}^{3} = \zeta_{71}^{3} = (1-\rho'')\zeta_{17}^{2} + \rho''\Delta\zeta_{17}^{2} = (1-0{,}1)0{,}009 + 0{,}1 * 0 = 0{,}0081$$

$$\zeta_{45}^{3} = \zeta_{54}^{3} = (1-\rho'')\zeta_{45}^{2} + \rho''\Delta\zeta_{45}^{2} = (1-0{,}1)0{,}008938 + 0{,}1 * 0{,}2708 = 0{,}03512$$

$$\zeta_{46}^{3} = \zeta_{64}^{3} = (1-\rho'')\zeta_{46}^{2} + \rho''\Delta\zeta_{46}^{2} = (1-0{,}1)0{,}00982 + 0{,}1 * 0{,}2708 = 0{,}03592$$

$$\zeta_{47}^{3} = \zeta_{74}^{3} = (1-\rho'')\zeta_{47}^{2} + \rho''\Delta\zeta_{47}^{2} = (1-0{,}1)0{,}009 + 0{,}1 * 0 = 0{,}00819$$

$$\zeta_{56}^{3} = \zeta_{65}^{3} = (1-\rho'')\zeta_{56}^{2} + \rho''\Delta\zeta_{56}^{2} = (1-0{,}1)0{,}009 + 0{,}1 * 0 = 0{,}00819$$

$$\zeta_{57}^{3} = \zeta_{75}^{3} = (1-\rho'')\zeta_{57}^{2} + \rho''\Delta\zeta_{57}^{2} = (1-0{,}1)0{,}00982 + 0{,}1 * 0{,}2708 = 0{,}03592$$

$$\zeta_{67}^{3} = \zeta_{76}^{3} = (1-\rho'')\zeta_{67}^{2} + \rho''\Delta\zeta_{67}^{2} = (1-0{,}1)0{,}00919 + 0{,}1 * 0{,}2708 = 0{,}03535$$

Matriz actualizada:

Tabla 27. Matriz actualizada globalmente en s'=2, k=2, ejemplo prototipo II

	1	4	5	6	7

1		0,0081	0,00819	0,03592	0,009
4	0,0081		0,03512	0,03592	0,00819
5	0,00819	0,03512		0,00819	0,03592
6	0,03592	0,03592	0,00819		0,03535
7	0,0081	0,00819	0,03592	0,03535	

ANEXO E. DESCRIPCIÓN DE VARIABLES USADAS EN LAS FUNCIONES DESARROLLADAS DEL MÉTODO ACO+ILS

En forma breve, se explican a continuación las variables empleadas en las funciones desarrolladas del método ACO+ILS.

Función: 'ProbabilidadSeleccion'

```
function [VectorProbabilidad,Correccion]=ProbabilidadSeleccion
(Tao_j,Nu_j,alpha,m)

P_hj=round(100.*(((Tao_j).*((Nu_j).^alpha))./(sum((Tao_j).*((Nu_j).^alpha
))))); %Determina el porcentaje de probabilidad de cada deposito a ser
seleccionado
            Correccion=sum(P_hj);

        Origen1(1)=1;
        for (jj=2:m)
             Origen1(jj)=Origen1(jj-1)+P_hj(jj-1);
        end

        for (ii=1:m)
             if (ii==1)
                for (jjj=1:P_hj(1))
                VectorProbabilidad(jjj)=1;
                end
             else
                for (jjj=Origen1(ii):Correccion)
                VectorProbabilidad(jjj)=ii;
                end
             end
        end
end
```

Descripcion: para el proceso de selección se crea un vector que contiene los depósitos disponibles una cantidad Ph_j para cada uno, de éste se selecciona al azar el depósito cuando se sigue la ecuación de exploración.

- **Entradas**
  Tao_j: valor de feromona para el depósito j.
  Nu_j: información heurística para el depósito j.
  alpha: parámetro que determina la influencia relativa entre la información heurística y el valor asociado de feromona para el proceso de selección.
  m: cantidad de depósitos candidatos.

- **Salidas**

Vector probabilidad: Crea un vector que contiene los depósitos disponibles una cantidad Ph_j para cada uno

Corrección: sumatoria de los elementos de Ph_j

Función: 'ILS'

```
function
[MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorOpcion,V
alor1,Valor3,Valor36]=ILS(Valor3,Valor1,MejorOpcionRuta1,MejorOpcionRuta,
Iteracion,MejorOpcionDepositos,CostoMejorOpcion,di,Rjj,Fjj,Clie_Dep,Clie_
Clie,T)
global T Valor1 Valor3 MejorOpcionRuta MejorOpcionRuta1
MejorOpcionDepositos CostoMejorOpcion di Rjj Fjj Clie_Dep Iteracion
Clie_Clie
global PosDepositosNulos DepositosNulos CantDepositosNulos


for Ronda21=1:T %Analisis N1
    disp('Analisis de la Estructura de Barrio N1: ')
    N1=Ronda21
    if Valor1>1 %Un intercambio solo es viable para mas de un deposito

    %%Analisis de la existencia de depositos sin clientes

[PosDepositosNulos,DepositosNulos,CantDepositosNulos]=DepositosVacios(Val
or1,MejorOpcionRuta,Iteracion,MejorOpcionDepositos)

    %%Intercambio de clientes entre depositos
    clearvars Valor5 Valor6 Valor7 Valor8 Valor9 Valor10 Valor11 Valor12
Valor13
    Valor5=ceil(rand*Valor1); %Deposito al azar
    Valor6=ceil(rand*Valor1); %Deposito al azar
    clearvars ViabilidadDeposito1
    [ViabilidadDeposito1]=find(PosDepositosNulos==Valor5);
    while sum(ViabilidadDeposito1)>0
    Valor5=ceil(rand*Valor1)
    clearvars ViabilidadDeposito1
    [ViabilidadDeposito1]=find(PosDepositosNulos==Valor5);
    end
    [ViabilidadDeposito2]=find(PosDepositosNulos==Valor6);
    while (Valor5==Valor6)||(sum(ViabilidadDeposito2)>0)
        Valor6=ceil(rand*Valor1) %Asi se evita que se repita el deposito
        clearvars ViabilidadDeposito2
        [ViabilidadDeposito2]=find(PosDepositosNulos==Valor6)
    end
    Valor7=ceil(rand*Valor3); %Fila de la matriz MejorOpcionRuta
    clearvars Contador
    Contador=0;
    while MejorOpcionRuta(Iteracion,Valor7,Valor5)==0
        if Contador<20
        Valor7=ceil(rand*Valor3); %Asi se evita que el cliente
seleccionado sea nulo
        elseif Contador==20
```

```
            Valor7=1
        else
            Valor5=ceil(rand*Valor1);
            clearvars ViabilidadDeposito1
            [ViabilidadDeposito1]=find(PosDepositosNulos==Valor5);
            while (sum(ViabilidadDeposito1)>0)||(Valor5==Valor6)
            Valor5=ceil(rand*Valor1);
            clearvars ViabilidadDeposito1
            [ViabilidadDeposito1]=find(PosDepositosNulos==Valor5);
            end
        end
    end
    %
    Valor8=ceil(rand*Valor3); %Fila de la matriz MejorOpcionRuta
    clearvars Contador
    Contador=0;
    while MejorOpcionRuta(Iteracion,Valor8,Valor6)==0
        if Contador<20
        Valor8=ceil(rand*Valor3); %Asi se evita que el cliente
seleccionado sea nulo
        elseif Contador==20
            Valor8=1
        else
            Valor6=ceil(rand*Valor1);
            clearvars ViabilidadDeposito1
            [ViabilidadDeposito1]=find(PosDepositosNulos==Valor5);
            while (sum(ViabilidadDeposito1)>0)||(Valor5==Valor6)
                Valor6=ceil(rand*Valor1);
                clearvars ViabilidadDeposito1
                [ViabilidadDeposito1]=find(PosDepositosNulos==Valor5);
            end
        end
    end
    Valor9=MejorOpcionRuta(Iteracion,Valor7,Valor5); %Se crean las
variables con los clientes a intercambiar
    Valor10=MejorOpcionRuta(Iteracion,Valor8,Valor6); %Se crean las
variables con los clientes a intercambiar
    MejorOpcionRuta(Iteracion,Valor7,Valor5)=Valor10; %Se intercambian
las variables en las casillas seleccionadas
    MejorOpcionRuta(Iteracion,Valor8,Valor6)=Valor9; %Se intercambian las
variables en las casillas seleccionadas

    %%Analisis de la viabilidad segun relacion Capacidad/Demanda
    clearvars  DemandaMejorOpcion CapacidadMejorOpcion Valor14
    for Ronda22=1:Valor1
        if (Ronda22==Valor5)||(Ronda22==Valor6)
            clearvars DemandaMejor
            Valor11=1;
            for Ronda23=1:Valor3
                if MejorOpcionRuta(Iteracion,Ronda23,Ronda22)>0

DemandaMejor(Valor11)=di(MejorOpcionRuta(Iteracion,Ronda23,Ronda22));
                    Valor11=Valor11+1;
                end
            end
            DemandaMejorOpcion(Ronda22)=sum(DemandaMejor);
```

```
CapacidadMejorOpcion(Ronda22)=Rjj(MejorOpcionDepositos(Iteracion,Ronda22)
);
                if
DemandaMejorOpcion(Ronda22)>CapacidadMejorOpcion(Ronda22)
                    Valor14(Ronda22)=1; %Esto significa que no se pudo
encontrar una mejor solucion
                else
                    Valor14(Ronda22)=0;
                end
        else
            Valor14(Ronda22)=0;
        end
    end

    %%Analisis de la viabilidad segun Costos

[Valor1,Valor3,MejorOpcionRuta1,MejorOpcionRuta,CostoMejorOpcion]=Analisi
sCostos(MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorO
pcion,Iteracion,Valor14,Valor1,Valor3,Fjj,Clie_Dep,PosDepositosNulos,Clie
_Clie)

    clearvars Valor5 Valor6 Valor7 Valor8 Valor9 Valor10 Valor11 Valor12
Valor13 Valor14
    clearvars ViabilidadDeposito1 ViabilidadDeposito2 Ronda22 Ronda23
    clearvars DemandaMejor DemandaMejorOpcion CapacidadMejorOpcion

     end
end
CostoMejorOpcion
MejorOpcionDepositos
MejorOpcionRuta
MejorOpcionRuta1
Valor1
Valor3
('Estructura de Barrio N1')
pause;

for Ronda28=1:T %Analisis N2
disp('Analisis de la Estructura de Barrio N2: ')
    N2=Ronda28
    if Valor1>1 %Un intercambio solo es viable para mas de un deposito

    %%Analisis de la existencia de depositos sin clientes

[PosDepositosNulos,DepositosNulos,CantDepositosNulos]=DepositosVacios(Val
or1,MejorOpcionRuta,Iteracion,MejorOpcionDepositos)

    %%Cambio de deposito para un cliente
    Valor16=ceil(rand*Valor1); %Deposito fuente
    Valor76=Valor16;
    Valor17=ceil(rand*Valor1); %Deposito destino
    clearvars ViabilidadDeposito1 ViabilidadDeposito2
    [ViabilidadDeposito1]=find(PosDepositosNulos==Valor16);
    [ViabilidadDeposito2]=find(PosDepositosNulos==Valor17);
    while (sum(ViabilidadDeposito1)>0)
```

```
        Valor16=ceil(rand*Valor1);
        [ViabilidadDeposito1]=find(PosDepositosNulos==Valor16);
        Valor76=Valor16;
    end
    while (Valor16==Valor17)||(sum(ViabilidadDeposito2)>0)
        Valor17=ceil(rand*Valor1);
        [ViabilidadDeposito2]=find(PosDepositosNulos==Valor17);
    end
    Valor78=Valor17;
    Valor18=ceil(rand*Valor3); %Cliente al azar del deposito fuente
    Contador=0;
    while MejorOpcionRuta(Iteracion,Valor18,Valor16)==0
        if Contador<20
        Valor18=ceil(rand*Valor3);
        elseif Contador==20
            Valor18=1; %Se obliga tomar el primer cliente asignado en
caso que no halle otro valido
        else
            Valor16=ceil(rand*Valor1);
            clearvars ViabilidadDeposito1
            [ViabilidadDeposito1]=find(PosDepositosNulos==Valor16);
            while
(Valor16==Valor17)||(Valor16==Valor76)||(sum(ViabilidadDeposito1)>0)
                Valor16=ceil(rand*Valor1);
                clearvars ViabilidadDeposito1
                [ViabilidadDeposito1]=find(PosDepositosNulos==Valor16);
            end
        end
        Contador=Contador+1;
    end
    Valor19=MejorOpcionRuta(Iteracion,Valor18,Valor16); %Guarda en una
variable el valor del cliente
    Valor20=Valor3+1; %Correccion del valor3 para la inclusion del nuevo
cliente
    Valor75=ceil(rand*Valor3); %Posicion del cliente en el deposito
destino
    clearvars Contador Contador2
    Contador=0;
    Contador2=1;
    while
(((MejorOpcionRuta(Iteracion,Valor75,Valor17))>=1)&&(Contador2<=Valor1))
        if Contador<30
        Valor75=ceil(rand*Valor3);
        elseif Contador==30
            Valor75=Valor3;
        else
            Valor17=ceil(rand*Valor1);
            clearvars ViabilidadDeposito2
            [ViabilidadDeposito2]=find(PosDepositosNulos==Valor17);
            while
(Valor16==Valor17)||(Valor17==Valor78)||(sum(ViabilidadDeposito2)>0)
                Valor17=ceil(rand*Valor1)
                clearvars ViabilidadDeposito2
                [ViabilidadDeposito2]=find(PosDepositosNulos==Valor17);
            end
            clearvars Contador
            Contador=-1;
```

```
            Contador2=Contador2+1;
        end
        Contador=Contador+1;
    end

    if MejorOpcionRuta(Iteracion,Valor75,Valor17)>0
        for NuevaColumnaCeros=1:Valor1
            MejorOpcionRuta(:,Valor3+1,NuevaColumnaCeros)=0;
            MejorOpcionRuta1(Valor3+1,NuevaColumnaCeros)=0;
        end
        Valor3=Valor20;
    end

    for Ronda60=1:Valor3
        if Ronda60<Valor75 %Correccion del deposito destino

MejorOpcionRuta(Iteracion,Ronda60,Valor17)=MejorOpcionRuta1(Ronda60,Valor
17);
        elseif Ronda60==Valor75

MejorOpcionRuta(Iteracion,Ronda60,Valor17)=MejorOpcionRuta1(Valor18,Valor
16);
        else

MejorOpcionRuta(Iteracion,Ronda60,Valor17)=MejorOpcionRuta1((Ronda60-
1),Valor17);
        end

        if Ronda60<Valor18 %Correccion del deposito fuente

MejorOpcionRuta(Iteracion,Ronda60,Valor16)=MejorOpcionRuta1(Ronda60,Valor
16);
        elseif (Ronda60>=Valor18)&&(Ronda60<Valor3)

MejorOpcionRuta(Iteracion,Ronda60,Valor16)=MejorOpcionRuta1((Ronda60+1),V
alor16);
        end
    end

    %%Analisis de la viabilidad segun relacion Capacidad/Demanda
    for Ronda29=1:Valor1
        if ((Ronda29==Valor17)||(Ronda29==Valor16))
            Valor21=1;
            for Ronda30=1:Valor3
                DemandaMejor2(Valor21)=0;
                if MejorOpcionRuta(Iteracion,Ronda30,Ronda29)>0

DemandaMejor2(Valor21)=di(MejorOpcionRuta(Iteracion,Ronda30,Ronda29));
                    Valor21=Valor21+1;
                end
            end
            DemandaMejorOpcion2(Ronda29)=sum(DemandaMejor2);

CapacidadMejorOpcion(Ronda29)=Rjj(MejorOpcionDepositos(Iteracion,Ronda29)
);
```

```
                if
DemandaMejorOpcion2(Ronda29)>CapacidadMejorOpcion(Ronda29)
                    Valor14(Ronda29)=1;
                else
                    Valor14(Ronda29)=0;
                end
        else
            Valor14(Ronda29)=0;
        end
    end

    %%Analisis de la viabilidad segun Costos

[Valor1,Valor3,MejorOpcionRuta1,MejorOpcionRuta,CostoMejorOpcion]=Analisi
sCostos(MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorO
pcion,Iteracion,Valor14,Valor1,Valor3,Fjj,Clie_Dep,PosDepositosNulos,
Clie_Clie)

    clearvars Valor14 Valor16 Valor17 Valor18 Valor19 Valor20 Valor21
Valor75 Valor76 Valor78
    clearvars ViabilidadDeposito1 ViabilidadDeposito2 Contador
NuevaColumnaCeros NuevaFilaCeros
    clearvars Ronda29 Ronda30 Ronda60 DemandaMejorOpcion
DemandaMejorOpcion2 DemandaMejor2
    end
end
CostoMejorOpcion
MejorOpcionDepositos
MejorOpcionRuta
MejorOpcionRuta1
Valor1
Valor3
('Estructura de Barrio N2')
pause;

for Ronda35=1:T %Analisis N3
    disp('Analisis de la Estructura de Barrio N3: ')
    N3=Ronda35

    %%Analisis de la existencia de depositos sin clientes

[PosDepositosNulos,DepositosNulos,CantDepositosNulos]=DepositosVacios(Val
or1,MejorOpcionRuta,Iteracion,MejorOpcionDepositos)
    DepositosNulos;
    PosDepositosNulos;
    CantDepositosNulos;
    %%Conteo de clientes por deposito: Solo se puede para depositos con
>= de 2 clientes
    clearvars Ronda40 Valor37 Ronda41
    for Ronda40=1:Valor1
    Valor37(Ronda40)=0;
    for Ronda41=1:Valor3
        if MejorOpcionRuta(Iteracion,Ronda41,Ronda40)>0
            Valor37(Ronda40)=Valor37(Ronda40)+1;
        end
    end
```

```
    end
    Valor28=ceil(rand*Valor1);
    clearvars ViabilidadDeposito1
    [ViabilidadDeposito1]=find(PosDepositosNulos==Valor28);
    while
(Valor37(Valor28)==0)||(Valor37(Valor28)==1)||(sum(ViabilidadDeposito1)>0
)
           Valor28=ceil(rand*Valor1);
    end

    %%Cambio de lugar de Clientes para un mismo deposito
    if Valor37(Valor28)==2
    Valor31=MejorOpcionRuta(Iteracion,1,Valor28);
    Valor32=MejorOpcionRuta(Iteracion,2,Valor28);
    MejorOpcionRuta(Iteracion,1,Valor28)=Valor32;
    MejorOpcionRuta(Iteracion,2,Valor28)=Valor31;
    else
    Valor34=Valor28;
    Valor29=ceil(rand*Valor3); %Seleccion del primer cliente
    Valor30=ceil(rand*Valor3); %Seleccion del segundo cliente
    clearvars Contador
    Contador=0;
    while (MejorOpcionRuta(Iteracion,Valor30,Valor28)==0)
        if Contador<20
        Valor30=ceil(rand*Valor3);
        elseif Contador==20
            Valor30=1;
        else
                Valor28=ceil(rand*Valor1);
                clearvars ViabilidadDeposito1
                [ViabilidadDeposito1]=find(PosDepositosNulos==Valor28);
            while
(Valor34==Valor28)||(sum(ViabilidadDeposito1)>0)||(Valor37(Valor28)<=2)
                Valor28=ceil(rand*Valor1);
                Valo34=Valor28;
                clearvars ViabilidadDeposito1
                [ViabilidadDeposito1]=find(PosDepositosNulos==Valor28);
            end
        end
        Contador=Contador+1;
    end
    clearvars Contador
    Contador=0;
    while
(MejorOpcionRuta(Iteracion,Valor29,Valor28)==0)||(Valor29==Valor30)
        if Contador<20
        Valor29=ceil(rand*Valor3);
        else
            Valor29=Valor37(Valor28);
        end
        Contador=Contador+1;
    end
    Valor31=MejorOpcionRuta(Iteracion,Valor29,Valor28)
    Valor32=MejorOpcionRuta(Iteracion,Valor30,Valor28)
    MejorOpcionRuta(Iteracion,Valor29,Valor28)=Valor32;
    MejorOpcionRuta(Iteracion,Valor30,Valor28)=Valor31;
    end
```

```
    %%Analisis de la viabilidad segun Costos
    clearvars Valor14
    Valor14=0;

[Valor1,Valor3,MejorOpcionRuta1,MejorOpcionRuta,CostoMejorOpcion]=Analisi
sCostos(MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorO
pcion,Iteracion,Valor14,Valor1,Valor3,Fjj,Clie_Dep,PosDepositosNulos,
Clie_Clie)
    clearvars Valor14

    clearvars Valor28 Valor29 Valor30 Valor31 Valor32 Valor34 Valor37
    clearvars Roda40 Contador2
end
CostoMejorOpcion
MejorOpcionDepositos
MejorOpcionRuta
MejorOpcionRuta1
Valor1
Valor3
('Estructura de Barrio N3')
pause;

for Ronda39=1:T %Analisis N4
disp('Analisis de la Estructura de Barrio N4: ')
N4=Ronda39

%%Analisis de la existencia de depositos sin clientes
[PosDepositosNulos,DepositosNulos,CantDepositosNulos]=DepositosVacios(Val
or1,MejorOpcionRuta,Iteracion,MejorOpcionDepositos)
    DepositosNulos;
    PosDepositosNulos;
    CantDepositosNulos;
%%Cambio de lugar de un solo cliente en un deposito
clearvars Ronda40 Ronda41 Valor37
Valor36=length(MejorOpcionRuta(Iteracion,:,1));
for Ronda40=1:Valor1
    Valor37(Ronda40)=0; %Arranca en cero para que la variable exista
    for Ronda41=1:Valor36
        if MejorOpcionRuta(Iteracion,Ronda41,Ronda40)>0
            Valor37(Ronda40)=Valor37(Ronda40)+1; %Contador de clientes
asignados a cada deposito
        end
    end
end
Valor35=ceil(rand*Valor1); %Seleccion del deposito a alterar
while (Valor37(Valor35)==1)||(Valor37(Valor35)==0) %En caso que se
seleccione un deposito con un solo cliente o con ninguno
    Valor35=ceil(rand*Valor1);
end
Valor35b=Valor35;
if Valor37(Valor35)==2 %Si solo tiene dos clientes, es un simple
intercambio de posiciones
Valor38=MejorOpcionRuta(Iteracion,1,Valor35);
Valor39=MejorOpcionRuta(Iteracion,2,Valor35);
MejorOpcionRuta(Iteracion,1,Valor35)=Valor39;
```

```
MejorOpcionRuta(Iteracion,2,Valor35)=Valor38;
else
    Valor40=ceil(rand*(Valor37(Valor35))); %Cliente a mover
    clearvars Contador
    Contador=0;
    while (MejorOpcionRuta(Iteracion,Valor40,Valor35)==0)
        if Contador<20
        Valor40=ceil(rand*(Valor37(Valor35)));
        elseif Contador==20
        Valor40=1;
        else
            Valor35=ceil(rand*Valor1);
            clearvars ViabilidadDeposito1
            [ViabilidadDeposito1]=find(PosDepositosNulos==Valor35);
        while
(Valor35b==Valor35)||(sum(ViabilidadDeposito1)>0)||(Valor37(Valor35)<3)
                Valor35=ceil(rand*Valor1);
                Valor35b=Valor35;
                clearvars ViabilidadDeposito1
                [ViabilidadDeposito1]=find(PosDepositosNulos==Valor35);
        end
        end
    end
    Valor41=ceil(rand*(Valor37(Valor35))); %Nueva posicion
    while Valor40==Valor41
        Valor41=ceil(rand*(Valor37(Valor35)));
    end
    clearvars MejorOpcionRuta5 Valor99 Valor100
    for Valor99=1:Valor1
        for Valor100=1:Valor3

MejorOpcionRuta5(Valor100,Valor99)=MejorOpcionRuta(Iteracion,Valor100,Val
or99)%Guardamos la matriz solucion
        end
    end
    clearvars Ronda97 Ronda98
    for Ronda97=1:Valor1
        for Ronda98=1:Valor3
            MejorOpcionRuta(Iteracion,Ronda98,Ronda97)=0; %Creamos una
matriz para ubicar los clientes redistribuidos
        end
    end
    Valor42=Valor40-Valor41;
    for Ronda42=1:Valor1
        for Ronda43=1:Valor36
            if Ronda42==Valor35
                if Valor42>0
                    if Ronda43<Valor41

MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda
42);
                        elseif Ronda43==Valor41

MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Valor40,Ronda
42);
                        elseif (Ronda43>Valor41)&&(Ronda43<=Valor40)
```

```
MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Ronda43-
1,Ronda42);
                    else

MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda
42);
                    end
                elseif Valor42<0
                    if Ronda43<Valor40

MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda
42);
                    elseif Ronda43==Valor40

MejorOpcionRuta(Iteracion,Valor41,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda
42);
                    elseif (Ronda43>Valor40)&&(Ronda43<=Valor41)
                        MejorOpcionRuta(Iteracion,Ronda43-
1,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda42);
                    else

MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda
42);
                    end
                end
            else

MejorOpcionRuta(Iteracion,Ronda43,Ronda42)=MejorOpcionRuta5(Ronda43,Ronda
42);
            end
        end
    end
end

%%Analisis de la viabilidad segun Costos
clearvars Valor14
Valor14=0;
[Valor1,Valor3,MejorOpcionRuta1,MejorOpcionRuta,CostoMejorOpcion]=Analisi
sCostos(MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorO
pcion,Iteracion,Valor14,Valor1,Valor3,Fjj,Clie_Dep,PosDepositosNulos,
Clie_Clie)
clearvars Valor14

clearvars Valor35 Valor37 Valor38 Valor39 Valor40 Valor41 Valor42 Valor99
Valor100
clearvars Ronda40 Ronda42 Ronda43 MejorOpcionRuta5
end
CostoMejorOpcion
MejorOpcionDepositos
MejorOpcionRuta
MejorOpcionRuta1
Valor1
Valor3
('Estructura de Barrio N4')
pause;
```

```
end
```

Descripción: método que aplica 4 estructuras de vecindad que varían la solución generada por el 'Sistema colonia de hormigas', con el objetivo de mejorarla iterativamente, estas estructuras se describen a continuación:

- Estructura N1: selecciona dos depósitos al azar y de cada uno, un cliente de forma arbitraria, para intercambiar sus posiciones de asignación y ruteo, se evalúa la capacidad de los depósitos para atender la demanda de los nuevos clientes asignados, evalúa el costo de esta nueva estructura y reemplaza la solución, en caso de obtener uno inferior.

- Estructura N2: selecciona un cliente al azar y lo traslada a otro depósito seleccionado arbitrariamente, se evalúa la capacidad de los depósitos para atender la demanda de los nuevos clientes asignados, evalúa el costo y reemplaza la solución si la estructura permite disminuir el costo del sistema.

- Estructura N3: selecciona un depósito al azar y de éste, dos clientes, intercambia sus posiciones en la ruta, se evalúa la capacidad de los depósitos para atender la demanda de los nuevos clientes asignados, evalúa el costo y si este resulta menor, se reemplaza la solución.

- Estructura N4: selecciona un cliente arbitrariamente y lo ubica en medio de dos clientes, dentro de la misma ruta, se evalúa la capacidad de los depósitos para atender la demanda de los nuevos clientes asignados, evalúa el costo, reemplazando la solución si éste resulta menor.

- **Entradas**

  MejorOpcionRuta1: matriz temporal que almacena la información de 'MejorOpcionRuta' antes de hacer las variaciones propias de cada estructura de vecindad.
  MejorOpcionRuta: hipermatriz que contiene la mejor ruta (aquella que genera el menor costo), encontrada hasta el momento, para cada depósito seleccionado.
  Valor 3: cantidad máxima de clientes asignados a un depósito (representa el número de filas de 'MejorOpcionRuta').
  Valor 1: cantidad de depósitos seleccionados.
  MejorOpcionDepositos: matriz en la cual cada fila corresponde a los depósitos seleccionados en cada iteración.

CostoMejorOpcion: vector cuyos elementos corresponden al menor costo obtenido en cada iteración.
Di: demanda de cada cliente i.
Rjj: capacidad de cada depósito j.
Fjj: costo asociado a la instalación de cada depósito j.
Clie_dep: matriz de costo de servicio de cada depósito j a cada cliente i.
Iteracion: variable que va desde 1 hasta s1.
Clie_Clie: matriz de costo asociado a la distancia entre clientes (costo de ruteo).
T: número de iteraciones que realiza el ILS y cada una de las estructuras de vecindad.

- **Dentro del proceso:**

Valor5: primer depósito seleccionado al azar para el intercambio de la estructura N1.
Valor6: segundo depósito seleccionado al azar para el intercambio de la estructura N1.
ViabilidadDeposito1: almacena la no viabilidad de 'Valor5' correspondiente a la ausencia de clientes asignados para realizar la estructura N1.
ViabilidadDeposito2: almacena la no viabilidad de 'Valor6' correspondiente a la ausencia de clientes asignados para realizar la estructura N1.
Valor7: posición del cliente correspondiente a 'Valor5' para hacer el intercambio de la estructura N1.
Valor8: posición del cliente correspondiente a 'Valor6' para hacer el intercambio de la estructura N1.
Valor9: primer cliente a intercambiar para la estructura N1.
Valor10: segundo cliente a intercambiar para la estructura N1.
DemandaMejor: Demanda de cada uno de los clientes asignados a un depósito.
DemandaMejorOpcion: demanda total de los clientes asignados a un depósito.
CapacidadMejorOpcion: capacidad del depósito seleccionado.
Valor16: depósito origen para la estructura N2.
Valor17: depósito destino para la estructura N2.
Valor18: posición del cliente seleccionado al azar del depósito fuente (Valor16) para la estructura N2.
Valor19: valor de la posición de 'Valor18'.

Valor20: corrección de ‘Valor3’ para la inclusión del nuevo cliente para la estructura N2.
Valor75: posición del cliente insertado en el depósito destino para la estructura N2.
DemandaMejor2: demanda de cada uno de los clientes asignados a un depósito en N2.
DemandaMejorOpcion2: Demanda de cada uno de los clientes asignados a un depósito en N2.
Valor28: selecciona el depósito en el cual se intercambian de lugar en la ruta 2 clientes.
Valor31: variable temporal donde se almacena el primer cliente en caso de que solo tenga dos clientes asignados.
Valor32: variable temporal donde se almacena el segundo cliente en caso que solo tenga dos clientes asignados.
Valor34: variable temporal donde se almacena el deposito seleccionado en caso que tenga más que dos clientes asignados.
Valor29: primer cliente seleccionado para el intercambio, en la estructura N3.
Valor30: segundo cliente seleccionado para el intercambio en la estructura de N3.
Valor35: depósito seleccionado al azar para el movimiento de inserción de la estructura N4.
Valor40: cliente asignado al depósito ‘Valor35’ que se inserta en medio de otros dos clientes en la misma ruta, para la estructura N4.
Valor41: nueva posición del cliente que se inserta, en la estructura N4.

- **Salidas**
  MejorOpcionRuta1: vector temporal que almacena la información de ‘MejorOpcionRuta’ antes de hacer las variaciones propias de cada estructura de vecindad.
  MejorOpcionRuta: vector que contiene la mejor ruta (que genera el menor costo) por cada depósito seleccionado encontrada hasta el momento.
  MejorOpcionDepositos: vector que contiene en orden cada uno de los depósitos seleccionados.
  CostoMejorOpcion: el menor costo en todo el sistema, por cada iteración.
  Valor1: cantidad de depósitos seleccionados.
  Valor3: tamaño del vector ‘MejorOpcionRuta’.

Función: 'AnalisisCostos'

```
function
[Valor1,Valor3,MejorOpcionRuta1,MejorOpcionRuta,CostoMejorOpcion]=Analisi
sCostos(MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorO
pcion,Iteracion,Valor14,Valor1,Valor3,Fjj,Clie_Dep,PosDepositosNulos,Clie
_Clie)
global MejorOpcionRuta1 MejorOpcionRuta MejorOpcionDepositos
CostoMejorOpcion Iteracion Valor14 Valor1 Valor3 Fjj Clie_Dep
PosDepositosNulos Clie_Clie

clearvars Costo8
 if sum(Valor14)>=1
      disp('-El analisis de la estructura de barrio N1 no es viable
porque no cumple el criterio de capacidad-')
 else
     clearvars Costo7
     for Ronda25=1:Valor1
            clearvars ViabilidadDeposito3
            [ViabilidadDeposito3]=find(PosDepositosNulos==Ronda25);
            if (sum(ViabilidadDeposito3)>0)
                Costo7(Ronda25)=0;
            else
            clearvars MejorOpcionRuta2
            Valor15=1;
            for Ronda26=1:Valor3
                if MejorOpcionRuta(Iteracion,Ronda26,Ronda25)>0

MejorOpcionRuta2(Valor15)=MejorOpcionRuta(Iteracion,Ronda26,Ronda25);
                    Valor15=Valor15+1;
                end
            end
            clearvars Costo6
            if Valor15==2 %Si Valor15 es igual a 2 significa que solo hay
un cliente por ese deposito
                Costo6=0;

Costo7(Ronda25)=(sum(Costo6))+(Clie_Dep(MejorOpcionRuta2(1),MejorOpcionDe
positos(Iteracion,Ronda25)))+(Clie_Dep(MejorOpcionRuta2(Valor15-
1),MejorOpcionDepositos(Iteracion,Ronda25)))+(Fjj(MejorOpcionDepositos(It
eracion,Ronda25)));
            elseif Valor15<=1 %Si Valor15 es igual a 1 significa que no
hay un cliente en este deposito
                Costo7(Ronda25)=0;
            else
            for Ronda27=1:(Valor15-2)

Costo6(Ronda27)=Clie_Clie((MejorOpcionRuta2(Ronda27)),(MejorOpcionRuta2(R
onda27+1)));
            end

Costo7(Ronda25)=(sum(Costo6))+(Clie_Dep(MejorOpcionRuta2(1),MejorOpcionDe
positos(Iteracion,Ronda25)))+(Clie_Dep(MejorOpcionRuta2(Valor15-
```

```
1),MejorOpcionDepositos(Iteracion,Ronda25)))+(Fjj(MejorOpcionDepositos(It
eracion,Ronda25)));
            end
            end
     end
     Costo8=sum(Costo7);
         if Costo8>CostoMejorOpcion(Iteracion)
            clearvars Ronda97 Ronda98
            for Ronda97=1:Valor1
                for Ronda98=1:Valor3
                    MejorOpcionRuta(Iteracion,Ronda98,Ronda97)=0;
                end
            end
               clearvars Valor3a Valor1a Ronda95 Ronda96
               Valor3a=length(MejorOpcionRuta1(:,1));
               Valor1a=length(MejorOpcionRuta1(1,:));
               for Ronda95=1:Valor1a
                   for Ronda96=1:Valor3a

MejorOpcionRuta(Iteracion,Ronda96,Ronda95)=MejorOpcionRuta1(Ronda96,Ronda
95);
                   end
               end
             disp('-El analisis de estructura de barrio N1 no optimiza la
respuesta-')
         else
             CostoMejorOpcion(Iteracion)=Costo8;
         end
  end
    clearvars MejorOpcionRuta1 Valor1 Valor3
    Valor1=length(MejorOpcionRuta(Iteracion,1,:));
    Valor3=length(MejorOpcionRuta(Iteracion,:,1));
    clearvars Ronda97 Ronda98
    for Ronda97=1:Valor1
        for Ronda98=1:Valor3

MejorOpcionRuta1(Ronda98,Ronda97)=MejorOpcionRuta(Iteracion,Ronda98,Ronda
97);
        end
    end

 clearvars Valor14 Valor15 Ronda25 Ronda26 Ronda27 Ronda95 Ronda96
Ronda97 Ronda98
 clearvars Valor1a Valor3a Costo6 Costo7 Costo8 ViabilidadDeposito3
PosDepositosNulos MejorOpcionRuta2
end
```

Descripción: analiza el costo obtenido en los cambios realizados por las estructuras de vecindad y lo compara con el costo que trae del proceso anterior, si este se logra disminuir con las modificaciones propias de la estructura, se reemplaza la solución.

- **Entradas**

MejorOpcionRuta1: vector temporal que almacena la información de 'MejorOpcionRuta' antes de hacer las variaciones propias de cada estructura de vecindad.
MejorOpcionRuta: vector que contiene la mejor ruta (aquella que genera el menor costo), encontrada hasta el momento, para cada depósito seleccionado.
MejorOpcionDepositos: vector que contiene en orden cada uno de los depósitos seleccionados.
CostoMejorOpcion: el menor costo en todo el sistema, por cada iteración.
Valor14: vector que indica la viabilidad del proceso de relación capacidad – demanda de un proceso de estructura de barrio.
Valor 1: Cantidad de depósitos seleccionados.
Valor 3: tamaño del vector 'MejorOpcionRuta'.
Fjj: Costo asociado a la instalación de cada depósito j.
Clie_dep: Matriz de costo de servicio de cada depósito j a cada cliente i.
PosDepositosNulos: posición dentro de 'MejorOpcionDepositos' de los depósitos que no poseen clientes asignados.
Clie_Clie: Matriz de costo asociado a la distancia entre clientes (costo de ruteo).

- **Dentro del proceso**
  ViabilidadDeposito3: vector donde se almacenan los depósitos que no tienen clientes asignados.
  MejorOpcionRuta2: vector temporal donde se almacena (sin ceros) los clientes asignados a un deposito respectivo.
  Costo6: vector que indica los costos del trayecto Cliente – Cliente por cada depósito de 'MejorOpcionDepositos'
  Costo7: vector que indica el costo de cada depósito y clientes por cada depósito de 'MejorOpcionDepositos'.
  Costo8: Costo final después del cambio de la estructura determinada.

- **Salidas**
  Valor1: cantidad de depósitos seleccionados.
  Valor3: tamaño del vector 'MejorOpcionRuta'.
  MejorOpcionRuta1: vector temporal que almacena la información de 'MejorOpcionRuta' antes de hacer las variaciones propias de cada estructura de vecindad.

MejorOpcionRuta: vector que contiene la mejor ruta (aquella que genera el menor costo), encontrada hasta el momento, para cada depósito seleccionado.
CostoMejorOpcion: el menor costo en todo el sistema, por cada iteración.

Función: 'DepositosVacios'

```
function
[PosDepositosNulos,DepositosNulos,CantDepositosNulos]=DepositosVacios(Val
or1,MejorOpcionRuta,Iteracion,MejorOpcionDepositos)
global PosDepositosNulos DepositosNulos CantDepositosNulos Valor1
MejorOpcionRuta Iteracion MejorOpcionDepositos

%Analisis de la existencia de depositos sin clientes
clearvars Ronda80 Ronda82 Valor80 Valor3b Ronda81 PosDepositosNulos
CantDepositosNulos DepositosNulos
DepositosNulos=[];
CantDepositosNulos=0;
PosDepositosNulos=[];
Valor3b=length(MejorOpcionRuta(1,:,1));
Valor80=zeros(1,Valor1);
for Ronda80=1:Valor1
    for Ronda82=1:Valor3b
        if MejorOpcionRuta(Iteracion,Ronda82,Ronda80)>0

Valor80(Ronda80)=Valor80(Ronda80)+MejorOpcionRuta(Iteracion,Ronda82,Ronda
80);
        end
    end
end
    [PosDepositosNulos]=find(Valor80==0);
    CantDepositosNulos=length(PosDepositosNulos);
    if CantDepositosNulos>0
        for Ronda81=1:CantDepositosNulos

DepositosNulos(Ronda81)=MejorOpcionDepositos(PosDepositosNulos(Ronda81));
        end
    end
clearvars Ronda80 Ronda82 Valor80 Valor3b Ronda81
end
```

Descripción: determina la cantidad, posición y depósitos que no tienen clientes asignados

- **Entradas**
  Valor1: Cantidad de depósitos seleccionados.
  MejorOpcionRuta: vector que contiene la mejor ruta (aquella que genera el menor costo), encontrada hasta el momento, para cada depósito seleccionado.

MejorOpcionDepositos: vector que contiene en orden cada uno de los depósitos seleccionados.

- **Dentro del proceso**
  Valor3b: cantidad de filas de 'MejorOpcionRuta'.
  Valor80: vector cuyos elementos contienen la sumatoria de los depósitos de determinada columna. Si 'Valor80'>0 significa que el depósito tiene clientes asignados.
- **Salidas**
  PosDepositosNulos: vector que almacena la posición de los depósitos nulos dentro del vector 'MejorOpcionDepositos'
  DepositosNulos: vector que almacena los depósitos de 'MejorOpcionDepositos' que no tiene clientes asignados.
  CantDepositosNulos: tamaño de 'DepositosNulos'.

CÓDIGO GENERAL ACO+ILS

```
%Parametros Iniciales
s1=input('Iteraciones totales del ACS: ');
b1=input('Hormigas totales del ACS: ');
r=input('Numero relacionado para determinar P_hs: ');
n=input('Numero de clientes: ');
m=input('Depositos candidatos: ');
T=input('Iteraciones para ILS: ');

%Parámetros que determinan la influencia relativa de la información
heurística vs la información de feromona
alpha=input('Parámetro "Alfa" para la selección: ');
beta=input('Parámetro "Beta" para la asignación: ');
gamma=input('Parámetro "Gama" para el VRP: ');
%Parámetros de evaporización de la feromona, en el rango de [0-1]. Los
tres parámetros siguientes pueden tomar el mismo valor.
rho1=input('Parametro "Rho1" de evaporizacion para la seleccion: ');
rho2=input('Parametro "Rho2" de evaporizacion para la asignacion: ');
rho3=input('Parametro "Rho3" de evaporizacion para la asignacion: ');

% Parámetros que determinan la importancia relativa de explotación vs
exploración.
q01=input('Parametro para la seleccion: '); %Este valor es necesario para
la comparación en cada colonia
q02=input('Parametro para asignacion: ');
q03=input('Parametro para VRP: ');

%Parametros adicionales para el VRP
s2=input('Iteraciones para el VRP: ');
b2=input('Hormigas para el VRP: ');
```

```
%Extraccion de datos de Documento Excel
dii=xlsread('DATOS DEL PROBLEMA','demanda','A1:A102');
Rjj=xlsread('DATOS DEL PROBLEMA','Capacidad depositos','A1:A20');
Fjj=xlsread('DATOS DEL PROBLEMA','Costo deposito','A1:A20');
Cil=xlsread('DATOS DEL PROBLEMA','Matriz costo cliente','B2:DQ101');

for Cliente=1:n
    di(Cliente)=dii(Cliente);
end

for Dep=1:m
    Rj(Dep)=Rjj(Dep);
end

for Cos=1:m
    Fj(Cos)=Fjj(Cos);
end

for ii=1:n

    for jj=1:n
        Clie_Clie(ii,jj)=Cil(ii,jj);
    end

    for(kk=1:20)
        Clie_Dep(ii,kk)=Cil(ii,(100+kk));
    end
end
%%
%Calculo de matrices iniciales de feromona
Tao_j=1./Fj;
Epsilon_ik=0.001*ones(n,m);
dseta=0.01*ones(n,n);

global Iteracion Hormiga Iteracion2 Hormiga2
for (Iteracion=1:s1)
    for (Hormiga=1:b1)
       %%
       %Calculo de la matriz P_hs
        U=ceil(rand*r);

P_hs(Iteracion,Hormiga)=(ceil((sum(di))/((sum(Rj))/(length(Rj)))))+(U);%M
atriz que determina los depositos a analizar
        if P_hs(Iteracion,Hormiga)>m %si P_hs es mayor que m, P_hs=m
           P_hs(Iteracion,Hormiga)=m;
        end
        clearvars U
%%
%Seleccion de los depositos
    Nu_j=Rj./Fj;
    for (Ronda1=1:P_hs(Iteracion,Hormiga))
        q1=rand;
    if (q1<=q01)

[ArgMax,j(Iteracion,Hormiga,Ronda1)]=max((Nu_j).*((Tao_j).^alpha)) %Lo
importante no es el Arg Maximo sino su posicion
    else
```

```
        [VectorProbabilidad,Correccion]=ProbabilidadSeleccion
(Tao_j,Nu_j,alpha,m);

j(Iteracion,Hormiga,Ronda1)=VectorProbabilidad(ceil(rand*Correccion))
%Halla una posicion al azar dentro del VectorProbabilidad y lee su valor
    end
    clearvars q1 ArgMax VectorProbabilidad Correccion

    Nu_j(j(Iteracion,Hormiga,Ronda1))=0; %Con esto se quita la
posibilidad que este deposito sea nuevamente elegido
    Capacidad_ik(Ronda1)=Rj(j(Iteracion,Hormiga,Ronda1))%Necesaria para
la asignacion de los clientes
    end
    %%
    %Comprobacion de la capacidad de los Phs depositos
    CapacidadTotal=sum(Capacidad_ik);
    DemandaTotal=sum(di);
     while (DemandaTotal>CapacidadTotal)
            P_hs(Iteracion,Hormiga)=P_hs(Iteracion,Hormiga)+1

            if (P_hs(Iteracion,Hormiga)>=m)
                NuevoDeposito2=ceil(P_hs(Iteracion,Hormiga)*rand)
            else
                NuevoDeposito2=ceil(m*rand);
            end

[DepositoRepetido]=find(j(Iteracion,Hormiga,:)==NuevoDeposito2)

        while (DepositoRepetido>=1)
            if (P_hs(Iteracion,Hormiga)>=m)
                NuevoDeposito2=ceil(P_hs(Iteracion,Hormiga)*rand)
            else
                NuevoDeposito2=ceil(m*rand)
            end

[DepositoRepetido]=find(j(Iteracion,Hormiga,:)==NuevoDeposito2)
        end

        NuevoDeposito=NuevoDeposito2;
        Rj(P_hs(Iteracion,Hormiga))=Rjj(NuevoDeposito)
        Fj(P_hs(Iteracion,Hormiga))=Fjj(NuevoDeposito)
        for (xx=1:n)

Clie_Dep(xx,P_hs(Iteracion,Hormiga))=Cil(xx,(101+NuevoDeposito)) %lee la
relación cliente-depósito del hallado nuevamente
        end
        Tao_j(P_hs(Iteracion,Hormiga))=1./Fj(P_hs(Iteracion,Hormiga));
        Epsilon_ik(:,P_hs(Iteracion,Hormiga))=0.01*ones(n,1);
        Capacidad_ik= [Capacidad_ik,Rj(P_hs(Iteracion,Hormiga))];
        CapacidadTotal=sum(Capacidad_ik);
        j(Iteracion,Hormiga,P_hs(Iteracion,Hormiga))=NuevoDeposito
        Fj(P_hs(Iteracion,Hormiga))=Fjj(NuevoDeposito)
     end

     clearvars NuevoDeposito xx NuevoDeposito2 DepositoRepetido
```

```
 %Actualizacion local de la feromona Tao
 for (Ronda8=1:P_hs(Iteracion,Hormiga))
    Tao_j(j(Iteracion,Hormiga,Ronda8))=((1-
rho1).*(Tao_j(j(Iteracion,Hormiga,Ronda8))))+(rho1.*(1./(Fj(j(Iteracion,H
ormiga,Ronda8)))));
 end
 %%
 %Asigancion de depositos a cada cliente (Cada cliente se asigna a un
 %depósito k seleccionado, teniendo en cuenta su capacidad)
 clearvars Costo2 Seleccion
     for (Cliente=1:n)
      clearvars Epsilon Costo
        for (Ronda2=1:P_hs(Iteracion,Hormiga))

Epsilon(Ronda2)=Epsilon_ik(Cliente,(j(Iteracion,Hormiga,Ronda2))) % Valor
inicial de feromona- (relación clientes-depósitos) Crea un vector con los
valores del analisis actual
            Costo(Ronda2)=
Clie_Dep(Cliente,(j(Iteracion,Hormiga,Ronda2))) %Crea un vector con los
costos del analisis actual- Matriz Cil
        end
        clearvars Costo3 Psi
        if (Cliente==1) %Halla el valor minimo de ruta (deposito o
cliente)- Es decir, para el cliente=1 el valor mímino se mide unicamente
con su relación entre depósitos
            Costo3=Costo %Porque en el cliente 1, no hay otras opciones
(cliente - deposito) mas economicas
        else
            Costo2=ones(n,(P_hs(Iteracion,Hormiga))) %Crea la matriz de
trabajo- matriz de unos
            for (Ronda3=1:P_hs(Iteracion,Hormiga))
                Costo2(:,Ronda3)=Costo(Ronda3) %Reemplaza los valores de
toda la columna con los valores
            end

            for (Ronda4=2:Cliente)
                Costo2(Ronda4,(k(Iteracion,Hormiga,(Ronda4-
1))))=Clie_Clie((Ronda4-1),Cliente) %Actualiza los valores de la matriz
con los de los depositos previamente ubicados
            end
            Costo3=min(Costo2) %Halla el minimo de cada columna = minimo
de cada grupo- de cada columna se elige el mínimo costo (una columna
tiene los clientes asignados por depósito)
        end

        q2=rand;
        if (q2<=q02)
            clearvars Ronda6
            for (Ronda6=1:P_hs(Iteracion,Hormiga))
                if (di(Cliente))> Capacidad_ik(Ronda6)
                    Costo3(:,Ronda6)=10^15 %Si la capacidad no soporta la
demanda pone un costo muy alto a toda la columna, sacandola del analisis
de costo minimo
                end
            end
            format long
            Psi=1./Costo3
```

```
[ArgMax,k(Iteracion,Hormiga,Cliente)]=max(Epsilon.*(Psi.^beta))
            while di(Cliente)>Capacidad_ik(k(Iteracion,Hormiga,Cliente))

k(Iteracion,Hormiga,Cliente)=ceil(rand*P_hs(Iteracion,Hormiga));
%Comprobacion de la capacidad
            end

Capacidad_ik(k(Iteracion,Hormiga,Cliente))=Capacidad_ik(k(Iteracion,Hormi
ga,Cliente))-(di(Cliente)); %Actualiza la capacidad disponible del
deposito
        else
            Psi=1./Costo3
            clearvars Ronda6
            for (Ronda6=1:P_hs(Iteracion,Hormiga))
                if (di(Cliente))> Capacidad_ik(Ronda6)
                    Psi(Ronda6)=0;%si la capacidad no soporta la demanda
anula la probabilidad de ser seleccionado
                end
            end
            clearvars P_hik VectorProbabilidad2

P_hik=round(100*((Epsilon.*(Psi.^beta))/(sum(Epsilon.*(Psi.^beta)))))
%probabilidad de selección
            Correccion2=sum(P_hik)

           Origen(1)=1;
           for (jj=2:P_hs(Iteracion,Hormiga))
               Origen(jj)=Origen(jj-1)+P_hik(jj-1); %Se usa para dar la
cantidad de veces que se escribe el depósito k en el vector probabilidad
           end

          for (i=1:P_hs(Iteracion,Hormiga))
          if (i==1)
             for (jjj=1:P_hik(1))
                 VectorProbabilidad2(jjj)=1; %crea el vector donde se
encontrarán los k depósitos a seleccionar, de allí el cliente es asignado
(ruleta)

             end
          else
             for (jjj=Origen(i):Correccion2)
                  VectorProbabilidad2(jjj)=i;
             end
          end
          end


k(Iteracion,Hormiga,Cliente)=VectorProbabilidad2(ceil(rand*Correccion2));
%k es elegido en la ruleta cuando q2 es mayor que q02
           while di(Cliente)>Capacidad_ik(k(Iteracion,Hormiga,Cliente))

k(Iteracion,Hormiga,Cliente)=ceil(rand*P_hs(Iteracion,Hormiga));
%Comprobacion de la capacidad
            end
```

```
Capacidad_ik(k(Iteracion,Hormiga,Cliente))=Capacidad_ik(k(Iteracion,Hormi
ga,Cliente))-(di(Cliente));%se actualiza la capacidad disponible
        end

        Seleccion(Cliente,k(Iteracion,Hormiga,Cliente))=Cliente %Crea una
matriz donde en cada columna representa el deposito determinado y se
encuentran los clientes ubicados en ellas
        CorreccionDepositosUtilizados=length(Seleccion(1,:)) %Es posible
que no se usen la mayoria de los depositos
        %Nota: El valor de K, representa la posicion del deposito en el
vector j

        %Actualizacion local de la feromona Epsilon

Epsilon_ik(Cliente,j(Iteracion,Hormiga,k(Iteracion,Hormiga,Cliente)))=((1
-
rho2).*(Epsilon_ik(Cliente,j(Iteracion,Hormiga,k(Iteracion,Hormiga,Client
e)))))+(rho2*0.001);
     end
     clearvars Origen VectorProbabilidad2 jj jjj i
    %%
    %Ruteo de clientes por deposito

     for Columna=1:CorreccionDepositosUtilizados
     clearvars cliXdep Cant_Clie Posicion Iteracion2 Hormiga2 Posicion
     Posicion=1;
     DepositoHabitado(Iteracion,Hormiga,Columna)=0;
     for(Fila=1:n)
         if ((Seleccion(Fila,Columna))>0)
             cliXdep(Posicion)=Seleccion(Fila,Columna); %Crea un vector
con los clientes de un deposito determinado
             Posicion=Posicion+1;

DepositoHabitado(Iteracion,Hormiga,Columna)=j(Iteracion,Hormiga,Columna);
         end
     end

     if (DepositoHabitado(Iteracion,Hormiga,Columna)>0)
         clearvars Cant_Clie
         Cant_Clie=length(cliXdep);

     for (Iteracion2=1:s2)
         for (Hormiga2=1:b2)
           clearvars FirstPosicion
           FirstPosicion=ceil(Cant_Clie*rand) %Genero el primer destino
de forma aleatoria
           Destino(1,Columna)=cliXdep(FirstPosicion)

          clearvars fi_iv Ronda7 dseta_iv Ronda12 Ronda13
          for (Ronda7=1:Cant_Clie)

fi_iv(Ronda7)=(Clie_Dep((Destino(1,Columna)),(j(Iteracion,Hormiga,Columna
))))+(Clie_Dep((cliXdep(Ronda7)),(j(Iteracion,Hormiga,Columna))))-
(Clie_Clie((Destino(1,Columna)),(cliXdep(Ronda7))))
```

```
                dseta_iv(Ronda7)=dseta(Destino(1,Columna),cliXdep(Ronda7))
%Feromona
            end

            Ruta_Seleccion=dseta_iv.*(fi_iv.^gamma)
            Ruta_Seleccion(FirstPosicion)=-1*(10^20) %Asi se evita que el
segundo cliente a visitar sea el mismo primero
            clearvars FirstPosicion

             for Ruta=1:(Cant_Clie-1)
                 q3=rand;
                 if (q3<=q03)
                     clearvars ArgMax PosArgMax
                     [ArgMax,PosArgMax]=max(Ruta_Seleccion)
                     Destino(Ruta+1,Columna)=cliXdep(PosArgMax)
                     %
                     clearvars fi_iv
                     for Ronda12=1:Cant_Clie

fi_iv(Ronda12)=(Clie_Dep((Destino(Ruta+1,Columna)),(j(Iteracion,Hormiga,C
olumna))))+(Clie_Dep((cliXdep(Ronda12)),(j(Iteracion,Hormiga,Columna))))-
(Clie_Clie((Destino(Ruta+1,Columna)),(cliXdep(Ronda12))))
                     end
                     Ruta_Seleccion=dseta_iv.*(fi_iv.^gamma)
                     %
                     for Ronda13=1:(Ruta+1)
                         clearvars PosClie
                         [PosClie]=find(cliXdep==Destino(Ronda13,Columna))
                         Ruta_Seleccion(PosClie)=-1*(10^20) %Hace fi=muy
pequeño para eliminar la probabilidad de dirigirse a un cliente ya
visitado
                     end
                 else


                     for Ronda13=1:(Ruta)
                         clearvars PosClie
                         [PosClie]=find(cliXdep==Destino(Ronda13,Columna));
                         fi_iv(PosClie)=0; %para cumplir con la condición
que un cliente es visitado una sola vez
                     end
                     %
                     if (sum(fi_iv)==0)
                         P_hiv=zeros(1,Cant_Clie);
                         Correccion3=0;
                     else

P_hiv=abs(real(round(100.*((dseta_iv.*(fi_iv.^gamma))./(sum(dseta_iv.*(fi
_iv.^gamma))))))));
                     Correccion3=sum(P_hiv);

                      Origen3(1)=1;
                      for (jj=2:Cant_Clie)
                          Origen3(jj)=Origen3(jj-1)+P_hiv(jj-1); %Creacion
del vector que indica en que posicion cambia la posibilidad de seleccion
                      end
```

```
                    for (ii=1:Cant_Clie)
                        if (ii==1)
                            for (jjj=1:P_hiv(1))

VectorProbabilidad3(jjj)=cliXdep(1);%vector probabilidad para seleccionar
el proximo cliente a visitar
                            end
                        else
                            for (jjj=Origen3(ii):Correccion3)
                            VectorProbabilidad3(jjj)=cliXdep(ii);
                            end
                        end
                    end

Destino(Ruta+1,Columna)=VectorProbabilidad3(ceil(rand*Correccion3))

                    clearvars fi_iv %borra la variable de información
heurística
                    for Ronda12=1:Cant_Clie

fi_iv(Ronda12)=(Clie_Dep((Destino(Ruta+1,Columna)),(j(Iteracion,Hormiga,C
olumna))))+(Clie_Dep((cliXdep(Ronda12)),(j(Iteracion,Hormiga,Columna))))-
(Clie_Clie((Destino(Ruta+1,Columna)),(cliXdep(Ronda12))));
                    end
                    Ruta_Seleccion=dseta_iv.*(fi_iv.^gamma)
                    for Ronda13=1:(Ruta+1)
                        [PosClie]=find(cliXdep==Destino(Ronda13,Columna));
                        Ruta_Seleccion(PosClie)=-1*(10^20) %para cumplir
con la condición que un cliente es visitado una sola vez
                    end
                    end

                  end

                clearvars VectorProbabilidad3 Origen3 ArgMax PosArgMax q3
Correccion3 P_hiv

                %Actualizacion local de la feromona dseta
                dseta(Destino(Ruta,Columna),Destino(Ruta+1,Columna))=((1-
rho3)*(dseta(Destino(Ruta,Columna),Destino(Ruta+1,Columna))))+(rho3*dseta
_0);%se 0.01=rho3 (igual para todos)
                end


            %Seleccion de la mejor solucion y peor solucion local
            clearvars ClienteDestino
            Posicion2=1;
            for (Ronda9=1:Cant_Clie)
                if (Destino(Ronda9,Columna)>0)
                    ClienteDestino(Posicion2)=Destino(Ronda9,Columna)
%Crea un vector con los clientes asignados a un deposito
                    Posicion2=Posicion2+1;
                end
            end
               clearvars Costo4
```

```
            for (Ronda10=1:(Cant_Clie-1))

Costo4(Ronda10)=Clie_Clie((ClienteDestino(Ronda10)),(ClienteDestino(Ronda
10+1)));
                Costo4
            end

            if Cant_Clie==1
            Costo4=0
            end

            Costo4

Costo5(Columna)=(sum(Costo4))+(Clie_Dep(ClienteDestino(1),j(Iteracion,Hor
miga,Columna)))+(Clie_Dep(ClienteDestino(Posicion2-
1),j(Iteracion,Hormiga,Columna)))+(Fj(j(Iteracion,Hormiga,Columna)));
%Costo de la ruta

            %Determinacion de la ruta mas economica por cada deposito
            %localmente
            if (Iteracion2==1)&&(Hormiga2==1)
                CostoMinDep(Columna)=Costo5(Columna);%Costo minimo por
cada bucle interno
                CostoMinDep2(Columna)=Costo5(Columna)-
(Fj(j(Iteracion,Hormiga,Columna)));
                for Ronda11=1:Cant_Clie

RutaFinalDep(Iteracion,Hormiga,Columna,Ronda11)=ClienteDestino(Ronda11);
%Ruta de costo minimo por cada bucle interno
                end
                %clearvars ClienteDestino  %estaba como comentario
            else
                if (CostoMinDep(Columna)>Costo5(Columna))
                    CostoMinDep(Columna)=Costo5(Columna)%Costo minimo por
cada bucle interno
                    CostoMinDep2(Columna)=Costo5(Columna)-
(Fj(j(Iteracion,Hormiga,Columna)));
                    for Ronda11=1:Cant_Clie

RutaFinalDep(Iteracion,Hormiga,Columna,Ronda11)=ClienteDestino(Ronda11);
%Ruta de costo minimo por cada bucle interno
                    end
                    %clearvars ClienteDestino
                end
            end

            %Determinacion de la ruta mas costosa por cada deposito
            %localmente
            if (Iteracion2==1)&&(Hormiga2==1)
                CostoMaxDep(Columna)=Costo5(Columna)%Costo maximo por cada
bucle interno
                CostoMaxDep2(Columna)=Costo5(Columna)-
(Fj(j(Iteracion,Hormiga,Columna)));
            else
                if (CostoMaxDep(Columna)<Costo5(Columna))
```

```
                    CostoMaxDep(Columna)=Costo5(Columna)%Costo maximo por
cada bucle interno
                    CostoMaxDep2(Columna)=Costo5(Columna)-
(Fj(j(Iteracion,Hormiga,Columna)));
                end
            end

             %Componentes para actualizacion global de la feromona dseta
             if (Iteracion2==1)&&(Hormiga2==1)
                 L_s=CostoMinDep2(Columna);
                 L_w=CostoMaxDep2(Columna)
             else
                 if L_s>CostoMinDep2(Columna)
                    L_s=CostoMinDep2(Columna);
                 end
                  if L_w<CostoMaxDep2(Columna)
                     L_w=CostoMaxDep2(Columna);
                  end
             end

          end

    %Actualizacion global de la feromona dseta
    if (Iteracion2==1)
        L_b=CostoMinDep2(Columna);
    else
        if L_b>CostoMinDep2(Columna)
           L_b=CostoMinDep2(Columna);
        end
    end
    Delta_dseta=((L_w-L_b)+(L_w-L_s))/L_w;
       Cant_Clie2=length(ClienteDestino);
       for Ronda14=1:Cant_Clie2
           for Ronda15=1:Cant_Clie2
               dseta(ClienteDestino(Ronda14),ClienteDestino(Ronda15))=((1-
rho3)*(dseta(ClienteDestino(Ronda14),ClienteDestino(Ronda15))))+(rho3*Del
ta_dseta);
           end
       end

      end


      else
          CostoMaxDep(Columna)=0;
          CostoMinDep(Columna)=0;
      end
      end

 %Determinacion de la solucion mas economica
 if (Iteracion==1)&(Hormiga==1)
     CostoMin=sum(CostoMinDep)
     IteracionMin=Iteracion
     HormigaMin=Hormiga
 else
     if (CostoMin>(sum(CostoMinDep)))
        CostoMin=sum(CostoMinDep)
```

```
        IteracionMin=Iteracion
        HormigaMin=Hormiga
    end
end

%Determinacion de la solucion mas costosa
if (Iteracion==1)&&(Hormiga==1)
    CostoMax=sum(CostoMaxDep)
else
    if(CostoMax<(sum(CostoMaxDep)))
        CostoMax=sum(CostoMaxDep)
    end
end

%Componentes de la Actualizacion Global de la Feromona Tao y Epsilon
Valor26=CorreccionDepositosUtilizados
clearvars Delta_Tao Delta_Epsilon
if (Iteracion==1)&&(Hormiga==1)
    L_w2=CostoMax;
else
    if L_w2<CostoMax
       L_w2=CostoMax;
    end
end

    end

% Actualizacion global de la feromona Tao y Epsilon
Valor26=CorreccionDepositosUtilizados
clearvars Delta_Tao Delta_Epsilon
if (Iteracion==1)
   L_s2=CostoMin;
   L_b2=sum(CostoMinDep);
else
   if L_s2>min(CostoMejorOpcion)
       L_s2=min(CostoMejorOpcion);
       L_b2=sum(CostoMinDep);
   end
end
Delta_Epsilon=((L_w2-L_b2)+(L_w2-L_s2))/(L_w2);

 for Ronda37=1:Valor26
     clearvars nj
     nj=length(RutaFinalDep(Iteracion,Hormiga,Ronda37,:));
     Delta_Tao=((L_w2-L_b2)+(L_w2-L_s2)).*(nj./L_w2);
     Tao_j(j(Iteracion,Hormiga,Ronda37))=((1-
rho1)*Tao_j(j(Iteracion,Hormiga,Ronda37)))+(rho1*Delta_Tao);
 end

 for Ronda38=1:Valor26
     Valor27=length(RutaFinalDep(Iteracion,Hormiga,Ronda38,:));
     for Ronda39=1:Valor27
         if RutaFinalDep(Iteracion,Hormiga,Ronda38,Ronda39)>0

Epsilon_ik(RutaFinalDep(Iteracion,Hormiga,Ronda38,Ronda39),j(Iteracion,Ho
rmiga,Ronda38))=((1-
```

```
rho2)*(Epsilon_ik(RutaFinalDep(Iteracion,Hormiga,Ronda38,Ronda39),j(Itera
cion,Hormiga,Ronda38))))+(rho2*Delta_Epsilon)
         end
     end
 end

%ILS Como optimizador del ACS

%Datos Necesarios:
%%% Vector con Depositos habilitados (Mejor Solucion) por Iteracion
%%% Matriz con Clientes ordenados segun ruta (Mejor Solucion) por
Iteracion
%%% Costo Ruta (Mejor Solucion) por Iteracion

clc;

%Para mayor facilidad los "depositos", "Clientes" y "Ruta" Mencionados a
%partir de este punto, corresponden a la mejor solucion del ACS por cada
%Iteracion

global T Valor1 Valor3 Valor36 MejorOpcionRuta MejorOpcionRuta1
MejorOpcionDepositos CostoMejorOpcion di Rjj Fjj Clie_Dep Clie_Clie

for Ronda20=1:T
disp('Iteracion de las Estructuras de Barrio ILS: ')
%ILS_2=Ronda20;

if Ronda20==1
    CostoMejorOpcion(Iteracion)=CostoMin
    Valor1_1=length(DepositoHabitado(Iteracion,HormigaMin,:));
    Valor1_3=1;
    for Valor1_2=1:Valor1_1
        if DepositoHabitado(Iteracion,HormigaMin,Valor1_2)>0

Valor1_4(Valor1_3)=DepositoHabitado(Iteracion,HormigaMin,Valor1_2);
        Valor1_3=Valor1_3+1;
        end
    end
    Valor1=length(Valor1_4); %Cantidad de depositos Mejor Opcion
    for Ronda17=1:Valor1

MejorOpcionDepositos(Iteracion,Ronda17)=DepositoHabitado(Iteracion,Hormig
aMin,Ronda17) %Depositos considerados la mejor opcion
    end
    for Ronda16=1:Valor1

Valor2(Ronda16)=length(RutaFinalDep(Iteracion,HormigaMin,Ronda16,:));
%Cantidad de clientes Mejor Opcion
    end
    Valor3=max(Valor2); %Clientes maximos asignados a un deposito
    for Ronda18=1:Valor1
        Valor4=1;
        for Ronda19=1:Valor3
            MejorOpcionRuta(Iteracion,Ronda19,Ronda18)=0;
            if RutaFinalDep(Iteracion,HormigaMin,Ronda18,Ronda19)>0
```

```
MejorOpcionRuta(Iteracion,Valor4,Ronda18)=RutaFinalDep(Iteracion,HormigaM
in,Ronda18,Ronda19) %Clientes considerados Mejor Opcion
                Valor4=Valor4+1;
            end
        end
    end
    Valor3
    Valor3_1=length(MejorOpcionRuta(Iteracion,:,1))
    if Valor3_1>Valor3
        Valor3=Valor3_1; %Optimizacion de los los clientes asignados a un
deposito
    end

    clearvars Ronda97 Ronda98
    for Ronda97=1:Valor1
        for Ronda98=1:Valor3

MejorOpcionRuta1(Ronda98,Ronda97)=MejorOpcionRuta(Iteracion,Ronda98,Ronda
97)
        end
    end

else
    Valor1=length(MejorOpcionRuta(Iteracion,1,:));
    Valor3=length(MejorOpcionRuta(Iteracion,:,1));
end
clearvars Ronda16 Ronda17 Ronda18 Ronda19 Valor2 Valor4

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorOpcion,V
alor1,Valor3,Valor36]=ILS(Valor3,Valor1,MejorOpcionRuta1,MejorOpcionRuta,
Iteracion,MejorOpcionDepositos,CostoMejorOpcion,di,Rjj,Fjj,Clie_Dep,Clie_
Clie,T)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


%Perturbacion de la solucion
%Nota: Para mayor efectividad de la solucion se usan dos opciones: la
%primera es abriendo un nuevo deposito y la segunda es trasladando los
%clientes de un deposito a otro.

if Valor1>1
%Analisis de la existencia de depositos sin clientes
[PosDepositosNulos,CantDepositosNulos,DepositosNulos]=DepositosVacios(Val
or1,MejorOpcionRuta,Iteracion,MejorOpcionDepositos)

Valor46=ceil(rand*Valor1); %Deposito fuente
while MejorOpcionDepositos(Iteracion,Valor46)==0
    Valor46=ceil(rand*Valor1);%Deposito Fuente
```

```
end

if (m>Valor1)
    OpcionPerturbacion=rand;
else
    OpcionPerturbacion=1;
end

if OpcionPerturbacion<0.5
    Valor48=ceil(m*rand); %Nuevo deposito "cerrado"

[PosDepositoRepetido2]=find(MejorOpcionDepositos(Iteracion,:)==Valor48);
%Para evitar repetir deposito con lo que estan seleccionados
    DepositoRepetido2=length(PosDepositoRepetido2);
    while DepositoRepetido2>=1
        Valor48=ceil(m*rand);

[PosDepositoRepetido2]=find(MejorOpcionDepositos(Iteracion,:)==Valor48);
        DepositoRepetido2=length(PosDepositoRepetido2);
    end
    %Analisis de la viabilidad segun la relacion de Demanda/Capacidad
    clearvars Valor49 Valor50
    Valor49=Rjj(Valor48); %Capacidad del deposito recien abierto
    Valor36=length(MejorOpcionRuta(Iteracion,:,1));
    for Ronda48=1:Valor36
        Valor50(Ronda48)=0;
        if MejorOpcionRuta(Iteracion,Ronda48,Valor46)>0
        Valor50(Ronda48)=di(MejorOpcionRuta(Iteracion,Ronda48,Valor46));
        end
    end
    clearvars Valor51
    Valor51=sum(Valor50); %Demanda de los clientes asignados al deposito
fuente
    Valor52=1;
    while (Valor52<=20)
        if Valor49<Valor51
            Valor48=ceil(20*rand);

[DepositoRepetido3]=find(MejorOpcionDepositos(Iteracion,:)==Valor48);
            while (length(DepositoRepetido3))>=1
                Valor48=ceil(M*rand);

[DepositoRepetido3]=find(MejorOpcionDepositos(Iteracion,:)==Valor48);
            end
            Valor49=Rjj(Valor48);
        end
        Valor52=Valor52+1;
    end
    if Valor49<Valor51
        disp('-No se encontro un deposito que cubra la demanda especifica
para esta Perturbacion, agregue mas depositos-')
    else
        %Analisis de la viabilidad segun el costo
        clearvars Costo19
        for Ronda49=1:Valor1
            clearvars ViabilidadDeposito1 PosViabilidadDeposito1
            [PosViabilidadDeposito1]=find(PosDepositosNulos==Ronda49);
```

```
            ViabilidadDeposito1=length(PosViabilidadDeposito1);
            if sum(ViabilidadDeposito1)>0
                Costo19(Ronda49)=0;
            else
            Valor53=1;
            for Ronda50=1:Valor36
                if MejorOpcionRuta(Iteracion,Ronda50,Ronda49)>0

MejorOpcionRuta7(Valor53)=MejorOpcionRuta(Iteracion,Ronda50,Ronda49);
                    Valor53=Valor53+1;
                end
            end

            clearvars Costo18
            if Valor53==2 %Si Valor15 es igual a 2 significa que solo hay
un cliente por ese deposito
                Costo18=0;
            if Ronda49==Valor46

Costo19(Ronda49)=sum(Costo18)+(Clie_Dep(MejorOpcionRuta7(1),Valor48))+(Cl
ie_Dep(MejorOpcionRuta7(Valor53-1),Valor48))+(Fjj(Valor48));
            else

Costo19(Ronda49)=sum(Costo18)+(Clie_Dep(MejorOpcionRuta7(1),MejorOpcionDe
positos(Iteracion,Ronda49)))+(Clie_Dep(MejorOpcionRuta7(Valor53-
1),MejorOpcionDepositos(Iteracion,Ronda49)))+(Fjj(MejorOpcionDepositos(It
eracion,Ronda49)));
            end
            elseif Valor53<=1 %Si Valor15 es menor o igual a 1 significa
que no hay un cliente en este deposito
                Costo19(Ronda49)=0
            else
            for Ronda51=1:(Valor53-2)

Costo18(Ronda51)=Clie_Clie(MejorOpcionRuta7(Ronda51),MejorOpcionRuta7(Ron
da51+1));
                Costo18
            end
            if Ronda49==Valor46

Costo19(Ronda49)=sum(Costo18)+(Clie_Dep(MejorOpcionRuta7(1),Valor48))+(Cl
ie_Dep(MejorOpcionRuta7(Valor53-1),Valor48))+(Fjj(Valor48));
            else

Costo19(Ronda49)=sum(Costo18)+(Clie_Dep(MejorOpcionRuta7(1),MejorOpcionDe
positos(Iteracion,Ronda49)))+(Clie_Dep(MejorOpcionRuta7(Valor53-
1),MejorOpcionDepositos(Iteracion,Ronda49)))+(Fjj(MejorOpcionDepositos(It
eracion,Ronda49)));
            end
            end

            end
        end
        clearvars Costo20
        Costo20=sum(Costo19);
        if Costo20>=CostoMejorOpcion(Iteracion)
             MejorOpcionRuta(Iteracion,:,:)=zeros(Valor36,Valor1);
```

```
                clearvars Valor3a Valor1a Ronda95 Ronda96
                Valor3a=length(MejorOpcionRuta1(:,1));
                Valor1a=length(MejorOpcionRuta1(1,:));
                for Ronda95=1:Valor1a
                    for Ronda96=1:Valor3a

MejorOpcionRuta(Iteracion,Ronda96,Ronda95)=MejorOpcionRuta1(Ronda96,Ronda
95);
                    end
                end
            disp('-La solucion no fue optimizada mediante la forma 1 de
Perturbacion-')
        else
            MejorOpcionDepositos(Iteracion,Valor46)=Valor48;
            CostoMejorOpcion(Iteracion)=Costo20;
        end
    end
    Valor1=length(MejorOpcionRuta(Iteracion,1,:));
    Valor3=length(MejorOpcionRuta(Iteracion,:,1));
    clearvars Ronda97 Ronda98
    for Ronda97=1:Valor1
        for Ronda98=1:Valor3

MejorOpcionRuta1(Ronda98,Ronda97)=MejorOpcionRuta(Iteracion,Ronda98,Ronda
97);
        end
    end
     %
MejorOpcionRuta(Iteracion,:,:)
MejorOpcionDepositos(Iteracion,:)
CostoMejorOpcion(Iteracion)
%
else
    Valor54=ceil(rand*Valor1); %Deposito Destino
    while
(Valor54==Valor46)||(MejorOpcionDepositos(Iteracion,Valor54)==0)
        Valor54=ceil(rand*Valor1);
    end

    %Analisis de la viabilidad segun relacion Demanda/Capacidad
    clearvars Valor55 Valor56 Valor57 Valor58 Valor59 Valor60
    Valor55=Rjj(Valor54); %Capacidad del deposito destino
    for Ronda52=1:Valor36
        Valor56(Ronda52)=0;
        if MejorOpcionRuta(Iteracion,Ronda52,Valor54)>0
        Valor56(Ronda52)=di(MejorOpcionRuta(Iteracion,Ronda52,Valor54));
        end
    end
    Valor57=sum(Valor56); %Demanda de los clientes previamente asignados
al deposito destino
    for Ronda53=1:Valor36
        Valor58(Ronda53)=0;
        if MejorOpcionRuta(Iteracion,Ronda53,Valor46)>0
        Valor58(Ronda53)=di(MejorOpcionRuta(Iteracion,Ronda53,Valor46));
        end
    end
```

```
    Valor59=sum(Valor58); %Demanda de los clientes a trasladar al
deposito destino
    Valor60=Valor57+Valor59; %Demanda total de los clientes
    if Valor60>Valor55
        disp('-El deposito seleccionado como destino no posee la
capacidad necesaria para suplir la demanda-')
    else
        %Analisis de la viabilidad segun costos
        Valor61=MejorOpcionRuta(Iteracion,:,Valor46);
        Valor62=MejorOpcionRuta(Iteracion,:,Valor54);
        Valor63=[Valor62,Valor61];
        Valor64=length(Valor63);
        Valor36=Valor64; %Cantidad maxima de clientes asignados a un
deposito
        for Ronda54=1:Valor64
            MejorOpcionRuta(Iteracion,Ronda54,Valor54)=Valor63(Ronda54);
            MejorOpcionRuta(Iteracion,Ronda54,Valor46)=0;
        end
        MejorOpcionDepositos(Iteracion,Valor46)=0;
        clearvars Costo22
        for Ronda55=1:Valor1
            clearvars ViabilidadDeposito1 PosViabilidadDeposito1
            [PosViabilidadDeposito1]=find(PosDepositosNulos==Ronda55);
            ViabilidadDeposito1=length(PosViabilidadDeposito1);
            if sum(ViabilidadDeposito1)>0
               Costo22(Ronda55)=0;
            else
            Valor65=1;
            if Ronda55==Valor46
                Costo22(Ronda55)=0;
            else
                for Ronda56=1:Valor64
                    if MejorOpcionRuta(Iteracion,Ronda56,Ronda55)>0

MejorOpcionRuta8(Valor65)=MejorOpcionRuta(Iteracion,Ronda56,Ronda55);
                        Valor65=Valor65+1;
                    end
                end

                clearvars Costo21
                if Valor65==2 %Si Valor15 es igual a 2 significa que solo
hay un cliente por ese deposito
                    Costo21=0;

Costo22(Ronda55)=sum(Costo21)+(Clie_Dep(MejorOpcionRuta8(1),MejorOpcionDe
positos(Iteracion,Ronda55)))+(Clie_Dep(MejorOpcionRuta8(Valor65-
1),MejorOpcionDepositos(Iteracion,Ronda55)))+(Fjj(MejorOpcionDepositos(It
eracion,Ronda55)));
                elseif Valor65<=1 %Si Valor15 es menor o igual a 1
significa que no hay un cliente en este deposito
                    Costo22(Ronda55)=0
                else
                for Ronda57=1:(Valor65-2)

Costo21(Ronda57)=Clie_Clie(MejorOpcionRuta8(Ronda57),MejorOpcionRuta8(Ron
da57+1));
                    Costo21
```

```
                end

Costo22(Ronda55)=sum(Costo21)+(Clie_Dep(MejorOpcionRuta8(1),MejorOpcionDe
positos(Iteracion,Ronda55)))+(Clie_Dep(MejorOpcionRuta8(Valor65-
1),MejorOpcionDepositos(Iteracion,Ronda55)))+(Fjj(MejorOpcionDepositos(It
eracion,Ronda55)));
                end
            end

            end
        end
       clearvars Costo23
        Costo23=sum(Costo22);
        if Costo23>CostoMejorOpcion(Iteracion)
            MejorOpcionRuta(Iteracion,:,:)=zeros(Valor36,Valor1);
            for Ronda83=1:Valor1
                for Ronda84=(Valor3+1):Valor36
                    MejorOpcionRuta1(Ronda84,Ronda83)=0;
                end
            end
               clearvars Valor3a Valor1a Ronda95 Ronda96
               Valor3a=length(MejorOpcionRuta1(:,1));
               Valor1a=length(MejorOpcionRuta1(1,:));
               for Ronda95=1:Valor1a
                   for Ronda96=1:Valor3a

MejorOpcionRuta(Iteracion,Ronda96,Ronda95)=MejorOpcionRuta1(Ronda96,Ronda
95);
                   end
               end
            disp('-La solucion no fue optimizada mediante la forma 2 de
Perturbacion-')
        else
            CostoMejorOpcion(Iteracion)=Costo23;
        end
    end
    Valor1=length(MejorOpcionRuta(Iteracion,1,:));
    Valor3=length(MejorOpcionRuta(Iteracion,:,1));
    clearvars Ronda97 Ronda98
    for Ronda97=1:Valor1
        for Ronda98=1:Valor3

MejorOpcionRuta1(Ronda98,Ronda97)=MejorOpcionRuta(Iteracion,Ronda98,Ronda
97);
        end
    end
%
MejorOpcionRuta(Iteracion,:,:)
MejorOpcionDepositos(Iteracion,:)
CostoMejorOpcion(Iteracion)
%
end
end

clearvars Valor46 Valor48 Valor49 Valor50 Valor51 Valor52 Valor53 Valor54
Valor55 Valor56 Valor57 Valor58 Valor59 Valor60 OpcionPerturbacion
```

```
clearvars ViabilidadDeposito1 PosViabilidadDeposito1 Costo18 Costo19
Costo20 Costo21 Costo22 Costo23 Ronda50 Ronda51 Ronda52 Ronda53 Ronda54
Ronda55
clearvars Ronda83 Ronda84 MejorOpcionRuta7 MejorOpcionRuta8 Valor61
Valor62 Valor63 Valor64 DepositoRepetido1 DepositoRepetido3
PosDepositoRepetido2

CostoMejorOpcion
MejorOpcionDepositos
MejorOpcionRuta
MejorOpcionRuta1
Valor1
Valor3
('Perturbacion')
pause;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
[MejorOpcionRuta1,MejorOpcionRuta,MejorOpcionDepositos,CostoMejorOpcion,V
alor1,Valor3,Valor36]=ILS(Valor3,Valor1,MejorOpcionRuta1,MejorOpcionRuta,
Iteracion,MejorOpcionDepositos,CostoMejorOpcion,di,Rjj,Fjj,Clie_Dep,Clie_
Clie,T)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

end
end
%%
%%Seleccion de la solucion definitiva
clc;
[MejorCostoDefinitivo,MejorIteracionDefinitiva]=min(CostoMejorOpcion)
MejorDepositosDefinitivos=MejorOpcionDepositos(MejorIteracionDefinitiva,:
)
MejorRutasDefinitivas=MejorOpcionRuta(MejorIteracionDefinitiva,:,:)
disp('Fin del algoritmo - ACS')
TiempoAnalisis=toc

%%Creacion de las graficas con la ruta
O=length(MejorDepositosDefinitivos);
coordenadaS1=xlsread('DATOS DEL PROBLEMA','coordenadas1','A2:AE2');
for clientex1=1:n
    coordenadax1(clientex1)=coordenadaS1(clientex1); %Coordenada x de los
clientes
end
coordenadaSS1=xlsread('DATOS DEL PROBLEMA','coordenadas1','A3:AE3');

for clientey1=1:n
    coordenaday1(clientey1)=coordenadaSS1(clientey1); %Coordenada y de
los clientes
end


coordenadaS2=xlsread('DATOS DEL PROBLEMA','coordenadas2','A2:K2');
```

```
for depositox2=1:O
    if MejorDepositosDefinitivos(depositox2)>0

coordenadax2(depositox2)=coordenadaS2(MejorDepositosDefinitivos(depositox
2));%coordenada x de los depositos
    end
end
coordenadaSS2=xlsread('DATOS DEL PROBLEMA','coordenadas2','A3:K3');
for depositoy2=1:O
    if MejorDepositosDefinitivos(depositoy2)>0

coordenaday2(depositoy2)=coordenadaSS2(MejorDepositosDefinitivos(deposito
y2)); %coordenada y de los depositos
    end
end

%depósitos
plot(coordenadax2,coordenaday2,'s')
hold on
%clientes
plot(coordenadax1,coordenaday1,'or')

P=length(MejorRutasDefinitivas(1,:,1));


for i=1:O
    if MejorDepositosDefinitivos(i)>0
    clearvars contador X1 Y1 Opcion
    contador=1;
    X1(contador)=coordenadax2(i);%(MejorDepositosDefinitivos(i));
    Y1(contador)=coordenaday2(i);%(MejorDepositosDefinitivos(i));
    for s=1:P

        if MejorRutasDefinitivas(1,s,i)>0
        contador=contador+1;

        X1(contador)=coordenadax1(MejorRutasDefinitivas(1,s,i));
        Y1(contador)=coordenaday1(MejorRutasDefinitivas(1,s,i));
        end
        if s==P
        Opcion=1;
        contador=contador+1;
        if Opcion==1
        Opcion=Opcion+1;
        X1(contador)=coordenadax2(i);
        Y1(contador)=coordenaday2(i);
        end
        end
    end
    plot(X1,Y1,'--*k')
    end
end
hold off
```

- **Descripción variables y parámetros empleados en el código general ACO+ILS**

  s1: número de iteraciones del algoritmo ACO+ILS.
  b1: número de hormigas del proceso general de ACO+ILS.
  r: parámetro relacionado a la determinación de P_hs.
  n: número de clientes.
  m: número de depósitos candidatos.
  T: número de iteraciones del método ILS y cada una de sus estructuras de vecindad.
  alpha: parámetro que determina la influencia relativa de la información heurística vs la información de feromona para el proceso de selección.
  beta: parámetro que determina la influencia relativa de la información heurística vs la información de feromona para el proceso de asignación.
  gamma: parámetro que determina la influencia relativa de la información heurística vs la información de feromona para el proceso de ruteo de vehículos.
  rho1: parámetro de evaporización de feromona para el proceso de selección de depósitos.
  rho2: parámetro de evaporización de feromona para el proceso de asignación de clientes a depósitos.
  rho3: parámetro de evaporización de feromona para el proceso de ruteo de vehículos.
  q01: parámetro que determina la importancia relativa entre las ecuaciones de exploración vs las ecuaciones de explotación para el proceso de selección.
  q02: parámetro que determina la importancia relativa entre las ecuaciones de exploración vs las ecuaciones de explotación para el proceso de asignación de clientes.
  q03: parámetro que determina la importancia relativa entre las ecuaciones de exploración vs las ecuaciones de explotación para el proceso de ruteo de vehículos.
  s2: número de iteraciones para el proceso de ruteo de vehículos.
  b2: número de hormigas para el proceso de ruteo de vehículos.
  di: demanda del cliente i.
  Rj: capacidad del depósito j.
  Fj: costo asociado a la instalación de cada depósito j.
  Clie_Clie: matriz de costo asociado a la distancia entre clientes, costo de ruteo.
  Clie_Dep: costo de servicio de cada depósito a los clientes.
  Tao_j: matriz de feromona para el proceso de selección de depósitos.

Epsilon_ik: matriz de feromona para el proceso de asignación de clientes a depósitos.
dseta: matriz inicial de feromona para el proceso de ruteo de vehículos.
U: número entero aleatorio entre [0,r], para determinar P_hs.
P_hs: número de depósitos seleccionados.
Costo2: costo asociado a la distancia entre clientes y el costo de servicio de depósitos a clientes.
Costo3: valor mínimo de cada columna de la matriz Costo2.
VectorProbabilidad2: Crea un vector de 100 elementos con los vectores a seleccionar repetidos de acuerdo a su porcentaje.
Capacidad_ik: capacidad disponible de cada depósito.
k: posición del depósito seleccionado.
Seleccion: matriz donde cada columna representa el depósito en orden de selección y en ella los clientes asignados a cada uno.
CorreccionDepositosUtilizados: corrige la cantidad de depósitos seleccionados, si a un depósito no se le asignan clientes, este no se tiene en cuenta en el análisis.
DepositoHabitado: Hipermatriz con la selección de los depósitos habitado seleccionados previamente.
cliXdep: vector que contiene los clientes asignados a un depósito.
Cant_clie: cantidad de clientes asignados a un depósito.
FirstPosicion: primer cliente seleccionado para iniciar la ruta.
Destino: matriz temporal con la ruta de los clientes para cada vector de depósitos.
PosClie: variable temporal que ayuda a determinar cuales clientes ya fueron ubicados y asi no repetirlos.
dseta_iv: matriz de feromona para el proceso de ruteo de vehículos.
fi_iv: matriz de feromona para el proceso de ruteo de vehículos.
ClienteDestino: vector que contiene los clientes asignados a un depósito.
Costo4: costo de ruteo por cada depósito seleccionado y sus clientes asignados.
Costo5: costo de ruteo por cada depósito seleccionado y sus clientes asignados más costo de instalación.
CostoMinDep: Costo mínimo interno en la sección de ruteo entre las rutas posibles para un deposito determinado.
CostoMinDep2: Costo mínimo interno en la sección de ruteo entre las rutas posibles para un deposito determinado excluyendo el valor de instalación del deposito.
RutaFinalDep: ruta final seleccionada a partir del costo minimo.
CostoMaxDep: costo máximo por cada depósito y sus clientes asignados. Instalación más ruteo.

CostoMaxDep: costo de ruteo máximo por cada depósito.
L_b: costo de ruteo mínimo encontrado hasta el momento.
L_s: costo de ruteo mínimo de la iteración s2 actual.
L_w: costo máximo de ruteo en la iteración s2 actual.
CostoMin: Determina el costo mínimo entre todos los costos calculados.
IteracionMin: Almacena la valor s1 en la cual fue calculado el 'CostoMin'.
HormigaMin: Almacena el valor b1 en el cual fue calculado 'CostoMin'
CostoMax: Determina el costo maximo entre todos los costos calculados, útil para la actualización de feromona.
L_b2: menor costo del sistema encontrado hasta el momento.
L_s2: menor costo del sistema encontrado en la iteración s1 actual.
L_w2: mayor costo de todo el sistema encontrado en la iteración s1 actual.
nj: cantidad máxima de clientes asignados a un depósito.

- **Relacionadas al proceso de perturbación de ILS**
  Valor46: deposito fuente para realizar la 'Perturbacion'.
  OpcionPerturbacion: determina mediante un número aleatorio cual opción de proceso de perturbación tomar.
  Valor48: Selecciona un nuevo depósito (sin utilizar) para almacena los clientes del depósito fuente, según indica la estructura de la perturbación.
  PosDepositoRepetido2: vector que indica si el deposito nuevo, está o no en uso.
  Valor49: capacidad del nuevo depósito seleccionado por la opción de perturbación el método ILS.
  Valor50: vector con la demanda de los clientes asignados al depósito fuente.
  Valor51: suma total de las demandas de los clientes asignados al depósito fuente.
  Costo19: costo total de cada depósito y su respectiva ruta de clientes.
  Costo18: costo del traslado Cliente – Cliente.
  Costo20: costo total del cambio.
  Valor54: deposito destino dentro de los depósitos en uso para almacenar los clientes del depósito fuente.
  Valor55: capacidad del depósito destino.
  Valor56: vector con la demanda de los clientes asignados al depósito destino previamente.
  Valor57: demanda total asignada inicialmente al depósito destino.
  Valor58: demanda de los nuevos clientes a asignar al depósito destino.
  Valor59: demanda total de los nuevos clientes inicialmente al depósito destino.
  Valor60: demanda total a cubrir por el deposito destino.

Valor63: nueva ruta sobre el depósito destino, luego de la perturbación.
Valor64: tamaño de la nueva ruta sobre el deposito destino luego de la perturbación.
Costo22: vector con el costo de servicio de cada depósito y sus clientes.
Costo21: vector con el costo de la ruta de clientes 'Clie-Clie'.
Costo23: valor total del costo de la nueva ruta.

- **Respuesta:**
  MejorCostoDefinitivo: corresponde al menor costo total encontrado en todo el recorrido.
  MejorDepositosDefinitivos: vector que contiene los valores correspondientes a los depósitos seleccionados.
  MejorRutasDefinitivas: muestra por vectores, en orden de selección, la disposición de clientes para las rutas correspondientes a cada depósito.

ANEXO F. VALIDACIÓN DEL ALGORITMO PROGRAMADO EN MATLAB

El proceso de validación se hace en 3 etapas, la primera para el proceso de selección-asignación, la segunda para el ruteo de vehículos y la última para el ILS.

ETAPA I VALIDACIÓN EN EL PROCESO DE SELECCIÓN DE DEPÓSITOS Y ASIGNACIÓN DE CLIENTES

Para esta etapa, se usan los datos del ejemplo Prototipo I, pero se debe hacer $q_0 = 1$ y $q_0^1 = 1$, para que el proceso sea guiado exclusivamente por las ecuaciones de explotación (Ec. 27 y Ec. 32), y evitar las variaciones generadas por la opción de probabilidad. También se hace $s = 1$, $b = 1$ y $r = 1$ (para garantizar un valor de $U(1,r) = 1$), así se genera una sola respuesta a comparar con el ejemplo escrito. Si se siguen estas indicaciones, se puede comprobar que los resultados arrojados en Matlab, son los siguientes:

- **Cantidad de depósitos a seleccionar**

  $P_s^h = 4$

- **Depósitos seleccionados** $W_1^1 = \{\mathbf{1}, \mathbf{5}, \mathbf{3}, \mathbf{4}\}$

Valor de argumento máximo	Depósito seleccionado
Argmax=0.0040	j(:,:,1) =1
Argmax=0.0027	j(:,:,2) =5
Argmax=0.0024	j(:,:,3) =3
Argmax=0.0022	j(:,:,4) =4

- **Asignación de clientes a depósitos**

  $Selección = \begin{vmatrix} 1 & 5 & 3 & 6 \\ 2 & 0 & 4 & 7 \end{vmatrix}$

ETAPA II VALIDACIÓN PARA EL PROCESO DE RUTEO DE VEHÍCULOS

Para la validación del proceso de ruteo de vehículos, se compara con el ejemplo Prototipo II. Para esto se tienen los siguientes datos: $r = 1$, $S1 = 1$, $S2 = 1$, $b1 = 1$, $b2 = 2$ y $q03 = 1$ (para que el proceso sea guiado por la ecuación de $argmax$. La Tabla 15. indica el costo asociado a la distancia entre clientes-clientes y clientes-depósitos, se observa que al correr el algoritmo en Matlab, los depósitos seleccionados son $k = 2$ y $k = 4$ y la ruta encontrada corresponde a los resultados obtenidos en el desarrollo del ejemplo prototipo II. Se corre el algoritmo tantas veces hasta elegir para el depósito $k = 4$ como cliente inicial $i = 5$.

Tabla 28. Matriz de costos para validación de VRP

	1	2	3	4	5	6	7	d1	d2	d3	d4
1	0	24	30	23	34	21	13	24	20	32	30
2	24	0	12	30	45	62	28	28	49	50	10
3	30	12	0	50	30	60	40	50	40	70	7
4	23	30	50	0	23	32	24	80	30	45	50
5	34	45	30	23	0	50	12	68	24	75	70
6	21	62	60	32	50	0	20	15	32	45	60
7	13	28	40	24	12	20	0	30	15	60	35

Tabla 29. Datos para validación VRP

n	m	R_1	R_2	R_3	R_4	F_1	F_2	F_3	F_4
7	4	20	78	24	34	60	100	80	60

d_1	d_2	d_3	d_4	d_5	d_6	d_7	α	β	ρ	ρ'	ρ''	γ
8	4	10	7	3	5	2	1	1	0, 1	0, 1	0, 1	4

La respuesta final en Matlab, que coincide con los resultados del ejemplo prototipo II, está dada por los siguientes vectores:

MejorDepositosDefinitivos = 4 2

MejorRutasDefinitivas(:,:,1) = 2 3 0 0 0

MejorRutasDefinitivas(:,:,2) = 5 4 6 1 7

En esta etapa, también se puede validar el costo total del problema, éste es:

MejorCostoDefinitivo = 317

ETAPA III VALIDACIÓN PARA EL ILS

Para la validación del ILS, se compara con el ejemplo Prototipo I, con una variación en la matriz de costos asociados a la distancia entre clientes-clientes y clientes-depósitos, para mejor verificación del funcionamiento del ILS. Para esto se tienen los siguientes datos: $r = 1$, $S1 = 1$, $S2 = 1$, $b1 = 1$, $b2 = 2$, $T = 1$. Los datos de costo, capacidad y demanda son los siguientes:

m	n	R_1	R_2	R_3	R_4	R_5	d_1	d_2	d_3	d_4	d_5	d_6	d_7
5	7	40	55	35	50	60	20	15	25	10	20	10	15
F_1	F_2	F_3	F_4	F_5									
100	200	120	150	150									

Y la matriz de costos asociados a la distancia entre clientes-clientes y clientes-depósitos es la siguiente:

	1	2	3	4	5	6	7	d1	d2	d3	d4	d5
1	0	50	80	120	180	190	210	50	100	200	220	180
2	50	0	30	70	130	140	160	50	80	140	190	140
3	80	30	0	40	100	110	130	80	30	30	100	100
4	120	70	90	0	60	70	90	110	20	100	30	70
5	180	130	100	60	0	20	30	120	90	60	70	30
6	190	140	110	70	20	0	20	140	110	50	10	60
7	210	160	130	90	30	20	0	200	120	90	20	100

Se observa que al correr el algoritmo en Matlab, los depósitos seleccionados y los clientes asignados a estos depósitos son:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 3 \quad 4\} \\ A = \begin{Bmatrix} 1 & 5 & 3 & 4 \\ 2 & & & 6 \\ & & & 7 \end{Bmatrix} \end{cases}$$

COSTO FINAL ACO: 930

Para la estructura de barrio N1 se escogen arbitrariamente: el depósito 5 y el deposito 3, y los clientes 5 y 3, para intercambiarlos. Obteniendo como resultado la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 3 \quad 4\} \\ A = \begin{Bmatrix} 1 & 3 & 5 & 4 \\ 2 & & & 6 \\ & & & 7 \end{Bmatrix} \end{cases}$$

Con la anterior modificación se obtiene que el costo es 1130, el costo es mayor que el obtenido por ACO, por lo tanto no se acepta la respuesta de esta estructura en la iteración y se continúa el proceso con los vectores iniciales.

Para la estructura de barrio N2 se escogen arbitrariamente: el depósito 5 y el deposito 4, y el cliente 4, para cambiarlo. Obteniendo como resultado la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 3 \quad 4\} \\ A = \begin{Bmatrix} 1 & 5 & 3 & 4 \\ 2 & 7 & & 6 \end{Bmatrix} \end{cases}$$

Se obtiene que el costo es 1000, el costo es mayor que el obtenido por ACO, por lo tanto no se acepta la respuesta de esta estructura en la iteración.

Para la estructura de barrio N3 se escogen arbitrariamente: el depósito 1 y los clientes 1 y 2, para intercambiar la posición. Obteniendo como resultado la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 3 \quad 4\} \\ A = \begin{Bmatrix} 2 & 5 & 3 & 4 \\ 1 & & & 6 \\ & & & 7 \end{Bmatrix} \end{cases}$$

Con este cambio, el costo del sistema es 930, el costo es igual al obtenido por ACO, por lo tanto esta nueva estructura se acepta y el proceso continúa.

Para la estructura de barrio N4 se escogen arbitrariamente: el depósito 4 y el cliente 4, para cambiarlo de posición entre 6 y 7. Se obtiene como resultado la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 3 \quad 4\} \\ A = \begin{Bmatrix} 2 & 5 & 3 & 6 \\ 1 & & & 4 \\ & & & 7 \end{Bmatrix} \end{cases}$$

Como el nuevo costo es 980, y este es mayor al obtenido en el proceso anterior, no se acepta la respuesta de esta estructura en la iteración.

Se toma un número mayor 0,5, para la variable OpciónPerturbación, para verificar el resultado de la perturbación tipo I, y posteriormente se toma un numero menor a 0,5 para la perturbación Tipo II y se corre nuevamente el algoritmos con este valor.

Perturbación I: se escogen arbitrariamente los depósitos 2 y 3; los clientes de 3 pasan al depósito 2 y el deposito 3 se cierra. Así, el resultado que se obtiene es el siguiente:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 2 \quad 4\} \\ A = \begin{Bmatrix} 2 & 5 & 3 & 4 \\ 1 & & & 6 \\ & & & 7 \end{Bmatrix} \end{cases}$$

Se obtiene un costo de 1010, que es inferior al obtenido en el proceso anterior, por lo tanto se acepta la respuesta de esta estructura.

También se realiza la validación con la perturbación tipo II, pasando todos los clientes del depósito 3 al depósito 5 y se obtiene un costo de 920, éste es menor al obtenido con la perturbación tipo I:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 4\} \\ A = \begin{Bmatrix} 2 & 5 & 6 \\ 1 & 3 & 4 \\ & & 7 \end{Bmatrix} \end{cases}$$

Se realiza nuevamente la estructura de barrio N1 con los mismos cambios que la anterior estructura N1, con la opción 2 de perturbación. Para este caso, como se cerró el deposito 3, entonces se procede a tomar los depósitos 1 y 5 para realizar el intercambio de clientes:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 4\} \\ A = \begin{Bmatrix} 5 & 2 & 6 \\ 3 & 1 & 4 \\ & & 7 \end{Bmatrix} \end{cases}$$

El costo que se obtiene con matlab es de 1200, por lo tanto, se mantiene el mismo costo de la perturbación Tipo II.

Con la Estructura de Barrio número 2 se obtiene la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 4\} \\ A = \begin{Bmatrix} 2 & 5 & 6 \\ 1 & 3 & 4 \\ & 7 & \end{Bmatrix} \end{cases}$$

Se obtiene un costo de 1020, que es mayor al obtenido en el proceso anterior, por lo tanto no se acepta la respuesta de esta estructura.

Con la Estructura de Barrio número 3 se obtiene la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 4\} \\ A = \begin{Bmatrix} 1 & 5 & 4 \\ 2 & 3 & 6 \\ & & 7 \end{Bmatrix} \end{cases}$$

Se obtiene un costo de 920, que es igual al obtenido en el proceso anterior, por lo tanto se acepta la respuesta de esta estructura.

Con la Estructura de Barrio número 4se obtiene la siguiente respuesta:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 4\} \\ A = \begin{Bmatrix} 1 & 5 & 6 \\ 2 & 3 & 4 \\ & & 7 \end{Bmatrix} \end{cases}$$

Se obtiene un costo de 970, que es mayor al obtenido en el proceso anterior, por lo tanto, no se acepta la respuesta de esta estructura.

Finalmente la respuesta es:

$$T_s \quad \begin{cases} V = \{1 \quad 5 \quad 4\} \\ A = \begin{Bmatrix} 1 & 5 & 4 \\ 2 & 3 & 6 \\ & & 7 \end{Bmatrix} \end{cases}$$

La respuesta final en Matlab, que coincide con los resultados del ejemplo prototipo I con las crrespondientes modificaciones, está dada por los siguientes vectores:

MejorCostoDefinitivo = 920

MejorIteracionDefinitiva = 1

MejorDepositosDefinitivos = 1 5 0 4

MejorRutasDefinitivas(:,:,1) = 1 2 0 0 0 0

MejorRutasDefinitivas(:,:,2) = 5 0 0 3 0 0

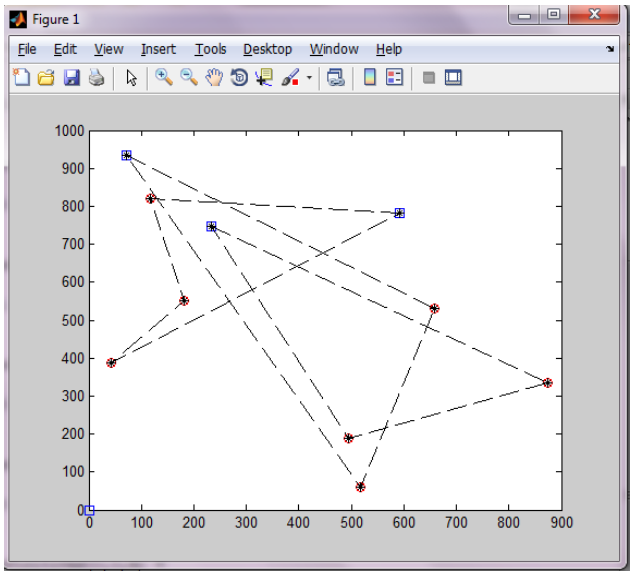
MejorRutasDefinitivas(:,:,4) = 4 6 7 0 0 0

Fin del algoritmo - ACS

TiempoAnalisis =

6.706539229708077e+02

Ilustración 30. Gráfica de los depósitos seleccionados, clientes asignados y rutas establecidas para la validación del ILS.



ANEXO G. ARTÍCULO DE INVESTIGACIÓN

ALGORITMO HÍBRIDO PARA RESOLVER EL PROBLEMA CONJUNTO DE LOCALIZACIÓN DE INSTALACIONES Y EL RUTEO DE VEHÍCULOS LRP

ANGÉLICA PATRICIA NIER OBREGÓN
ESTUDIANTE DE INGENIERÍA INDUSTRIAL
UNIVERSIDAD INDUSTRIAL DE SANTANDER
angelicanier@hotmail.com
BUCARAMANGA, COLOMBIA

JENIFER ALEXANDRA NIÑO HERNÁNDEZ
ESTUDIANTE DE INGENIERÍA INDUSTRIAL
UNIVERSIDAD INDUSTRIAL DE SANTANDER
jealnihe@yahoo.es
BUCARAMANGA, COLOMBIA

RESUMEN

PALABRAS CLAVE: Problema de localización-enrutamiento (LRP), optimización por colonia de hormigas (ACO), búsqueda local iterativa (ILS), metaheurística.

El problema conjunto de localización de instalaciones y enrutamiento de vehículos, con capacidad limitada en depósitos y un vehículo por depósito (LRP, por sus siglas en inglés), es definido como un caso especial del problema de ruteo de vehículos (VRP), donde se determina de forma simultánea la localización de depósitos y las rutas de distribución. En este proyecto, se propone un algoritmo híbrido que emplea optimización por colonia de hormigas (ACO, por sus siglas en inglés), con el uso de tres colonias (selección de depósitos, asignación de clientes y VRP), para generar una solución inicial que es mejorada con búsqueda local iterativa (ILS, por sus siglas en inglés) que consta de cuatro estructuras de vecindad y una perturbación. El algoritmo ACO+ILS es implementado en MATLAB y con la propuesta de un banco de pruebas se emplea un diseño factorial fraccionado 2^{k-2} con el fin de determinar el efecto y la mejor combinación en valores de los factores. Los resultados del algoritmo propuesto son comparados con los del ACO tradicional mostrando mejoras significativas frente a este en cuanto a disminución costo total del sistema.

ABSTRACT

KEYWORDS: Location routing problema (LRP), ant colony optimization, iterated local search (ILS), metaheuristic.

The location routing problema (LRP), is defined a special case of the vehicle routing problema (VRP), which is determined simultaneously locating facilities and distribution routes. In this research work, a hybrid algorithm that uses Ant Colony Optimization (ACO), with the use of three colonies (location selection, customer assignment and VRP) to generate an initial solution is proposed that is enhanced iterated local search (ILS) consisting of four structures neighborhood and disturbance. ACO+ILS algorithm is implemented in Matlab and a test is proposed. A fractional factorial design 2^(k-2) is used to determine the effect and the best combination of values of the factors utilized. The results of the proposed algorithm are compared with the traditional ACO and significant improvements over this, in terms of the reduction of total system cost, are showed.

1 INTRODUCCIÓN

El problema conjunto de localización de instalaciones y ruteo de vehículos (LRP por sus siglas en inglés) es un problema de optimización combinatoria NP-Hard, en el cual la ubicación de instalaciones se determina simultáneamente con las rutas de distribución [1]. En los sistemas logísticos se busca mejorar aspectos como los tiempos de entrega y proximidad al mercado, y así, aumentar la calidad del servicio.

Este problema puede describirse de la siguiente manera: dado un conjunto de posibles sitios de distribución y un conjunto de clientes con una demanda esperada, se seleccionan los depósitos, se asigna cada cliente a uno de estos, desde donde envía un vehículo para cumplir con su demanda. El objetivo es minimizar los costos asociados a la apertura de los depósitos y al ruteo de vehículos. Algunas variantes del problema están relacionadas con la capacidad de la instalación y/o del vehículo, limitación en el tiempo total de la ruta, la distancia total recorrida por el vehículo, o los datos de entrada [2].

El LRP ha sido cuestionado debido a que contiene dos problemas con horizontes de planeación diferentes (localización-largo plazo y ruteo-mediano plazo); sin embargo, se ha comprobado que tomando una decisión estratégica pero con modificaciones en las rutas se pueden obtener mejores resultados en la solución del problema [3].

En el presente trabajo se aborda el problema de localización de instalaciones y ruteo de vehículos con múltiples depósitos con limitaciones de capacidad y un vehículo por depósito. Para dar solución a este problema se plantea un algoritmo de Optimización basado en Colonia de Hormigas (ACO) combinado con la heurística de la Búsqueda Local Iterativa (ILS), que permite mejorar las soluciones dadas por el ACO.

A continuación se presenta la descripción del problema, de ACO y de ILS; así como el desarrollo del algoritmo híbrido y los resultados obtenidos en su implementación en el lenguaje de programación Matlab.

2 PROBLEMA CONJUNTO DE LOCALIZACIÓN DE INSTALACIONES Y RUTEO DE VEHÍCULOS

El LRP es un conjunto de problemas que tienen gran impacto en las decisiones de los sistemas logísticos y afectan tanto la rentabilidad de las empresas como la calidad del servicio prestado a los clientes. El problema está integrado por el problema de localización de instalaciones que se relaciona con las decisiones de apertura, ubicación y asignación de clientes a las instalaciones; y por el problema de ruteo de vehículos en el cual se obtienen las rutas para satisfacer la demanda de los clientes [4].

La forma general del LRP, busca minimizar los costos de apertura de los depósitos y de las rutas, seleccionar la ubicación de las instalaciones, asignar los clientes y construir las rutas que satisfagan las siguientes restricciones [5]:

- Satisfacer la demanda de los clientes sin exceder la capacidad de las instalaciones o de los vehículos.
- Cada ruta empieza y termina en la misma instalación.
- La longitud de una ruta y el número de vehículos tiene un número límite predefinido.

Los trabajos relacionados con este problema se han realizado desde hace casi cincuenta años. Maranzana (1964) [6], presenta un trabajo en el cual la ubicación de fábricas, almacenes y puntos de suministro, es influenciada por los costos de transporte. Se incorpora la trayectoria más corta en un problema de localización, en lugar del problema de enrutamiento de vehículos, sin embargo, este trabajo se ha considerado como una de las primeras publicaciones que tienen en cuenta el LRP.

Watson-Gandy y Dorhrn (1973) [7], estudian claramente el LRP por medio de un modelo no lineal. Por otro lado, Salhi y Rand (1989) [8], presentan los beneficios obtenidos al combinar los problemas de localización de depósitos y ruteo de vehículos.

Las variantes del problema que se encuentran en la literatura, están asociadas con la naturaleza de la demanda, es decir, si es determinística o estocástica; el periodo de planeación, estático o dinámico; el tipo de flota de los vehículos, homogénea o heterogénea; la cantidad de instalaciones; la capacidad de instalaciones y/o vehículos; y el tipo de instalación: primaria, si estas son los orígenes y destinos de los recorridos de los vehículos, y secundarias cuando sólo pueden ser depósitos intermedios [9].

En cuanto a los métodos de solución para el LRP se tienen los métodos exactos y los métodos aproximados. Los métodos exactos sólo pueden plantear instancias relativamente pequeñas debido a la complejidad del problema. Laporte y Norbert (1981) [10], desarrollan el primer algoritmo exacto utilizando un algoritmo *branch-and-bound* para dar solución al LRP general en el cual se selecciona un solo depósito y se utiliza un número fijo de vehículos. Laporte, Nobert, y Arpin (1986) [11], plantean un algoritmo *branch and cut*, para un LRP con limitaciones de capacidad para el vehículo.

Recientemente, algunos trabajos exponen algoritmos exactos. Entre estos está el algoritmo *branch and price* propuesto por Akca, Berger y Ralphs (2009) [12], en el cual es solucionado el LRP con limitaciones de capacidad en depósitos y vehículos (CLRP). Belenger, Benavent y Prins (2011) [13], aportan un algoritmo *branch and cut* basado en un modelo lineal cero-uno. Karaoglan, Aliparmak, Kara y Dengiz (2011) [14], también abordan el enfoque de *branch and cut* para resolver LRP con entrega y recogida simultaneas y, adicionalmente se emplea recocido simulado.

En cuanto a los métodos aproximados existe un gran número de investigaciones y trabajos publicados que utilizan Heurísticas y Metaheurísticas, puesto que, en problemas amplios se pueden obtener soluciones factibles en tiempos de cómputo razonables. Wu, Low, y Bai (2002) [15], desarrollan un algoritmo iterativo de búsqueda tabú combinado con recocido simulado, en el cual se consideran los puntos finales de las rutas para entrar a la fase de localización.

Albareda-Sambola et al. (2005) [9], proponen como solución inicial de la heurística Búsqueda tabú, la solución de la programación lineal del

modelo considerado en un procedimiento de redondeo. Además, esta solución ofrece una cota inferior inicial. Los autores proponen una cota inferior diferente basada en la estructura del problema, con dos términos: los costos de ubicación y los costos de enrutamiento. Albareda-Sambola, Fernández, y Laporte (2007) [16], plantean una cota inferior y una heurística para la solución de un LRP estocástico, con varios depósitos con limitación de capacidad y un vehículo para cada depósito.

Dentro de los métodos de solución utilizados en el LRP, algunos autores desarrollan algoritmos híbridos. Es el caso de Prins y Prodhon (2008) [17], que resuelven un LRP con limitaciones de capacidad tanto en depósitos como en vehículos, y utilizan un algoritmo memético con gestión de la población, combinan técnicas de búsqueda local y gestión de la población.

Derbel, Jarboui, Hanafi y Cabchoub (2012) [18], desarrollan un algoritmo híbrido combinando un algoritmo genético (GA) con búsqueda local iterativa (ILS) para intensificar el espacio de búsqueda. Se da solución a un LRP con múltiples almacenes con limitaciones de capacidad y un vehículo por depósito con capacidad ilimitada. Dentro del ILS utilizan cuatro estructuras de vecindad. Este trabajo logra mejores soluciones en cuanto a costos, que los obtenidos por Albreda-Sambola (2005).

Escobar, Linfati y Toth (2012) [19], tratan el problema con capacidad limitada tanto en vehículos como en depósitos, CLRP, con demandas deterministas. El algoritmo propuesto combina la heurística de Lin y kemighan, con la heurística de búsqueda tabú. El algoritmo consta de dos fases, la primera es una fase de construcción y la segunda es una fase de mejora, en esta última interviene la búsqueda tabú.

2.1 FORMULACIÓN MATEMATICA

Existe un costo fijo F_j asociado con la apertura de un depósito en cada sitio potencial y un costo variable relacionado con las rutas que atienden la demanda de cada cliente. El objetivo es minimizar el costo total de localización de instalaciones y el de distribución. Cada cliente es asignado a un depósito que envía un solo vehículo para cumplir con su demanda. Una ruta de vehículo debe partir y terminar en el mismo depósito[59]. La capacidad de un depósito abierto también representa la capacidad del vehículo para la ruta asociada con el depósito[60].

En este trabajo se aborda el LRP considerando múltiples depósitos con limitaciones de capacidad y un solo vehículo asignado por depósito. La formulación matemática para la solución de este problema, está basada en Wu, Low, y Bai (2002) y Yu et al. (2010).

Sea $G = (V, E)$ un grafo no dirigido, donde V es un conjunto de nodos que comprenden un subconjunto J de m sitios candidatos de depósito y un subconjunto $I = V \setminus J$ de n clientes. E es un conjunto de aristas que conectan cada par de nodos en V.

Los conjuntos, parámetros, y variables utilizadas en el modelo matemático se definen a continuación[61]:

Conjuntos

- $\boldsymbol{I}$ Conjunto de todos los clientes
- $\boldsymbol{J}$ Conjunto de todos los sitios potenciales de depósito
- $\boldsymbol{K}$ Conjunto de rutas que equivale al total de depósitos seleccionados

Parámetros

- $\boldsymbol{N}$ Número de clientes
- $\boldsymbol{C_{ij}}$ Matriz de costos asociados a la distancia entres los puntos i, j. $i, j \in I \cup J$

[59] Lee W., Lin Y., Ting C. & YuV. (2010). A simulated annealing heuristic for the capacitated location routing problem. Computers & Industrial Engineering 58 (2010) 288–299.

[60] Albareda-Sambola. M., Díaz. J. & Fernández. E. A compact model and tight bounds for a combined location-routing problem. Computers & Operations Research 32 (2005)

[61] Bai J., Low C. & Wu T. Heuristic solutions to multi-depot location-routing problems. Computers & Operations Research 29 (2002)

F_j Costo fijo por instalación del depósito j
R_j Capacidad máxima del depósito j
d_i Demanda del cliente i

Variables de decisión

$x_{ijk} = 1$ Si el nodo i precede inmediatamente el nodo j en la ruta k , 0 en caso contrario. $(i, j \in I \cup J)$

$y_j = \begin{cases} 1, \\ 0, \end{cases}$ Si se selecciona el depósito j En caso contrario

$z_{ji} = \begin{cases} 1, \\ 0, \end{cases}$ Si el cliente i se asigna al depósito j En caso contrario

U_{lk}: Variable auxiliar para la eliminación de las restricciones del sub tour en la ruta k

Z: Costo total

En el modelo no se contemplan los costos variables asociados al uso de vehículos y almacenamiento en depósitos.

Función Objetivo

$$Min\ Z = \sum_{j \in J} F_j\, y_j + \sum_{i \in I \cup J} \sum_{j \in I \cup J} \sum_{k \in K} C_{ij} x_{ijk} \quad (1)$$

La función objetivo minimiza la suma del costo fijo por seleccionar un depósito y el costo de las rutas, que se define como la suma de los costos de cada viaje en la ruta.

Sujeto a:

$$\sum_{k \in K} \sum_{j \in I \cup J} x_{ijk} = 1, \quad i \in I \quad \textbf{(2)}$$

Cada cliente es visitado por exactamente una ruta.

$$U_{lk} - U_{ik} + N x_{lik} \leq N - 1, \quad l, i \in I,\ k \in K, \quad (3)$$

Conjunto de restricciones para la eliminación de sub-tour.

$$\sum_{i \in I \cup J} x_{jik} - \sum_{i \in I \cup J} x_{ijk} = 0, \quad k \in K,\ i \in I \cup J, \quad (4)$$

$$\sum_{j \in J} \sum_{i \in I} x_{jik} \leq 1, \quad k \in K, \quad (5)$$

Garantizan la continuidad de cada ruta. Cada ruta termina en el depósito donde inició.

$$\sum_{i \in I} d_i\, z_{ij} - R_j y_j \leq 0, \quad j \in J, \quad (6)$$

Restricción de capacidad para los depósitos.

$$\sum_{u \in I} x_{juk} + \sum_{u \in V \setminus \{i\}} x_{uik} \leq 1 + z_{ji}, \quad \forall j \in J,\ i \in I, k \in K, \quad (7)$$

Especifica que un cliente se puede asignar a un depósito sólo si se abre una ruta que los conecta.

$$x_{ijk} \in \{0,1\}, \quad \forall i \in I,\ j \in J,\ k \in K, \quad (8)$$

$$y_j \in \{0,1\}, \quad \forall j \in J, \quad \textbf{(9)}$$

$$z_{ij} \in \{0,1\}, \quad \forall i \in I, \quad j \in J, \quad \textbf{(10)}$$

Requisitos binarios en las variables de decisión.

$$U_{lk} \geq 0, \quad l \in I,\ k \in K, \quad \textbf{(11)}$$

La variable auxiliar U_{lk}, toma valores positivos.

3 ALGORITMO DE OPTIMIZACIÓN BASADO EN COLONIA DE HORMIGAS

La metaheurística de optimización basada en colonia de hormigas (ACO) se ha clasificado como parte de la computación evolutiva, más específicamente dentro de la inteligencia de enjambres y está inspirada a raíz del comportamiento de algunas especies de hormigas. Las hormigas reales encuentran rutas más cortas entre sus colonias y la fuente de alimento mediante la explotación de información de la feromona. ACO es un algoritmo en el que un conjunto de hormigas artificiales cooperan para la solución de un problema mediante el intercambio de información través de la feromona depositada en los bordes del grafo [22].

La metaheurística ACO presenta tres fases después de que se ha iniciado el algoritmo. Estas fases son: en cada iteración, las hormigas construyen un número de soluciones; estas soluciones se mejoran a través de una búsqueda local, y, finalmente, se actualiza la feromona [23].

En ACO, una hormiga artificial construye una solución al atravesar el grafo de construcción totalmente conectado G_C (V, E), donde V es un conjunto de vértices y E es un conjunto de aristas. Las hormigas artificiales se mueven de vértice a vértice a lo largo de los bordes del grafo, construyendo incrementalmente una solución

parcial. Las hormigas depositan una cierta cantidad de feromona, ya sea en los vértices o en los bordes que atraviesan. La cantidad de feromona depositada puede depender de la calidad de la solución encontrada [23]. Por lo tanto, el valor de la feromona puede aumentar, cuando las hormigas depositan la feromona en los componentes o conexiones que utilizan; o puede disminuir debido a la evaporación de feromonas [24]. Las hormigas posteriores utilizan la información de la feromona como una guía hacia las regiones prometedoras del espacio de búsqueda [23].

Dentro de la metaheurística ACO se han desarrollado diferentes algoritmos como el *Sistema de Hormigas*, el *Sistema Colonia de Hormigas*, el *Sistema de Hormigas Max-Min*, y el *Sistema Mejor-Peor Hormiga*, entre otros.

4 BÚSQUEDA LOCAL ITERATIVA

El ILS es una metaheurística que consiste en la aplicación iterativa de una heurística de búsqueda local (LS) y el uso de un criterio de perturbación como mecanismo que favorece la exploración, permitiendo así encontrar mejores soluciones. En cada iteración, se genera una nueva solución inicial empleando la heurística Búsqueda local (LS), y ésta se convierte en el nuevo punto de partida para la búsqueda. En cada iteración, se aplica perturbación a la solución encontrada hasta el momento, y su ejecución se detiene cuando alcanza un número máximo de iteraciones o cuando cumple un criterio de parada específico [25].

La búsqueda local es un método de optimización clásica que consiste en generar un óptimo local, mediante la exploración en los alrededores de una solución dada. Uno de los componentes principales en el diseño de una búsqueda local es la elección de estructuras de barrio, éstas se construyen mediante la modificación de algunos de los componentes de una solución dada para generar una nueva y evitar el estancamiento en óptimos locales [26].

El LS emplea cuatro estructuras de barrio denotadas N_1, N_2, N_3 y N_4, que tienen en cuenta únicamente los depósitos abiertos [26]: N_1 y N_2 se llevan a cabo entre dos rutas diferentes con un intento de mejorar la solución. En el barrio N_1, se realiza un movimiento de intercambio (*swap move*), se seleccionan al azar dos clientes asignados a dos depósitos diferentes y se intercambian, se modifican las rutas, pero no el número ni el orden.

En el barrio N_2 *(insertion move)*, se selecciona un cliente de una ruta y se inserta en otra ruta, se debe verificar que la capacidad del depósito donde estará el cliente pueda satisfacer su demanda.

N_3 y N_4 se llevan a cabo en las mismas rutas. Para el barrio N_3 *(swap move)*, se intercambian las posiciones de dos clientes y para el barrio N_4 *(insertion move)* se inserta un cliente entre otros dos de estos.

La perturbación es un factor clave del algoritmo ILS porque determina la porción de la solución localmente óptima que se modifica. Si la perturbación es demasiado pequeña, el algoritmo de ILS tiende a quedarse atrapado en una solución localmente óptima. Por otra parte, si la perturbación es demasiado grande, la información contenida dentro de la solución localmente óptima se pierde y el algoritmo ILS se comporta como si una solución inicial al azar se utilizara en cada iteración [27].

5 DESCRIPCIÓN DEL ALGORITMO HÍBRIDO

Para la solución del LRP se construye un algoritmo híbrido, que consta de dos etapas, en la primera etapa se encuentra soluciones por medio de la heurística sistema colonia de hormigas (ACS, por sus siglas en inglés), y en la segunda etapa la solución hallada se mejora con la heurística búsqueda local iterativa (ILS). El presente trabajo se basa inicialmente en los planteamientos de Thing y Chen (2013) [28].

La primera parte utiliza tres colonias de hormigas: la primera selecciona los depósitos que pueden ser abiertos, la segunda colonia asigna los clientes a los depósitos y la última colonia realiza el ruteo de vehículos. Las colonias trabajan de forma iterativa hasta cumplir un criterio de parada. En el presente trabajo, el criterio de parada será un número de iteraciones previamente establecido.

5.1 SELECCIÓN DE DEPÓSITOS

Para la selección de depósitos, se tiene una colonia con una cantidad b de hormigas, que debe ser la misma para la asignación de clientes y un número de iteraciones S.

En cada iteración s, inicialmente cada hormiga h, elige la cantidad de depósitos P_s^h que deben ser abiertos, dada por la ecuación (12):

$$P_s^h = \left[\frac{\sum_{i=1}^{n} d_i}{\sum_{j=1}^{n} \frac{R_j}{m}}\right] + U(1,r) \quad (12)$$

Donde, d_i es la demanda del cliente i, R_j es la capacidad del depósito j, m es el número de depósitos candidatos, $[x]$ es el menor entero mayor o igual a x, $U(1,r)$ es un número aleatorio (entero) que sigue una distribución uniforme en el intervalo $[1,r]$; y r es un número predeterminado que contribuye a definir la cantidad de depósitos que se desea abrir, la cantidad P_s^h puede ser mayor (probablemente), cuanto mayor sea el valor de r que se elige.

Se debe cumplir que $P_s^h \leq m$. Si esto no se cumple, P_s^h se hace igual a m.

Los P_s^h depósitos son escogidos por cada hormiga h sucesivamente de los m depósitos candidatos, de acuerdo a las reglas de selección que se muestran en las ecuaciones (13) y (14).

$$j = \begin{cases} \arg\max_{j \epsilon O_s^h} [\tau_j^s (\eta_j)^\alpha], & Si\ q \leq q_0 \\ J, & Caso\ contrario \end{cases} \quad (13)$$

Para cada cliente i, dentro de O_s^h se elige el depósito j que de mayor valor a la función $\tau_j^s (\eta_j)^\alpha$ si $q \leq q_0$. En caso contrario, los depósitos se eligen del conjunto O_s^h usando la distribución de probabilidad dada por la siguiente ecuación:

$$J: P_j^h = \frac{\tau_j^s (\eta_j)^\alpha}{\sum_{j \epsilon O_s^h} \tau_j^s (\eta_j)^\alpha} \quad (14)$$

Donde, $\boldsymbol{O_s^h}$ es el conjunto de depósitos candidatos que no se han seleccionado aún por la hormiga h en la iteración s, $\boldsymbol{\tau_j^s}$ es la cantidad de feromona en el depósito j en la iteración s, $\boldsymbol{\eta_j}$ es la relación entre capacidad (R_j) y el costo fijo por instalación (F_j) del depósito j:

$$\eta_j = \frac{R_j}{F_j} \quad (15)$$

$\boldsymbol{\alpha}$ es el parámetro que determina la influencia relativa de τ_j^s versus η_j $(\alpha > 0)$, $\boldsymbol{q}$ variable aleatoriamente que se distribuye uniformemente entre $[0,1]$ con una frecuencia $\boldsymbol{q_0}$ previamente definida.

La solución generada por cada hormiga, en cada iteración se representa en un conjunto llamado $\boldsymbol{W_s^h}$.

5.1.1 Regla de actualización local de feromona

Las hormigas en su recorrido van dejando rastros de feromona. Para el presente trabajo una hormiga artificial actualiza la feromona para la selección de depósitos mediante la siguiente regla:

$$\tau_j^s \leftarrow (1-\rho)\tau_j^s + \rho\tau_j^0 \quad si\ j \in T_h \quad (16)$$

Donde, $\boldsymbol{\tau_j^0}$ es el valor inicial de la feromona, dado por la ecuación:

$$\tau_j^0 = \frac{1}{F_j} \quad (17)$$

$\boldsymbol{T_h}$ es la solución construida por la hormiga$h$, $\boldsymbol{\rho}$ es el parámetro de actualización de la feromona $(\rho \geq 0)$.

5.2 ASIGNACION DE CLIENTES

Para la asignación de clientes a depósitos, se emplea otra colonia de hormigas con una cantidad b de hormigas (igual que para selección) que se encarga de asignar a cada cliente i un depósito k de los ya seleccionados $(i \neq k)$. Para cada conjunto W_s^h habrá una hormiga que haga la respectiva asignación.

Este proceso es uno a uno, es decir, cada cliente i se asigna a un depósito k (se denotan con k los depósitos que hacen parte de W_s^h), de acuerdo con la ecuación (18):

$$k = \begin{cases} \operatorname{argmax}_{k \epsilon W_s^h} [\xi_{ik}^s (\Psi_{ik}^s)^\beta], & Si\ q' \leq q_0' \\ K, & Caso\ contrario \end{cases} \quad (18)$$

Si $q' \leq q'_0$, el cliente i se asigna al depósito k que genere el mayor valor de la función $\xi_{ik}^{s}(\Psi_{ik}^{s})^{\beta}$, en caso contrario se elige el depósito k dentro del conjunto W_s^h usando la distribución de probabilidad inducida por la ecuación (19):

$$K: P_{ik}^{h} = \frac{\xi_{ik}^{s}(\psi_{ik}^{s})^{\beta}}{\sum_{k \epsilon W_s^h} \xi_{ik}^{s}(\psi_{ik}^{s})^{\beta}} \quad (19)$$

Donde, $\boldsymbol{W_s^h}$ es el conjunto de depósitos seleccionados por la hormiga h en la iteración s, $\boldsymbol{\xi_{ik}^s}$ es la cantidad de feromona entre el cliente i y el depósito k en la iteración s, $\boldsymbol{\beta}$ es el parámetro que determina el efecto relativo de ξ versus Ψ, $\boldsymbol{q'}$ es una variable aleatoria entre [0,1], $\boldsymbol{q'_0}$ es el parámetro que determina la importancia relativa de la ecuación de explotación (18) versus la ecuación de exploración (19).

Ψ_{ik}^{s} es el reciproco deD_{ik}, donde:

$$D_{ik} = \min_{l \epsilon A_k^{sh}} C_{il} \quad (20)$$

A_k^{sh} es el conjunto de nodos (incluyendo el depósito k) que han sido asignados al sitio k por la hormiga h en la iteración s, (inicialmente sólo el depósito k).

C_{il} es el costo asociado a la distancia entre los nodos i y l.

5.2.1 Regla de actualización local de feromona

La regla de actualización de feromona para la asignación de clientes, se realiza mediante la ecuación (10):

$$\xi_{ij}^{s} \leftarrow (1-\rho')\xi_{ij}^{s} + \rho'\,\xi_0 \;\; si\ el\ borde\ (i,j) \in T_h \quad (21)$$

Donde $\boldsymbol{T_h}$ es la solución construida por la hormiga h. $\boldsymbol{\rho'}$es el parámetro de evaporación de la feromona. $\boldsymbol{\xi_0}$ es el nivel inicial de feromona. Se puede tomar un número muy pequeño.

5.3 RUTEO DE VEHÍCULOS

La construcción de rutas de vehículos para cada depósito, se considera como un problema de ruteo de vehículos independiente para cada uno de los depósitos seleccionados y sus respectivos clientes asignados. Así, para cada depósito k, se tiene un valor de S' iteraciones y de b' que representa el número de hormigas que se emplean para el VRP de este depósito.

La hormiga h', parte desde el depósito a un nodo de manera aleatoria, se mueve desde este nodo i a un nodo v siguiendo la regla de construcción dada por la ecuación (22). Una vez visitados todos los nodos retorna al depósito correspondiente. Antes que una hormiga escoja el siguiente nodo a visitar, se genera un número aleatorio q".

$$v = \begin{cases} \underset{v \epsilon Z_{s'}^{h'}}{\operatorname{argmax}} \left[\zeta_{iv}^{s'}(\varphi_{iv}^{s'})^{\gamma}\right] & q" \leq q_0'' \\ V, & o\ contrario \end{cases} \quad (22)$$

Si $q'' \leq q_0''$ la hormiga h' se mueve del nodo i al nodo v que genere el mayor valor a la función $\zeta_{iv}^{s'}(\varphi_{iv}^{s'})^{\gamma}$, en caso contrario se elige el nodo dentro del conjunto $Z_{s'}^{h'}$ usando la distribución de probabilidad dada por la ecuación (23):

$$P_{iv}^{h'} = \frac{\zeta_{iv}^{s'}(\varphi_{iv}^{s'})^{\gamma}}{\sum_{v \epsilon Z_{s'}^{h'}} \zeta_{iv}^{s'}(\varphi_{iv}^{s'})^{\gamma}}, \quad (23)$$

Donde, $\boldsymbol{Z_{is'}^{h'}}$ es el conjunto de nodos aún no visitados aún por la hormiga h' en la iteración s', $\boldsymbol{\zeta_{iv}^{s'}}$ es la cantidad de feromona en la arista (i,v) en la iteración s', $\boldsymbol{\gamma}$ es el parámetro que denota la influencia relativa de la información de la feromona versus la información heurística ($\gamma > 0$), $\boldsymbol{q''}$ es una variable aleatoria dentro de una distribución [0,1]; y $\boldsymbol{q_0''}$ es el parámetro que determina la importancia relativa entre la ecuación de explotación y la ecuación de exploración.

$\boldsymbol{\varphi_{iv}^{s'}}$ es el ahorro en la combinación de dos nodos i,v en una excursión en lugar de servirlos en dos excursiones diferentes y es calculado con la ecuación (24):

$$\varphi_{iv}^{s'} = C_{io} + C_{0v} - C_{iv} \quad (24)$$

Donde, $\boldsymbol{C_{iv}}$ es la distancia entre los nodos i,v.

Nodo 0 denota el depósito seleccionado.

5.3.1 Regla de actualización local de feromona

Para el VRP, la regla de actualización local, está representada por la ecuación (25), esta se aplica inmediatamente después de que cada hormiga cruza el nodo (i,v) durante la construcción de la ruta:

$$\zeta_{iv}^{s'} \leftarrow (1-\rho'')\zeta_{iv}^{s'} + \rho''\zeta_0 \; si\; la\; arista\; (i,j) \in T_{h'} \quad (25)$$

Donde, $\boldsymbol{T_{h'}}$ es la ruta construida por la hormiga h', $\boldsymbol{\rho''}$ es el parámetro de evaporación de la feromona en un rango de $[0,1]$, $\boldsymbol{\zeta_0}$ es el nivel inicial en la matriz de feromona Se puede tomar un número muy pequeño.

5.3.2 Regla de actualización global para VRP

Para las mejores rutas, la mejor ruta global T_b' y la mejor ruta de la iteración $T_{s'}'$ del VRP, se les permite poner feromona en los bordes que les pertenecen. La regla de actualización global se define:

$$\zeta_{iv}^{s'+1} = (1-\rho'')\zeta_{iv}^{s'} + \rho''\Delta\zeta_{iv}^{s'} \quad (26)$$

Donde,

$$\Delta\zeta_{iv}^{s'} = \begin{cases} \frac{\left[(L_w' - L_b') + (L_w' - L_{s'}')\right]}{L_w'}, & Si\ \{(i,v) \in T_b'\ o\ T_{s'}'\} \\ 0, & Caso\ contrario \end{cases} \quad (27)$$

Donde $\boldsymbol{L_b'}$ y $\boldsymbol{L_{s'}'}$ definen el costo de la ruta de la mejor solución global y la mejor solución de la iteración del VRP, respectivamente. $\boldsymbol{L_w'}$ es el costo del recorrido de la peor solución de la iteración actual.

5.4 BÚSQUEDA LOCAL ITERATVA

Cuando se hace el VRP para todos los depósitos se genera la solución conjunta, es decir aquella que muestre selección, asignación y ruteo con su respectivo costo, de esta forma se elige la mejor solución de la iteración a la cual se le aplica ILS. Posterior a esto, se hace la actualización global de feromona para las colonias de selección y asignación.

La aplicación de ILS se hace siguiendo el marco general planteado por Derbel, Jarboui, Hanafi y Chabchoub (2010) [26]:

- T_s es la solución inicial para el proceso de ILS.
- Se aplica búsqueda local a T_s generando una mejor solución T_s'. En la búsqueda local se emplean las estructuras de barrio planteadas por Derbel, Jarboui, Hanafi y Chabchoub (2012) [26].
- Se hace perturbación a T_s' para obtener otra solución T_s''.
- Después de esto, se aplica de nuevo búsqueda local a T_s'' para obtener un nuevo óptimo local $T_s^{\sim}$.

5.5 ACTUALIZACIÓN GLOBAL DE FEROMONA

5.5.1 Regla de actualización global para la selección

La actualización global de feromona tanto para la selección como para la asignación de clientes, se usa teniendo en cuenta la mejor solución global T_b y la mejor solución de la iteración T_s.

La regla de actualización está dada por la siguiente ecuación:

$$\tau_j^{s+1} = (1-\rho)\tau_j^s + \rho\Delta\tau_j^s \quad (28)$$

Donde,

$$\Delta\tau_{j=}^{s} \begin{cases} [(l_w - l_b)] + [(l_w - l_s)] * \frac{n_j}{l_w} & si\ j \in T_b\ o\ T_s \\ 0, & Caso\ contrario \end{cases} \quad (29)$$

Donde, $\boldsymbol{l_b}\ y\ \boldsymbol{l_s}$ definen el costo total de la mejor solución global y la mejor solución de la iteración del LRP, y $\boldsymbol{l_w}$ costo total de la peor solución en la iteración actual, $\boldsymbol{n_j}$ es el número de clientes asignados a la ubicación j.

5.5.2 Regla de actualización global para la asignación de clientes

La regla de actualización está dada por la ecuación:

$$\xi_{ij}^{s+1} = (1-\rho')\xi_{ij}^{s} + \rho'\,\Delta\xi_{ij}^{s} \quad (30)$$

Donde,

$$\Delta\xi_{ij}^{s} = \begin{cases} [(l_w - l_b) + (l_w - l_s)]/l_w & Si\ \{j \in T_b\ o\ T_s\} \\ 0 & Caso\ contrario \end{cases} \quad (31)$$

Donde, $\boldsymbol{l_b}\ y\ \boldsymbol{l_s}$ definen el costo total de la mejor solución global y la mejor solución de la iteración del LRP, y $\boldsymbol{l_w}$ costo total de la peor solución en la iteración actual. Se definen de la misma manera que para la regla de actualización global para la selección de instalaciones.

5.6 REPRESENTACIÓN DE LA SOLUCIÓN

La representación de la solución para el LRP consiste en un vector y una matriz. El vector A, representa los depósitos seleccionados, la matriz V indica la asignación de clientes y su posición en la ruta de distribución.

Por ejemplo, si en la mejor solución encontrada por el algoritmo híbrido, son seleccionados 3 depósitos: $2{,}4\ y\ 6$; para atender 10 clientes, la solución está representada así:

$$A = \{2\ 4\ 6\}$$

$$V = \begin{cases} 3 & 1 & 9 \\ 5 & 4 & 7 \\ 2 & 8 & 10 \\ 0 & 0 & 6 \end{cases}$$

Donde el depósito $j = 2$ atiende los clientes $3{,}5\ y\ 2$ (en ese orden), $j = 4$ atiende los clientes $1, 4\ y\ 8$ y $j = 6$ atiende los clientes $9, 7, 10\ y\ 6$.

6 RESULTADOS

6.1 BANCO DE PRUEBAS

A través del banco de pruebas se crea el conjunto de instancias, donde, se dan valores específicos a los parámetros del problema, permitiendo así evaluar los resultados obtenidos. Se propone el siguiente banco de pruebas:

Tabla 1. Banco de pruebas

Instancia	Dep.	Cli	C
$\boldsymbol{m1n1r1ap\_1}$	5	30	1000x1000
$\boldsymbol{m2n1r1ap\_2}$	10	30	1000X1000
$\boldsymbol{m1n2r1ap\_3}$	5	50	1000x1000
$\boldsymbol{m2n2r1ap\_4}$	10	50	1000X1000
$\boldsymbol{m1n1r1ag\_5}$	5	30	Clusters
$\boldsymbol{m2n1r1ag\_6}$	10	30	Clusters
$\boldsymbol{m1n2r1ag\_7}$	5	50	Clusters
$\boldsymbol{m2n2r1ag\_8}$	10	50	Clusters

Para todos los casos se toma una relación demanda/capacidad=0,5, para esto, se generan valores de demanda aleatorios dentro de una distribución uniforme [500,1000]. C se refiere a la distribución de los clientes de forma aleatoria en un cuadro de 1000X1000 o en clusters, la fórmula que se emplea para el costo de instalación de depósitos es:

$$F_j = \left(1 + \frac{D}{R_j}\right) * \tilde{c} \qquad (32)$$

6.2 DISEÑO DE EXPERIMENTOS

Con el fin de estudiar el efecto de la variación de los parámetros del algoritmo híbrido ACO+ILS, sobre el valor de la función objetivo y establecer aquella configuración que lleve a mejores resultados del LRP, se emplea un diseño factorial fraccionado 2^{k-2}, que permite variar cada uno de los parámetros en dos niveles.

De los parámetros descritos, los valores de , α, β , γ, se toma el mismo valor para todos los procesos y se denota α. ρ, ρ', ρ'', se denota en general ρ. Sucede igual con S' y S , que se definen como $S.q_0$ representa el mismo valor para los parámetros: q_0, q_0', y q_0''. b indica la cantidad de hormigas tanto para VRP como para el proceso general. El valor de $r = 3$, como en los experimentos preliminares de Ting y Hochen (2012) de donde también se obtienen los niveles alto y bajo. Con esto, se tienen en cuenta 6 factores, y se aplica un diseño factorial fraccionado 2^{6-2} .

Tabla 2. Factores del diseño experimental

Factor	(+)	(-)
A:q_0	0,9	0,1
B:α	3	1
C:S	5	2
D:b	5	2
E:T	5	2
F: ρ	0,9	0,1

Los resultados del diseño experimental ye efectos estimados se pueden observar en el anexo A y B respectivamente.

El efecto de los factores A, B, C, D y E tiene un valor negativo, es decir que al pasar de un nivel bajo al nivel alto, el valor de la función objetivo disminuye, para el factor F, se observa este comportamiento en la mayoría de los casos. Dado que el objetivo es minimizar el costo asociado a la instalación de depósitos y ruteo de vehículos, todos los factores deben asumir un nivel alto.

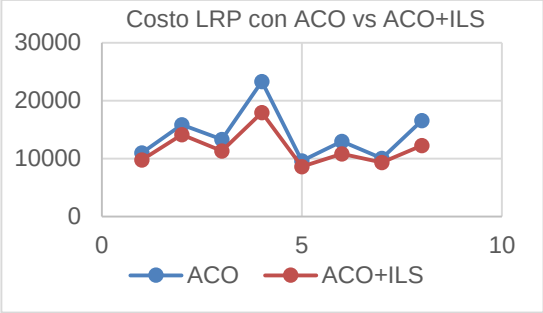
De acuerdo a los efectos estimados y los resultados del diseño experimental, se establecen los siguientes valores para los parámetros ACO+ILS. $q_0 = 0{,}9, q_0' = 0{,}9, q_0'' = 0{,}9, \alpha = 3, \beta = 3, \gamma = 3, S = 5, S' = 5, b = 5, b' = 5, t =$

$5, \rho = 0{,}9, \quad \rho' = 0{,}9\ y\ \rho'' = 0{,}9$. Para los parámetros que no tienen un efecto significativo, se toma sus valores en el nivel alto dado que en estos casos se observan mejores resultados.

6.3 COMPARACIÓN ACO vs ACO+ILS

Teniendo en cuenta la mejor combinación de factores encontrada con el diseño experimental, se evalúan las 8 instancias planeadas para ACO y ACO+ILS por separado, con el fin de analizar el comportamiento en cuanto a costo y tiempo computacional cuando se emplean las técnicas de búsqueda local y perturbación dadas por ILS al ACO. Esto se observa en el anexo C.

Ilustración 1. Comparación costo LRP con ACO vs ACO+ILS 8 instancias



En todos los casos, se evidencian disminuciones en el costo cuando se emplean técnicas de ILS, sin embargo, el aumento en el tiempo computacional no supera el 40%, con esto, el algoritmo híbrido propuesto asegura una mejora significativa en la disminución de costos de instalación y ruteo, sin afectar considerablemente el tiempo requerido, se puede decir entonces que se tiene un algoritmo que además de ser eficaz es eficiente. El buen desempeño se logra gracias al uso de las estructuras de vecindad y perturbación, empleadas por cada iteración, las cuales permiten explorar óptimos locales generados con el algoritmo ACO que a su vez emplea las mejores soluciones encontradas para hacer actualización global de feromona y retroalimentar el proceso.

7 CONCLUSIONES Y TRABAJOS FUTUROS

De acuerdo con los resultados del diseño experimental, el factor con mayor influencia sobre la función objetivo, es el parámetro q_o, que representa la importancia de las ecuaciones de exploración vs las de explotación, donde su valor más alto genera mejores resultados, es decir cuando aumenta la probabilidad de emplear las ecuaciones de explotación. Los parámetros α (relación de importancia entre información heurística y de feromona) y S (número de iteraciones) también representan alta influencia sobre la función objetivo. En general todos los factores arrojan mejores resultados en sus niveles, esto se evidencia en el signo negativo en la mayoría de los efectos.

Los resultados evidencian mayor rendimiento del algoritmo híbrido propuesto frente al algoritmo ACO tradicional con respuestas que generan una disminución del costo hasta en un 25,91%.

Para trabajos futuros se recomienda Emplear el algoritmo híbrido propuesto en las diferentes variaciones del problema conjunto de localización de instalaciones y ruteo de vehículos.Integrar las estructuras y perturbación basadas en la metaheurística ILS para cada ciclo u hormiga ‘b’ y comparar su desempeño con el algoritmo propuesto.

Realizar futuras investigaciones basadas en el presente trabajo, empleando el banco de pruebas propuesto y a su vez crear más bancos de pruebas que permitan aumentar las instancias para el benchmark.

REFERENCIAS

[1] Mohammaadi, M. (2013). Multi-objective invasive weed optimization for stochastic 211reen hub location routing problema with simultaneous pick-ups and deliveries.

[2] Yu, V. F., Lin, S.-W., Lee, W., & Ting, C.-J. (2010). A simulated annealing heuristic for the capacitated location routing problema. Computers & Industrial Engineering 58 (2010) 288–299

[3] Nagy, G., & Salhi, S. (2007). Location-routing: Issues, models and methods. European Journal of Operational Research, 177(2), 649–672. [Accessed 01 March 2014

[4] Escobar, J., Gatica, G., & Linfati, R. (2014). Un algoritmo metaheurístico para el problema de localización y ruteo con flota heterogénea. Ingeniería y Ciencia. vol. 10, no. 19, pp. 55–76,

[5] Berguer, R., Coullard, C. & Daskin, M. (2007). Location-Routing Problems with Distance Constraints. Transportation Science 41(1), pp. 29–43.

[6] Maranzana, F.E., 1964. On the location of supply points to minimise transport costs. Operational Research Quarterly 15

[7] Watson-Gandy, C.D.T. Depot location with van salesmen – a practical approach. Omega. [Online] Junio 1973.

[8] RAND, Graham K. SALHI Said. The effect of ignoring routes when locating depots. European Journal of Operational Research. [online] Marzo 1989.

[9] Albareda-sambola, Maria. Diaz, Juan. Fernández, Elena. A compact model and tight bounds for a combined location routing problem. Computers and Operations Research 32. [Online] 2005.

[10] Laporte, G., Nobert, Y. An exact algorithm for minimizing routing and operating costs in depot location. European Journal of Operational Research. [Online] Febrero de 1981.

[11] Laporte, G., Nobert, Y., Arpin, D., 1986. An exact algorithm for solving a capacitated location-routing problem. Annals of Operations Research 6. [Online] 1986.

[12] Z. Akca, R.T. Berger, and T.K. Ralphs. A Branch-and-Price Algorithm for Combined Location and Routing Problems Under Capacity Restrictions. Operations Research and Cyber-Infrastructure, Operations 309 Research/Computer Science Interfaces Series. 2009.

[13] Belenguer, J., Benavent, E., Prins, C., Prodhon, C., & Wolfler-Calvo, R. A branch-and-cut method for the capacitated location-routing problem. Computers and Operations Research. [Online], 2011. [Citado 28 de marzo de 2014].

[14] Karaoglan, Aliparmak, Dengizc, I., Karac, I. A branch and cut algorithm for the location-routing problem with simultaneous pickup and delivery. European Journal of Operational Research. [Online] 2011.

[15] WU, Tai-His. LOW, Chinyao. BAI Jiunn-Wei. Heuristic solutions to multi-depot location-routing problems. Computers and Operations. [Online] 2002.

[16] Albareda-Sambola, M., Fernández, E. & Laporte, G. Heuristic and lower bound for a stochastic location-routing problem. European Journal of Operational Research. [Online] 2007.

[17] Prins, C. & Prodhon, C. A memetic algorithm with population management (MAjPM)) for the periodic location-routing problem. In M. J. Blesa, C. Blum, G. Raidl, A. Roli, & M. Sampels (Eds.), Hybrid metaheuristics. Lecture notes in computer science. [Online] 2008.

[18] Chabchoub, H., Derbel, H., Jarboui, B. & Hanafi, S. An iterated local search for solving a location-routing problem. Electronic notes in discrete mathematics. [Online], 2011.

[19] Escobar, J., Linfati, R., & Toth, P. A two-phase hybrid heuristic algorithm for the capacitated location-routing problem. Computers and operations research. [Online], Junio 2012.

[20] Prins, C., Prodhon, C. & Calvo. R.W. Solving the capacitated location-routing problem by a GRASP complemented by a learning process and a path relinking.

[21] Arpin, D., Laporte, G. & Nobert. An exact algorithm for solving a capacitated location routing problem. Annids of Operations Research. [Online] 1986.

[22] Marco Dorigo y Luca Maria Gambardella (1997). Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem. Ieee transactions on evolutionary computation, vol. 1, no. 1, april 1997.

[23] Dorigo, M., Birattari, M. y Stützle, T. Ant Colony Otimization artificial ants as a computational intelligence technique. Ieee Computational Intelligence Magazine. November 2006.

[24] Dorigo, M. Stützle. Thomas. Ant Colony Optimization. Bradford Books. MIT Press, Cambridge, MA. [Online] 2004.

[25] Cuervo, D., Goos, P., Sörensen, K. & Arráiz, E. An iterated local search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research. p. 3. 2014.

[26] Derbel, H., Jarboui, B., Hanafi, S., & Chabchoub, H. Genetic algorithm with iterated local search for solving a location-routing problem. Expert systems with applications. p. 5-6. 2012.

[27] Cuervo, D., Goos, P., Sörensen, K. & Arráiz, E.(2014) An iterated local search algorithm for the vehicle routing problem with backhauls. European Journal of Operational Research. p. 5.
[28] Ching-Jung Ting, Chia-HoChen. A multiple ant colony optimization algorithm for the capacitated location routing problema. Int. J. Production Economics 141 (2013).

ANEXO A. RESULTADOS DEL DISEÑO EXPERIMENTAL

Factores						Valor promedio de la función objetivo							
A	B	C	D	E	F	$m1n1\ r1ap\_1$	$m2n1\ r1ap\_2$	$m1n2\ r1ap\_3$	$m2n2\ r1ap\_4$	$m1n1\ r1ag\_5$	$m2n1\ r1ag\_6$	$m1n2\ r1ag\_7$	$m2n2\ r1ag\_8$
-	-	-	-	-	-	14521	18611	18755	25216	12497	16771	20587	20952
+	-	-	-	+	-	11329	15202	13077	18531	9191	12058	12858	13867
-	+	-	-	+	+	11982	16429	13429	19675	10311	13025	15652	14218
+	+	-	-	-	+	11149	16239	13615	18921	9224	12009	13272	12846
-	-	+	-	+	+	13770	18975	16416	22590	12328	12492	18774	19119
+	-	+	-	-	+	10990	15248	13239	19460	9146	11711	11531	12905
-	+	+	-	-	-	10732	16429	13017	19784	10803	12507	13915	13380
+	+	+	-	+	-	10630	16251	12582	19201	8630	11562	12050	11924
-	-	-	+	-	+	12634	19395	18181	23390	12712	15538	18973	19193
+	-	-	+	+	+	11168	14488	13277	19263	8722	12004	11339	12605
-	+	-	+	+	-	12306	15742	13794	19489	9738	12770	14346	14508
+	+	-	+	-	-	11220	15866	12562	19974	8693	11581	12643	11807
-	-	+	+	+	-	13347	17207	16485	19093	10848	14016	18486	16922
+	-	+	+	-	-	11123	16397	12834	22023	9210	12431	11688	12145
-	+	+	+	-	+	12183	15799	12909	17916	9029	12203	13063	13214
+	+	+	+	+	+	9832	14334	12242	16913	8474	11447	11208	11673

ANEXO B. EFECTOS ESTIMADOS

Factores	Efecto estimado							
	$m1n1\ r1ap\_1$	$m2n1\ r1ap\_2$	$m1n2\ r1ap\_3$	$m2n2\ r1ap\_4$	$m1n1\ r1ag\_5$	$m2n1\ r1ag\_6$	$m1n2\ r1ag\_7$	$m2n2\ r1ag\_8$
A: q_0	-1754	-1821	-2445	-1608	-2122	-1814	-4651	-3967
B: alpha	-1106	-1016	-2264	-2212	-1219	-1240	-2261	-3017
C: S	-463	-128	-871	-935	-328	-923	-1119	-1089
D: b	-161	-520	-231	-665	-588	-18	-862	-893
E: t	-23	-669	-476	-1491	-384	-672	-120	-201
F: rho	-187	-100	25	-648	42	-408	-345	33

ANEXO C. COMPARACIÓN ACO vs ACO+ILS

INSTANCIAS	ACO		ACO+ILS		COMPARACIÓN	
	Costo	Tiempo(s)	Costo	Tiempo(s)	Δ Tiempo(s)	Δ Costo
$m1n1r1ap\_1$	10985	1204	9764	1654	450	1221
$m2n1r1ap\_2$	15814	994	14118	1374	380	1696
$m1n2r1ap\_3$	13289	3459	11293	4003	544	1996
$m2n2r1ap\_4$	23287	2463	17919	2884	421	5368

$m1n1r1ag\_5$	9597	1409	8577	1708	299	1020
$m2n1r1ag\_6$	12942	1086	10810	1198	112	2132
$m1n2r1ag\_7$	10033	3070	9306	3980	910	727
$m2n2r1ag\_8$	16570	2490	12276	2701	211	4294

ANEXO H RÉPLICAS DEL DISEÑO EXPERIMENTAL Y COMPARACIÓN ACO vs ACO-ILS

Tabla 30. Réplicas para la instancia 1

$m1n1r1ap\_1$									
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	14521	15683	13360	14521	14521, 25
+	-	-	-	+	-	11783	12236	10421	10876	11329
-	+	-	-	+	+	12462	11024	12941	11502	11982, 25
+	+	-	-	-	+	12487	10255	12041	9812	11148, 75
-	-	+	-	+	+	15423	12669	14872	12117	13770, 25
+	-	+	-	-	+	11870	9671	10110	12309	10990
-	+	+	-	-	-	12020	9874	11591	9444	10732, 25
+	+	+	-	+	-	11905	11481	9355	9780	10630, 25
-	-	-	+	-	+	14151	11624	13645	11116	12634
+	-	-	+	+	+	10275	12063	9823	12509	11167, 5
-	+	-	+	+	-	13783	11322	13291	10829	12306, 25
+	+	-	+	-	-	10772	10320	12118	11669	11219, 75
-	-	+	+	+	-	12814	12280	14414	13881	13347, 25
+	-	+	+	-	-	10679	10013	12230	11568	11122, 5
-	+	+	+	-	+	11695	11209	13158	12671	12183, 25
+	+	+	+	+	+	9439	9046	10619	10222	9831, 5

Tabla 31. Réplicas para la instancia 2

$m2n1r1ap\_2$									
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	17867	17123	20100	19354	18611
+	-	-	-	+	-	14419	13986	16593	15811	15202, 25
-	+	-	-	+	+	15772	15087	17743	17115	16429, 25
+	+	-	-	-	+	15590	14940	17536	16889	16238, 75
-	-	+	-	+	+	18216	17457	20493	19734	18975
+	-	+	-	-	+	14639	14029	16468	15856	15248
-	+	+	-	-	-	15772	15112	17744	17087	16428, 75

+	+	+	-	+	-	15598	14951	17552	16902	16250, 75
-	-	-	+	-	+	20171	18620	20947	17840	19394, 5
+	-	-	+	+	+	15068	13909	15648	13327	14488
-	+	-	+	+	-	16372	17002	14482	15110	15741, 5
+	+	-	+	-	-	16501	15230	17136	14597	15866
-	-	+	+	+	-	17896	16519	18580	15831	17206, 5
+	-	+	+	-	-	15086	17709	15740	17053	16397
-	+	+	+	-	+	16431	15168	17063	14534	15799
+	+	+	+	+	+	14908	13761	15481	13184	14333, 5

Tabla 32. Réplicas para la instancia 3

$m1n2r1ap\_3$										
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	19506	18005	20254	17255	18755
+	-	-	-	+	-	13601	12554	14124	12027	13076, 5
-	+	-	-	+	+	13967	12892	14500	12355	13428, 5
+	+	-	-	-	+	13071	14159	14705	12526	13615, 25
-	-	+	-	+	+	15760	17073	17730	15100	16415, 75
+	-	+	-	-	+	12710	13769	11651	14825	13238, 75
-	+	+	-	-	-	12497	13538	11452	14580	13016, 75
+	+	+	-	+	-	12079	13086	11073	14091	12582, 25
-	-	-	+	-	+	17454	18909	16000	20360	18180, 75
+	-	-	+	+	+	12746	13809	11683	14871	13277, 25
-	+	-	+	+	-	14346	15450	13240	12139	13793, 75
+	+	-	+	-	-	14069	11055	13065	12060	12562, 25
-	-	+	+	+	-	16485	16485	14506	18464	16485
+	-	+	+	-	-	12834	12834	11294	14375	12834, 25
-	+	+	+	-	+	13942	11877	11359	14459	12909, 25
+	+	+	+	+	+	13222	11263	10770	13712	12241, 75

Tabla 33. Réplicas para la instancia 4

$m2n2r1ap\_4$									
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	26225	23197	27234	24208	25216
+	-	-	-	+	-	17790	20013	17049	19273	18531, 25
-	+	-	-	+	+	20460	18101	21249	18888	19674, 5
+	+	-	-	-	+	19675	17408	20435	18165	18920, 75
-	-	+	-	+	+	21687	24398	20783	23491	22589, 75
+	-	+	-	-	+	21017	18682	17900	20239	19459, 5
-	+	+	-	-	-	20576	18202	21365	18993	19784
+	+	+	-	+	-	19970	17662	20738	18433	19200, 75
-	-	-	+	-	+	24326	21519	25260	22455	23390
+	-	-	+	+	+	17720	20805	18493	20034	19263
-	+	-	+	+	-	20265	18710	21049	17930	19488, 5
+	+	-	+	-	-	20773	19176	21568	18377	19973, 5
-	-	+	+	+	-	19857	18330	20619	17566	19093
+	-	+	+	-	-	22901	21143	23785	20262	22022, 75
-	+	+	+	-	+	18633	17200	19347	16483	17915, 75
+	+	+	+	+	+	17590	16237	18267	15559	16913, 25

Tabla 34. Réplicas para la instancia 5

$m1n1r1ag\_5$										
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	12997	11498	13497	11995	12496, 75
+	-	-	-	+	-	8824	9926	8459	9556	9191, 25
-	+	-	-	+	+	10724	9486	11136	9899	10311, 25
+	+	-	-	-	+	9962	8856	8485	9593	9224
-	-	+	-	+	+	12822	11342	13314	11835	12328, 25
+	-	+	-	-	+	8781	9510	8415	9878	9146
-	+	+	-	-	-	11236	9939	11370	10668	10803, 25
+	+	+	-	+	-	8976	7921	9339	8285	8630, 25
-	-	-	+	-	+	13220	11695	13729	12203	12711, 75
+	-	-	+	+	+	9070	8025	9420	8374	8722, 25
-	+	-	+	+	-	10127	8959	10518	9349	9738, 25
+	+	-	+	-	-	9041	8344	9389	7998	8693
-	-	+	+	+	-	11280	9981	11716	10415	10848

+	-	+	+	-	-	9579	8473	9947	8842	9210, 25
-	+	+	+	-	+	8668	9751	8306	9390	9028, 75
+	+	+	+	+	+	8813	7796	9152	8136	8474, 25

Tabla 35. Réplicas para la instancia 6

$m2n1r1ag\_6$										
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	17442	15430	18113	16100	16771, 25
+	-	-	-	+	-	12541	11094	13022	11576	12058, 25
-	+	-	-	+	+	13546	11983	14066	12504	13024, 75
+	+	-	-	-	+	12490	11049	12970	11528	12009, 25
-	-	+	-	+	+	12992	11493	13490	11993	12492
+	-	+	-	-	+	12180	10775	12647	11243	11711, 25
-	+	+	-	-	-	13008	11507	13506	12007	12507
+	+	+	-	+	-	12025	10637	12487	11100	11562, 25
-	-	-	+	-	+	16160	14295	16782	14916	15538, 25
+	-	-	+	+	+	12485	11044	12965	11522	12004
-	+	-	+	+	-	12259	13792	11749	13281	12770, 25
+	+	-	+	-	-	12045	10655	12507	11118	11581, 25
-	-	+	+	+	-	14577	12895	15138	13455	14016, 25
+	-	+	+	-	-	12929	11437	13425	11932	12430, 75
-	+	+	+	-	+	12692	11225	13180	11715	12203
+	+	+	+	+	+	10990	12362	10532	11905	11447, 25

Tabla 36. Réplicas para la instancia 7

$m1n2r1ag\_7$										
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	21411	18940	22234	19764	20587, 25
+	-	-	-	+	-	13373	11830	13886	12344	12858, 25
-	+	-	-	+	+	15026	16905	14400	16278	15652, 25
+	+	-	-	-	+	13803	12211	14334	12740	13272
-	-	+	-	+	+	19525	17271	20276	18022	18773, 5
+	-	+	-	-	+	11993	10609	12450	11070	11530, 5
-	+	+	-	-	-	14470	12802	15029	13359	13915
+	+	+	-	+	-	12530	11086	13014	11568	12049, 5
-	-	-	+	-	+	18213	20491	17456	19732	18973

+	-	-	+	+	+	11793	10430	12247	10886	11339
-	+	-	+	+	-	14920	13199	15492	13773	14346
+	+	-	+	-	-	13149	11632	13654	12138	12643, 25
-	-	+	+	+	-	19226	17008	19964	17747	18486, 25
+	-	+	+	-	-	12156	10752	12624	11221	11688, 25
-	+	+	+	-	+	13586	12017	14109	12541	13063, 25
+	+	+	+	+	+	11657	10312	10760	12103	11208

Tabla 37. Réplicas para la instancia 8

$m2n2r1ag\_8$										
A	**B**	**C**	**D**	**E**	**F**	Réplica 1	Réplica 2	Réplica 3	Réplica 4	Promedio
-	-	-	-	-	-	21276	19791	22114	20628	20952, 25
+	-	-	-	+	-	13313	14977	12758	14420	13867
-	+	-	-	+	+	14787	13081	15356	13647	14217, 75
+	+	-	-	-	+	12333	13874	11819	13358	12846
-	-	+	-	+	+	19884	17590	20647	18355	19119
+	-	+	-	-	+	13422	11870	12389	13938	12904, 75
-	+	+	-	-	-	13916	12310	12285	14450	13240, 25
+	+	+	-	+	-	11448	12878	10971	12400	11924, 25
-	-	-	+	-	+	19960	17658	20729	18426	19193, 25
+	-	-	+	+	+	13110	11597	13614	12100	12605, 25
-	+	-	+	+	-	15668	13928	13348	15089	14508, 25
+	+	-	+	-	-	12279	10863	12752	11335	11807, 25
-	-	+	+	+	-	17599	15567	18276	16246	16922
+	-	+	+	-	-	11660	13115	11174	12631	12145
-	+	+	+	-	+	12684	14272	12157	13743	13214
+	+	+	+	+	+	12140	10740	12607	11206	11673, 25

ACO+ILS								
	Rép. 1	Rép. 2	Rép. 3	Costo Prom	Rép. 1	Rép. 2	Rép. 3	Tiempo (s) Prom
Ins1	9767	9828	9697	9764	1605	1670	1688	1654, 33333
Ins2	13975	13800	14578	14117, 6667	1260	1402	1461	1374, 33333
Ins3	11178	11417	11284	11293	3963	3560	4485	4002, 66667
Ins4	17740	17278	18739	17919	2942	2854	2857	2884, 33333
Ins5	8390	8580	8760	8576, 66667	1691	1700	1733	1708
Ins6	11026	10690	10714	10810	1176	1190	1228	1198
Ins7	9103	9495	9320	9306	3910	3975	4054	3979, 66667
Ins8	12060	12270	12499	12276, 3333	2670	2680	2752	2700, 66667

Tabla 38. Pruebas para ACO

ACO								
	Rép. 1	Rép. 2	Rép. 3	Costo Prom	Rép. 1	Rép. 2	Rép. 3	Tiempo Prom(s)
Ins1	10544	11534	10877	10985	1128	1290	1193	1203, 66667
Ins2	15655	15658	16129	15814	984	995	1003	994
Ins3	13554	13156	13156	13288, 6667	3424	3475	3478	3459
Ins4	23452	23354	23054	23286, 6667	2412	2540	2436	2462, 66667
Ins5	9601	9689	9500	9596, 66667	1395	1490	1342	1409
Ins6	12812	13589	12424	12941, 6667	1100	1080	1077	1085, 66667
Ins7	9933	10032	10134	10033	3125	3044	3040	3069, 66667
Ins8	16500	16610	16601	16570, 3333	2401	2520	2550	2490, 33333

Tabla 39. Pruebas para ACO+ILS

ACO								
	Rép. 1	Rép. 2	Rép. 3	Costo Prom	Rép. 1	Rép. 2	Rép. 3	Tiempo Prom
Ins1	10544	11534	10877	10985	1128	1290	1193	1203, 66667
Ins2	15655	15658	16129	15814	984	995	1003	994
Ins3	13554	13156	13156	13288, 6667	3424	3475	3478	3459
Ins4	23452	23354	23054	23286, 6667	2412	2540	2436	2462, 66667
Ins5	9601	9689	9500	9596, 66667	1395	1490	1342	1409
Ins6	12812	13589	12424	12941, 6667	1100	1080	1077	1085, 66667
Ins7	9933	10032	10134	10033	3125	3044	3040	3069, 66667
Ins8	16500	16610	16601	16570, 3333	2401	2520	2550	2490, 33333