UN MÉTODO EVOLUTIVO DE COLONIA DE HORMIGAS PARA LA SOLUCIÓN DEL PROBLEMA DE RUTEO DE VEHÍCULOS CON DEMANDAS ESTOCÁSTICAS

FRANCIA DEL PILAR ACOSTA NIÑO
DANIELA ANGÉLICA OSORIO OSORIO

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍA FÍSICO MECÁNICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2013

UN MÉTODO EVOLUTIVO DE COLONIA DE HORMIGAS PARA LA SOLUCIÓN DEL PROBLEMA DE RUTEO DE VEHÍCULOS CON DEMANDAS ESTOCÁSTICAS

FRANCIA DEL PILAR ACOSTA NIÑO
DANIELA ANGÉLICA OSORIO OSORIO

Trabajo de grado para optar el título de Ingeniero Industrial

Director:
HENRY LAMOS DÍAZ
PhD. En Física-matemática

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍA FÍSICO MECÁNICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2013

AGRADECIMIENTOS

Al Doctor Henry Lamos Díaz por depositar su confianza en nosotras, orientarnos y apoyarnos en el desarrollo de este proyecto.

A la ingeniera Silvia Galván Nuñez por las asesorías brindadas que sirvieron de base para la realización de esta investigación.

A la Universidad Industrial de Santander, Escuela de Estudios Industriales y Empresariales y grupo de investigación OPALO por la formación académica y respaldo brindado.

DEDICATORIA

A Dios, por brindarme sabiduría y fortaleza.

A mis padres y hermana, por su esfuerzo, amor y apoyo incondicional.

A mi familia y amigos, por los buenos momentos y las palabras de aliento en los tiempos difíciles.

A Edinsson Castro, por su amor y comprensión.

Francia del Pilar Acosta Niño

A Dios por regalarme la sabiduría y fortaleza para alcanzar este logro.

A mis padres y hermanos por su amor, confianza y apoyo constante.

A mis tías Cruz María Osorio y Carmen Osorio por sus sabios consejos y respaldo incondicional.

A mis familiares y amigos que me bridaron su compañía y apoyo en el transcurso de mi carrera.

Daniela Angélica Osorio Osorio

TABLA DE CONTENIDO

Pág.

INTRODUCCIÓN 17

1 OBJETIVOS 21

1.1 OBJETIVO GENERAL 21

1.2 OBJETIVOS ESPECÍFICOS 21

2 MARCO TEÓRICO 22

2.1 LOGÍSTICA DE DISTRIBUCIÓN 22

2.2 PROBLEMAS DE OPTIMIZACIÓN COMBINATORIA 24

2.2.1 Definición de problemas de optimización combinatoria 24

2.2.2 Complejidad computacional 26

2.2.3 Métodos de solución de un problema de optimización combinatoria 27

2.3 PROBLEMA DE RUTEO DE VEHÍCULOS 35

2.3.1 Definición de un problema de ruteo de vehículos 36

2.3.2 Modelación matemática del VRP 37

2.3.3 Variantes del VRP 38

2.3.4 Métodos de solución del VRP 41

2.4 PROBLEMA DE RUTEO DE VEHÍCULOS ESTOCÁSTICO 49

2.4.1 Variantes del SVRP 50

2.5 PROBLEMA DE RUTEO DE VEHÍCULOS CON DEMANDAS ESTOCÁSTICAS (VRPSD) 53

2.5.1 Definición del VRPSD 53

2.5.2 Políticas de servicio....54
2.5.3 Métodos de solución del VRPSD....58
3 OPTIMIZACIÓN COLONIA DE HORMIGAS....68
3.1 VARIANTES DEL ACO....68
3.1.1 Sistema de hormigas....68
3.1.2 Sistema colonia de hormigas....70
3.1.3 Sistema de hormiga max-min....73
4 ALGORITMOS GENÉTICOS....75
4.1 OPERADORES GENÉTICOS....76
4.1.1 Selección....76
4.1.2 Operador cruce- crossover....77
4.1.2.2 Cruce en dos puntos.....78
4.1.3 Mutación....80
5 PROCEDIMIENTOS DE BÚSQUEDA LOCAL....81
5.1 PROCEDIMIENTO **k**- INTERCAMBIO....82
5.2 PROCEDIMIENTO $\mathbf{Or-Opt}$....83
6 DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN....85
6.1 DEFINICIÓN DEL VRPSD....85
6.2 DEFINICIÓN DE LA POLÍTICA DE SERVICIO....86
6.3 MODELACIÓN MATEMÁTICA DEL VRPSD....87
6.4 CÁLCULO DE LA FUNCIÓN OBJETIVO....89
6.5 DEFINICIÓN DEL NÚMERO DE ESCENARIOS DE DEMANDA....91
6.6 INSTANCIAS....92

7 MÉTODO EVOLUTIVO DE COLONIA DE HORMIGAS PARA LA SOLUCIÓN DEL VRPSD....96
7.1 INICIALIZACIÓN DE PARÁMETROS....96
7.2 CONSTRUCCIÓN DE SOLUCIONES....97
7.3 MEJORAR LAS SOLUCIONES....98
7.3.1 Operador de mutación....98
7.3.2 Heurística de intercambio 2-Opt....100
7.4 COMPARAR Y CONTRASTAR LAS SOLUCIONES GENERADAS....101
7.5 ACTUALIZACIÓN DE FEROMONA....101
8 DISEÑO DE EXPERIMENTOS....103
8.1 ANÁLISIS DE PARÁMETROS DEL MÉTODO EACO....103
8.2 RESULTADOS DEL DISEÑO EXPERIMENTAL....104
8.3 ANÁLISIS DEL DISEÑO EXPERIMENTAL....105
8.4 COMPARACIÓN DEL RENDIMIENTO DEL ALGORITMO....108
9 CONCLUSIONES....110
10 RECOMENDACIONES....112
BIBLIOGRAFÍA....114
ANEXOS....123

LISTA DE FIGURAS

Pág.

Figura 2. Subsistemas que conforman la logística.....23
Figura 3. Clasificación de Metaheurísticas 30
Figura 4. Ejemplo del VRP con 1 depósito, 8 clientes y 2 vehículos.....36
Figura 5. Métodos de solución del VRP.....41
Figura 6. Variantes del SVRP 50
Figura 7. Cruce en un punto 78
Figura 8. Cruce en dos puntos.....78
Figura 9. Cruce uniforme 79
Figura 10. Procedimiento $\mathbf{2-Opt}$ 82
Figura 11. Procedimiento $\mathbf{3-Opt}$ 83
Figura 12. Modificaciones de una secuencia inicial aplicando Or-Opt.....84
Figura 13. Ejemplo del cálculo de la función objetivo 90
Figura 14. Costo promedio vs número de escenarios para una ruta de 100 clientes 91
Figura 15. Costo promedio vs número de escenarios para una ruta de 100 clientes 92
Figura 16. Ejemplo mutación flip.....99
Figura 17. Ejemplo mutación Swap 99
Figura 18. Ejemplo mutación Slide 99
Figura 19. Diagrama de Pareto Instancia 1 105
Figura 20. Diagrama de Pareto Instancia 2 106
Figura 21. Diagrama de Pareto Instancia 3 106
Figura 22. Diagrama de Pareto Instancia 4 106
Figura 23. Diagrama de Pareto Instancia 5 106
Figura 24. Diagrama de Pareto Instancia 6 107
Figura 25. Diagrama de Pareto Instancia 7 107
Figura 26. Diagrama de Pareto Instancia 8 107

LISTA DE TABLAS

Pág.

Tabla 1.Factores y niveles 94
Tabla 2. Banco de pruebas 95
Tabla 3.Configuración de los valores de los niveles de cada factor 104
Tabla 4. Parámetros del diseño experimental 104
Tabla 5. Resultado del diseño experimental 104
Tabla 6. Efectos estimados para la función objetivo 105
Tabla 7. Resultados del valor de la función objetivo AS-EACO 108

LISTA DE ANEXOS

Pág.

ANEXO A. Descripción de variables para cada función desarrollada del método EACO.....123
ANEXO B. Código de programación método EACO.....135

RESUMEN

TÍTULO:
"UN MÉTODO EVOLUTIVO DE COLONIA DE HORMIGAS PARA LA SOLUCIÓN DEL PROBLEMA DE RUTEO DE VEHÍCULOS CON DEMANDAS ESTOCÁSTICAS"[1]

AUTORES:
ACOSTA NIÑO, Francia del Pilar
OSORIO OSORIO, Daniela Angélica[2]

PALABRAS CLAVE:
VRP con Demandas Estocásticas, Método Evolutivo ACO.

DESCRIPCIÓN:

En el presente trabajo de investigación, se desarrolla un Método Evolutivo de Colonia de Hormigas (EACO) para resolver el Problema de Ruteo de Vehículos con Demandas Estocásticas (VRPSD) con reabastecimiento preventivo para el caso de un solo vehículo. El método propuesto se fundamenta principalmente en el algoritmo Sistema de Hormigas tradicional – AS, desarrollado por Dorigo et al., en el año 1996. Además, se utiliza la mutación, un operador evolutivo, para ampliar la exploración de los agentes (hormigas artificiales) hacia nuevas zonas del espacio de búsqueda y, una heurística de búsqueda local, intercambio 2-Opt, con el fin de explotar las mejores soluciones.

Para estudiar el efecto de la variación de los parámetros del algoritmo Evolutivo de Colonia de Hormigas en el valor de la función objetivo, y establecer la configuración que lleve a mejores soluciones del Problema de Ruteo de Vehículos con Demandas Estocásticas, se realizó un diseño experimental. Se utilizó un banco de pruebas previamente definido para evaluar la calidad de las soluciones obtenidas por el algoritmo EACO, las cuales fueron comparadas con la metaheurística Sistema de Hormigas tradicional. Los resultados revelan que el método propuesto supera el rendimiento del algoritmo Sistema de hormigas tradicional en las ocho instancias comparadas.

[1] Proyecto de Grado.
[2] Facultad de Ingenierías Físico mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Ph.D. Henry Lamos Díaz.

ABSTRACT

TITLE:
"AN EVOLUTIONARY METHOD BASED ON ANT COLONY OPTIMIZATION FOR SOLVING VEHICLE ROUTING PROBLEM WITH STOCHASTIC DEMANDS"[3]

AUTHORS:
ACOSTA NIÑO, Francia del Pilar
OSORIO OSORIO, Daniela Angélica[4]

KEY WORDS:
VRP with stochastic demands, Evolutionary Method ACO.

DESCRIPTION:

In the current research project, an Evolutionary Method based on Ant Colony Optimization (EACO) is developed in order to solve the Single Vehicle Routing Problem with Stochastic Demands (VRPSD) with preventive restocking. The proposed method is principally based on the traditional Ant system algorithm, developed by Dorigo et al. in 1996. Moreover, It uses mutation, evolutionary operator, to expand the exploration of agents (artificial ants) towards new areas of the search space, and a local search heuristics, exchange 2-Opt, in order to exploit the best solutions.

An experimental design was developed to the effect of the variation of Evolutionary Method based on Ant Colony Optimization Parameters in the objective function value and set the configuration that may lead to better solutions for Vehicle Routing Problem with Stochastic Demands. A previously defined test was employed to evaluate the quality of the solutions obtained by the Evolutionary Method based on Ant Colony Optimization, which were compared with the traditional metaheuristic Ant Sstem. The results show that the proposed method outperforms the traditional ant system algorithm in the eight instances compared.

[3] Proyecto de Grado.
[4] Facultad de Ingenierías Físico mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Ph.D. Henry Lamos Díaz.

INTRODUCCIÓN

Diseñar eficazmente la red de distribución de un sistema logístico es uno de los retos a los cuales se enfrentan las organizaciones para lograr competitividad mediante la eficiencia del servicio al cliente y la reducción de "costos logísticos relacionados con el costo de transporte, que representa cerca del 50% de estos"[5]. Para ello las empresas comerciales y manufactureras han tenido que plantearse estrategias para modelar eficazmente la red de distribución. Sin embargo factores externos como la infraestructura logística de un país tiene un alto impacto en los costos logísticos.

En Colombia "las principales desventajas para el sector de transporte de carga radican en la carencia y deficiencia de la infraestructura vial y logística"[6], así lo señala el presidente de Colfecar. Ante esta problemática se han realizado "acuerdos comerciales con diversos países, en especial con Estados Unidos y la Unión Europea, motivando la puesta en marcha del desarrollo de los sectores transportador, logístico y de infraestructura"[7]; al mismo tiempo, en "el Plan Nacional de Desarrollo 2010-2014, se plantea la provisión de infraestructura bajo criterios de conectividad, accesibilidad e intermodalidad para soportar la apuesta de aumentar la competitiva nacional, debido a la participación positiva registrada por la rama de transporte, almacenamiento y comunicación en el PIB total de 2011"[8].

---

[5] R.H. Ballou. Business Logistics/Supply Chain Management.5° ed. Pretince Hall.Upper Saddle River.New Jersey. 2004. p.14. Citapo por: CORREA, Andrés Felipe, et al. Características del diseño de rutas de distribución de alimentos en el Valle de Aburrá. En: Revista Facultad de Ingeniería de la Universidad de Antioquia. Septiembre, 2008. No 45. p. 173.

[6] Revista de logística [en línea]. LEGIS. Mayo-Julio 2012. [citado: 14 Mayo 2013]. Disponible en internet: http://www.revistadelogistica.com/transporte-de-carga-una-cuantia-aun-a-medio-pagar.asp

[7] Ibid.

8 Informe al Congreso. [en línea].Presidencia de la Republica de Colombia, Departamento de Planeación. 2012. p. 54. [citado: 14 Mayo 2013]. Disponible en internet: https://sinergia.dnp.gov.co/sinergia/Documentos/Informe_Senor_Presidente_al_Congreso_de_la_Rep%C3%BAblica_2012.pdf.

En cuanto a la disponibilidad y uso de software para el diseño de rutas, según el estudio realizado en la universidad EAFIT, "se evidencia que de la totalidad de operadores logísticos y empresas transportadoras consultados, ninguna ofrece herramientas de software de diseño de ruta y menos del 10% utilizan tales herramientas. Este hecho refleja la situación de las operaciones de diseño de rutas en Colombia, y constituye una oportunidad de investigación y de mejoramiento para el sistema de transporte en general"[9]. Herramientas computacionales que sirven de soporte en la elaboración de rutas son una opción dentro del proceso de búsqueda de ventajas competitivas en las organizaciones.

El problema de ruteo de vehículos (*Vehicle Routing Problem,* VRP), es un problema dentro de la cadena de abastecimiento, en el cual los parámetros de entrada como: demanda de los clientes, tiempo de entrega, tiempo de viaje del vehículo, son de naturaleza determinística. Sin embargo, el comportamiento estocástico de algunos de estos parámetros genera variantes conocidas como problemas de ruteo de vehículos estocásticos (*Stochastic Vehicle Routing Problem*, SVRP).

Diferentes situaciones empresariales o comerciales que surgen en la logística de distribución, donde la demanda de los clientes sólo se conoce al llegar a la ubicación de éstos se han modelado como un problema de ruteo de vehículos con demandas estocásticas (*Vehicle Routing Problem with Stochastic Demand*, VRPSD). El VRPSD es "un problema de optimización combinatoria debido a que las variables son de naturaleza discreta y se relacionan entre sí de diferentes maneras, donde cada configuración obtenida es una posible solución al problema"[10].

9 GUTIÉRREZ, Valentina; PALACIO, Juan David y VILLEGAS, Juan Guillermo. Reseña del software disponible en Colombia para el diseño de rutas de distribución y servicios. En: Revista Universidad EAFIT. 2007. vol, 43. no, 145. p, 18.

[10]TRIPATHI, Mukul; KURIGEr, Glenn y WAN, Hung-da. An ant based simulation optimization for vehicle routing problem with stochastic demands. En: Actas de la Conferencia de Simulación de Invierno. San Antonio: Universidad de Texas. 2009. P, 2479.

La solución de un problema combinatorio requiere en muchos casos un tiempo computacional no polinomial; “el VRPSD se clasifica como un problema NP-hard”[11]. Con el propósito de obtener soluciones cercanas al óptimo global en tiempos computacionales razonables, se han desarrollado diferentes metaheurísticas como la Optimización Colonia de Hormigas (*Ant Colony Optimization*, ACO), que se basa en el comportamiento de las hormigas que “trabajan de manera cooperativa y se comunican mediante rastros de feromonas”[12] buscando el camino más corto.

La investigación se enfoca en el uso de la metaheurística ACO y operadores evolutivos que aceleran la convergencia del algoritmo para la solución del VRPSD. La validación de la eficacia del modelo propuesto, se hará a través de un conjunto de instancias definidas en la propuesta de investigación “Optimización por simulación basado en un sistema evolutivo de optimización de enjambre de partículas para el problema de ruteo de vehículos con demandas estocásticas” presentado en el 2012 por la ingeniería Silvia Adriana Galván Núñez[13].

En el presente documento se exponen aspectos teóricos de la optimización combinatoria, del problema de ruteo de vehículo con demandas estocásticas, de los algoritmos basados en la metaheurísticas Optimización Colonia de Hormigas y operadores evolutivos. Se elabora un algoritmo EACO para la solución del VRPSD, con el propósito de mejorar la calidad de la solución. Por último se publicara los análisis, resultados y conclusiones del tema estudiado en un artículo académico.

[11]MARINAKIS, Yannis; LORDANIDOU, Georgia-Roumbini y MARINAKI, Magdalene Marinaki Particle Swarm Optimization for the Vehicle Routing Problem with StochasticDemands. En:Applied Soft Computing, 2013, vol. 13, p. 1693–1704.

[12] ALONSO, Sergio, et al. La metaheurística de optimización basada en colonia de hormigas: modelos y nuevos enfoques. Trabajo realizado en el marco del proyecto “Mejora de Metaheurísticas mediante hibridación y sus aplicaciones” de la Universidad de Granada.

[13]GALVÁN NÚÑEZ, Silvia Adriana. Optimización por simulación basado en un sistema evolutivo de optimización de enjambre de partículas para el problema de ruteo de vehículos con demandas estocásticas. Tesis de maestría de ingeniería industrial. Bucaramanga. Universidad Industrial de Santander. 2012.

TABLA DE CUMPLIMIENTO DE OBJETIVOS	
OBJETIVO	**CUMPLIMIENTO**
Seleccionar el algoritmo para la solución del VRPSD de acuerdo a la revisión bibliográfica previamente realizada sobre algoritmos basados en la metaheurística Optimización Colonia de Hormigas (ACO).	Subcapítulo 2.5.3. Capítulo 3.
Desarrollar el algoritmo EACO que resuelve el problema de ruteo de vehículos con demandas estocásticas.	Capítulo 4. Capítulo 5. Capítulo 7.
Diseñar un toolbox en el software Matlab® con el algoritmo EACO que solucione el modelo.	Anexo A. Anexo B.
Evaluar la calidad de la solución obtenida por el algoritmo mediante instancias definidas previamente.	Capítulo 8. Capítulo 6.
Publicar los análisis, resultados y conclusiones del tema estudiado en un artículo académico.	Anexo C.

1 OBJETIVOS

1.1 OBJETIVO GENERAL

Resolver el Problema de Ruteo de Vehículos con Demandas Estocásticas (VRPSD) por medio de un algoritmo Evolutivo de Colonia de Hormigas (EACO).

1.2 OBJETIVOS ESPECÍFICOS

- Seleccionar el algoritmo para la solución del VRPSD de acuerdo a la revisión bibliográfica previamente realizada sobre algoritmos basados en la metaheurística Optimización Colonia de Hormigas (ACO).

- Desarrollar el algoritmo EACO que resuelve el problema de ruteo de vehículos con demandas estocásticas.

- Diseñar un toolbox en el software Matlab® con el algoritmo EACO que solucione el modelo.

- Evaluar la calidad de la solución obtenida por el algoritmo mediante instancias definidas previamente.

- Publicar los análisis, resultados y conclusiones del tema estudiado en un artículo académico.

2 MARCO TEÓRICO

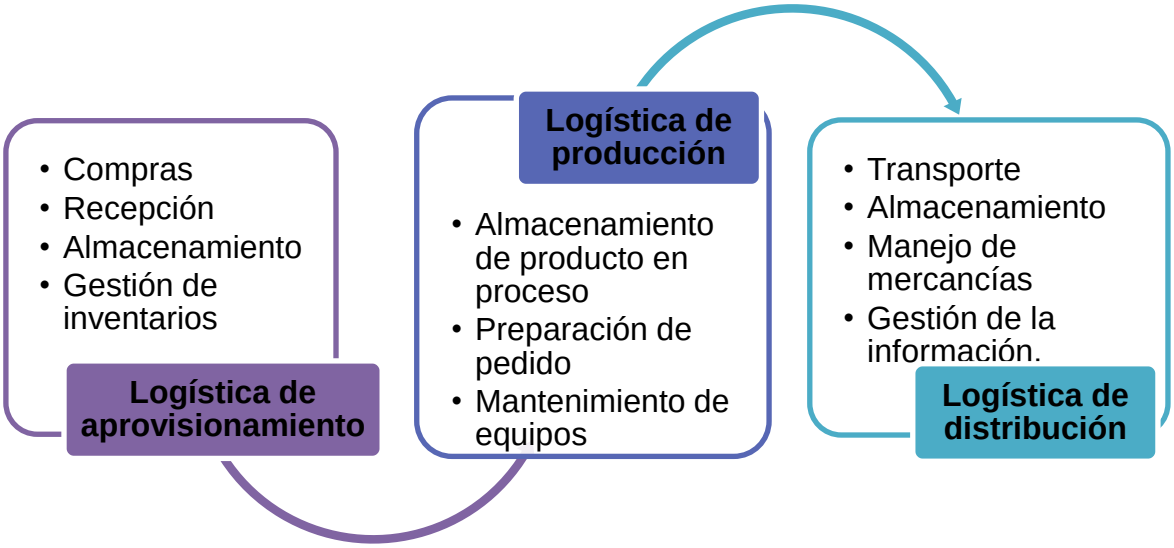
2.1 LOGÍSTICA DE DISTRIBUCIÓN

"La logística es el proceso de planear, implementar y controlar eficaz y efectivamente el flujo, almacenamiento de bienes o servicios e información de origen a destino"[14]. Comprende una serie de actividades que deben integrarse con el fin de generar valor para el cliente. "El valor en la logística puede ser expresado en términos de tiempo y de lugar, ya que los productos o servicios no tienen valor si los clientes no los encuentran en el tiempo y lugar exacto en el que quieren consumirlos"[15].

Las actividades logísticas deben desarrollarse como un proceso integrado, para ofrecer mayor velocidad de respuesta al mercado con costos mínimos. Por tal razón, como se muestra en la Figura 2, la logística se puede considerar como un sistema conformado por tres subsistemas: logística de abastecimiento, logística de producción y logística de distribución.

[14] CASTELLANOS RAMIREZ, Andrés. Manual de la gestión logística del transporte y distribución de mercancías. Barranquilla: Ediciones Uninorte, 2009. p. 55. ISBN 987-958-741-001-3.
[15] BALLOU, Ronald. Logística: Administración de la cadena de suministro. 5 ed. México: Pearson, educación, 2004. p. 11. ISBN: 9702605407. Citado por: GARCÍA MERINO, Felipe y RUIZ DALPIVAR, Álvaro. Análisis y propuesta de un modelo logístico para la empresa Productos Agropecuarios Santa Cecilia S. A. de C. V. Licenciatura en Administración de Empresas. Cholula, Puebla: Universidad de las Américas Puebla, 2004. p.12.

Figura 1. Subsistemas que conforman la logística



En la logística de abastecimiento se realizan todas las funciones relacionadas con la disposición de productos, bienes y servicios externos necesarios para que una empresa lleve a cabo sus operaciones. Comprende las actividades de compras, recepción, almacenamiento y gestión de inventarios; así mismo, se encarga de la selección y seguimiento de proveedores.

La logística de producción se enfoca en optimizar el flujo físico de materiales dentro de la planta. Considera el almacenamiento de producto en proceso, la preparación de pedidos, y el mantenimiento de los equipos de producción.

Por último, la logística de distribución se ocupa de gestionar las actividades relacionadas con el flujo de productos terminados y de información, desde que finaliza el proceso productivo hasta que estos llegan al cliente. Abarca las actividades de: transporte, almacenamiento, manejo de mercancías y gestión de la información. “El transporte es una de las actividades que implica mayor costo en la

logística de distribución"[16], por ello, se debe gestionar adecuadamente mediante la selección del método de transporte y la creación de rutas óptimas.

El diseño de rutas óptimas para una flota de vehículos que cumple la tarea de servir la demanda a un conjunto de clientes distribuidos espacialmente, se puede formular como un problema de optimización combinatoria, "*problema de ruteo de vehículos*". Una solución a este problema es la elección de la mejor ruta, que cumpla con las restricciones de capacidad del vehículo, tiempo del conductor y mínimo costo total de transporte.

2.2 PROBLEMAS DE OPTIMIZACIÓN COMBINATORIA

2.2.1 Definición de problemas de optimización combinatoria. En términos generales, un problema de optimización consiste en encontrar el valor de ciertas variables de decisión para maximizar o minimizar determinada función objetivo, sujeta a un conjunto de restricciones. Comúnmente, cuando se trata de maximizar se hace referencia a utilidades o ganancias, y cuando se trata de minimizar se habla de costos. Siendo así, un problema de ruteo de vehículos es un problema de minimización, cuyo objetivo es diseñar una ruta adecuada para encontrar el costo mínimo, que es función de la distancia total recorrida.

Para un problema de optimización existen diferentes soluciones dentro de un espacio de búsqueda y un criterio para discriminar entre ellas, depende del objetivo que se persiga; maximizar o minimizar. Dependiendo del dominio del espacio de búsqueda se pueden definir problemas de optimización binaria (si las variables de decisión son de naturaleza binaria, es decir, toman únicamente dos

[16] R.H. Ballou. Business Logistics/Supply Chain Management.5° ed. Pretince Hall.Upper Saddle River.New Jersey. 2004. p.14. Citapo por: CORREA, Andrés Felipe, et al. Características del diseño de rutas de distribución de alimentos en el Valle de Aburrá. En: Revista Facultad de Ingeniería de la Universidad de Antioquia. Septiembre, 2008. No 45. p. 173.

valores), entera (si las variables toman valores enteros positivos), continua (si las variables toman valores reales) o heterogénea (si las variables son de naturaleza binaria, entera y continua).

Cuando el espacio de búsqueda está limitado por un conjunto finito de soluciones denotado por S, donde cada elemento puede ser una solución candidata del problema, se denominan problemas de optimización combinatoria. Dada la condición de finitud las variables son de naturaleza discreta, con un domino restringido a un número entero, un subconjunto, una permutación o una estructura gráfica.

Un problema de optimización combinatoria puede ser definido por[17]:

- Un conjunto de variables $X = \{x_1, \dots, x_n\}$;
- El dominio de las variables $D_1, \dots D_n$;
- Restricciones entre variables;
- Una función objetivo f para ser maximizada o minimizada, donde $f: D \rightarrow R^+$

El conjunto de todas las posibles asignaciones factibles es:

$$S = \{s = (x_1, v_1), \dots, (x_n, v_n)\} |\ x_1 \in D_i, \quad s \ \ satisface\ todas\ las\ restricciones\}$$

Así resolver un problema de optimización combinatoria consiste en encontrar una solución $s^* \in S$ tal que, $f(s^*) \leq f(s)$, *si se minimiza* o $f(s^*) < f(s)$, *si se maximiza.*

Ejemplos de problemas de optimización combinatoria son el problema del agente viajero (TSP), el problema de ruteo de vehículos (VRP), la asignación cuadrática

[17] BLUM, Christian, y ROLI, Andrea. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. En: *ACM Computing Surveys (CSUR), 2008., vol.* 35. no. 3., p. 268.

(APQ), y problemas programación, como la programación de horarios y programación de diseño de redes de telecomunicaciones. Debido a la importancia práctica de estos problemas se han desarrollado diferentes algoritmos para obtener soluciones óptimas. Sin embargo, un problema de optimización puede ser categorizado como un problema de fácil o de difícil solución de acuerdo a la complejidad computacional del algoritmo.

2.2.2 Complejidad computacional. Un problema desde el punto de vista computacional está conformado por "un conjunto de datos de entrada, un conjunto de datos de salida y una función que asigna a cada dato del problema, una salida correcta del mismo"[18]. Los valores particulares que toman los datos de entrada (parámetros) son definidos como instancias del problema. Este puede ser resuelto mediante la implementación de un algoritmo, que es una secuencia de pasos definidos, que trasforma los datos de entrada en datos de salida.

El término complejidad computacional indica la eficiencia del algoritmo en la solución de un problema. Se pueden diferenciar principalmente dos criterios de eficiencia: tiempo y espacio, los cuales son recursos computacionales necesarios para su generación o cálculo. A partir de esta idea, matemáticamente los problemas pueden ser caracterizados por su grado de dificultad de acuerdo con la complejidad computacional del algoritmo que permite establecer su resolución. De esta manera los problemas pueden ser clasificados en tratables e intratables. Los problemas intratables no se pueden resolver en tiempo polinomial, esto quiere decir que si un algoritmo trata de solucionarlos, la cantidad de tiempo computacional que requeriría los convertiría en inaccesible aun cuando los tamaños de los problemas sean pequeños.

[18] PÉREZ JIMÉNEZ, Mario de J y SANCHO CAPARRINI, Fernando. Máquinas moleculares basadas en ADN. [En línea]. España: Universidad de Sevilla. 2003. [Citado 22 Mayo 2013]. Disponible en internet: http://goo.gl/TNprv

Los problemas tratables también conocidos como problemas ***P***, presentan una complejidad computacional polinómica, donde el tiempo de ejecución crece de forma polinomial con el tamaño de los datos de entrada.

Existen problemas que no se pueden resolver en tiempo polinómico, sin embargo, las posibles soluciones pueden ser examinadas en tiempo polinomial para determinar si son o no soluciones correctas, estos problemas son los denominados problemas ***NP***. Dentro de la clase ***NP***, se encuentra la subclase ***NP-completo***, que son los más difíciles de resolver. La característica principal es que cualquier problema ***NP*** se puede transformar en él con un coste adicional en tiempo polinomial. Otra clase de problema son los ***NP-duros (NP-Hard)***, para los cuales no existe un algoritmo que determine su solución óptima en tiempo polinomial.

A nivel práctico se pueden encontrar una gran variedad de problemas NP-Hard, como los problemas de optimización combinatoria para el diseño de rutas. Dada su complejidad las técnicas de optimización exactas no pueden establecer alguna solución óptima dentro de un intervalo de tiempo, por tanto en los últimos años se ha promovido el desarrollo de procedimientos que encuentren buenas soluciones en un tiempo computacional razonable, llamados métodos aproximados, dentro de los cuales se encuentran las heurísticas y metaheurísticas.

2.2.3 Métodos de solución de un problema de optimización combinatoria.

Los problemas de optimización combinatoria han representado un atractivo de investigación gracias a la complejidad computacional que poseen y a las diversas aplicaciones del mundo real. Se han propuesto múltiples métodos para determinar su solución. Los métodos pueden clasificarse básicamente en: métodos exactos y métodos aproximados.

2.2.3.1 Métodos exactos. Garantizan "encontrar la solución óptima para cualquier instancia de cualquier problema en un tiempo acotado"[19]. Sin embargo, poseen un inconveniente cuando se trata de un problema de optimización combinatoria de la clase NP-Hard, el tiempo que se requiere para determinar su solución crece exponencialmente con el tamaño del problema. A continuación se describe las características básicas de algunos métodos exactos.

La familia de algoritmos ***Branch and X***[20], que abarcan los procedimientos Branch and Bound, Branch and Cut, Branch and Price, entre otros. Básicamente utilizan una estrategia de fraccionar y dominar, donde se divide el espacio de solución en sub-problemas, los cuales luego son optimizados de manera individual.

La ***Programación dinámica***, es un método exacto que no está orientado a la optimización global, opera de acuerdo a decisiones secuenciales en etapas sucesivas, las decisiones que se toman en una etapa afectan el desarrollo futuro del sistema. La programación dinámica junto con el método Branch and Bound, generan lo que se conoce como ***programación dinámica acotada.***

Otro procedimiento comúnmente utilizado, es la ***programación lineal,*** su principal característica es que las expresiones de las restricciones y la función objetivo a optimizar no deben ser expresiones polinómicas, exponenciales, y no puede existir multiplicación entre las variables. Esto hace que el método exacto sea muy limitante, puesto que requiere de gran cantidad de variables para modelar las restricciones de un problema.

[19]CHICANO GARCÍA, José Francisco. Metaheurísticas e Ingeniería del Software. Tesis doctoral. Universidad de Málaga, Lenguajes y ciencias de la computación. 2007. p, 46.
[20] JOURDAN, BASSEUR, y TALBI. Hybridizing exact methods and metaheuristics: A taxonomy. En: European Journal of Operational Research. Diciembre, 2009. vol, 119. no, 3. p, 620–629.

2.2.3.2 Métodos aproximados. Ofrecen la opción de encontrar soluciones de alta calidad en tiempos razonables para problemas tipo NP-Hard, aunque estas soluciones no sean óptimas. Las técnicas aproximadas se clasifican en: heurísticas y metaheurísticas.

a. HEURÍSTICAS: Reeves (1995) define heurística, como "una procedimiento que busca buenas soluciones a un costo computacional razonable, aunque no se garantice su optimalidad, e incluso, en algunos casos no se llegue a establecer que tan cerca este de la optimalidad o factibilidad"[21]. Las técnicas heurísticas pueden dividirse principalmente en dos categorías: Heurísticas constructivas y métodos de búsqueda local.

Las **heurísticas constructivas** determinan las soluciones mediante un procedimiento que incorpora iterativamente elementos a una estructura inicialmente vacía, hasta obtener una solución completa, satisfaciendo las restricciones del problema. Este tipo de heurísticas "se pueden utilizar siempre que el problema se pueda resolver mediante decisiones sucesivas, como rutas, secuenciaciones, líneas de montaje, entre otros"[22]. El ***algoritmo de ahorro*** de ***Clarke*** y ***Wright***, es un ejemplo de este tipo de método, conocido porque se ha implementado al problema de ruteo de vehículo clásico.

Los **métodos de búsqueda local**, comienzan su procedimiento desde una solución inicial, y emplean un operador para construir un vecindario, y avanzan buscando mejoras hasta que encuentran una mejor solución en el vecindario, que reemplaza la actual. Dentro de estas técnicas se puede reconocer un concepto,

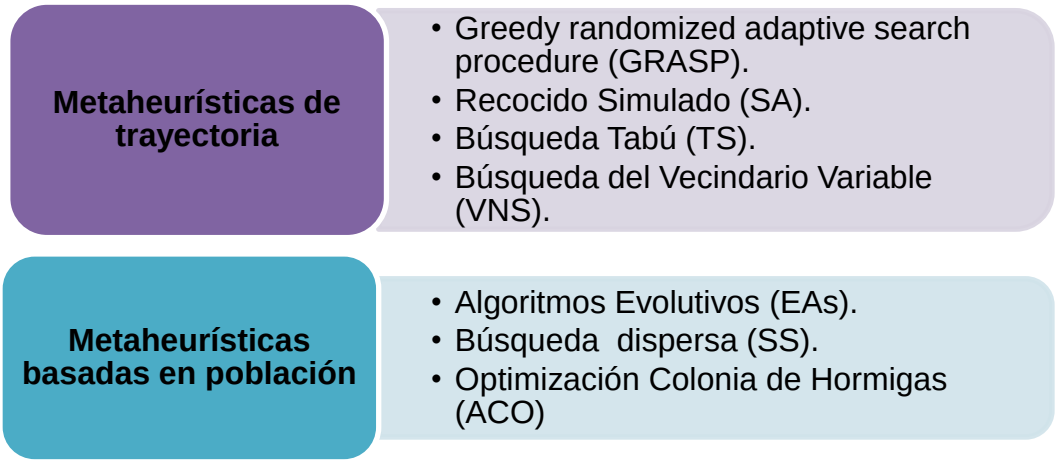
[21]REEVES, C. Modern Heuristic Techniques for Combinatorial Problems. 1995. Mc-Greenhill. Citado por: NASSER A, El-Sherbeny. Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. En: Journal of King Saud University. 2010. vol, 22. p, 126 - 127.
[22]GARCÍA SABATER, José Pedro y MAHEUT, Julien. Modelos y Métodos de Investigación de Operaciones. Procedimientos para Pensar. p,103.

óptimo local, que puede definirse como una solución mejor o igual que cualquier otra solución de un vecindario.

b. METAHEURÍSTICAS: Durante los últimos años ha surgido un nuevo tipo de algoritmo de aproximación, denominado metaheurística. Puede definirse como un proceso de generación que guía una heurística subordinada, combinando de forma inteligente diferentes conceptos para explorar y explotar el espacio de búsqueda y encontrar soluciones eficientes cercanas al óptimo. El principal objetivo es evitar la desventaja de las heurísticas, escapar de los óptimos locales.

En la Figura 3, se aprecia un esquema de clasificación de las metaheurísticas publicado en el 2005 por Michel Gendreau y Jean-Yves Potvin. Es importante resaltar que existen diversas formas de clasificar las metaheurísticas, y depende del enfoque que los autores determinen.

Figura 2. Clasificación de Metaheurísticas



Fuente: Adaptado a partir de GENDREAU, Michel y POTVIN, Jean-Yves. Metaheuristics in Combinatorial Optimization. En: Annals of Operations Research. 2005. vol,140. p, 189–213.

Metaheurísticas de trayectoria: Estas metaheurísticas básicamente parten de una solución, y a través de la exploración del vecindario se va actualizando la

solución actual, formando así una trayectoria. A diferencia de los procedimientos de población, los métodos de trayectoria manipulan una solución del espacio de búsqueda por iteración.

- **Procedimiento de búsqueda adaptativa aleatorizada codiciosa (GRASP)**

Es un método de múltiples comienzos, donde a cada uno de estos se le aplica una heurística de construcción codiciosa aleatoria con el propósito de generar una solución inicial, que será mejorada posteriormente mediante la implementación de la búsqueda local. Este procedimiento se repite un número determinado de reinicios y la mejor solución global se toma al final. En cada etapa de la heurística, los elementos que aún no se han incorporado en la solución parcial son evaluados utilizando una función codiciosa, y los mejores se mantienen en una Lista Candidata Restringida (*Restricted Candidate List*, RCL). A partir de esta lista se elige aleatoriamente un elemento para ser incorporado en la solución, generando diversidad de soluciones.

- **Recocido Simulado (SA)**

La metaheurística de Recocido Simulado fue propuesta por primera vez en 1983 por Kirkpatrick, Gelatt y Vecchi, y luego tratada nuevamente por Cerny en 1985. Para evitar el atrapamiento del método en mínimos locales se permite el movimiento a soluciones con valores de la función objetivo peores que el valor de la solución actual. El movimiento que se realiza se denomina movimiento ascendente. En cada iteración la solución actual s se modifica mediante la selección aleatoria de una solución $s' \in N(s)$. Si s' tiene un valor de la función objetivo más bajo que s, se acepta como solución actual. Pero si el movimiento de s a s' es ascendente, s' se admite de acuerdo a una probabilidad, que está en función de un parámetro T_K denominado temperatura. El parámetro es actualizado en cada iteración de acuerdo a un esquema de enfriamiento, es decir, se define el

valor de T_K en cada iteración K. El valor inicial de la probabilidad de aceptación del movimiento ascendente debe ser alto, y a medida que avanza la búsqueda gradualmente se disminuye.

- **Búsqueda Tabú (TS)**

Las primeras ideas sobre la metaheurística Búsqueda Tabú fueron introducidas por Glover en 1986. La Búsqueda Tabú simple utiliza una memoria a corto plazo para realizar seguimiento a las soluciones recientemente visitadas, excluyéndolas del vecindario de la solución actual. Esta memoria se conoce comúnmente como Lista Tabú. En cada iteración del procedimiento se selecciona a la mejor solución del vecindario de la solución s, como la nueva solución actual. Esta es adicionada a la lista tabú, y si ha llegado a su máxima capacidad, se elimina de acuerdo al sistema PEPS (primeras en entrar, primeras en salir) una de las soluciones que ya se encontraban en la lista. El algoritmo se detiene cuando se cumple un criterio de terminación, o cuando el conjunto de soluciones se encuentra vacío.

La longitud de la lista tabú controla la memoria del proceso de búsqueda. Una longitud pequeña obliga al proceso a explorar regiones pequeñas, y por el contrario una longitud grande permite la exploración de regiones más grandes, puesto que prohíbe volver a visitar a un mayor número de soluciones.

- **Búsqueda del Vecindario Variable (VNS)**

Esta metaheurística principalmente emplea una heurística de búsqueda local para cambiar sistemáticamente la estructura del vecindario. A partir de un conjunto pre-seleccionado de estructuras del vecindario, se genera aleatoriamente una solución en el primer vecindario de la solución actual, desde la cual se realiza un descenso local. Si el óptimo local obtenido no es mejor que la solución actual, el procedimiento se repite en el siguiente vecindario. Cuando se encuentra una solución mejor que la actual o cada estructura del vecindario ha sido explorada, la búsqueda se reinicia desde el primer vecindario.

Metaheurísticas basadas en población: En cada iteración estas metaheurísticas trabajan con un conjunto de soluciones que son mejoradas mediante un proceso inteligente que se encarga de explorar el espacio de búsqueda. Dentro de estos métodos se pueden distinguir dos grupos principalmente: algoritmos evolutivos (*Evolutionary Algorithms*, EAs) y procedimientos alternativos que presentan algunos rasgos de los EAs, pero tienen características particulares que los diferencian.

- **Algoritmos evolutivos (EAs)**

Los algoritmos evolutivos están inspirados en el principio Darwiniano de la selección natural, fundamentado en la capacidad que tienen las especies de evolucionar y adaptarse a su medio ambiente. A partir de este principio, una población de soluciones generada aleatoriamente o mediante un método heurístico, evoluciona desde una generación a otra mediante la aplicación de operadores genéticos: selección, cruce y mutación. El operador de selección, probabilísticamente toma a las mejores soluciones para ser padres y crear descendencia. El operador de cruce selecciona dos soluciones padre para combinar las características más atractivas y deseables con el propósito de generar una o dos soluciones descendencia. El proceso se realiza iterativamente hasta que se establece una nueva población de soluciones. Antes de sustituir la antigua población, cada individuo de la nueva población es alterado mediante el operador de mutación. Los EAs se caracterizan por explotar una población de soluciones y crear nuevas a través de la recombinación de los buenos atributos de dos padres soluciones.

Debido a que estos métodos presentan el inconveniente de que la población mejora en promedio de generación en generación, pero a menudo no generan soluciones casi óptimas, se han desarrollado híbridos en donde se emplean operadores de búsqueda local para reemplazar el proceso de mutación, mejorando la solución hasta encontrar un óptimo local.

- **Búsqueda dispersa (SS)**

La idea básica del SS (*Search Scatter*, SS) fue desarrollada por Glover entre 1963 y 1997. Fundamentalmente combina un conjunto de soluciones, llamado conjunto referencia, para crear nuevas soluciones. El procedimiento comienza a partir de un conjunto referencia inicial, generalmente creado mediante heurísticos. Se realizan combinaciones convexas y no convexas de un número variable de soluciones para generar candidatos dentro y fuera de las regiones convexas abarcadas por el conjunto referencia. Estas soluciones heurísticamente son mejoradas y las mejores de acuerdo a su calidad y diversidad, son adicionadas al conjunto referencia. Este procedimiento se repite hasta que el conjunto referencia no cambie más.

- **Optimización Colonia de Hormigas**

Marco Dorigo junto con otros investigadores, introdujeron los primeros algoritmos ACO en la década de 1990. ACO es una metaheurística basada en el comportamiento de forrajeo de las hormigas reales. En cada ciclo, un número de hormigas artificiales secuencialmente construyen soluciones de manera aleatoria. Cada hormiga elige el siguiente elemento a ser incorporado en su solución actual de acuerdo a la información heurística y la cantidad de feromona asociada con ese elemento. La feromona representa la memoria del sistema, y se relaciona con soluciones previamente construidas. Fundamentalmente, se define una distribución de probabilidad llamada probabilidad de transición, sobre todos los elementos a ser incorporados en la solución parcial actual. Cada vez que un elemento se selecciona por una hormiga, el nivel de feromona es actualizado adicionando nueva cantidad de feromona y utilizando la tasa de evaporación de feromona, que se encarga de eliminar parte de esta. Cuando todas las hormigas han construido una solución completa, el procedimiento se reinicia con los niveles de feromonas actualizados. Esto se repite para un determinado número de ciclos o hasta que se produce el estancamiento de búsqueda.

En la literatura existen variaciones sobre la metaheurística ACO, entre las más representativas se encuentran: Sistema de Hormigas, Sistema Colonia de Hormigas y MAX - MIN Sistema de Hormigas.

2.3 PROBLEMA DE RUTEO DE VEHÍCULOS

El VRP es una generalización del problema del agente viajero (*Traveling Salesman Problem,* TSP), que fue introducido por primera vez en un artículo publicado por "**Dantzig** y **Ramser (1959)** que trataba el problema de despacho de camiones"[23]. Es un problema de optimización combinatoria, "clasificado como un problema NP-hard debido a su complejidad"[24].

El problema de ruteo de vehículos, consiste en el diseño de rutas óptimas, para la entrega o recolección de productos de uno o varios depósitos, a un conjunto de clientes dispersos geográficamente, con la finalidad de cumplir un objetivo planteado y ciertas restricciones operativas, tales como; la capacidad de los vehículos, ventanas de tiempo y períodos de trabajo del conductor.

Algunos objetivos que comúnmente se plantean para resolver el VRP son[25]:

- Minimización del costo de transporte total, dependiendo de la distancia total de viaje (o el tiempo total de viaje) y el costo fijo asociado con los vehículos usados (y con el correspondiente conductor).
- Minimización del número de vehículos (o conductores) requeridos para servir a todos los clientes, teniendo en cuenta el equilibrio de las rutas, para el tiempo de viaje y la carga del vehículo.

[23] CORDEAU, Jean François, et al. Vehicle Routing. En: Handbooks in Operations Research & Management Science: Transportation. 2007. p.367

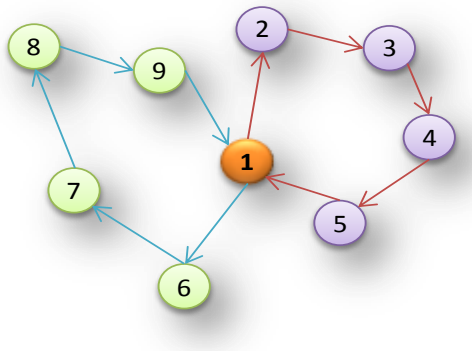
[24] Lenstra, J. K., &Rinnooy, K. Complexity of vehicle routing and scheduling problems.Networks, 1981.Vol. 11. p. 221–227. Citadopor: JEON, Geonwook; LEEP, Hermna R. y SHIM, Jae Young. En: A vehicle routing problem solved by using a hybrid genetic algorithm. Junio, 2007.vol. 53.p.681.

[25] TOTH, Paolo y VIGO, Daniele. The Vehicle Routing Problem. En: Society for Industrial and Applied Mathematics, 1987. vol.9., p. 4.

- Minimización de la penalización asociada con el servicio parcial de los clientes. Situación que se presenta cuando la capacidad del vehículo no puede suplir la demanda total del cliente.
- O la combinación del peso de estos objetivos, conocidos como problemas multi-objetivos.

2.3.1 Definición de un problema de ruteo de vehículos. En términos generales, el VRP puede ser formulado de la siguiente manera: "Sea $\boldsymbol{G} = (\boldsymbol{V}, \boldsymbol{A})$ un gráfico completo, donde $\boldsymbol{V} = \{\mathbf{1}, \dots, \boldsymbol{n} + \mathbf{1}\}$ define un conjunto de vértices asignados a $\boldsymbol{n}$ clientes con el depósito localizado en el vértice *1,* y $\boldsymbol{A}$ un conjunto de arcos"[26]. "El costo no negativo $\boldsymbol{C_{ij}}$ asociado a cada arco $(\boldsymbol{i}, \boldsymbol{j}) \in \boldsymbol{A}$ con $\boldsymbol{i} \neq \boldsymbol{j}$, representa el costo de viaje entre los vértices $\boldsymbol{i}$ y $\boldsymbol{j}$"[27]. Dependiendo del contexto, este costo puede ser interpretado como distancia o tiempo de viaje. Además, en algunos casos se considera que el problema es simétrico, es decir, $\boldsymbol{C_{ij}} = \boldsymbol{C_{ji}}$ para cada arco $(\boldsymbol{i}, \boldsymbol{j})$. Todos los clientes $\boldsymbol{i}$ están asociados a una demanda conocida no negativa, $\boldsymbol{d_i}$ para ser entregada por una flota de $\boldsymbol{m}$ vehículos que inician y terminan sus recorridos en el depósito.

Figura 3. Ejemplo del VRP con 1 depósito, 8 clientes y 2 vehículos.



[26] LAPORTE, Gilbert. The Vehicle Routing Problem: An overview of exact and approximate algorithms. En: European Journal of Operational Research, 1992., vol. 59., p. 345
[27] ALABAS-USLU, Cigdem, DENGIZ, Berna. A self-adaptive local search algorithm for the classical vehicle routing problem. En: Expert Systems with Applications, 2011., vol. 38., p. 8990.

El problema consiste en determinar las rutas de los respectivos vehículos con el menor costo posible, tal que cada cliente, excepto el depósito, es visitado una sola vez por un vehículo para servir a su demanda.

2.3.2 Modelación matemática del VRP. El problema de ruteo de vehículos es formulado de acuerdo a Marinakis, y Marinaki (2009), de la siguiente manera[28]:

- ***Variable de decisión:*** Es de naturaleza binaria. $x_{ijk}, \forall i, j$

  $1 \leq i, j \leq n+1, \quad i \neq j, \quad k = 1, \dots, m$

  - x_{ijk}=1 , si en la solución, el camino (i, j)es recorrido por el vehículo k
  - x_{ijk}=0, si en la solución, el camino (i, j)no es recorrido por el vehículo k

- ***Función objetivo:*** Se pretende minimizar el costo total de la ruta. La función objetivo es calculada así:

  $$Min\ z(x) = \sum_{i=1}^{n+1} \sum_{j=1}^{n+1} \sum_{k=1}^{m} C_{ij} x_{ijk} \qquad [1]$$

- **Restricciones:** Se plantean las siguientes restricciones de conservación del flujo:

- $\sum_{k=1}^{m} \sum_{j=1}^{n+1} x_{ijk} \leq 1 \qquad \forall i \epsilon V \backslash \{1\} \qquad [2]$
- $\sum_{k=1}^{m} \sum_{i=1}^{n+1} x_{ijk} \leq 1 \qquad \forall j \epsilon V \backslash \{1\} \qquad [3]$
- $\sum_{j=1}^{n+1} x_{1jk} = 1 \qquad \forall k \qquad [4]$
- $\sum_{i=1}^{n+1} x_{i1k} = 1 \qquad \forall k \qquad [5]$
- $\sum_{i=1}^{n+1} x_{ifk} - \sum_{j=1}^{n+1} x_{fik} = 0 \qquad \forall k; f \epsilon V \qquad [6]$

[28] MARINAKIS, Yannis y MARINAKI, Magdalene. A hybrid genetic – Particle Swarm Optimization Algorithm for the vehicle routing problem.En: Expert Systems with Applications. Marzo, 2009. vol. 37, no, 2., p. 1447.

- $x_{ijk} \epsilon \{0,1\}$ $\forall i, j$ [7]

Las restricciones [2] y [3] indican que para cada cliente sólo se puede asignar una ruta vehicular. Las restricciones [4] y [5] limitan el número de rutas por vehículo a uno. [4] hace referencia al número de veces que el vehículo sale del depósito y [5] indica el número de veces que el vehículo llega al depósito. La ecuación [6] determina la continuidad de la ruta, es decir, el número de vehículos que llega a un cliente debe ser igual al número de vehículos que sale. La restricción [7] designa a la variable x_{ijk} los números enteros 0 y 1.

Esta definición del VRP clásica, es un problema idealizado que presenta poca aplicabilidad en escenarios reales, por ello se han incorporado variantes más flexibles, adicionando nuevas restricciones que permitan representar diferentes situaciones prácticas.

2.3.3 Variantes del VRP. Dependiendo del objetivo a ser optimizado y de los tipos de restricciones que deben cumplirse, se despliegan un gran número de variantes del VRP que se describen a continuación.

- ***Capacitated Vehicle Routing Problem (CVRP):*** En el CVRP se restringe la capacidad del vehículo, y así mismo la demanda total asignada a cada ruta. Las características básicas de este problema son: las demandas del cliente son deterministas y se conocen con anticipación a su visita, las entregas no se pueden dividir; esto quiere decir que una orden no puede ser servida por más de un vehículo, la flota de vehículos es homogénea y sólo hay un depósito. Además, el objetivo es disminuir el costo total de viaje que está en función de la distancia total recorrida para visitar a todos los clientes.

- ***Vehicle Routing Problem with Time Windows (VRPTW).*** Este problema se caracteriza por limitar el tiempo de inicio y fin del servicio y la llegada y salida

del vehículo del depósito. Se pueden presentar dos situaciones relevantes: que el vehículo pueda llegar después del tiempo establecido para el inicio del servicio, o en caso contrario, que el vehículo llegue antes. En la primera situación, se incurre en una penalización del objetivo y en la segunda, se introduce un tiempo de espera en el horario de la ruta. En la mayoría de problemas de aplicaciones del VRPTW se considera una función multiobjetivo, minimizando el número de vehículos, la distancia y el tiempo de viaje. Un caso especial del VRPTW se presenta cuando se limita únicamente el tiempo de inicio del servicio; a este problema se le conoce como VRP con plazos de tiempo (*VRP with time Deadlines, VRPTD*).

- ***Vehicle Routing Problem with Backhauls (VRPB):*** El VRP con viajes de regreso, es una extensión del CVRP. Los clientes son divididos en dos grupos: clientes lineahaul, que requieren una cantidad dada de producto desde el depósito y clientes backhaul, para los que una cantidad dada de producto entrante debe ser recogida y llevada al depósito. Con el objetivo de ahorrar costos, se permite que los vehículos vacíos recojan los productos entrantes y los transporten a la estación. Para evitar que se dé un reordenamiento de mercancía en el interior del vehículo, se establece una restricción de precedencia, que determina que todos los clientes linehaul deben ser visitados antes que los clientes backhaul. Cuando se admite que un vehículo recoja los productos, sin antes haber visitado todos los clientes linehaul, la restricción de precedencia se relaja resultando otra variante del VRP. También existe el VRPB mixto, que mezcla libremente las operaciones de entrega y recogida de mercancías, siempre y cuando se cumpla la restricción de capacidad. Otra variante es el VRPBTW que se obtiene cuando ventanas de tiempo son asociadas a los clientes.

- ***Time-Dependent Vehicle Routing Problem (TDVRP):*** Es una extensión interesante del VRP, principalmente cuando es aplicado a problemas de ruteo

en entornos urbanos, donde los tiempos de viajes no son fijos; sino que dependen tanto de la distancia entre los clientes como de la carga de tráfico, la cual varía con la hora del día.

- ***Pick-Up and Delivery Problem (PDP):*** En el VRP con recogida y entrega, se cumple con una serie de solicitudes de transporte. A diferencia de los otros problemas, los artículos transportados no se almacenan inicialmente en el depósito, sino que se distribuyen a través de los nodos de una ruta. Una solicitud de transporte consiste en la transferencia de la demanda desde el punto de recogida hasta el punto de entrega. Cuando las ventanas de tiempo están asociadas a las operaciones de entrega y recogida, se obtiene el PDPTW. Una aplicación del PDP se da en el servicio de transporte público. Sin embargo, en este caso se limitan los tiempos de viaje, por ello es conocido como el problema Dial-A-Ride (DARP).

- ***Dynamic Vehicle Routing Problem (DVRP):*** Otra extensión del VRP, que es común en muchos casos prácticos, se presenta cuando se obtiene nueva información durante el proceso de distribución, la cual debe integrarse con las rutas actuales. Por ejemplo, cuando los nuevos pedidos llegan de forma dinámica, las rutas tienen que volverse a planificar incluyendo el nuevo servicio. Otra caso se puede presentar cuando los tiempos de viaje cambian debido a acontecimientos imprevistos.

- ***Multiple-depot problems:*** Se plantea el problema del transporte de materias primas procedentes de múltiples depósitos, a un conjunto de clientes, posibilitando al vehículo finalizar su recorrido en una estación diferente a la de inicio.

- ***Mixed fleet and size problems:*** Consiste en determinar el tamaño y composición de una flota de transporte de acuerdo a los requerimientos del

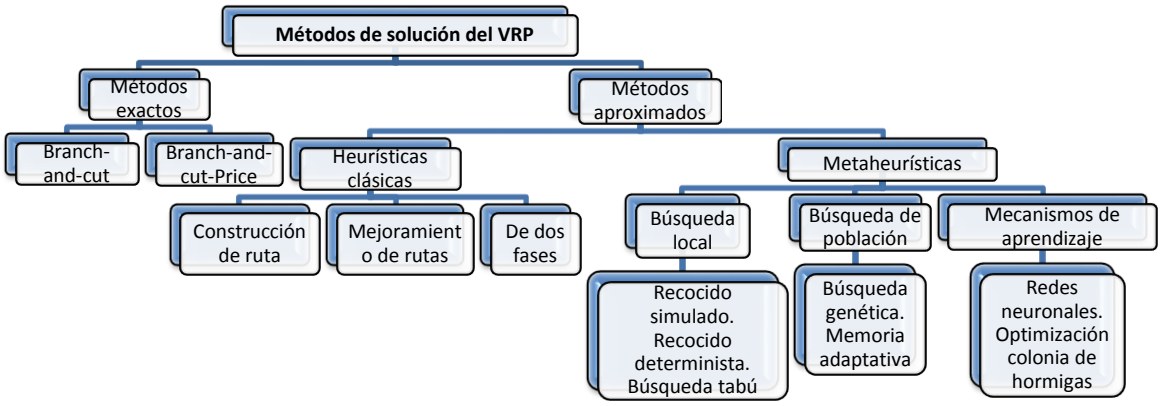
cliente. En cuanto a la composición se pueden tener flotas constituidas por vehículos idénticos o heterogéneos (de distinta capacidad).

- ***Location-Routing Problems:*** En estos problemas se toman decisiones estratégicas sobre las ubicaciones de los depósitos y almacenes, y al mismo tiempo se diseñan las rutas.

- ***Stochastic Vehicle Routing Problems (SVRP):*** Estos problemas se caracterizan por el comportamiento estocástico de algunos parámetros, tales como; la demanda de los clientes, ubicación de los clientes, y tiempo de viaje. En este trabajo se estudia el problema de ruteo de vehículos con demandas estocásticas.

2.3.4 Métodos de solución del VRP. El VRP ha sido uno de los problemas NP-hard más estudiado debido a su importancia teórica y práctica. Para la solución del VRP se han usado diferentes métodos exactos y aproximados, que son representados en la siguiente figura:

Figura 4. Métodos de solución del VRP



Fuente: Adaptado a partir de CORDEAU, Jean François, et al. Vehicle Routing. En: Handbooks in Operations Research & Management Science: Transportation. 2007. p.367

Un estudio detallado sobre métodos exactos, es presentado por **Toth** y **Vigo** en el **2002,** sobre dos principales métodos: "Branch-and-Cut (BC), basado en un modelo de flujo de dos índices y Branch-and-Cut-and-Price (BCP), basado en un modelo de fraccionamiento en conjunto (Set Partitioning)"[29].

Para los métodos aproximados, el más conocido de los algoritmos de heurísticas clásicas para construcción de ruta es el de "**Clarke** y **Wright** desarrollado en **1964**, que se basa en el concepto de ahorro"[30]. Para el mejoramiento de rutas, una heurística común es *λ-opt*, propuesta por **Lin** en **1965** para el problema del agente viajero, donde λ arcos son eliminados de la solución actual y sustituidos por otros. Otra alternativa que toma en cuenta un valor de *λ* más grande, es la heurística *Or-intercambios*, desarrolla por **Or** en **1976**. Por otra parte, para las heurísticas de dos fases, que se fundamentan en solucionar el VRP a partir de dos sub-problemas: agrupación y enrutamiento, se destaca el algoritmo de barrido propuesto en **1974** por **Gillet y Millet** "que se enfoca en el sub-problema de agrupación, clasificando y conectando los puntos de demanda por medio de su ángulo en coordenadas polares"[31].

En cuanto a las metaheurísticas, se puede destacar el trabajo realizado por **Osman** en **1993**, sobre la heurística de recocido simulado, que se fundamenta "en el enfoque de 2-intercambios para la definición de vecindarios y el uso de una regla decreciente en el cambio de la función de temperatura"[32].

Entre los estudios de metaheurísticas de búsqueda tabú que han tenido gran éxito en su aplicación cabe resaltar la investigación propuesta por **Taillard** en **1993,**

[29] BALDACCI, Roberto, et al. An exact solution framework for a broad class of vehicle routing problems.En: Computational Management Science. Julio, 2010.vol, 7. p.230.
[30] CORDEAU, Jean François, et al. Op. cit., p 376.
[31]YOUSEFIK HOSHBAKHt, Majid y KHORRAM, Esmaile. Solving the vehicle routing problem by a hybrid meta-heuristic algorithm. En: Journal of Industrial Engineering International. Diciembre, 2012.vol, 8. no, 1. p.5
[32]OSMAN, Ibrahim Hassan. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. En: Annals of Operations Research. 1993. vol.41. p.423

sobre el algoritmo de búsqueda tabú aplicado al CVRP (*Capacited Vehicle Routing Problem*), que se basa en un mecanismo de 1-intercambio para definir soluciones vecinas y que por medio de un algoritmo exacto del TSP, propuesto por Volgenant y Jonker en 1983 se re optimizan las rutas periódicas. En el algoritmo Taburoute formulado por **Gendreau, Hertz** y **Laporte (1994)**, en cada iteración un subconjunto de vértices es seleccionado aleatoriamente, moviendo un vértice desde su ruta actual a otra ruta que es una de sus vecinos más cercanos"[33]. El algoritmo de búsqueda del vecindario más grande (VLNS, por sus inglés) desarrollado por **Ergun et al**. en el **2003** y el algoritmo de búsqueda tabú granular (Granular Tabu Search) de **Toth** y **Vigo (2003).**

En la literatura, también se encuentran diversos estudios que combinan las características de la búsqueda tabú con otros algoritmos, entre ellos se encuentran: El algoritmo EVE-OPT propuesto por **Perboli, Guido; Pezzella, Ferdinando** y **Tadei, Roberto, (2008)**[34] para resolver el CVRP, el cual consiste en un algoritmo híbrido que combina la capacidad del algoritmo genético de explorar el espacio de búsqueda, y la ventaja para converger rápidamente en buenas soluciones de la búsqueda Tabú. El algoritmo híbrido a gran escala propuesto por **Du, Lingling** y **He, Ruhan,** en el **2012**[35], enlaza las características de la búsqueda del vecino más cercano y búsqueda tabú en un procedimiento de dos etapas. En la primera etapa las rutas iniciales se construyen a partir de la búsqueda del vecino más cercano y en la segunda etapa se optimizan las intra-rutas y las inter-rutas mediante la búsqueda tabú.

Para el algoritmo de recocido determinista, el primer trabajo aplicado al VRP, fue desarrollado por **Golden et al.** en **1998** y recientemente en el **2005, Lie et al.**

[33]CORDEAU, Jean François, y LAPORTE, Gilbert. . Op. cit., p. 150.
[34]PERBOLI, Guido; PEZZELLA, Ferdinando y TADEI, Roberto. EVE-OPT: a hybrid algorithm for the capacitated vehicle routing problem. En: Mathematical Methods of Operations Research. Octubre, 2008.vol, 68. no. 2. p. 371.
[35]DU, Lingling y HE, Ruhan.Combining Nearest Neighbor Search with Tabu Search forLarge-Scale Vehicle Routing Problem. En: PhysicsProcedia. 2012. vol. 25. p. 1536-1546.

elaboraron un algoritmo que “combina el principio de grabar para recordar de Dueck (1993) con el GTS y los vecinos se definen a través movimientos 2-opt”[36].

Una importante contribución de las heurísticas de búsqueda de población aplicadas al problema de ruteo de vehículos, es presentada por **Rochat** y **Taillard** en **1995,** quienes formulan un procedimiento de memoria adaptativa (AMP, por sus siglas en inglés).

En la mayoría de los estudios sobre algoritmos genéticos (*Genetic Algorithm*, GA) se realiza una combinación de este algoritmo con algún tipo de procedimiento de mejora, resultando un GA híbrido o algoritmo memético, dentro de estos estudios se encuentran: la investigación de **Prins, Christian**[37] en el **2004** en la cual se presenta un algoritmo genético sin limitaciones de rutas, hibridado con un procedimiento de búsqueda local que evita la convergencia prematura, mediante la exploración de vecindarios pequeños que son compensados por el paralelismo intrínseco del GA, además utiliza un procedimiento especial de separación que permite la realización de cruces de orden OX. Y el trabajo presentado en el **2009** por **Marinakis, Yannis** y **Marinaki, Magdalene**[38], que se fundamenta principalmente en la combinación de un algoritmo genético y un algoritmo de optimización de enjambre de partícula (*Particle Swarm Optimization*-PSO).

Respecto a los mecanismos de aprendizaje se han propuesto un número limitado de heurísticas. Sin embargo, recientemente se han realizado diferentes estudios sobre metaheurísticas híbridas de la optimización colonia de Hormigas, para obtener mejores soluciones. El primer algoritmo de optimización colonia de hormigas “fue desarrollado por **Clolrni et al. (1991)** y aplicado con éxito al

[36]CORDEAU, Jean François, et al. Op. cit., p. 383
[37]PRINS, Christian.A simple and effective evolutionary algorithm for the vehicle routing problem.En:Computers& Operations Research. Octubre, 2004.vol.31. p.1985-2002.
[38]MARINAKIS, Yannis y MARINAKI, Magdalene. Op. cit., p. 1449.

problema del agente viajero en base a las capacidades de búsqueda de trayectoria de las hormigas reales (Dorigo y Gambardella, 1997)"[39].

En el trabajo realizado por **Mazzeo, Silvia** y **Loiseau, Irene**[40] en el **2004,** proponen un algoritmo de colonia de hormigas para el CVRP. Las características importantes del algoritmo son: construcción de varias soluciones al mismo tiempo, intercambio de información durante el proceso y uso de información de iteraciones anteriores.

Bell, John E. y **MC. Mullen Patrick R**.[41], en este mismo año, **2004**, aplicaron la metaheurística ACO a un conjunto establecido de problemas de ruteo de vehículos. Utilizaron múltiples colonias de hormigas con depósitos únicos de feromona para cada vehículo y la heurística 2-opt como estrategia de mejoramiento.

En el **2008, Wen, Peng** y **Chang-Yu, Zhou**[42], combinaron el algoritmo genético y la colonia de hormiga (*Ant Colony*, AC) para resolver el VRP. En el AC, cuando la intensidad de las feromonas de una ruta es más alta que otras, este algoritmo es prematuro y no puede encontrar la solución más óptima, para evitar esto se introduce un algoritmo genético, mejorando la débil capacidad de explotación local. El momento en el que entra a actuar el algoritmo genético es después de que AC genera las soluciones óptimas.

[39]ZHANG,Xiaoxiay LANG, Lixin. A new hybrid ant colony optimization algorithm for the vehicle routing problem.En: Pattern Recognition Letters. 2009. vol. 30. p. 848
[40] MAZZEO, Silvia y LOISEAU, Irene. An Ant Colony Algorithm for the CapacitatedVehicle Routing.En:Electronic Notes in Discrete Mathematics. 2004. vol 18. p. 181-186
[41]BELL, John E. y MC.MULLEN Patrick R. Ant colony optimization techniques for the vehicle routing problem.En:Advanced Engineering Informatics. 2004. vol. 18. p. 41-48.
[42]WEN, Peng y CHANG-YU, Zhou.Solving Vehicle Routing Problem Using Ant Colony and Genetic Algorithm. En: International Conference on Intelligent Computing (4: 15-18, Septiembre: Shanghai, China). 2008. p. 23-30.

En el **2009**, **Zhang, Xiaoxia** y **Tang, Lixin** desarrollaron un algoritmo que combina el mecanismo de construcción de rutas de ACO y la búsqueda diversa llamada SS_ACO. El SS durante el proceso de búsqueda mantiene algunas soluciones de alta calidad y diversifica el espacio de búsqueda cuando el proceso está atrapado en un mínimo local. "El procedimiento de transferencias cíclicas está incrustado en un marco de búsqueda dispersa como búsqueda local para mejorar las soluciones mediante la transferencia de clientes entre las rutas de una manera cíclica"[43].

Yousefi Khoshbakht, Majid y **Sedighpour, Mohammad**[44], en el **2011** realizaron un trabajo para resolver el VRP, modificando el modelo de evaluación de feromonas de la metaheurística sistema colonia de hormigas (*Ant Colony System*, ACS). En el proceso de mejora del ACS, se encontró que utilizando una función trigonométrica se obtienen mejores resultados que con el factor de evaporación de feromona (ρ), permitiendo que el algoritmo olvide las malas decisiones tomadas anteriormente. Adicionalmente, se incorpora la heurística 3-Opt para mejorar el rendimiento de la búsqueda.

Otro estudio reciente fue elaborado por **Xiao, Zhang** y **Wang, Jiang-qing**[45] en el **2012,** sobre la base tradicional ACO y la heurística del vecino más cercano. Durante la fase de construcción de la solución, cada hormiga de la colonia intenta construir una solución. La cual se somete a la operación de la mutación y la heurística 2-opt, para mejorar la solución y evitar la convergencia prematura.

[43] ZHANG, Xiao-xia y TANG, Lixin. College of Information Science and Engineering,Northeastern University,Shenyang.
[44] YOUSEFI KHOSHBAKHT, Majid y SEDIGHPOUR, Mohammad. An Optimization Algorithm for the Capacitated Vehicle Routing Problem Based on Ant Colony System. En: Australian Journal of Basic and Applied Sciences. 2011. vol, 5. no, 12. p. 2729-2737.
[45]XIAO, Zhang y WANG, Jiang-qing.Hybrid Ant Algorithm and Applications for Vehicle RoutingProblem.En:Physics Procedia. 2012. vol. 25. p. 1892 – 1899

En la Escuela de Estudios Industriales y Empresariales de la Universidad Industrial de Santander se han desarrollado algunos trabajos de investigación basados en el Problema de Ruteo de Vehículos. Entre estos se destacan:

El trabajo realizado por **Pertuz Montenegro**, **Alfredo José** y **Rojas Silva, Kimberly** en el **2007**[46] sobre un método para la optimización del CVRP. La metodología se basa en la metaheurística Búsqueda Tabú, y además emplea el algoritmo de ahorros de Clark y Wright para la generación de soluciones iniciales y la técnica de Lin Kernighan para la construcción del vecindario de las soluciones. Con el fin de comparar solución para el CVRP generada por el algoritmo propuesto con la mejor solución determinada en la literatura, se empleó un conjunto de problemas seleccionados por los autores, y se desarrolló el método en una plataforma Java. Y a partir de la respuesta generada por el algoritmo propuesto en dos problemas, se empleó el diseño de experimentos para determinar los factores relevantes en la calidad de la solución.

En el **2011 Galvis Mejia, Yessica Tatiana**; **Jaimes Momsalve**, **Gladys Andrea** y **Quiroga Berdugo, Nataly Jhoana**[47], estudiaron la distribución de correspondencia, la distribución de almuerzos de la división de Bienestar Universitario y la recolección de residuos sólidos de la Universidad Industrial de Santander como aplicaciones del problema de ruteos de vehículos, implementando para cada caso métodos exactos como Branch and Bound, algunas heurísticas como el algoritmo del vecino más cercano, el algoritmo de barrido, el algoritmo de ahorros y el algoritmo 2 Opt, y la metaheurística Búsqueda Tabú.

[46] PERTUZ MONTENEGRO, Alfredo José y ROJAS SILVA, Kimberly. Formulación y evaluación de un algoritmo basado en la metaheurística búsqueda tabú para la optimización del Problema de Ruteo de Vehículos con Capacidad. Trabajo de grado Ingeniero Industrial. Bucaramanga: Universidad Industrial de Santander. 2007.

[47] GALVIS MEJIA, Yessica Tatiana; JAIMES MOMSALVE, Gladys Andrea y QUIROGA BERDUGO, Nataly Jhoana. Estudio cuantitativo de tres aplicaciones diferentes del problema de ruteo de vehículos (VRP) en la Universidad Industrial de Santander. Trabajo de grado Ingeniero Industrial. Bucaramanga: Universidad Industrial de Santander. 2011.

Las heurísticas y la metaheurística se desarrollaron en el software Matlab y se utilizó la herramienta Solver de Excel en la implementación del método exacto. Los resultados arrojados por cada uno de las metodologías se compararon con el fin de determinar la mejor solución para cada caso estudiado. En el caso de la distribución de correspondencia la mejor solución arrojada fue por la heurística 2 Opt. En la aplicación de distribución de almuerzos casi todos los métodos generaron soluciones óptimas, y la ruta generada en la recolección de basuras por la técnica Branch and Bound presentó una reducción de longitud.

Gómez Atuesta, David Fernando y **Rangel Carvajal, Carlos Eduardo** en el **2011**[48] formularon las metaheurísticas búsqueda tabú y recocido simulado para la solución del CVRP. El software usado para la programación de los algoritmos fue Matlab® versión R2009a.

Para el algoritmo de búsqueda tabú se programaron dos estrategias de vecindad: inserción e intercambio. Así mismo, para el algoritmo de recocido simulado se programaron dos estrategias cada una de ellas contenía una función de enfriamiento diferente: una decrecía linealmente y la otra decrecía de acuerdo a un factor definido. Ambos algoritmos usaban como solución inicial la mejor solución encontrada por las heurísticas: Clarke and Wrigth y Vecino más cercano.

Además, realizaron un diseño de experimentos con 10 problemas encontrados en las librerías concluyendo que: en el algoritmo de búsqueda tabú el tratamiento tabú intercambio arroja resultados de menor calidad, el factor de tamaño de la lista tabú no tiene un efecto significativo en la respuesta dada por el algoritmo de búsqueda tabú al igual que el número de iteraciones y la función de enfriamiento en el algoritmo de recocido simulado.

[48] GÓMEZ ATUESTA, David Fernando y RANGEL CARVAJAL, Carlos Eduardo. Formular las metaheurísticas búsqueda tabú y recocido simulado para la solución del CVRP (Capacitated Vehicle Routing Problem). Tesis de grado para Ingeniero Industrial. Universidad Industrial de Santander. 2011.

Lozada Díaz, Adriana y **Cadena González, Ricardo Andrés** en el **2012**[49] presentan una investigación donde se estudiaron y sistematizaron cinco heurísticas de construcción para dar solución al problema de ruteo de vehículos con ventanas de tiempo. La eficiencia de los algoritmos se midió con instancias de tamaño de 100 nodos propuestas por Solomon (1987), en función de la distancia recorrida en la ruta.

Los algoritmos programados fueron: algoritmo de ahorros de Clarke y Wright en versión secuencial, algoritmo del vecino más cercado con enfoque temporal, heurística de inserción I1, heurística de inserción I2, y la heurística de inserción I3 de Solomon.

2.4 PROBLEMA DE RUTEO DE VEHÍCULOS ESTOCÁSTICO

El problema de ruteo de vehículos estocástico (SVRP) surge cuando algunos parámetros del VRP son aleatorios. "Los ejemplos más comunes son las demandas estocásticas y tiempos de viaje estocásticos. Los métodos de solución de estos problemas son considerados más complejos que para el VRP determinista, debido a que combinan características de los programas estocásticos y enteros"[50].

En el marco de la programación estocástica, el SVRP es modelado en dos etapas. En una primera etapa, se determina una solución planificada o "a priori". Después de que las variables aleatorias son reveladas, en una segunda etapa, se aplica una acción de recurso o acción correctiva, que por lo general, genera un costo o

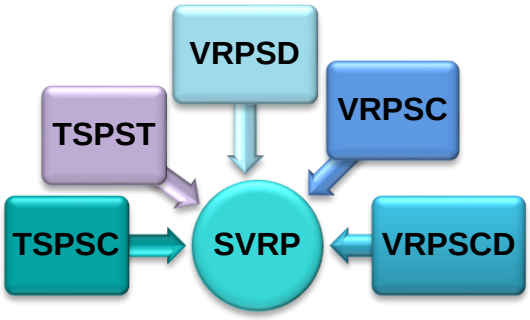
[49] LOZADA DÍAZ, Adriana y CADENA GONZÁLEZ, Ricardo Andrés. Solución del problema de ruteo de vehículos con ventanas de tiempo (VRPTW). Tesis de grado para Ingeniero Industrial. Universidad Industrial de Santander. 2012.
[50] GENDREAU, Michel; LAPORTE, Gilbert y SÉGUIN, René. Invited Review: Stochastic vehicle routing. En: European Journal of Operational Research, 1996, vol. 88., P. 4

un ahorro que debe ser considerado en el diseño de la solución de la primera etapa.

Las dos formulaciones básicas de programas estocásticos son: “programas limitados de oportunidad (CCP) y los programas estocásticos con recurso (SPR). En CCP, la probabilidad de fallo en la primera etapa está limitada a estar por debajo de un umbral dado, y no se tiene en cuenta los costos de las acciones correctivas en caso de fallo. En los SPR, el objetivo es determinar en la primera etapa, una solución que minimice el costo esperado de la segunda fase de solución”[51]. Estos costos se componen de: costo de la solución de la primera fase, más el costo neto esperado de recurso. Otro enfoque para modelar el SVRP es a través de un proceso de decisión de Markov. Algunos métodos de modelización más recientes incluyen la programación neurodinámica o metodología de aprendizaje por refuerzo.

2.4.1 Variantes del SVRP. Los problemas SVRP han sido estudiados en diferentes formulaciones. En la Figura 6 se presenta un esquema general de las variantes del SVRP.

Figura 5. Variantes del SVRP



[51] LARSEN, Allan, y MADSEN, Oli BG. The dynamic vehicle routing problem. En: Diss. Technical University of DenmarkDanmarks Tekniske Universitet, Department of TransportInstitut for Transport, Logistics & ITSLogistik & ITS, 2000.P. 27.

- ***The Traveling Salesman Problem with Stochastic Customers (TSPSC):*** El TSPSC se define en un gráfico $G = (N, A)$, donde N, es el conjunto de nodos que representan las ciudades a ser visitadas por el agente viajero, y A es el conjunto de arcos que detalla la secuencia en que serán visitadas estas ciudades. Cada nodo i está presente con una probabilidad P_i. En una primera etapa de la solución, se construye un recorrido para todos los nodos. El conjunto de nodos presentes es revelado a continuación. En la segunda etapa de la solución, los nodos que no están presentes se pueden saltar y el vendedor puede viajar al siguiente nodo. "El problema consiste en encontrar un recorrido de longitud mínima esperada. El TSPSC se introdujo por Jaillet en 1985, que describe una serie de modelos matemáticos, límites y propiedades teóricas"[52]. Las aplicaciones del PTSP se pueden presentar, cuando se realizan entregas diarias a un conjunto de clientes que no requieren el servicio todos los días.

- ***The Traveling Salesman Problem with Stochastic Travel Times (TSPST):*** "En este problema se conocen los clientes, pero la longitud de los arcos varía, es decir, los coeficientes $C_{ij,}$ que representan los tiempos de viaje son variables aleatorias. El objetivo es por lo general maximizar la probabilidad de completar el recorrido en un plazo determinado"[53].

- ***The Vehicle Routing Problem with Stochastic Demands (VRPSD):*** En el VRPSD la demanda de los clientes son variables aleatorias y se supone que son independientes. Tillman (1969) fue el primero en proponer un algoritmo para resolver este problema en casos donde hay muchos vehículos y se incurre en penalizaciones cuando la demanda de la ruta excede la capacidad del vehículo.

[52] GENDREAU, Michel; LAPORTE, Gilbert y SÉGUIN, René. Op. cit., p 6.
[53] LARSEN, Allan, y MADSEN, Oli BG. Op. cit., p. 27.

- ***The Vehicle Routing Problem with Stochastic Customers (VRPSC)****:* El VRPSC es una extension del TSPSC. Se caracteriza porque los clientes se encuentran con una probabilidad P_i pero tienen demandas determinísticas. La capacidad del vehículo se debe respetar, sin embargo cuando su capacidad es alcanzada se devuelve al depósito para el reabastecimiento. Al igual que en el TSPSC, los clientes ausentes se omiten en la segunda etapa de la solución.

- ***The Vehicle Routing Problem with Stochastic Customers and Demands (VRPSCD):*** El VRPSCD es una combinación del VRPSC y el VRPSD. Una importante definición del VRPSCD es propuesta por Bertsimas en 1992. En una primera etapa, se determina un conjunto de rutas las cuales inician y terminan en el depósito, con la restricción de que cada cliente puede ser visitado exactamente una vez. "El conjunto de clientes con una demanda cero, es decir, clientes ausentes, se revela poco a poco, pero la demanda positiva de los clientes restantes se da a conocer sólo cuando el vehículo llega a la ubicación del cliente"[54]. En la segunda etapa de la solución, las rutas obtenidas de la anterior etapa se siguen de acuerdo a lo planeado, con las siguientes dos excepciones: (1) los clientes ausentes se omiten; (2) cuando la demanda de la ruta es superior a la capacidad del vehículo, se devuelve al depósito para el reabastecimiento, y reanuda su recorrido a partir del último cliente visitado; si la demanda de cualquier cliente completa exactamente la capacidad del vehículo, éste regresa al depósito y reanuda su recorrido en el próximo cliente de la ruta planeada.

[54] GENDREAU, Michel; LAPORTE, Gilbert y SÉGUIN, René. Op. cit., p 9.

2.5 PROBLEMA DE RUTEO DE VEHÍCULOS CON DEMANDAS ESTOCÁSTICAS (VRPSD)

2.5.1 Definición del VRPSD. El VRPSD es un problema NP-hard, en el que la demanda de los clientes es de naturaleza estocástica, es decir, sólo se conoce su distribución estadística, y la demanda real es revelada en el momento que el vehículo llega al cliente. Debido a este comportamiento aleatorio, se pude presentar el caso en que se exceda la capacidad del vehículo, produciendo un "fallo de ruta", para lo cual es necesario tomar una acción correctiva o de recurso, que permita satisfacer la demanda de todos los clientes. El objetivo de este problema es determinar el conjunto óptimo de rutas con el mínimo costo, bajo las siguientes restricciones[55]:

- Todas las rutas comienzan y finalizan en el depósito.
- Cada vehículo tiene una capacidad de carga máxima, que se considera igual para todos los vehículos.
- Cada cliente tiene una demanda (estocástica) que debe ser servida.
- Cada cliente es atendido por un solo vehículo.
- Un vehículo no puede visitar dos veces al mismo cliente, sin incurrir en un costo de penalización.

Comúnmente, el VRPSD es modelado como un problema de dos etapas. En la primera etapa, se especifica una ruta para cada vehículo, conocidas como rutas a priori, y las demandas de los clientes son reveladas en cada parada a lo largo de las rutas planeadas. En la segunda etapa, se especifica la ruta real de cada vehículo, teniendo en cuenta las acciones de recurso que se implementaron

[55] JUAN, Angel A, et al. Applying Simulation and Reliability to Vehicle Routing Problems with Stochastic Demands. En: *XI Conference of the Italian Association for Artificial Intelligence (AIIA09), Reggio-Emilia (Italia)*. 2009. P. 2

cuando se presentaron fallos de rutas. En conclusión, el objetivo del VRPSD es diseñar un conjunto de rutas planificadas, minimizando la suma de los costos de las rutas previstas y costos esperados de la acción de recurso.

2.5.2 Políticas de servicio. La mayoría de las investigaciones sobre el VRPSD pueden clasificarse ya sea como un enfoque de optimización a priori o como un enfoque de reoptimización.

a. ENFOQUE DE OPTIMIZACIÓN A PRIORI*:* También conocido como política de servicio estática. Bajo este enfoque se diseñan rutas antes de conocer las demandas reales, y en caso de que se presente un fallo de ruta se requieren acciones de recurso, sin necesidad de modificar la secuencia planeada.

Dentro del enfoque de optimización a priori se plantean las siguientes estrategias de recurso:

- *"**Stewart et al.*** en ***1983*** proponen dos modelos sencillos para incorporar sanciones a la función objetivo en los casos de fallo de ruta. En el primer modelo se impone una penalización fija para cada ruta fracasada. En el segundo, se determina una multa por cada unidad demandada en exceso de la capacidad del vehículo"[56]. Ambas versiones suponen que todos los clientes deben ser visitados en la ruta y por lo tanto la función objetivo incluye el coste total determinista de la ruta más el término de penalización.

- *"**Dror*** y ***Trudeau*** en ***1986***, plantean que después de un fallo de ruta, todos los demás clientes deben ser atendidos con entregas directas individuales desde el depósito"[57].

[56] NOVOA, Clara., et al. A set-partitioning-based model for the stochastic vehicle routing problem. En: Lehigh University, Bethlehem, 2006, p. 4.
[57] Ibid., p. 4

- ***Estrategia de recurso tradicional*:** Propuesta por **Dror et al**. en **1989.** Se asume que cuando se presenta un fallo de ruta, los vehículos retornan al depósito para descargar / reponer capacidad y luego regresar a la posición de fracaso para seguir la ruta prevista.

- ***Chepuri*** y ***de Mello*** en el ***2005,*** proponen una acción de recurso que consiste en terminar la ruta en caso de que se presente un fallo. Evalúan la penalización como la pérdida de ingresos y / o los gastos por no servir al cliente. Este recurso es práctico cuando las entregas son de emergencia, es decir, el plazo en que la demanda se debe cumplir es crítico y no hay tiempo de volver al depósito para el reabastecimiento.

- ***Estrategia de reabastecimiento preventivo:*** También conocida como la política mixta, debido a que combina elementos de las políticas estática y dinámica. En esta estrategia, el vehículo visita a los clientes en el orden dado por la ruta a priori, y tiene que elegir, de acuerdo a la demanda del cliente actual, si procede al siguiente cliente o realiza un viaje al almacén para el reabastecimiento. "Algunas veces la mejor opción es elegir el reabastecimiento, así el vehículo no este vacío, o si su capacidad es mayor que la demanda esperada del próximo cliente programado, esta acción se llama reabastecimiento preventivo"[58]. El objetivo de esta estrategia es evitar que la carga del vehículo no sea suficiente para atender a un cliente, y por lo tanto se tenga que realizar un viaje de ida y vuelta al depósito para completar la entrega.

- ***Estrategia de recurso ampliada:*** Bajo esta esta estrategia, se permiten dos nuevas acciones de recurso cuando se da un fallo de ruta. "En primer lugar, un vehículo que ha completado su ruta original y tiene capacidad

[58] BIANCHI, Leonora; et al. Metaheuristics for the Vehicle Routing Problem with Stochastic Demands. En: Parallel Problem Solving from Nature-PPSN. Enero, 2004. vol. 8, p. 450

restante, puede atender a los clientes de las rutas fallidas directamente sin retornar al depósito. En segundo lugar, un vehículo que experimenta un fallo de ruta puede volver al depósito y luego realizar un "viaje extra" para atender la demanda de los clientes faltantes de rutas fracasadas"[59].

- ***Estrategia de recurso de parejas de vehículos:*** **Ak** y **Erera** en el **2006**, muestran una estrategia de recurso que elimina el supuesto de que los vehículos operan independientemente en rutas a priori. El estudio presenta dos vehículos en el cual uno de ellos sirve a una cantidad grande de demanda y otro a una cantidad más baja. "En caso de que la capacidad del primer vehículo sea superada por la demanda, el segundo vehículo añade cualquier cliente sin servicio al final de su recorrido. Si el segundo vehículo no está en condiciones de servir a todos los clientes adicionales en su ruta original, entonces la estrategia de **recurso tradicional** se utiliza para servir a estos clientes"[60].

b. ENFOQUE DE REOPTIMIZACIÓN: También conocido como política de servicio dinámica. Bajo este enfoque no se planea anticipadamente las rutas, sino que cada vez que el vehículo llega a la ubicación del cliente y atiende su demanda, se decide sobre el próximo cliente a visitar o si se regresa al depósito para el reabastecimiento. Este problema es modelado como un proceso de decisión de Markov, donde la elección del siguiente lugar a visitar depende del estado actual del sistema.

En un estudio presentado por Dror, Moshe; *Laporte, Gilbert* y *Pierre, Truedeau* en *1989*[61]*,* destacan dos políticas de servicio principales:

[59] NOVOA, Clara., et al. Op cit., p. 3.
[60] Ibid., p. 4.
[61] DROR, Mosher; LAPORTE, Gilbet y TRUDEAU, Pierre. Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks. En: Transportation Science. Agosto, 1989. Vol. 23, no.3,. p. 168.

- ***Entrega completa:*** Se asume que cuando un vehículo llega a la ubicación de un cliente debe servir toda su demanda. Por lo tanto, es necesario en este caso, hacer una suposición adicional $\xi_i \leq Q(i = 1, \dots, n)$, de otro modo, el problema no es factible. Esta política es la más usada en la literatura del VRPSD.

- ***Entregas divididas:*** Esta política se caracteriza porque la demanda de un cliente puede ser suministrada por varios vehículos o dividida entre varias visitas del mismo vehículo. Aquí se omite la restricción de que un cliente puede ser visitado exactamente una vez por el mismo vehículo. Además, no es necesario suponer que $\xi_i \leq Q(i = 1, \dots, n)$. Otra posibilidad que pertenece a este categoría es cuando la demanda se puede dividir, pero sólo en fracciones predefinidas.

Dror, Moshe; *Laporte, Gilbert y Pierre, Truedeau (1989)*[62]*,* también proponen una clasificación de las políticas de servicio de acuerdo al tiempo en el cual se conoce la demanda real del cliente.

- ***Información anticipada completamente:*** Todas las demandas son conocidas completamente antes de que se planifiquen las rutas de los vehículos, lo que conduce al VRP.

- ***Información de última:*** La demanda del cliente sólo se conoce después de que el vehículo llega a las instalaciones del cliente, lo que conduce al SVRP. Se pueden presentar dos casos: que la demanda sea conocida inmediatamente antes de la entrega o inmediatamente después de que se realice el servicio completo.

[62] Ibid., p. 168.

Relacionando estas dos políticas, con la anterior clasificación propuesta por los mismo autores. Se define que para la política de servicio de entrega completa, la demanda se conoce después de que el vehículo llega a la ubicación del cliente pero antes de la entrega. Para la política de entrega dividida, si no hay un servicio de mayor prioridad se permite el caso de información de última, donde la cantidad entregada, está determinada por la demanda y la cantidad residual en el vehículo en el momento que llega a la ubicación de cliente; esto significa que al iniciar el servicio, este finalizará cuando se entregue la demanda total o el vehículo quede vacío. Si hay un servicio de preferencia, entonces una nueva variable de decisión se debe adicionar. La cual determina el nivel máximo de reposición en cada visita a un cliente.

2.5.3 Métodos de solución del VRPSD. El primer trabajo de investigación presentado sobre el VRPSD fue elaborado por **Tillman** en **1969**[63], quien propuso un método basado en el algoritmo de ahorro de Clarke y Wirght. A partir de esta investigación se han desarrollado diversos métodos de solución para el VRPSD, además de la extensión de las políticas de servicio. A continuación se presentan algunas de las contribuciones al estudio del VRPSD.

Dror, Mosher; Laporte, Gilbet y **Trudeau, Pierre[64],** realizaron en **1989** una investigación que describe dos marcos de solución: programación estocástica y proceso de decisión de Markov. En la programación estocástica, el problema se modela de dos formas: programación estocástica restringida con oportunidad (Chance Constrainted Stochastic Programing) y programación con recurso

[63] SHEN, Zhihong; ORDÓÑEZ, Fernando y DESSOUKY, Maged M. The Stochastic Vehicle Routing Problem for Minimum Unmet Demand. En: Optimization and Logistics Challenges in the Enterprise Springer Optimization and Its Applications. 2009. vol, 30.p 353.

[64] DROR, Mosher; LAPORTE, Gilbet y TRUDEAU, Pierre. Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks. En: Transportation Science. Agosto, 1989. Vol. 23, no.3,.p. 166 - 176.

(Stochastic Programming with Recourse). Ambas programaciones manejan una política de servicio sin re-optimización, es decir, cuando ocurre una falla en la ruta, el vehículo viaja hasta el depósito antes de reanudar su viaje y no se realiza la planeación de una secuencia nueva con los clientes restantes.

El modelamiento del VRPSD a través de un proceso de decisión de Markov, consiste de un gran número de estados. El proceso de decisión se lleva a cabo en cada uno de los nodos, y comprende un estado inicial, donde el vehículo se encuentra en el depósito con n clientes por atender, y un estado final en el que todos los clientes han sido atendidos.

Secomandi, Nicola en **1998,** propone una formulación dinámica del VRPSD "como un problema del camino más corto estocástico (Stochastic Shortest Path Problem, SSPP) y un algoritmo exacto de Programación Dinámica (*Dynamic Programming*, DP) que puede resolver con optimalidad pequeñas instancias del VRPSD (alrededor de 10 clientes)"[65]. En el **2002**, **Secomandi**[66] estudia la aplicabilidad de la metodología de la programación neuro-dinámica (*Neuro Programation Dinamic*, NPD) a la formulación SSPP del VRPSD con un solo vehículo con el fin de calcular una política dinámica óptima. El autor compara el desempeño de dos algoritmos NDP: iteración de política aproximada optimista (Optimistic Approximate Policy Iteration, OAPI) y política de despliegue (*Rollout Policies*, RP). De los resultados computacionales obtenidos se concluyó que la OAPI mejora el rendimiento de una política de vecino más cercano en un 2,3%, sin embargo la RP genera soluciones de mayor calidad.

[65] SECOMANDI, Nicola. Comparing neuro-dynamic programming algorithms for the vehicle routing problem with stochastic demands. En: Computers & Operations Research., 2000. vol. 27. P. 1203.
[66] Ibid., p. 1201-1225

Yang, W; Mathur, K. y **Ballou, R. H** en el **2000**[67], investigaron el VRPSD con uno y múltiples vehículos. Probaron dos algoritmos heurísticos: primero enrutar y luego agrupar, y primero agrupar y después enrutar, comparándolos con soluciones del método exacto Branch and Bound, determinando que estos son eficientes y robustos para instancias de tamaño pequeño, de hasta 15 clientes. Así mismo, adaptan la búsqueda local Or-Opt, proponiendo un cálculo rápido de aproximación para el valor de la función objetivo de una solución modificada con un movimiento de búsqueda local.

Bianchi, Leonora, et al. en **2006**[68] determinaron que un aspecto importante para resolver el VRPSD es la implementación de funciones objetivo aproximadas que sean rápidas y efectivas en el proceso de búsqueda. Para ello tomaron en cuenta el concepto de ruta a priori en el contexto del VRPSD, y la política de reabastecimiento preventivo. Se cataloga a la ruta a priori como una solución factible para el VRPSD y el TSP, debido a que la función objetivo del problema del agente viajero puede ser definida como una aproximación factible de la función del VRP con demandas estocásticas, considerando que si en el VRPSD la capacidad del vehículo es infinita, este se convierte en un TSP.

La investigación muestra el estudio de la hibridación de cinco metaheurísticas: recocido simulado, búsqueda tabú, búsqueda local iterada, optimización de colonias de hormigas y algoritmos evolutivos, intercambiando la función objetivo exacta del VRPSD por dos aproximaciones (VRPSD aproximation- TSP aproximation). Se trabajaron dos tipos de hibridación para cada metaheurística: una con un algoritmo de búsqueda local (Or-Opt) aplicado a la aproximación VRPSD y a la aproximación TSP. Y otra con la analogía TSP (TSP analogy)

[67] YANG, Wen-Huei; MATHUR,Kamlesh y BALLOU, Ronald H.Stochastic vehicle routing problem with restocking.En: Transportation ScienceTrans. Sci. 2000. vol, 34. No, 1. p. 99–112.
[68] BIANCHI, Leonora, et al. Hybrid Metaheuristics for the Vehicle Routing Problem with Stochastic Demands. En: Journal of Mathematical Modelling and Algorithms. 2006. p. 91-110.

utilizando el operador de búsqueda local 3-opt, efectiva para el TSP, pero no para el VRPSD.

K.C. Tan, C.Y. Cheong, y **C.K. Goh,**[69]en el **2007** estudiaron un algoritmo evolutivo multiobjetivo (*Multiobjective Evolutionary Algorithm*, MOEA) para resolver el problema del VRPSD con restricciones de capacidad y de tiempo. Mediante la aplicación de los conceptos de la optimalidad de Pareto, el algoritmo permite resolver el VRPSD minimizando simultáneamente tres objetivos: el recorrido, la remuneración del conductor, y el número de vehículos necesarios, y así mismo, permite mejorar la solución y explorar aún más el espacio de búsqueda.

En el algoritmo propuesto la búsqueda de nuevas soluciones se realiza mediante dos procesos: cruce, que consiste en la combinación de genes de diferentes cromosomas, y mutación, que altera los cromosomas existentes de la población. Debido a que utiliza una política de servicio a priori, en la búsqueda local se incorporan dos heurísticas: Shortest Path Search (SPS) y Which Directional Search (WDS). El SPS, aprovecha que los fallos de ruta tienen mayor probabilidad de ocurrencia al final de cada trayectoria, de este modo intenta cambiar el orden en que serán visitados los clientes disminuyendo los costos de transporte adicionales en que se puede incurrir. El WDS diseñado para aprovechar el hecho de que el costo de transporte previsto de una ruta depende de la dirección en que se recorre, por ello construye una nueva ruta que se ejecuta en la dirección opuesta, eligiendo la de menor costo.

Wang, Dedong; Chen, Qijun y **Lili**, en el **2008**[70] presentaron un estudio sobre un tipo de SVRP, en donde el número de clientes y la demanda de los clientes son

[69]TAN, K.C.; CHEONG, C.Y. y GOH, C.K. Solving multiobjective vehicle routing problemwith stochastic demand via evolutionary computation.En:European Journal of Operational Research, 2007, vol177, p.813–839

[70]WANG, Dedong; CHEN, Qijun y Lili. Stochastic Vehicle Routing Problems and their Solution Algorithm. En: International Conference on Automation and Logistics. (Septiembre, 2008: Qingdao,

estocásticos VRPSCD (*Vehicle Routing Problem with Stochastic Customers and Demands*). En este problema sólo se sabe en el nodo precedente si aparecerá o no el siguiente cliente a ser visitado. Y cuando se presenta una falla en la ruta se asume que el vehículo debe retornar al depósito para cargar y reanudar el recorrido desde el punto precedente a la falla. La investigación plantea un método de búsqueda aleatoria en el que se aprovecha la fuerte capacidad de búsqueda local del algoritmo genético para evitar los errores del recocido simulado de converger lentamente y caer fácilmente en el mínimo local. Esto sucede gracias a que el algoritmo propuesto implementa la operación de herencia a cada generación que se obtiene mediante la operación de recocido simulado.

Young, Peng y **Ha-Ying, Zhu**[71] en el **2008** solucionan el problema de ruteo de vehículos con demandas estocásticas a partir del algoritmo DP - PSO con operador Inver-over. El PSO con operador Inver-over se utilizó para optimizar la ruta a priori. Y DP fue usado para calcular el valor de la función objetivo.

Novoa, Clara y **Storer, Robert**[72] en el **2009** examinan los algoritmos de programación dinámica aproximadas para el VRPSD desde una perspectiva dinámica o de re optimización. La política dinámica, que se eligió supone entregas divididas cuando se produce un fallo de ruta, esto quiere decir que el vehículo completa la demanda del cliente en otra visita no necesariamente inmediata. Este estudio se extiende al algoritmo de despliegue mediante la aplicación de diferentes secuencias de bases (es decir, soluciones a priori), políticas de pre análisis y esquemas de poda.

China). Memorias. Qingdao: Instituto de Ingenieros Eléctricos y Electrónicos – IEEE, 2008. p. 958 – 962.

[71] YOUNG, Peng y HA-YING, Zhu. Research on Vehicle Routing Problem with Stochastic Demandand PSO-DP Algorithm with Inver-over Operator. En: Systems Engineering — Theory & Practice, 2008, vol. 10 p. 76-81

[72] NOVOA, Clara y STORER, Robert. An approximate dynamic programming approach for the vehicle routingproblem with stochastic demands. En: European Journal of Operational Research, 2009, vol. 196, p. 509–515.

Las principales contribuciones de este trabajo son: el desarrollo de un algoritmo de despliegue en dos etapas que permitió reducir el costo en un 1,6%, el uso de la simulación de Montecarlo para el cálculo del costo esperado de las secuencias bases de actualización como una alternativa al cálculo exacto en Secomandi (1998, 2000, 2001) redujo el tiempo de cálculo en un 65%, y el desarrollo de mejores secuencias de bases y sistemas de poda que conducen a una reducción de costes de alrededor de 4% sobre métodos anteriores.

Ismail, Zuhaimy y **Irhamah** en el **2010**[73], presentaron una modelación para un problema de diseño de rutas de recolección de basuras sólidas, basada en el VRPSD. Proponen un híbrido (HGATS) entre algoritmo genético y búsqueda tabú (*Tabu Search*, TS), combinando la ventaja del método basado en población (GA) para identificar áreas prometedoras en el espacio de búsqueda y la capacidad de exploración en el espacio de búsqueda del método de trayectoria (TS). En el estudio se encontró que para problemas de tamaño pequeño, el híbrido y el algoritmo genético presentan superioridad en la calidad de las soluciones en comparación con la búsqueda tabú, aunque el HGATS sea más robusto. Este también muestra su ventaja para problemas de tamaño grande, puesto que obtiene mejores soluciones que el GA.

Goodson, Justin C.; Ohlmann, Jeffrey W. y **Thomas, Barrett W**[74] en el **2012** diseñaron un vecindario utilizando la codificación de solución de orden cíclica de Ryan et al. (1993) para el VRP. Para resolver el VRPSD se integran los vecindarios de orden cíclico en un procedimiento de recocido simulado. Se emplea la política a priori de Laporte et al. (2002), por tanto el costo esperado de la ruta es

[73]ISMAIL, Zuhaimy y IRHAMAH.Genetic Algorithm and Tabu Search for Vehicle Routing Problems with Stochastic Demand.En: International Conference on Mathematical Science. (2010). Instituto Americano de Física. p. 488 – 504.

[74] GOODSON, Justin C.; OHLMANN, Jeffrey W. y THOMAS, Barrett W. Cyclic-order neighborhoods with application to the vehicle routing problemwith stochastic demand. En:European Journal of Operational Research, 2012, vol. 217, p. 312–323.

calculado a partir de la suma de los gastos de viaje previsto y el costo esperado por los fallos de ruta.

El procedimiento de recocido simulado se ejecuta en dos fases. En la primera fase se resuelve el VRP clásico, por lo tanto el costo de la ruta candidata para un determinado orden cíclico es calculado teniendo en cuenta únicamente los gastos esperados del viaje. En la segunda fase, se mejora la solución incluyendo los costos por las acciones de recurso.

Marinakis, Yannis; Lordanidou, Georgia-Roumbini y **Marinaki, Magdalene Marinaki**[75] en el **2013,** incorporan en un esquema híbrido: la optimización de enjambre de partículas, las metaheurísticas de búsqueda local, 2-opt y 3-opt y la estrategia de volver a vincular el camino. En el PSO en cada iteración se debe calcular dos variables para cada miembro de la población: la posición y la velocidad. Para la ecuación de velocidad de las partículas se utilizaron diferentes enfoques determinándose que la optimización de enjambre de partículas de constricción fue la que mejor funcionaba. Además, la política de servicio se fundamentó en el valor límite de la carga residual.

2.5.3.1 Métodos de solución para el VRPSD basados en ACO. ACO inicialmente fue aplicado al problema del agente viajero. A partir de esta implementación, esta metaheurística se ha utilizado para determinar la solución a diferentes problemas de optimización como: “problemas de asignación cuadrática, problemas de ordenamiento secuencial, problemas de coloración de grafos y problemas de enrutamiento de vehículos”[76]. Otras aplicaciones destacables se

[75] MARINAKIS, Yannis; LORDANIDOU, Georgia-Roumbini y MARINAKI, Magdalene Marinaki Particle Swarm Optimization for the Vehicle Routing Problem with StochasticDemands. En:Applied Soft Computing, 2013, vol. 13, p. 1693–1704.
[76]BLUM, Christian. Ant colony optimization: Introduction and recent trends. En: Physics of Life Reviews. 2005. vol, 2. p, 363.

presentan en el “diseño de redes de comunicación, en la bioinformática y en la biomedicina”[77].

Dentro de las aplicaciones a los problemas de ruteos de vehículos, se encuentra las empleadas al VRPSD, debido a que permiten modelar diferentes situaciones que surgen en la gestión empresarial y el servicio de transporte. A continuación se presentan algunas contribuciones para la solución del VRPSD implementando la metaheurística ACO.

Reimann, Marc, en el **2005**[78], presenta un estudio donde se resuelve el VRPSD como un programa estocástico con recurso, utilizando una versión modificada del ahorro basado en el algoritmo de optimización colonia de hormigas propuesto por Tillman (1969). La estrategia de recurso que se asume consiste en que cada vez que el vehículo se quede sin carga, retorne al depósito para el reaprovisionamiento y continúe con la ruta planeada.

En el trabajo presentado se utilizaron dos modelos de riesgos para la estimación de este valor real de la demanda, uno basado en la demanda y otro basado en la capacidad del vehículo. Además, para evaluar la solución construida por cada hormiga, se comparan dos enfoques; analítico y etiquetado de Monte Carlo. Con el primero, se realiza una aproximación analítica de los costos de las rutas planeadas, y con el segundo la evaluación se realiza a través de una muestra *SampeSize* de diferentes demandas. Se concluye que se obtienen mejores soluciones con este último enfoque.

[77] Ibid., p. 363.
[78] REIMANN, Marc. Analyzing a vehicle routing problem with stochastic demands using ant colony optimization. En: Advanced OR and AI Methods in Transportation, Publishing House of Poznan University of Technology, 2005. p. 764-769.

Teodorovic, Dusan y **Lucic, Panta** durante el **2007**[79] dieron a conocer el Sistema de Hormigas Fuzzy (*Fuzzy Ant System*, FAS), una modificación del cálculo de las probabilidades de transición de la metaheurística colonia de hormigas. Básicamente el modelo desarrollado se fundamente en las reglas aritméticas Fuzzy, lógica Fuzzy y Sistema de hormigas. Esto implica el uso de un algoritmo de razonamiento aproximado que puede reemplazar la ecuación original para calcular las probabilidades de transición.

Para resolver el VRPSD a través del FAS, la regla de transición considerada se lleva a cabo por dos etapas principalmente. Primero se disminuye las dimensiones del problema utilizando un algoritmo aproximado, empleado para elegir un grupo de nodos candidatos a ser visitados por la hormiga. En esta etapa la capacidad disponible, el incremento de la longitud de la ruta y la preferencia por el nodo, se consideran como números fuzzy triangulares. En el segundo paso, el nodo a ser visitado por la hormiga es determinado desde el conjunto de nodos candidatos, a través de un segundo algoritmo de razonamiento aproximado. Y la distancia entre la ubicación actual de la hormiga, el camino de feromona y la utilidad de la hormiga por elegir un nodo, se asumen como números fuzzy triangulares.

Tripathi, **Mukul**; **kuriger**, **Glenn** y **wan**, **Hung-da** en **2009**[80], introducen una metodología que combina la simulación y un método mejorado de la metaheurística optimización colonia de hormigas para resolver el VRPSD. Básicamente el algoritmo de optimización se emplea para optimizar los parámetros de configuración en el modelo de simulación y determina las configuraciones en cada iteración. Emplean un algoritmo de hormiga adaptativo integrado de búsqueda vecindario (n*eighborhood-search embedded Adaptive Ant*

[79]TEODOROVIC, Dusan y LUCIC, Panta. The fuzzy ant system for the vehicle routing problem when demand at nodes is uncertain. En: International Journal on Artificial Intelligence Tools. 2007. vol, 16. no, 5,. p. 751 – 770

[80]TRIPATHI, Mukul; KURIGER, Glenn y WAN, Hung-da. An ant based simulation optimization for vehicle routing problem with stochastic demands. En: Actas de la Conferencia de Simulación de Invierno. San Antonio: Universidad de Texas. 2009. P, 2476-2487.

Algorithm, ns-AAA), para mejorar la capacidad de búsqueda del algoritmo a través del uso de una técnica de barrido que realiza movimientos compuestos. La política de servicio que se considera en la investigación es el reabastecimiento preventivo. Se realizaron una serie de experimentos de acuerdo a un banco de pruebas, para estudiar el efecto de siete parámetros seleccionados sobre el valor de la función objetivo. Además, se compara el desempeño del algoritmo propuesto con la metaheurística ACO y un algoritmo genético. Los resultados experimentales muestran que el ns-AAA supera en rendimiento a los dos algoritmos.

Moghadam, **Sadjadi** y **Seyedhosseini** en **2010**[81], compararon la solución robusta que genera el procedimiento Branch and Cut, con la solución determinada por la metaheurística sistema colonia de hormigas. El algoritmo ACS se fundamenta en la nivelación de la capacidad no utilizada de cada ruta, calculada mediante parámetros determinísticos y empleada para la actualización de feromona. La política de servicio seleccionada establece que el vehículo debe regresar al depósito una vez que se haya cumplido la restricción de capacidad del mismo, o cuando todos los clientes sean visitados. Se utilizaron entre 5 y 100 nodos para evaluar los métodos de solución. Según los resultados el ACS propuesto en el estudio proporciona una solución aproximadamente igual a la solución exacta para el problema con tamaños grandes en un tiempo computacional aceptable.

En la Escuela de Estudios Industriales y Empresariales de la Universidad Industrial de Santander se han desarrollado diferentes investigaciones sobre el Problema de Ruteo de Vehículos. Hasta el momento la investigación presentada por la ingeniera Silvia Adriana Galván Núñez es el único estudio del VRP con demandas estocásticas, donde se implementó un sistema evolutivo de optimización de enjambre de partículas para la solución de este problema.

[81]MOGHADAM, B. FARHANG; Sadjadi, S. J. y SEYEDHOSSEINI, S. M. Comparing mathematical and heuristic methods for robust vehicle routing problem. En: International Journal of Research and Reviews in Applied Sciences. Febrero, 2010. vol,2. no,2. p, 108 – 116.

3 OPTIMIZACIÓN COLONIA DE HORMIGAS

Básicamente la metaheurística toma como fuente de inspiración el comportamiento de las hormigas reales para encontrar caminos más cortos entre las fuentes de alimento y el nido. Las hormigas gracias a que son insectos sociales, trabajan a favor de la supervivencia de la colonia, y emplean un mecanismo de *comunicación indirecta* mediante rastros de feromonas para influir en el comportamiento de las demás hormigas, a este mecanismo se le conoce como *estigmergía*.

El modelamiento del algoritmo se basa en el concepto de *estigmergía artificial*, definido por Dorigo y Di Caro. La *feromona artificial* se considera como una *variable estigmergica* encargada de guardar la información empleada por las hormigas para comunicarse, además imita la evaporación de feromona real, disminuyendo el rastro que deja una hormiga cuando se desplaza por el camino construido. Cuanto mayor sea la cantidad de feromona depositada en un camino, mayor será la probabilidad de que una hormiga lo elija.

3.1 VARIANTES DEL ACO

3.1.1 Sistema de hormigas. Dorigo et al. en 1996 desarrollaron el primer algoritmo denominado sistema de hormigas. En este algoritmo la probabilidad de transición $\boldsymbol{p_{ij}^{k}}$ incluye información heurística y una lista tabú, relacionada con la capacidad de memoria.

Probabilidad de transición desde el nodo i hacia el nodo j:

$$p_{ij}^{k}(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{u \in N_i^k(t)} \tau_{iu}^{\alpha}(t)\eta_{iu}^{\beta}(t)} & si\ j \in N_i^k(t) \\ 0 & si\ \ j \notin N_i^k(t) \end{cases}$$

τ_{ij} representa la eficacia a-posteriori del movimiento desde i a j, expresada en términos de la concentración de feromona en el arco (i,j). η_{ij} constituye la eficacia a priori del desplazamiento desde el nodo i al nodo j, es decir, la deseabilidad del movimiento, que depende de la información del problema.

La probabilidad de transición propuesta en AS, plantea un balance entre la información heurística η_{ij} y la concentración de feromona τ_{ij}, a partir de la determinación de dos parámetros α y β. Si $\alpha = 0$**,** la información sobre la intensidad de feromona no se utiliza, por tanto la experiencia de búsqueda se descuida. Si $\beta = 0$ la deseabilidad del movimiento no se toma en cuenta.

En el AS se incorporan en N_i^k todos los nodos que aún no han sido visitados por la hormiga k. Cuando una hormiga visita un nodo, este es agregado a la lista tabú de la hormiga, y eliminado del conjunto N_i^k para asegurar que ningún nodo es visitado más de una vez.

La actualización de la feromona en cada arco se realiza después de que cada una de las hormigas ha construido un camino.

Actualización de feromona*:*

$$\boldsymbol{\tau}_{ij}\,(t+1) = (1-\rho)\,\tau_{ij}\,(t) + \Delta\boldsymbol{\tau}_{ij}(t)$$

$$\Delta\boldsymbol{\tau}_{ij}(t) = \sum_{k=1}^{n_k} \Delta\boldsymbol{\tau}_{ij}^{k}(\mathrm{t}),$$

Donde:

$\tau_{ij}\,(t)$, es la concentración de feromona que se encuentra en el arco (i,j).

ρ, es el factor de evaporación. Toma valores entre 0 y 1.

$\Delta\tau_{ij}^{k}(t)$ es la cantidad de feromona depositad en el arco (i,j) por la hormiga k en el tiempo t.

$$\Delta\boldsymbol{\tau}_{ij}^{k}(\mathrm{t}) = \begin{cases} \dfrac{\mathrm{Q}}{\mathrm{f}\left(\mathrm{x}^{\mathrm{k}}(\mathrm{t})\right)}; & \text{si el arco } (\mathrm{i},\mathrm{j}) \text{ se encuentra en el camino } \mathrm{x}^{\mathrm{k}}(\mathrm{t}). \\ 0; & De\ lo\ contrario. \end{cases}$$

Dependiendo del objetivo que se quiera alcanzar las hormigas pueden ser ubicadas de dos maneras posibles. Si el objetivo es encontrar el camino más corto entre el nodo origen y el nodo destino, todas las hormigas n_k se ubican en el nodo de origen. Pero si se quiere construir una ruta que conecte todos los nodos una sola vez, las hormigas se localizan de manera aleatoria en todos los nodos del grafo. Esta última opción permite que el algoritmo mejore la capacidad de exploración de búsqueda.

3.1.2 Sistema colonia de hormigas. Para mejorar la capacidad de búsqueda del algoritmo Sistema de Hormigas, Dorigo y Gambardella sugirieron el sistema colonia de hormigas (ACS) en 1997. El cual difiere del AS en las siguientes características:

1. “Emplea una probabilidad de transición diferente.
2. Define una regla de actualización de feromona diferente.
3. Introduce la actualización de feromona local.

4. Lista de candidatos son implementadas para favorecer a nodos específicos"[82].

La regla o probabilidad de transición utilizada en ACS, se desarrolló con el propósito de generar un balance entre la explotación y exploración del algoritmo.

Regla de transición:

$$j = \begin{cases} argmax_{u \in N_i^k(t)} \left\{ \tau_{iu}(t) \eta_{iu}^{\beta}(t) \right\}; si\ r\ \leq r_0 \\ J;\ si\ r\ > r_0 \end{cases}$$

Donde $r \sim U(0,1)$, y $r_0 \in [0,1]$ es un parámetro especificado por el usuario; $N_i^k(t)$ es un conjunto de nodos válidos para ser visitados; $J \in N_i^k(t)$ y es un nodo seleccionado aleatoriamente a partir de la siguiente probabilidad:

$$p_{iJ}^k = \frac{\tau_{iJ}(t) \eta_{iJ}^{\beta}(t)}{\sum_{u \epsilon N_i^k} \tau_{iu}(t) \eta_{iu}^{\beta}(t)}$$

Para establecer el balance entre la exploración y la exploración en la regla de probabilidad se emplea el parámetro r_0. Su funcionamiento se presenta de la siguiente forma: si $r\ \leq r_0$ el algoritmo realiza la explotación a favor de los mejores arcos. Si $r\ > r_0$ se ejecuta la exploración en busca de nuevas soluciones. En esta regla $\alpha = 1$, por tanto se decide eliminar de la formulación.

En el ACS, sólo la mejor hormiga a nivel global puede reforzar las concentraciones de feromonas en los arcos de la ruta correspondiente (mejor camino). La regla de actualización global en ACS, permite que las hormigas busquen en la vecindad de la mejor solución encontrada hasta el momento, favoreciendo así la explotación.

[82] ENGELBRENCHT, Andries P. Computational Intelligence. 2 ed. South Africa: John Wiley & Sons, Ltda. 2007. ISBN 978-0-470-03561-0. p, 372.

Esta regla se aplica después de que todas las hormigas han construido una solución.

Regla de actualización global de feromona:

$$\tau_{ij}(t+1) = (1-\rho_1)\tau_{ij}(t) + \rho_1 \Delta\tau_{ij}(t)$$

Donde:

$$\Delta\tau_{ij}(t) = \begin{cases} \dfrac{1}{f\left(x^+(t)\right)}; si\ (i,j)\ \in\ x^+(t) \\ 0; De\ lo\ contrario \end{cases}$$

Con: $f\left(x^+(t)\right) = |x^+(t)|$, en el caso de encontrar caminos cortos.

$x^+(t)$ representa la mejor solución. Dorigo y Gambrella determinaron dos métodos para seleccionar el camino $x^+(t)$.

1. ***Mejor iteración***: donde $x^+(t)$ indica el mejor camino encontrado en la iteración actual t. Denotado por $\tilde{x}$(t).
2. ***Mejor global*:** donde $x^+(t)$ representa el mejor camino hallado desde la primera iteración del algoritmo. Denotada por $\hat{x}(t)$.

ρ_1 indica tasa de evaporación de feromona. Para valores pequeños, las concentraciones de feromonas en los arcos se evaporan lentamente, disminuyendo la influencia de la mejor ruta. Mientras que con valores grandes, la cantidad de feromonas se evaporan rápidamente, generando que exista mayor influencia de la mejor ruta.

Además de la regla de actualización global de feromona, el algoritmo ACS emplea una actualización local.

Regla de actualización local de feromona:

$$\tau_{ij}(t+1) = (1-\rho_2)\tau_{ij}(t) + \rho_2\tau_0$$

Donde ρ_2 toma valores entre (0,1), y τ_0 es una constante positiva pequeña.

Otra característica del ACS que mejora el desempeño del algoritmo respecto a AS, es la definición del conjunto de nodos próximos a ser seleccionados. En el ACS el conjunto de nodos $N_i^k(t)$ contiene adicionalmente una lista de nodos candidatos, compuesta por nodos preferidos para ser visitados por primera vez.

3.1.3 Sistema de hormiga max-min. Stützle y Hoos propusieron el algoritmo sistema de hormiga Max-Min, con el propósito de evitar el estancamiento prematuro del AS, que se presenta cuando se explora y explota las concentraciones de feromonas más altas. Las características principales del MMAS son: la intensidad de feromonas está limitada por intervalos dados, sólo la mejor hormiga puede reforzar el rastro de feromona, las concentraciones iniciales de feromonas se determinan según el valor máximo permitido y adicionalmente se emplea un mecanismo que regula los rastros de feromonas**.**

La actualización de feromona se realiza de acuerdo a la regla planteada en ACS, sin embargo, el cálculo del $\Delta\tau_{ij}(t)$ se realiza tomando en cuenta dos métodos: *Mejor global y/o Mejor iteración*. Inicialmente el MMAS empleaba el primer método de cálculo*,* no obstante las últimas versiones de este algoritmo utilizan el *Mejor Global* con diferentes estrategias para actualizar las feromonas.

Estrategias Mejor Global:

- Emplea sólo el mejor camino encontrado desde la primera iteración del algoritmo ($\hat{x}(t)$), presentando la desventaja de limitar la exploración.
- Utiliza una estrategia mixta, para favorecer la exploración. Esta estrategia implementa el método *Mejor Iteración*, y además emplea periódicamente el procedimiento *Mejor Global*.
- "En el punto de estancamiento, todas las concentraciones de feromonas, τ_{ij} ,se reinicializan al valor máximo permitido, después de esto se utiliza el método *Mejor Iteración* para un número limitado de iteraciones

Por otra parte en el MMAS todas las concentraciones de feromonas se encuentran dentro de un rango definido previamente. En los planteamientos iniciales del MMAS, la concentración de feromona (τ_{ij}) de todos los arcos (i, j) pertenece a un intervalo definido $[\tau_{min}, \tau_{max}]$. Estos límites son parámetros estáticos dependientes del problema. El límite superior evita que la concentración de feromona pueda crecer en exceso, limitando la probabilidad de convergencia rápida al mejor camino. El límite inferior $\tau_{min} > 0,$ permite que todos los arcos tengan una probabilidad mayor a cero de ser incluidos en una solución, y también facilita la exploración.

En la inicialización del algoritmo MMAS, todas las concentraciones de feromonas toman el valor máximo permitido ($\boldsymbol{\tau_{min}, \tau_{max}}$). Además en el MMAS se emplea una estrategia de suavizado para disminuir la diferencia entre las concentraciones altas y bajas de feromonas, y evitar de esta manera el estancamiento del algoritmo.

4 ALGORITMOS GENÉTICOS

El método de solución a desarrollar en el trabajo de investigación se fundamenta en la metaheurística Optimización Colonia de Hormigas mejorada mediante la implementación de determinada operación propia de los algoritmos genéticos, con el propósito de aumentar la capacidad de exploración local. Los algoritmos genéticos son métodos inspirados en fenómenos que se encuentran en la naturaleza, como la selección y la genética natural. Estos algoritmos son eficaces para encontrar soluciones a problemas de optimización complejos como los NP-Hard.

La metodología básica del algoritmo genético inicia con una *población*, que es un conjunto inicial de *cromosomas* generados de manera aleatoria. Un cromosoma es una codificación de una solución, representado generalmente mediante una cadena de símbolos o números, cada uno de estos se denomina *gen.* Todos los cromosomas son evaluados por medio de una función *fitness,* que puede ser definida como la medida de la calidad de una solución candidata para el problema. De acuerdo a esta evaluación, el proceso de selección determina que cromosomas se deben reproducir. La reproducción se realiza mediante operadores de cruce y mutación, que generan nuevos individuos a partir de los existentes. Cada ciclo del algoritmo genera una nueva población de cromosomas conocida como *generación*. El algoritmo continúa su procedimiento hasta que se encuentra con un cromosoma que satisfaga las restricciones del problema o posea un nivel fitness suficientemente alto.

4.1 OPERADORES GENÉTICOS

Selección, cruce y mutación son los operadores genéticos empleados para asegurar la conservación de la carga genética de la población de generación en generación.

4.1.1 Selección. Se emplea para generar una nueva población de soluciones candidatas con mejores fitness, que formaran la próxima generación. Existen principalmente tres esquemas de selección: Selección ruleta, Selección de rango y selección torneo.

- **Selección Ruleta:** La selección ruleta también se le conoce como *selección proporcionada fitness*. En este método la probabilidad de que un individuo sea elegido para la reproducción es equivalente al fitness que posee. La búsqueda del método inicialmente se enfoca en la explotación de cromosomas con mayor fitnness, impidiendo la exploración de otros espacios de búsqueda, lo cual genera convergencia prematura.

- **Selección de Rango:** Se diseñó con el fin de evitar la convergencia de la selección ruleta. La selección de los cromosomas depende de una clasificación o rango asignado a cada individuo y no de sus fitness (aptitud) absolutos.

- **Selección Torneo:** Este método escoge aleatoriamente de la población de padres un subconjunto pequeño de cromosomas, pueden ser dos o tres, los cuales se comparan de acuerdo a sus fitness y se elige al cromosoma más apto de este subconjunto para reproducirse. El procedimiento continúa hasta que se obtengan todos los padres necesarios.

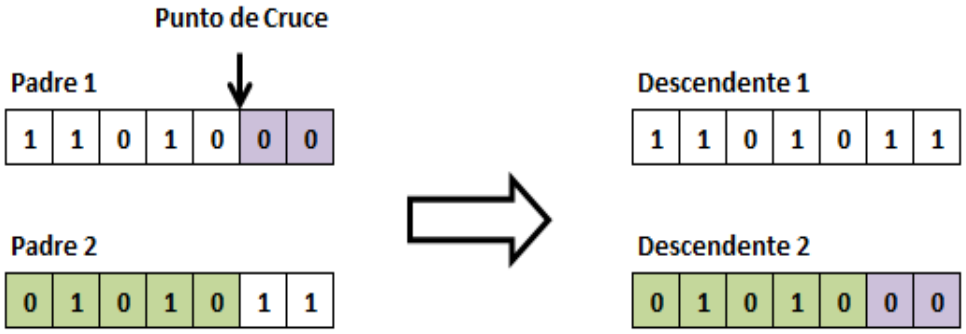
4.1.2 Operador cruce- crossover. El proceso de selección requiere de otros procedimientos para producir nuevas poblaciones con mejor carga genética. El operador de cruce es una técnica de recombinación empleada para la creación de uno o más descendientes a través del intercambio genético que poseen los padres. El operador crossover puede dividirse en tres categorías de acuerdo al número de padres empleados:

- **Asexual:** un padre genera una descendencia.
- **Sexual:** dos padres producen una o más descendencias.
- **Multi-recombinación:** más de dos padres se utilizan para generar una o más descendencias.

El proceso de recombinación puede realizarse a través de diversas técnicas que se han desarrollado en la literatura, las más empleadas son: cruce en un punto, cruce en dos puntos y cruce uniforme. La codificación más común utilizada en estos procedimientos es la binaria. A continuación se detallan y representan cada uno de estas técnicas:

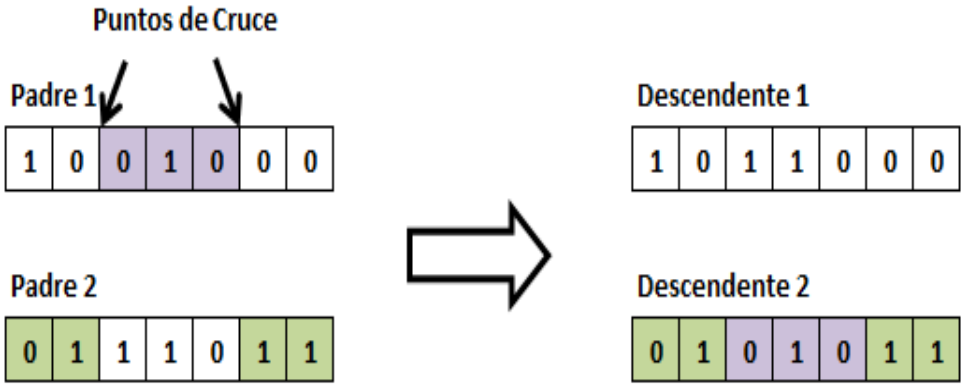
4.1.2.1 Cruce en un punto. En esta operación se seleccionan dos padres, y aleatoriamente se elige un punto entre dos genes (*punto de cruce*), en donde se realizará el corte de ambos cromosomas en dos partes. Entonces, la primera parte del primer padre y la segunda parte del segundo padre se unen para formar la primera descendencia. De manera similar, se combinan las partes aún no utilizadas o restantes para generar el segundo descendiente. De esta manera se obtienen dos individuos que pertenecen a la siguiente población. El proceso anteriormente descrito se ilustra en la Figura 7.

Figura 6. Cruce en un punto



4.1.2.2 Cruce en dos puntos. En este caso dos posiciones o puntos de cruce se seleccionan al azar, y las cadenas de genes que los conforman se intercambian, como se puede apreciar en la Figura 8. Se debe tomar en cuenta que ninguno de los puntos de cruce coincidan con los extremos de las cadenas para garantizar la combinación entre cromosomas. La primera descendencia se genera asignando el segmento central del segundo padre a los segmentos extremos del primer padre. La combinación de los demás segmentos produce la segunda descendencia.

Figura 7. Cruce en dos puntos



4.1.2.3 Cruce uniforme. Es una técnica especial, en la cual cada gen de la descendencia tiene las mismas probabilidades de pertenecer a cualquiera de los dos padres. El proceso se basa en la generación de una máscara de cruce con valores binarios. Al igual que en las técnicas anteriores se generan dos descendientes. El primer descendiente se forma de la siguiente manera: se toma del primer padre los genes que se encuentren en la posición 1 de la máscara, y los que se encuentren en la posición 0 se toman del segundo padre. Para producir el segundo descendiente, "se intercambian los papeles de los padres, o se intercambia la interpretación de los unos y los ceros de la máscara de cruce"[83]. El esquema de representación de un cruce uniforme se detalla en la Figura 9.

Figura 8. Cruce uniforme

[83] GESTAL POSE, Marcos. Introducción a los algoritmos genéticos [En línea]. Departamento de Tecnologías de la información y las comunicaciones. Universidad de Coruña. [Citado: 8 Junio 2013]. Disponible en: http://sabia.tic.udc.es/mgestal/cv/aaggtutorial/tutorialalgoritmosgeneticos.pdf

4.1.3 Mutación. La mutación es un operador que permite introducir material genético nuevo en un individuo existente, con el propósito de añadir diversidad a las características genéticas de la población. A menudo se utiliza en conjunto con el operador de cruce. Después de realizada esta operación, a cada uno de los genes de la descendencia se les aplica la mutación de acuerdo a cierta probabilidad *Pm*. La probabilidad de mutación usualmente toma valores entre 0 y 1. La mutación más común es el reemplazo aleatorio, el cual varía aleatoriamente un gen de un cromosoma.

Con la mutación se logra la diversidad o ampliación de la información contenida en la población, lo cual permite que el algoritmo explore regiones prometedoras del espacio de búsqueda. Para problemas de gran magnitud, emplear gran cantidad de mutaciones en las primeras generaciones mejora la capacidad de exploración, sin embargo, "cuando está finalizando la búsqueda se puede llegar a destruir buen material genético. Por tanto, la probabilidad o tasa de mutación no debe permanecer constante durante el proceso de búsqueda del algoritmo. Esta es la razón por la cual se han desarrollado diversos procesos dinámicos, como inclusión lineal, paso línea y Gauss"[84].

[84] ČEPIN, Marko. Genetic Algorithm. En: Assessment of Power System Reliability. Editor: Springer London. 2011. p, 266.

5 PROCEDIMIENTOS DE BÚSQUEDA LOCAL

La solución de un problema hallada mediante un procedimiento puede mejorarse mediante un método o algoritmo de búsqueda local. Los métodos de búsqueda local parten de una solución x y mediante la aplicación de una operación llamada movimiento, transforma esa solución en una solución diferente x' denominada solución vecina. El conjunto de soluciones que se pueden definir a partir de la solución actual x, mediante la implementación del movimiento es llamado vecindario $N(x)$. Si una solución vecina $x,'$ tiene mejor evaluación (fitness) de la función objetivo que la solución actual, x' reemplaza a x. El proceso se repite hasta que la solución no puede ser mejorada, y la solución encontrada se denomina óptimo local en el vecindario

En los procedimientos de búsqueda local, no se garantiza que el óptimo local encontrado sea el óptimo global del problema de optimización. A pesar de esto, se emplean junto con el diseño de metaheurísticas, puesto que son sencillos, brindan buenas soluciones y pueden ser rápidos.

La complejidad que caracteriza al problema de ruteo de vehículo con demandas estocásticas, genera que se desarrollen métodos que combinen las ventajas que ofrecen diferentes procedimientos para encontrar una buena solución. El método que se propone en el trabajo de investigación combina las ventajas que ofrece la metaheurística ACO, los operadores evolutivos y los procedimientos de búsqueda local. “Dos algoritmos clásicos de búsqueda local que se han empleado para el VRPSD son: los procedimientos Or-opt y 3-opt”[85].

[85] PELLEGRINI, Paola y BIRATTARI, Mauro. Out-of-the-Box and Custom Implementation of Metaheuristics. A Case Study: The Vehicle Routing Problem with Stochastic Demand. En: Intelligent Computational Optimization in Engineering. Springer Berlin Heidelberg. vol, 366. 2011. p 277.

5.1 PROCEDIMIENTO k- INTERCAMBIO

La búsqueda local con k-intercambio, conocido, como $k - opt$ consiste en eliminar k (k >1) arcos de la solución y reconectar los k segmentos restantes de tal manera que se conforme una solución diferente a la inicial. En una trayectoria o camino con n nodos, existen $\binom{n}{k}$ maneras posibles de eliminar k arcos, y $2^{k-1}(k-1)!$ formas de construir nuevas trayectorias (incluyendo la ruta original). Una solución es óptima siempre y cuando esta no puede ser mejorada empleando $k - intercambios$. El tiempo computacional que se requiere para examinar un movimiento $k - intercambio$ tiene complejidad por lo menos O(n^k).

Los procedimiento k-intercambio, generalmente más implementados son 2*-intercambio y 3- intercambio*. Hay que tener en cuenta que no todos las posibilidades de nuevas rutas se van a evaluar, debido a que estas heurísticas deben satisfacer las restricciones y condiciones del problema. En las siguientes figuras se observa los intercambios de arcos para $2 - Opt$ (Figura 10) y algunos movimientos para $3 - Opt$ (Figura 11).

Figura 9. Procedimiento $\mathbf{2 - Opt}$

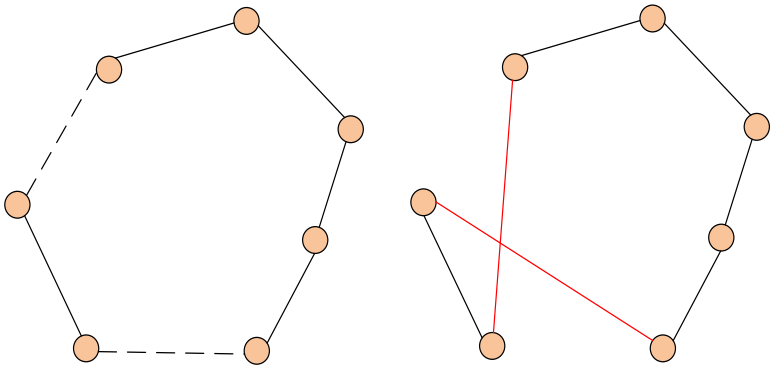
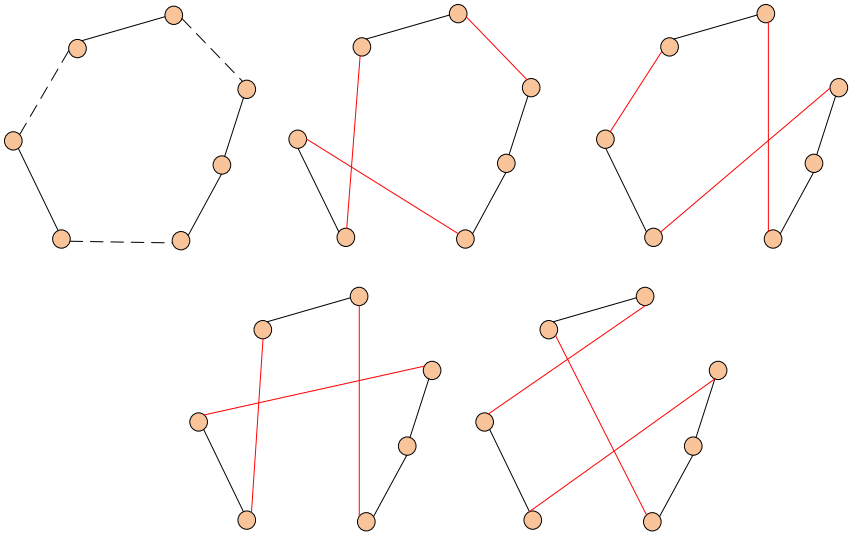


Figura 10. Procedimiento $\mathbf{3-Opt}$

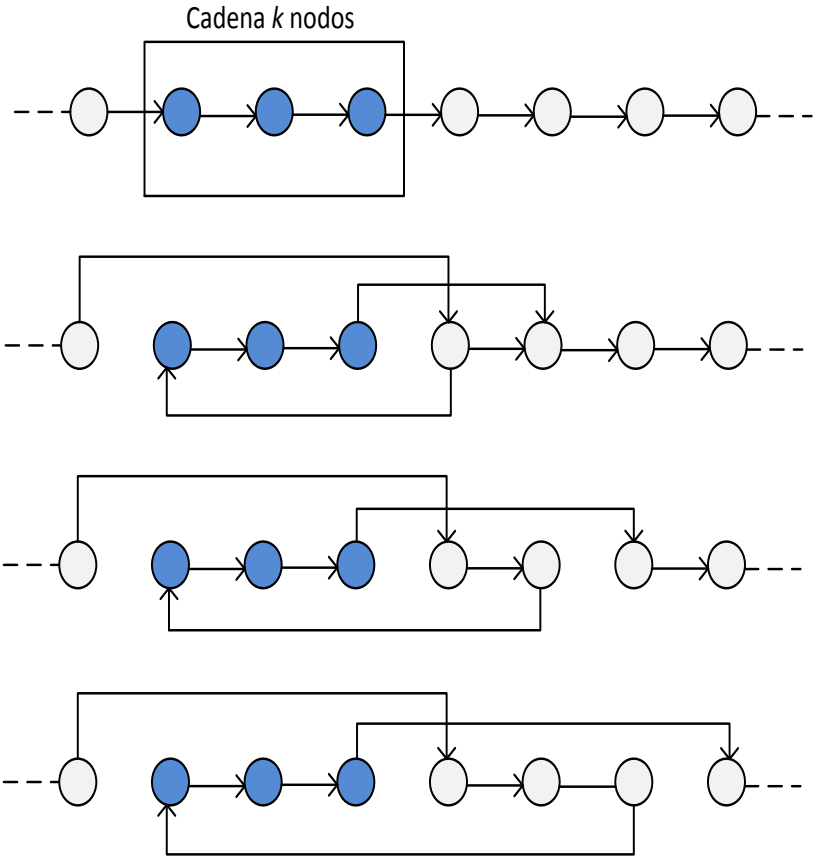


5.2 PROCEDIMIENTO $\mathbf{Or-Opt}$

Este algoritmo fue propuesto por Or en 1976. Consiste en la extracción de una cadena de k nodos consecutivos de la secuencia de una solución inicial, y son insertados en una posición diferente, de tal forma que permanezcan en el mismo orden.

Para determinar si un movimiento Or-Opt genera una mejor solución que la actual, primero se calcula el ahorro de extraer la cadena de nodos, y segundo se determina el costo de reinsertar los nodos en la ruta. En los problemas estocásticos estos cálculos son computacionalmente demandantes. El tiempo computacional para un movimiento puede ser de $O(nKQ)$. En la figura 12 se presenta una trayectoria inicial y las posibles re-ubicaciones de la secuencia de $(k = 3)$ nodos.

Figura 11. Modificaciones de una secuencia inicial aplicando Or-Opt



6 DEFINICIÓN DEL PROBLEMA DE INVESTIGACIÓN

6.1 DEFINICIÓN DEL VRPSD

El VRPSD se define mediante un grafo completo $G = \{V, A, C\}$, donde[86]:

- $V = \{0,1 \dots, n\}$ es un conjunto de nodos. El nodo 0 representa el depósito y los nodos 1,2,…, n corresponde a los clientes que serán visitados.
- $A = \{(i,j): i,j \in V, i \neq j\}$ es el conjunto de arcos que conectan los nodos (i,j).
- $C = \{c_{ij}: i,j \in V, i \neq j\}$ representa la matriz de costos de viaje entre los nodos. Cada arco tiene asociado un costo no negativo c_{ij}. La matriz de costos C es simétrica, es decir $c_{ij} = c_{ji}$, y satisface la desigualdad triangular $c_{ij} \leq c_{ik} + c_{kj}$.

Un vehículo con capacidad Q tiene que entregar bienes a los clientes de acuerdo a su demanda, minimizando el costo esperado total de viaje. Las demandas se definen como variables estocásticas $\xi_i, i = 1, \dots n$ con distribuciones de probabilidad conocidas. La demanda actual de cada cliente sólo se conoce cuando el vehículo llega a la localización del cliente y no debe exceder la capacidad del vehículo. En el presente trabajo de investigación se considera que la demanda sigue una distribución de probabilidad uniforme y discreta $p_{ik} = Prob(\xi_i = k), k = 0,1,2 \dots, K \leq Q$.

[86] BIANCHI, Leonora, et al. Op. cit.,p. 450.

6.2 DEFINICIÓN DE LA POLÍTICA DE SERVICIO

El VRPSD es modelado a partir de dos enfoques que determinan la política de servicio a ejecutar por el vehículo para satisfacer las necesidades de los clientes, *optimización a priori* y *reoptimización* o *servicio dinámico.*

"La estrategia de reoptimización tiene el potencial de obtener una solución de mayor calidad, sin embargo, existen varias dificultades con este enfoque, por ejemplo; si el problema de optimización combinatoria es de tipo NP-hard se tendría que resolver exponencialmente muchas instancias del problema"[87]. Cuando el tamaño del problema es muy grande el rediseño de rutas no justifica el esfuerzo y el costo computacional requerido. De este modo, la estrategia de reoptimización obtiene soluciones factibles para problemas de tamaño pequeño.

En el enfoque a priori, con la implementación de la estrategia de reabastecimiento preventivo se obtienen buenas soluciones con menor esfuerzo computacional, resultando atractiva frente a la política de reoptimización. Las investigaciones realizadad por Dag Haugland, Sin C.Ho y Gilbert Laporte, [88] Wen-Huei Yang, Kamlesh Mathur y Ronald H. Ballou [89] y Leonora Bianchi, et al.,[90] demuestran la preferencia por éste enfoque para el VRPSD.

[87] FAN, Jianhua, WANG, Xiufeng y NING, Hongyun. A Multiple Vehicles Routing Problem Algorithm with Stochastic Demand. En: Journal of Mathematical Modelling and Algorithms. Abril, 2006. vol., 5. p., 1689.
[88] HAUGLAND, Dag y HO,Sin C y LAPORTE, Gilbert. Designing delivery districts for the vehicle routing problem with stochastic demands. En: European Journal of Operational Research. Agosto, 2007. vol, 180. p, 997–1010.
[89] YANG, Wen-Huei, MATHUR, Kamlesh y BALLOU, Ronald. Stochastic vehicle routing problem with restocking. En: Transportation Science. 2000. vol., 34. p., 99-112.
[90] BIANCHI, Leonora, et al. Hybrid Metaheuristics for the Vehicle Routing Problem with Stochastic Demands. En: Journal of Mathematical Modelling and Algorithms. 2006. P., 91-110.

Por estas razones, el objetivo de esta investigación es estudiar el VRPSD con la estrategia de reabastecimiento preventivo, modelado a partir de un método evolutivo de la optimización colonia de hormigas.

6.3 MODELACIÓN MATEMÁTICA DEL VRPSD

La definición formal del problema contempla la estrategia de reabastecimiento preventivo. En donde la secuencia que sigue el vehículo para visitar a los clientes depende de la ruta a priori, la cual se define como una permutación de clientes $s = \big(s(1), s(2), \dots, s(n)\big)$, que inicia y finaliza en el depósito $s(1) = 0$. El vehículo de acuerdo a la ruta a priori y a la demanda actual de cada cliente, decide si continúa con el siguiente cliente o regresa al depósito para realizar la descarga o recarga preventiva, con el fin de evitar un fallo de ruta, que se genera cuando la capacidad del vehículo es excedida por la demanda total de los clientes.

En el VRPSD, el objetivo es encontrar una ruta a priori que minimice el costo esperado de viaje total, conformado por el costo de viaje planeado c_{ij} entre los nodos de la ruta a priori y el costo de recurso que depende de la política de servicio. El costo asociado a la política de servicio de recarga preventiva se calcula de la siguiente manera. Dada una ruta a priori, el vehículo "después de servir completamente a un cliente j, se considera que tiene una carga residual q, y sea $f_j(q)$ que denota el costo esperado total desde el nodo j en adelante. Si L_j representa el conjunto de todas las posibles cargas que un vehículo puede tener después de servir completamente al cliente j, entonces $f_j(q)$ para $q \in L_j$ satisface:

$$f_j(q) = \textit{minimizar } \{f_j^p(q), f_j^r(q)\}"[91].$$

[91] BIANCHI, Leonora, et al. Op. Cit. p., 94.

Donde:

$$f_j^p(q) = c_{j,j+1} + \sum_{k:k\leq q} f_{j+1}(q-k)p_{j+1,k} + \sum_{k:k>q} [2c_{j+1,0} + f_{j+1}(q+Q-k)]p_{j+1,k}$$

$$f_j^r(q) = c_{j,0} + c_{0,j+1} + \sum_{k=1}^{K} f_{j+1}(Q-k)p_{j+1,k}$$

Con la condición límite:

$$f_{n(q)=d_{n,0},\ q\epsilon L_n}$$

Siendo:

- $f_j^p(q)$: El costo esperado si el vehículo decide ir al próximo cliente de la ruta a priori. Además, se contempla el costo de fallo de ruta, relacionado con el costo de viaje de regresar al depósito para atender la demanda del cliente.
- $f_j^r(q)$: El costo esperado si el vehículo retorna al depósito para el reabastecimiento preventivo.
- c_{ij}: Costo de ir del nodo i al nodo j.
- $p_{j+1,k}$: Determina la probabilidad de que la demanda en el nodo $j+1$ sea igual k.

El criterio de decisión del vehículo está determinado por un umbral (h_j) para la carga residual después del servir al cliente j. De modo que, si $q \geq h_{j,}$ se procede al próximo cliente de la ruta, o si $q < h_j$ el vehículo vuelve a la estación para el reabastecimiento preventivo.

Las ecuaciones que se presentaron anteriormente se utilizan para determinar el valor de la función objetivo de la ruta planeada. En la siguiente sección se describe de manera general el cálculo de la función objetivo tomando en cuenta la modelación matemática del VRPSD.

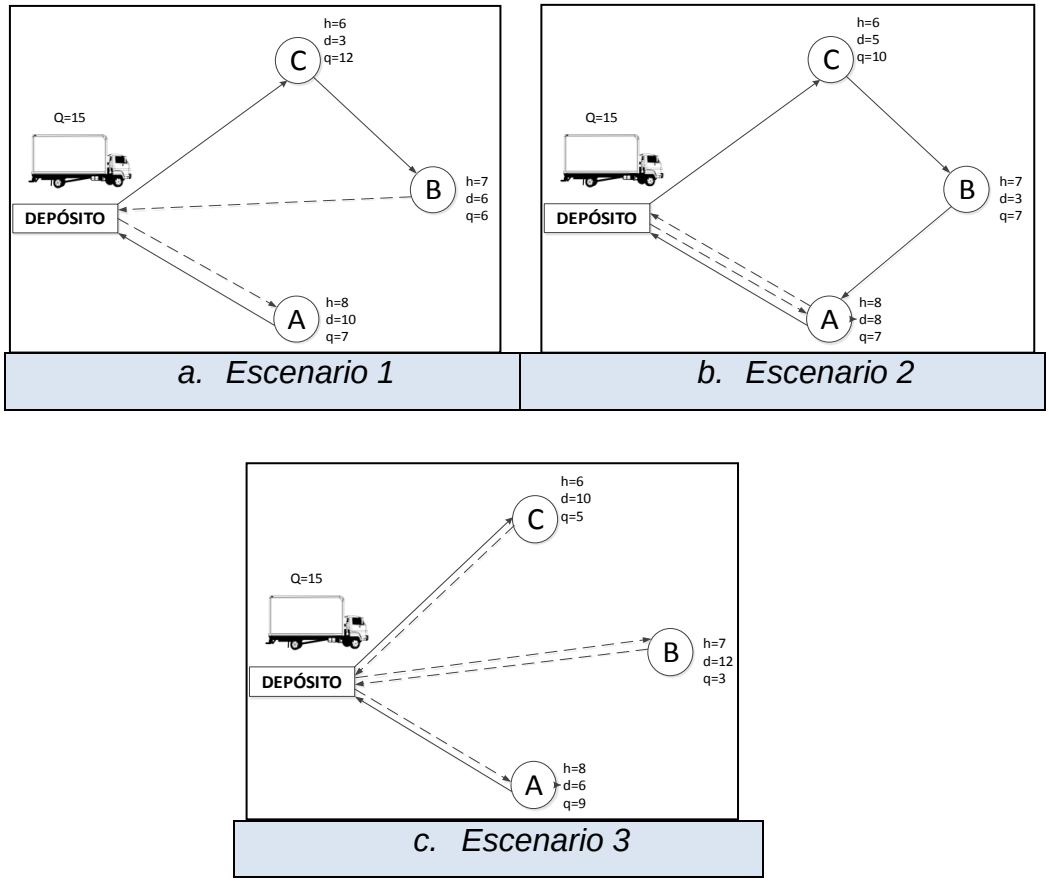
6.4 CÁLCULO DE LA FUNCIÓN OBJETIVO

Debido a la naturaleza estocástica de las demandas en el VRPSD, la ruta real que debe realizar el vehículo sólo se conoce con certeza cuando el vehículo la ejecuta. Para determinar el costo esperado de viaje total con la ejecución de una ruta, Tan, Cheong y Goh[92], proponen un método donde se emplean diferentes escenarios de demandas que son simulados a partir de la distribución de probabilidad. De esta manera el costo esperado de viaje total, equivale a la media de los costos por cada conjunto de demandas.

En la figura 13, se muestra un ejemplo del cálculo de la función objetivo. La secuencia de la ruta a priori es: Depósito → C → B → A → Depósito. Para cada cliente se genera un conjunto de demandas aleatorias. El conjunto de demandas para el cliente A es {10,8,6}, para el cliente B {6,3,12} y para el cliente C {3,5,10}. En este caso existen tres posibles escenarios de demandas para la ruta {C,B,A}; estos son: escenario1= {3,6,10}, escenario2={5,3,8}, escenario3={10,12,6}. El umbral de capacidad de cada cliente está determinado por el promedio del conjunto de demandas, de este modo el umbral de capacidad para los clientes A, B y C son {8,7,6} respectivamente. Se supone que la capacidad del vehículo (Q) es 15, cada flecha indica una unidad de distancia.

[92] TAN, K.C., CHEONG, C.Y. y GOH, C.K. Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. En: European Journal of Operational Research. 2007. vol, 177. no,2. p 813–839.

Figura 12. Ejemplo del cálculo de la función objetivo



a. Escenario 1. Se realiza un viaje de regreso al depósito para el reabastecimiento, debido a que la carga residual *q* después de servir al cliente B es mayor que su umbral de capacidad. El vehículo retoma la ruta en el siguiente cliente a visitar. El costo total de viaje es de: 5 unidades.

b. Escenario 2. Se presenta un fallo de ruta cuando se atiende al cliente A, es decir, la carga residual no es suficiente para satisfacer la demanda del cliente, y se debe que realizar un viaje de regreso al depósito. El costo total de viaje es de: 6 unidades.

c. *Escenario 3.* Después de servir a los clientes C y B se realiza un viaje al depósito para el reabastecimiento. El costo total de viaje es de: 6 unidades.

El costo total de viaje esperado de la ruta es el promedio de los costos de la ruta evaluado con los diferentes escenarios, es decir, 5,67 unidades.

6.5 DEFINICIÓN DEL NÚMERO DE ESCENARIOS DE DEMANDA

El costo de una solución evaluada con diferentes escenarios de demanda presenta alta variabilidad. Por ello, el cálculo del costo promedio puede verse afectado por valores extremos cuando el número de escenarios es pequeño. Sin embargo, establecer y evaluar un gran número de escenarios requiere una cantidad significativa de tiempo computacional. Considerando esto, el número total de escenarios de demanda se definió a partir de un análisis gráfico del comportamiento del costo promedio de una ruta para 100 y 200 clientes.

Figura 13. Costo promedio vs número de escenarios para una ruta de 100 clientes

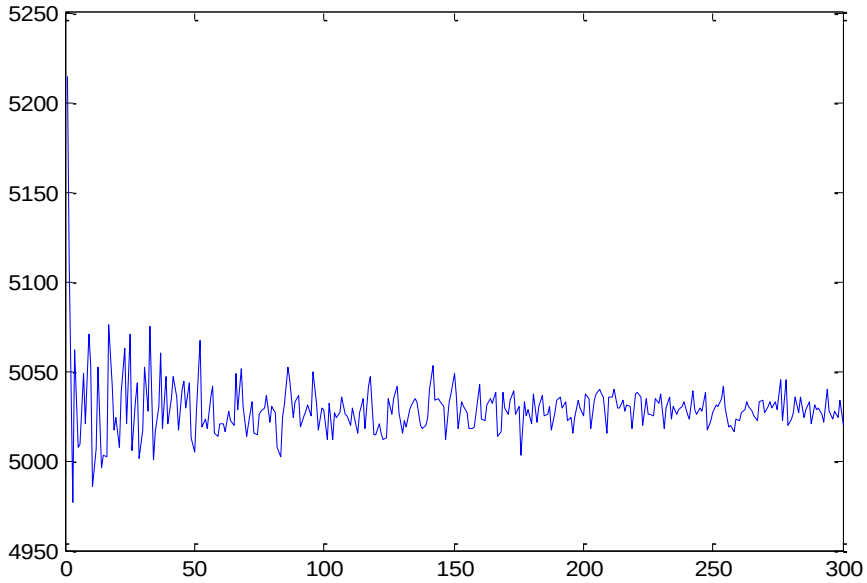
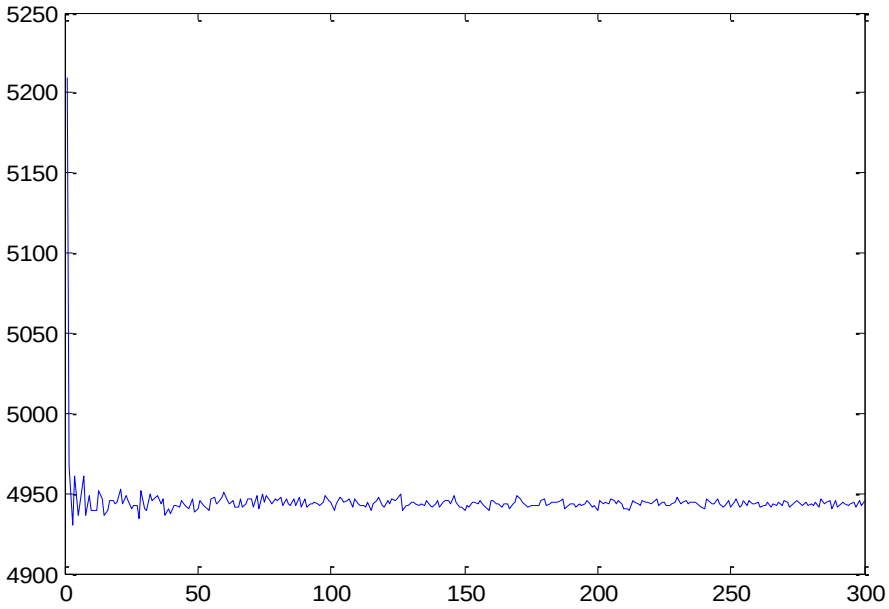


Figura 14. Costo promedio vs número de escenarios para una ruta de 100 clientes



Como se observa en las figuras 14 y 15, a mayor número de escenarios el comportamiento del costo promedio tiende a estabilizarse, es decir, disminuye el efecto de la variabilidad de los datos, por tanto se determinó que el número total de escenarios de demandas adecuado para evaluar las rutas es 50.

6.6 INSTANCIAS

Una instancia comprende la asignación de valores específicos al conjunto de parámetros que caracterizan el problema. A través de las instancias se puede evaluar la calidad de los resultados obtenidos por las técnicas de solución propuestas.

Los parámetros relacionados con el planteamiento de una instancia para el VRP con demandas estocásticas tratado en el presente trabajo son:

- Número de clientes
- Ubicación de los clientes
- Demanda de los clientes
- Capacidad del vehículo
- Política de descargue

En el caso de problemas clásicos como el VRP, existen diversas fuentes de instancias o conjuntos de datos en la literatura, sin embargo, Bianchi *et al.*[93] señalan que comúnmente no se utiliza el Benchmark para el VRPSD. Por tanto, varios autores han desarrollado sus propios bancos de pruebas, por ejemplo "Yang *et al*. y Teodorovic y Lucic, generaron aleatoriamente la ubicación del depósito y de los clientes, la demanda de cada cliente, la media y la varianza."[94]

El banco de pruebas que se empleará para validar la eficacia y eficiencia del método EACO es el generado en la propuesta de investigación realizada por Silvia Galván[95]. Para la generación de las instancias se determinaron dos niveles en cada uno de los factores contemplados. El número de clientes toma el valor de 100 en el nivel 1 y de 200 en el nivel 2. La ubicación de los clientes del nivel 1 se determina por una distribución uniforme dentro del intervalo [0,100], y a diferencia del nivel 1, en el nivel 2 los clientes se encuentran ubicados aleatoriamente siguiendo una distribución normal alrededor de dos centros. La desviación estándar y media para cada cluster se determinó de acuerdo a una distribución uniforme en el intervalo [0,100].

La demanda de los clientes en los niveles 1 y 2 sigue una distribución de probabilidad uniforme discreta definida por la demanda promedio (D_i) y la

[93]BIANCHI, Leonora, et al. Op. cit. p. 98.
[94] TAN, K.C.; CHEONG, C.Y. y GOH, C.K. Solving multiobjective vehicle routing problem with stochastic demand via evolutionary computation. En: European Journal of Operational Research, 2007, vol177, p.813–839
[95] GALVÁN, Silvia; ARIAS, Javier y LAMOS, Henry. En: DYNA. Junio, 2013. Ed,179. pp. 60-69.

desviación (S_i). Por tanto, la demanda de cada cliente i toma valores en el intervalo [D_i - S_i , D_i + S_i].

La capacidad del vehículo se calcula de acuerdo a la siguiente formulación[96]:

$$Q = \frac{\sum_{i=1}^{n}(D_i) * r}{n}$$

Donde:

r es el número promedio de clientes atendidos por el vehículo antes de realizar la descarga preventiva. r toma el valor de 4 para el nivel 1 y de 8 para el nivel 2. La formulación de la capacidad se basa en que la diferencia entre la capacidad del vehículo y la demanda total promedio de los clientes influye significativamente en la dificultad del problema.

El último factor a tratar es el umbral o política de descargue, que determina el nivel de carga residual del vehículo para el descargue preventivo. Para el nivel 1 el umbral es igual a la demanda promedio de cada cliente i, y para el nivel 2 es equivalente al primer cuartil dentro de la distribución de demanda del cliente i.

Tabla 1.Factores y niveles

FACTOR	NIVEL 1	NIVEL 2
Número de clientes (n)	100	200
Ubicación de los clientes (P)	Distribución Uniforme	Distribución normal con dos clusters
Demanda promedio (D_i)	[50,70]	[80,100]
Desviación (S_i)	[1,5]	[10,15]
Número promedio de clientes antes de la descarga preventiva (r)	4	8

[96] GENDREAU, M., LAPORTE, G. y SÉGUIN, R., An Exact Algorithm for the Vehicle Routing Problem with Stochastic Demands and Customers. En: Transportation Science, 2, pp. 143-155, 1995. Ibid., p. 66.

FACTOR	NIVEL 1	NIVEL 2
Umbral (h_i)	Demanda promedio	Primer cuartil de la demanda

Fuente: GALVÁN, Silvia; ARIAS, Javier y LAMOS, Henry. En: DYNA. Junio, 2013. Ed,179. p, 65.

El banco de pruebas está conformado por ochos instancias, cada una contiene una combinación de factores y niveles. A continuación en la Tabla 2 se presenta la información de las instancias.

Tabla 2. Banco de pruebas

INSTANCIA	NO. CLIENTES (n)	UBICACIÓN CLIENTES (P)	DEMANDA PROMEDIO (D_i)	DESVIACIÓN (S_i)	r	h_i
1	Nivel 1	Nivel 1	Nivel 1	Nivel 1	Nivel 1	Nivel 1
2	Nivel 1	Nivel 1	Nivel 2	Nivel 2	Nivel 2	Nivel 2
3	Nivel 1	Nivel 2	Nivel 1	Nivel 2	Nivel 2	Nivel 1
4	Nivel 1	Nivel 2	Nivel 2	Nivel 1	Nivel 1	Nivel 2
5	Nivel 2	Nivel 1	Nivel 1	Nivel 2	Nivel 1	Nivel 2
6	Nivel 2	Nivel 1	Nivel 2	Nivel 1	Nivel 2	Nivel 1
7	Nivel 2	Nivel 2	Nivel 1	Nivel 1	Nivel 2	Nivel 2
8	Nivel 2	Nivel 2	Nivel 2	Nivel 2	Nivel 1	Nivel 1

7 MÉTODO EVOLUTIVO DE COLONIA DE HORMIGAS PARA LA SOLUCIÓN DEL VRPSD

El método propuesto en el presente proyecto se fundamenta principalmente en el algoritmo Sistema de Hormigas AS, primer procedimiento desarrollado por Dorigo et al. en 1996, basado en los conceptos generales de la metaheurística optimización colonia de hormigas. Con el propósito de ampliar la exploración de los agentes (hormigas artificiales) hacia nuevas zonas del espacio de búsqueda, se incorpora un operador evolutivo que permite la diversidad de las soluciones. Además, se utiliza una estrategia de movimientos básicos de intercambio entre los arcos para explotar las soluciones generadas por las hormigas. Cada una de las etapas para resolver el VRPSD se detallan a continuación.

Etapa 1 Inicialización de parámetros. Cálculo inicial del nivel feromona sobre todos los arcos (i, j) a partir de la heurística del vecino más cercano. Cálculo de la información heurística.
Etapa 2 Construcción de soluciones empleando el algoritmo AS.
Etapa 3 Mejorar la solución aplicando operadores de mutación y heurística 2-opt.
Etapa 4 Comparar y contrastar las soluciones generadas.
Etapa 5 Actualización de feromona local y global.
Etapa 6 Acumular el número de iteraciones
Etapa 7 Si el número de la iteración actual es mayor que el total de iteraciones, entonces la solución del VRPSD es la mejor ruta encontrada. Sino ir a la etapa 2.

7.1 INICIALIZACIÓN DE PARÁMETROS

En esta etapa se establecen los parámetros tales como: (α) y (β) que determinan la relación entre el nivel de feromona y la información heurística respectivamente,

tasa de evaporación local ($\rho_{1)}$ y global (ρ_2), número de agentes (m) y número de iteraciones ($Iter_EACO$).

El nivel de feromona se inicializa sobre todos los arcos (i,j). $\tau_{ij}(0) = \frac{1}{(n-1)*T_{nn}}$. Donde T_{nn} representa la distancia de viaje total de una solución factible generada por la heurística del vecino más cercano. Además, se inicializa la información heurística. $\eta_{ij}(0) = \frac{1}{Dist_{ij}}$. Donde $Dist_{ij}$ representa la distancia entre cada par de nodos (i,j).

7.2 CONSTRUCCIÓN DE SOLUCIONES

El algoritmo principalmente construye las soluciones de manera probabilística, empleando un conjunto de agentes artificiales, y una estructura dinámica (feromona) que contiene información sobre la calidad de soluciones previas.

En cada iteración m agentes (hormigas) se ubican de manera aleatoria en las localizaciones de los clientes e iterativamente construyen rutas que inician y finalizan en el depósito. Las hormigas cuentan con una memoria llamada lista tabú $N_i^k(t)$, para evitar la transición de cada k hormiga a los clientes que ya ha visitado. En el AS, la hormiga ubicada en el cliente i probabilísticamente selecciona a j de acuerdo a la siguiente probabilidad:

$$P_{ij}^k(t) = \begin{cases} \frac{\tau_{ij}^{\alpha}(t)\eta_{ij}^{\beta}(t)}{\sum_{u \in N_i^k(t)} \tau_{iu}^{\alpha}\eta_{iu}^{\beta}} & si \quad j \in N_i^k(t) \\ o & si\ j \notin N_i^k(t) \end{cases}$$

Donde t indica el número de la iteración actual y $N_i^k(t)$ es el conjunto de nodos válidos para ser visitados. El término $\tau_{ij}(t)$ se relaciona con la información

obtenida del nivel de feromonas sobre el arco de los nodos i y j de la t^{th} iteración, y el término $\eta_{ij}^{\beta}(t) = \frac{1}{Dist_{ij}}$ representa la información heurística propia del problema, asociada con el inverso de la distancia entre pares de nodos. Y los parámetros α y β establecen la relación entre el nivel de feromona y la información heurística, respectivamente.

7.3 MEJORAR LAS SOLUCIONES

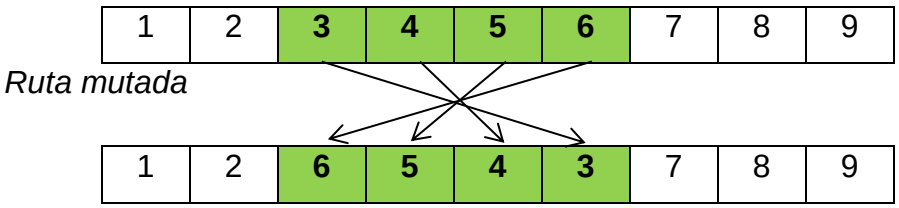
7.3.1 Operador de mutación. La diversidad es necesaria para garantizar que gran parte de las regiones del espacio de búsqueda sean exploradas por el algoritmo, ampliando de esta manera la información a transmitir entre generaciones de agentes, cuya función principal es elaborar las posibles soluciones. Mediante la implementación de un operador evolutivo como la mutación, se busca alcanzar este objetivo de aprendizaje para el algoritmo.

En el presente trabajo de investigación el operador de mutación se aplica cada dos iteraciones a todas las rutas construidas por las $\boldsymbol{m}$ hormigas, y se ejecuta de tres maneras posibles: mutación flip (vuelta), mutación swap (intercambio) o mutación slide (deslizamiento). Las nuevas rutas que se obtengan de la implementación de la mutación son evaluadas y comparadas con las rutas construidas por las hormigas, seleccionando las $\boldsymbol{m}$ soluciones con menor costo promedio.

En el operador de mutación, un cromosoma es una codificación de una solución, es decir, cada ruta se representa mediante una cadena de números enteros, que indican la secuencia en que serán visitados los clientes (nodos). Para aplicar los tres tipos de mutación se seleccionan aleatoriamente dos puntos ($\boldsymbol{I}$ y $\boldsymbol{J}$) del cromosoma. A continuación en las Figuras 16, 17 y 18, se presenta el funcionamiento de los operadores.

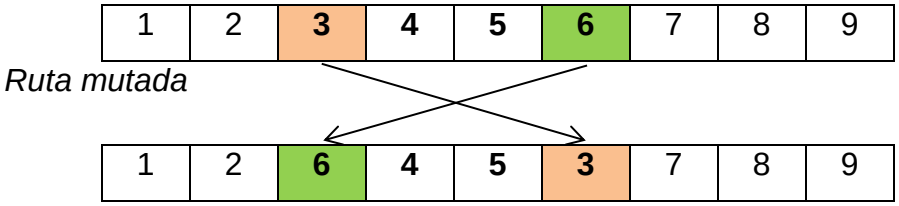
- ***Mutación flip:*** Se invierte el orden de recorrido entre los nodos I y J.

Figura 15. Ejemplo mutación flip



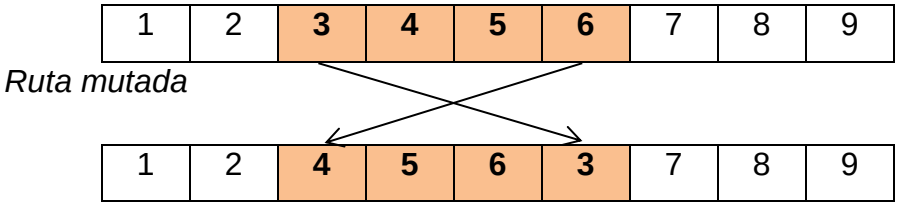
- ***Mutación Swap:*** Se intercambia el nodo en la posición I por el que se encuentra en J, y viceversa.

Figura 16. Ejemplo mutación Swap



- ***Mutación Slide***: Dentro del intervalo definido por I y J, el segundo nodo, toma la posición del primero, este pasa al final del intervalo y los nodos restantes conservan el sentido inicial.

Figura 17. Ejemplo mutación Slide



7.3.2 Heurística de intercambio 2-Opt. El costo esperado de viaje total de una ruta, está conformado por el costo de viaje planeado según la ruta a priori y el costo asociado a la política de servicio. “Debido a que normalmente el costo total de una solución está dominado por el costo de recurso”[97] (política de servicio), una buena solución para el VRPSD se basa principalmente en generar buenas rutas planeadas (rutas a priori). Bajo este supuesto se emplea la heurística de búsqueda local al 50% de las rutas con menor costo, con el propósito de disminuir el valor de la función objetivo.

La heurística de intercambio 2-Opt, a partir de una solución x elimina 2 arcos no adyacentes y reconecta los 2 segmentos restantes de tal manera que se conforme una solución diferente a la inicial x', denominada *solución vecina.* Durante el procedimiento de la búsqueda local, se evalúa la función objetivo de cada una de las soluciones vecinas que se obtienen, y se compara con el valor de la ruta inicial, seleccionando la solución que presente el mejor valor de la función objetivo.

Debido al método utilizado para determinar el valor de la función objetivo, se observó en pruebas preliminares, que evaluar el costo de cada una de las rutas vecinas que se generan en el intercambio de los pares de arcos, requiere de tiempo computacional significativo, por tanto, en el método desarrollado la heurística de mejoramiento se emplea de tal manera que tan pronto como se encuentre una solución vecina mejor que la inicial o no existan más arcos para intercambiar se detiene el proceso de búsqueda.

[97] REIW,Gendreau M, Soriano P.A hybrid Monte Carlo local branching algorithm for the single vehicle routing problem with stochastic demands. Technical Report CIRRELT-2007-24, Centre Interuniversitaire de Recherche sur les Réseaux d'Entreprise, la Logistique et le Transport (CIRRELT); 2007. Citado por: MENDOZA, Jorge E. et al. A memetic algorithm for the multi-compartment vehicle routing problem with stochastic demands. En: Computers & Operations Research. vol, 37. no, 2. p, 1890.

7.4 COMPARAR Y CONTRASTAR LAS SOLUCIONES GENERADAS

En cada iteración, después de aplicar el intercambio 2-opt se comparan las rutas con menor costo promedio para seleccionar la mejor ruta de la iteración actual ($Iteration_Best$). Si el costo de $Iteration_Best$ es menor que el costo de la mejor solución encontrada hasta la iteración actual ($Global_Best$), entonces $Global_Best = Iteration_Best$. Este contraste permite determinar las rutas con las que se realizará la actualización global de feromona.

7.5 ACTUALIZACIÓN DE FEROMONA

La estrategia que se empleó en el método EACO para actualizar feromona es una modificación de la actualización presentada en el algoritmo Sistema Colonia de Hormigas (Dorigo y Gambardella, 1997). En ACS, se utilizan dos tipos de actualización de feromona: local y global.

En la ***regla de actualización local*** de feromona se asume que las hormigas depositan determinada cantidad de feromona de acuerdo a la siguiente ecuación:

Regla de actualización local de feromona:

$$\tau_{ij}(t+1) = (1-\rho_1)\tau_{ij}(t) + \rho_1\tau_0$$

Donde:

ρ_1**:** representa la constante de evaporación local y toma valores entre (0,1).

τ_0: es una constante positiva pequeña equivalente al nivel inicial de feromona sobre los arcos.

La actualización de feromona local se implementa con el fin de transmitir la información de una generación a otra de hormigas, sobre los m mejores caminos construidos después de aplicar el intercambio 2-Opt.

"Para problemas complejos el AS se puede estancar prematuramente, es decir, todas las hormigas siguen el mismo camino, explorando poco el espacio de búsqueda y explotando rápidamente altas concentraciones de feromonas"[98]. Para evitar este comportamiento, se emplea una estrategia mixta para la ***actualización de feromona global***, en donde la mejor solución en cada iteración ($Iteration_Best$) se utiliza en conjunto con la mejor solución encontrada hasta la iteración actual ($Global_Best$). La frecuencia de uso de $Global_Best$ se incrementa durante el progreso del algoritmo, empleándose sólo esta solución en las últimas iteraciones. La actualización global se realiza según la siguiente ecuación:

Regla de actualización global de feromona:

$$\tau_{ij}(t+1) = (1-\rho_2)\tau_{ij}(t) + \rho_2 \Delta\tau_{ij}^{Global} + \rho_2 \Delta\tau_{ij}^{Iteration}$$

Donde:

ρ_2*:* representa la constante de evaporación global y toma valores entre (0,1).

$$\Delta\tau_{ij}^{Global} = \begin{cases} \dfrac{1}{f(\hat{x}(t))}; & si\ (i,j)\ \in\ \hat{x}(t) \\ 0; & De\ lo\ contrario \end{cases}$$

$\hat{x}(t)$: representa la mejor solución encontrada hasta la iteración actual.

$$\Delta\tau_{ij}^{Iteration} = \begin{cases} \dfrac{1}{f(\tilde{x}(t))}; & si\ (i,j)\ \in\ \tilde{x}(t) \\ 0; & De\ lo\ contrario \end{cases}$$

$\tilde{x}(t)$: representa el mejor camino encontrado durante la iteración actual.

[98] ENGELBRENCHT, Andries P. Computational Intelligence. 2 ed. South Africa: John Wiley & Sons, Ltda. 2007. ISBN 978-0-470-03561-0. p, 375.

8 DISEÑO DE EXPERIMENTOS

Para estudiar el efecto de la variación de los parámetros del algoritmo EACO en el valor de la función objetivo, y establecer la configuración que lleve a mejores soluciones del VRPSD, se utilizó un diseño factorial fraccionado 2^{k-1}, que permite variar cada uno de los parámetros en dos niveles, disminuyendo el número de corridas experimentales.

8.1 ANÁLISIS DE PARÁMETROS DEL MÉTODO EACO

El desempeño del algoritmo EACO está determinado por los siguientes parámetros: α y β que establecen la relación entre el nivel de feromona y la información heurística respectivamente, tasa de evaporación local (ρ_1) y global (ρ_2), número de agentes (m), y número de iteraciones $(Iter_EACO)$.

A partir de las investigaciones realizadas por Tripathi, Kuriger y Wan[99], Bianchi,et al[100]., y Moghadam, Sadjadi y Seyedhosseini[101], sobre la aplicación de algoritmos ACO para la solución del VRPSD, se determinó que α y β asumirían los valores de 1 y 3, respectivamente. De esta manera, el diseño experimental toma en cuenta sólo cuatro factores, es decir, se aplicó un diseño factorial fraccionado 2^{4-1} .

Para establecer los niveles de los factores se determinó un rango que incluyera los valores empleados en las investigaciones tomadas como referencia. En la tabla 3 se muestran los factores con sus respectivos niveles.

[99] TRIPATHI, Mukul; KURIGER, Glenn y WAN, Hung-da. Op.cit.
[100] BIANCHI, Leonora, et al. Op.cit.
[101] MOGHADAM, B. FARHANG; Sadjadi, S. J. y SEYEDHOSSEINI, S. M. Op.cit.

Tabla 3.Configuración de los valores de los niveles de cada factor

Factor	Nivel Bajo	Nivel Alto
Tasa de evaporación local (ρ_1)	0.1	0.6
Tasa de evaporación global (ρ_2)	0.1	0.6
Número de agentes (m)	10	20
Número de iteraciones($Iter_EACO$)	100	200

8.2 RESULTADOS DEL DISEÑO EXPERIMENTAL

El algoritmo fue programado en Matlab® 8.0 y ejecutado en un equipo procesador Intel Core i5 con 4GB de RAM. En la tabla 5 se muestra los resultados del diseño experimental. Para el cálculo del valor promedio de la función objetivo, cada tratamiento fue replicado cinco veces.

Tabla 4. Parámetros del diseño experimental

Parámetros	Nivel bajo(-)	Nivel alto(+)
A: m	10	20
B: Iter_EACO	100	200
C: Rho_local	0,1	0,6
D: Rho_global	0,1	0,6

Tabla 5. Resultado del diseño experimental

Corridas				Valor promedio de la función objetivo							
A	B	C	D	Instancia 1	Instancia 2	Instancia 3	Instancia 4	Instancia 5	Instancia 6	Instancia 7	Instancia 8
-	-	-	-	4723,5918	2784,0410	2112,0990	3947,1804	9216,5815	4957,4086	4140,1544	7287,3768
+	-	-	+	4716,2074	2803,1044	2150,5072	3970,8297	9320,4546	5095,1872	4215,2146	7374,0422
-	+	-	+	4756,5058	2857,6881	2165,0530	3966,2998	9359,5199	5125,2365	4258,4185	7402,4803
+	+	-	-	4678,2146	2765,4196	2081,3764	3902,9030	9163,4355	4851,2313	4090,5883	7242,3137
-	-	+	+	4793,5612	2849,6645	2166,8533	4005,7483	9397,1129	5119,2389	4269,7333	7358,9349
+	-	+	-	4693,7305	2759,2045	2100,2756	3944,6603	9201,5669	4895,5791	4102,9839	7273,2216
-	+	+	-	4679,4635	2767,3193	2116,8937	3934,7138	9229,7489	4902,9230	4134,4313	7280,5054
+	+	+	+	4665,5453	2750,5475	2095,2116	3910,2688	9234,3851	4988,5539	4159,5655	7276,8727

8.3 ANÁLISIS DEL DISEÑO EXPERIMENTAL

A cada uno de las instancias que conforman el banco de pruebas se le aplicó el diseño de experimento seleccionado, el cual consta de ocho tratamientos y cinco replicas. En la siguiente tabla se muestra los efectos estimados de cada uno de los factores sobre el valor de la función objetivo.

Tabla 6. Efectos estimados para la función objetivo

Factores	Efecto estimado							
	Instancia 1	Instancia 2	Instancia 3	Instancia 4	Instancia 5	Instancia 6	Instancia 7	Instancia 8
A: m	-49,8561	-45,1092	-33,3821	-31,3201	-70.7803	-68,5639	-58.5963	-40,7118
B: Iter_EACO	-36,8404	-13,76	-17,8001	-38,5583	-37.1566	-49,8673	-21.2707	-22,8508
C: Rho_local	-10,5548	-20,8794	-7,45034	-2,04459	0.705595	-30,6922	-9.41543	-29,1696
D:Rho_global	39,2048	46,255	41,7451	30,9222	125.035	180,269	108.693	82,2282

Si el valor estimado de un efecto tiene un signo positivo, indica que al pasar del nivel bajo al nivel alto el valor de la función objetivo aumenta, este comportamiento se evidencia con el factor D. Por otra parte, el efecto de los factores A, B y C es negativo. Debido a que el objetivo es minimizar el costo esperado de viaje total, Rho_global debe asumir el valor del nivel bajo y los factores m, $Iter_EACO$ y Rho_local el valor del nivel alto.

Los efectos de cada una de las instancias se grafican en un diagrama de Pareto para visualizar cuáles tienen un mayor impacto sobre la variable respuesta.

Figura 18. Diagrama de Pareto Instancia 1

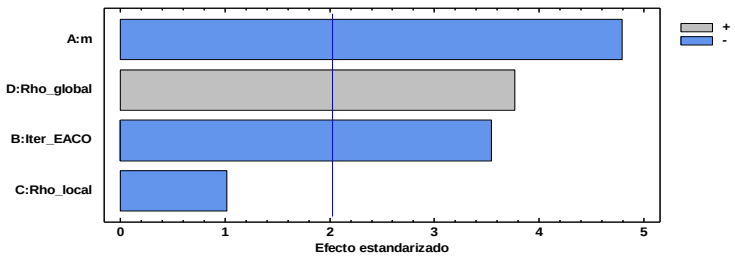


Figura 19. Diagrama de Pareto Instancia 2

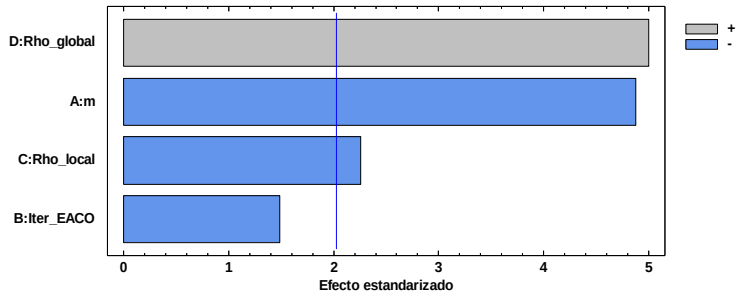


Figura 20. Diagrama de Pareto Instancia 3

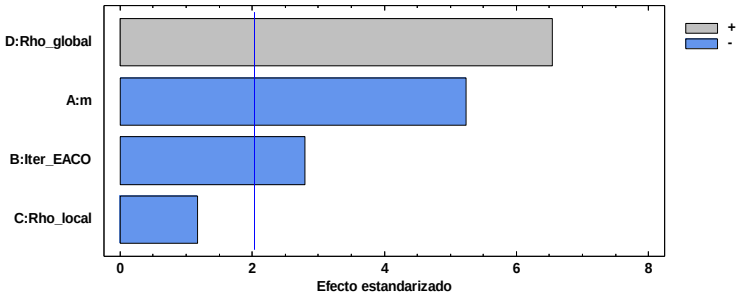


Figura 21. Diagrama de Pareto Instancia 4

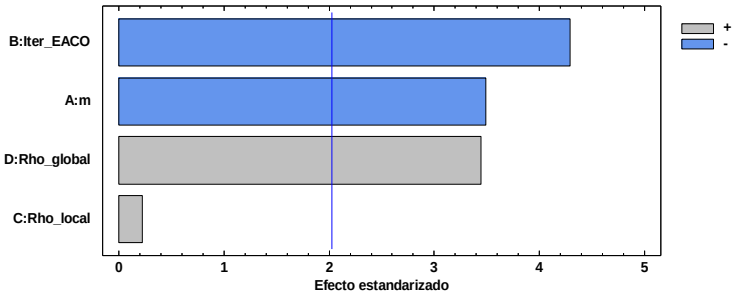


Figura 22. Diagrama de Pareto Instancia 5

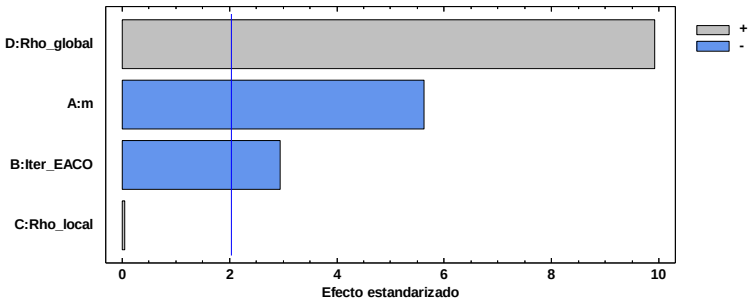


Figura 23. Diagrama de Pareto Instancia 6

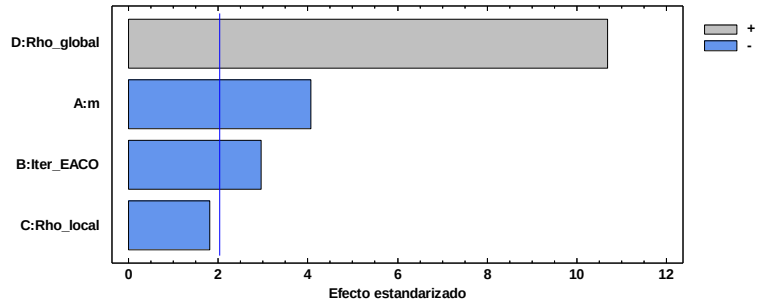


Figura 24. Diagrama de Pareto Instancia 7

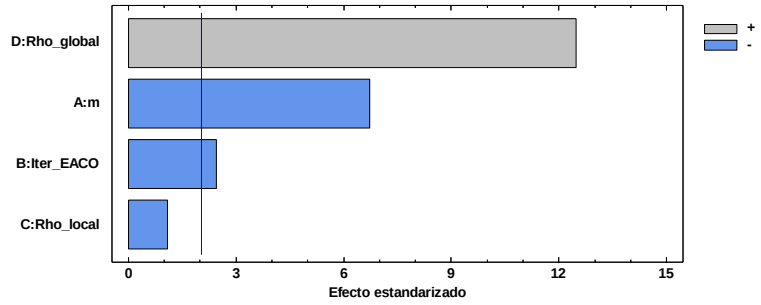
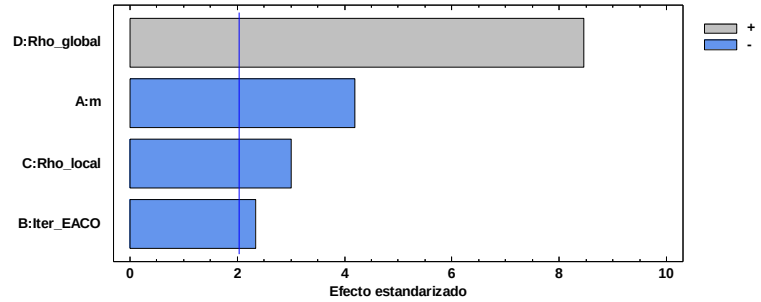


Figura 25. Diagrama de Pareto Instancia 8



En el diagrama de Pareto se puede observar el orden de importancia para cada uno de los efectos. En la mayoría de las instancias los efectos que tienen mayor impacto sobre la variable respuesta son: tasa de evaporación global, número de agentes y número de iteraciones. La tasa de evaporación local no representa en la mayoría de los casos un efecto significativo sobre el valor de la función objetivo, por tanto puede tomar el valor de cualquiera de los dos niveles.

De acuerdo a los efectos estimados de los factores y los resultados del diseño experimental, se establece que $m = 20$, $Iter_EACO = 200$, $\rho_2 = 0.1$ y $\rho_1 = 0.1$. El valor de la tasa de evaporación local se estableció finalmente en el nivel bajo, puesto que con este valor se presentan mejores resultados.

8.4 COMPARACIÓN DEL RENDIMIENTO DEL ALGORITMO

Para evaluar el rendimiento del algoritmo propuesto, los resultados del valor esperado de la función objetivo obtenidos con el método EACO son comparados con los de la metaheurística Sistema de Hormigas tradicional, empleando la misma configuración de parámetros. Por tanto, se implementó el AS para las ocho instancias evaluadas con el EACO. En la Tabla 7 se muestran los resultados hallados con los dos métodos.

Tabla 7. Resultados del valor de la función objetivo AS-EACO

Instancia	AS	EACO	Diferencia Porcentual
1	4729.32736	4678.2146	1.09%
2	2806.22244	2765.41956	1.48%
3	2118.35438	2081.37636	1.78%
4	3974.62098	3902.90298	1.84%
5	9241.34282	9163.43546	0.85%
6	4915.04214	4851.2313	1.32%
7	4118.36054	4090.5883	0.68%
8	7281.5023	7242.3137	0.54%

Los resultados revelan que el método propuesto supera el rendimiento del algoritmo AS tradicional en las ocho instancias comparadas. Este mejor desempeño se alcanza gracias a la capacidad que presenta el algoritmo de equilibrar la exploración y explotación. Mediante la aplicación del operador de

mutación se aumenta el espacio de búsqueda permitiendo diversificar las soluciones y, con la heurística 2-Opt se explotan las mejores soluciones generadas. Además, la estrategia de actualización de feromona global logra que la colonia explote las mejores soluciones, evitando la convergencia prematura del algoritmo.

9 CONCLUSIONES

Los objetivos propuestos en esta investigación se alcanzaron en su totalidad. Se desarrolló el método EACO en el software Matlab® para resolver el problema de ruteo de vehículos con demandas estocásticas, arrojando resultados factibles.

Para el desarrollo del método EACO fue necesario seleccionar el algoritmo basado en la metaheurística optimización colonia de hormigas que presentará un buen desempeño en las instancias probadas.

A partir de la revisión bibliográfica, inicialmente se decidió implementar el algoritmo Sistema Colonia de Hormigas modificando la estrategia de actualización global de feromona. A través de pruebas preliminares se observó la convergencia prematura del algoritmo en óptimos locales, por ello, se aplicó la metaheurística Sistema de Hormigas con la estrategia de actualización global para disminuir la explotación. Además, en la construcción de las rutas, los agentes se ubicaron de manera aleatoria en la localización de los clientes, para mejorar la capacidad de exploración.

Con el objetivo de mejorar el rendimiento del algoritmo se estudiaron las bases teóricas de los operadores genéticos y heurísticas de búsqueda local aplicados al VRPSD, seleccionando el operador de mutación para aumentar la capacidad de exploración y la heurística de búsqueda local 2-Opt para explotar las mejores soluciones.

Debido a la diversidad de soluciones que genera el operador evolutivo, las rutas mutadas se compararon con las rutas construidas por las hormigas, escogiendo las que presentaron menor costo. Por otra parte, la heurística de mejoramiento se empleó de tal manera que tan pronto como se encontrará una mejor solución que la inicial o no existieran más arcos para intercambiar, el proceso de búsqueda

finalizará debido al tiempo computacional requerido para evaluar un intercambio de arco.

Tomando en cuenta los resultados arrojados por el diseño de experimentos se puede afirmar que en la mayoría de las instancias los efectos que tienen mayor impacto sobre la variable respuesta son: tasa de evaporación global, número de agentes y número de iteraciones. La tasa de evaporación local resultó ser un efecto no significativo sobre la variable respuesta.

De acuerdo a los efectos estimados, el método propuesto arroja mejores resultados al aumentar el valor de los parámetros: número de agentes y número de iteraciones. Y considerando que la actualización de feromona se realiza con las mejores rutas construidas en el procedimiento, resulta conveniente establecer pequeños valores para las tasas de evaporación local y global.

Los resultados revelan que el método propuesto supera el rendimiento del algoritmo AS tradicional en las ocho instancias comparadas. Este rendimiento se logra gracias a la capacidad que presenta el algoritmo de equilibrar la exploración y explotación. Sin embargo se observa que el algoritmo presentó mejor desempeño en las instancias con 100 clientes, con una diferencia porcentual superior al 1%.

10 RECOMENDACIONES

Para futuras investigaciones se recomienda:

- Implementar el método EACO para un problema de ruteo de vehículos con demandas estocástica multiobjetivo, en donde se considere minimizar la distancia total de viaje, número de vehículos y penalización asociada con el servicio parcial de los clientes, para modelar diferentes situaciones empresariales o comerciales que surgen en la logística de distribución.

- Emplear un operador evolutivo que permita intercambiar segmentos entre rutas, para la generación de soluciones con mejor calidad de la función objetivo.

- Aplicar otra heurística de búsqueda local que requiera menor tiempo computacional. Aunque “el algoritmo 2-opt genera mejores resultados, requiere significativamente mayor tiempo computacional (aproximadamente más del 150%) en comparación con el algoritmo Or-opt”[102].

- Crear bancos de pruebas para el VRPSD, con el fin de aumentar el número de instancias benchmark que permitan validar los métodos desarrollados para solucionar este problema.

- Desarrollar un nuevo software comercial en base a la herramienta computacional creada, con estructuras informáticas y lenguajes de programación más robustos, que permita aplicar el método propuesto en la solución de problemas del mundo real.

[102] YANG, Wen-Huei, MATHUR, Kamlesh y BALLOU, Ronald. Op. cit. p., 104.

- Realizar futuras investigaciones para la línea de investigación de ruteo de vehículos del grupo OPALO, a partir de lo estudiado en el presente proyecto.

BIBLIOGRAFÍA

ALEJANDRO ARITO, Franco Luis. Algoritmos de Optimización basados en Colonias de Hormigas aplicados al Problema de Asignación Cuadrática y otros problemas relacionados. Trabajo Final para alcanzar el grado de Licenciado en Ciencias de la Computación. Argentina: Universidad Nacional de San Luis. Abril de 2010. 83 p.

ALONSO, Sergio, et al. La metaheurística de optimización basada en colonia de hormigas: modelos y nuevos enfoques. Trabajo realizado en el marco del proyecto "Mejora de Metaheurísticas mediante hibridación y sus aplicaciones" de la Universidad de Granada. 50 p.

BALDACCI, Roberto, et al. An exact solution framework for a broad class of vehicle routing problems.En: Computational Management Science. Julio, 2010.vol, 7. no, 3. p 229-268.

BELL, John E. y MC.MULLEN Patrick R. Ant colony optimization techniques for the vehicle routing problem. En:Advanced Engineering Informatics. 2004. vol. 18. p. 41-48.

BIANCHI, Leonora, et al. Hybrid Metaheuristics for the Vehicle Routing Problem with Stochastic Demands. En: Journal of Mathematical Modelling and Algorithms. 2006. p. 91-110.

BIANCHI, Leonora, et al. Metaheuristics for the Vehicle Routing Problem with Stochastic Demands. En: Parallel Problem Solving from Nature - PPSN VIII, Lecture Notes in Computer Science, 2004, vol. 3242, p. 450-460.

BLUM, Christian y LI, Xiaodong. Swarm Intelligence in Optimization. En: Swarm Intelligence. 2008. Springer Berlin Heidelberg. p 43.

BLUM, Christian, y ROLI, Andrea. Metaheuristics in combinatorial optimization: Overview and conceptual comparison. En: ACM Computing Surveys (CSUR), 2008., vol. 35. no. 3., p. 268-308.

BLUM, Christian. Ant colony optimization: Introduction and recent trends. En: Physics of Life Reviews. 2005. vol, 2. p, 353–373.

BÖHNLEIN, Dominik; GAHM, Christian y TUMA, Axel. A hybrid meta-heuristic for the VRPTW with cluster-depent tour stars in the newspaper industry.42nd Hawaii International Conferences on System Sciences.2009.

CABALLERO, Rafael, et al. 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 177. Elsevier. Noviembre, 2005.p,1751–1763.

CASTELLANOS RAMIREZ, Andrés. Manual de la gestión logística del transporte y distribución de mercancías. Barranquilla: Ediciones Uninorte, 2009. p. 55. ISBN 987-958-741-001-3.

ČEPIN, Marko. Genetic Algorithm. En: Assessment of Power System Reliability. Editor: Springer London. 2011. p, 266.

CHICANO GARCÍA, José Francisco. Metaheurísticas e Ingeniería del Software. Tesis doctoral. Universidad de Málaga, Lenguajes y ciencias de la computación. 2007. p, 46.

CORDEAU, Jean François, et al. Vehicle Routing. En: Handbooks in Operations Research & Management Science: Transportation. 2007.

CHANG, Mei-Shiang. A Vehicle Routing Problem With Time Windows And Stochastic Demands. En: Journal of the Chinese Institute of Engineers. 2005, Vol. 28, No 5, pp. 783-794.

CHUN YEW, Cheong y KAY CHEN, Tan. Hybridizing Problem-Specific Operators with Meta-heuristics for Solving the Multi-objetive Vehicle Routing Problem with Stochastic Demand. En: BAPTISTA PEREIRA, Francisco y TAVARES, Jorge. Bio-inspired Algorithms for the Vehicle Routing Problem. Springer, Nov 17, 2008 . p.101 – 119.

DROR, Mosher; LAPORTE, Gilbet y TRUDEAU, Pierre. Vehicle Routing with Stochastic Demands: Properties and Solution Frameworks. En: Transportation Science. Agosto, 1989. Vol. 23, no.3,. p, 166-176.

DU, Lingling y HE, Ruhan.Combining Nearest Neighbor Search with Tabu Search forLarge-Scale Vehicle Routing Problem. En: PhysicsProcedia. 2012. vol. 25. p. 1536-1546.

ENGELBRENCHT, Andries P. Computational Intelligence. 2 ed. South Africa: John Wiley & Sons, Ltda. 2007. ISBN 978-0-470-03561-0. 630, p.

GALVÁN NÚÑEZ, Silvia Adriana. Optimización por simulación basado en un sistema evolutivo de optimización de enjambre de partículas para el problema de ruteo de vehículos con demandas estocásticas. Tesis de maestría de ingeniería industrial. Bucaramanga. Universidad Industrial de Santander. 2012.

GARCÍA SABATER, José Pedro y MAHEUT, Julien. Modelos y Métodos de Investigación de Operaciones. Procedimientos para Pensar. p,103.

GARCÍA MERINO, Felipe y RUIZ DALPIVAR, Álvaro. Análisis y propuesta de un modelo logístico para la empresa Productos Agropecuarios Santa Cecilia S. A. de C. V. Licenciatura en Administración de Empresas. Cholula, Puebla: Universidad de las Américas Puebla, 2004. 816 p.

GENDREAU, Michel; LAPORTE, Gilbert y SÉGUIN, René. Invited Review: Stochastic vehicle routing. En: European Journal of Operational Research, 1996, vol. 88., p. 3–12.

GESTAL POSE, Marcos. Introducción a los algoritmos genéticos [En línea]. Departamento de Tecnologías de la información y las comunicaciones. Universidad de Coruña. [Citado: 8 Junio 2013]. Disponible en: http://sabia.tic.udc.es/mgestal/cv/aaggtutorial/tutorialalgoritmosgeneticos.pdf

GOODSON, Justin C.; OHLMANN, Jeffrey W. y THOMAS, Barrett W. Cyclic-order neighborhoods with application to the vehicle routing problemwith stochastic demand. En:European Journal of Operational Research, 2012, vol. 217, p. 312–323.

GUTIÉRREZ, Valentina; PALACIO, Juan David y VILLEGAS, Juan Guillermo. Reseña del software disponible en Colombia para el diseño de rutas de distribución y servicios. En: Revista Universidad EAFIT. 2007. vol, 43. no, 145. p, 60-80.

Informe al Congreso. [en línea].Presidencia de la Republica de Colombia, Departamento de Planeación. 2012. p. 54. [citado: 14 Mayo 2013]. Disponible en internet:

https://sinergia.dnp.gov.co/sinergia/Documentos/Informe_Senor_Presidente_al_Congreso_de_la_Rep%C3%BAblica_2012.pdf.

ISMAIL, Zuhaimy y IRHAMAH.Genetic Algorithm and Tabu Search for Vehicle Routing Problems with Stochastic Demand.En: International Conference on Mathematical Science. (2010). Instituto Americano de Física. p. 488 – 504.

JUAN, Angel A, et al. Applying Simulation and Reliability to Vehicle Routing Problems with Stochastic Demands. En: XI Conference of the Italian Association for Artificial Intelligence (AIIA09), Reggio-Emilia (Italia). 2009. 14, p.

LAPORTE, Gilbert. The Vehicle Routing Problem: An overview of exact and approximate algorithms. En: European Journal of Operational Research, 1992., vol. 59., p, 345–358.

LARSON, Allan, y MADSEN, Oli BG. The dynamic vehicle routing problem. En: Diss. Technical University of DenmarkDanmarks Tekniske Universitet, Department of TransportInstitut for Transport, Logistics & ITSLogistik & ITS, 2000. 189, p.

MANFRIN, Max. Ant Colony Optimization for the Vehicle Routing Problem.Memoria presentada para la obtención del Diploma de Estudios Avanzados de Ciencias Aplicadas. Universidad Libre de Bruxellles, Facultad de Ciencias Aplicadas. 2004.

MARINAKIS, Yannis y MARINAKI, Magdalene. A hybrid genetic – Particle Swarm Optimization Algorithm for the vehicle routing problem. En: Expert Systems with Applications. Marzo, 2009. vol. 37, no, 2., p, 1446–1455.

MAZZEO, Silvia y LOISEAU, Irene. An Ant Colony Algorithm for the CapacitatedVehicle Routing.En:Electronic Notes in Discrete Mathematics. 2004. vol 18. p. 181-186

MOGHADAM, B. FARHANG; Sadjadi, S. J. y SEYEDHOSSEINI, S. M. Comparing mathematical and heuristic methods for robust vehicle routing problem. En: International Journal of Research and Reviews in Applied Sciences. Febrero, 2010. vol,2. no,2. p, 108 – 116.

NOVOA, Clara y STORER, Robert. An approximate dynamic programming approach for the vehicle routingproblem with stochastic demands. En: European Journal of Operational Research, 2009, vol. 196, p. 509–515.

NOVOA, Clara., et al. A set-partitioning-based model for the stochastic vehicle routing problem. En: Lehigh University, Bethlehem, 2006, 29, p.

OSMAN, Ibrahim Hassan. Metastrategy simulated annealing and tabu search algorithms for the vehicle routing problem. En: Annals of Operations Research. 1993. vol.41. p, p 421-451.

PELLEGRINI, Paola y BIRATTARI, Mauro. Out-of-the-Box and Custom Implementation of Metaheuristics. A Case Study: The Vehicle Routing Problem with Stochastic Demand. En: Intelligent Computational Optimization. 2011. Vol, 366. P, 273-295.

PERBOLI, Guido; PEZZELLA, Ferdinando y TADEI, Roberto. EVE-OPT: a hybrid algorithm for the capacitated vehicle routing problem. En: Mathematical Methods of Operations Research. Octubre, 2008.vol, 68. no. 2. p, 361-382.

PÉREZ JIMÉNEZ, Mario de J y SANCHO CAPARRINI, Fernando. Máquinas moleculares basadas en ADN. [En línea]. España: Universidad de Sevilla. 2003. [Citado 22 Mayo 2013]. Disponible en internet: http://goo.gl/TNprv

PRINS, Christian.A simple and effective evolutionary algorithm for the vehicle routing problem.En:Computers& Operations Research. Octubre, 2004.vol.31. p.1985-2002.

R.H. Ballou. Business Logistics/Supply Chain Management.5° ed. Pretince Hall.Upper Saddle River.New Jersey. 2004. p.14. Citapo por: CORREA, Andrés Felipe, et al. Características del diseño de rutas de distribución de alimentos en el Valle de Aburrá. En: Revista Facultad de Ingeniería de la Universidad de Antioquia. Septiembre, 2008. No 45. p. 173.

NASSER A, El-Sherbeny. Vehicle routing with time windows: An overview of exact, heuristic and metaheuristic methods. En: Journal of King Saud University. 2010. vol, 22. p, 126 -127.

REIMANN, Marc. Analyzing a vehicle routing problem with stochastic demands using ant colony optimization. En: Advanced OR and AI Methods in Transportation, Publishing House of Poznan University of Technology, 2005. p. 764-769.

Revista de logística [en línea]. LEGIS. Mayo-Julio 2012. [citado: 14 Mayo 2013]. Disponible en internet: http://www.revistadelogistica.com/transporte-de-carga-una-cuantia-aun-a-medio-pagar.asp

SHEN, Zhihong; ORDÓÑEZ, Fernando y DESSOUKY, Maged M. The Stochastic Vehicle Routing Problem for Minimum Unmet Demand. En: Optimization and Logistics Challenges in the Enterprise Springer Optimization and Its Applications. 2009. vol, 30.p, 349-371.

SUGIYAMA, Hiroshi. Principle of dynamic programming as a natural law discovered by Richard Bellman. En: Journal of Mathematical Analysis and Applications. Octubre - Noviembre, 1986. vol, 119. no, 1-2.

TAN, K.C.; CHEONG, C.Y. y GOH, C.K. Solving multiobjective vehicle routing problemwith stochastic demand via evolutionary computation.En:European Journal of Operational Research, 2007, vol177, p.813–839.

TEODOROVIC, Dusan y LUCIC, Panta. The fuzzy ant system for the vehicle routing problem when demand at nodes is uncertain. En: International Journal on Artificial Intelligence Tools. 2007. vol, 16. no, 5,. p. 751 – 770.

TOTH, Paolo y VIGO, Daniele. The Vehicle Routing Problem. En: Society for Industrial and Applied Mathematics, 1987. vol.9.

TRIPATHI, Mukul; KURIGER, Glenn y WAN, Hung-da. An ant based simulation optimization for vehicle routing problem with stochastic demands. En: Actas de la Conferencia de Simulación de Invierno. San Antonio: Universidad de Texas. 2009. P, 2476-2487.

VELEZ, Mario Cesar y MONTOYA, José Alejandro. Metaheurísticos: Una alternativa para la solución de problemas combinatorios en administración de operaciones. En:EIA. Diciembre, 2007, No 8, p. 99-115.

WANG, Dedong; CHEN, Qijun y Lili. Stochastic Vehicle Routing Problems and their Solution Algorithm. En: International Conference on Automation and Logistics. (Septiembre, 2008: Qingdao, China). Memorias. Qingdao: Instituto de Ingenieros Eléctricos y Electrónicos – IEEE, 2008. p. 958 – 962.

WEN, Peng y CHANG-YU, Zhou.Solving Vehicle Routing Problem Using Ant Colony and Genetic Algorithm. En: International Conference on Intelligent Computing (4: 15-18, Septiembre: Shanghai, China). 2008. p. 23-30.

XIAO, Zhang y WANG, Jiang-qing.Hybrid Ant Algorithm and Applications for Vehicle RoutingProblem.En:Physics Procedia. 2012. vol. 25. p. 1892 – 1899.

YANG, Wen-Huei; MATHUR,Kamlesh y BALLOU, Ronald H.Stochastic vehicle routing problem with restocking.En: Transportation ScienceTrans. Sci. 2000. vol, 34. No, 1. p. 99–112.

YOUNG, Peng y HA-YING, Zhu. Research on Vehicle Routing Problem with Stochastic Demandand PSO-DP Algorithm with Inver-over Operator. En: Systems Engineering — Theory & Practice, 2008, vol. 10 p. 76-81.

YOUSEFI KHOSHBAKHT, Majid y SEDIGHPOUR, Mohammad. An Optimization Algorithm for the Capacitated Vehicle Routing Problem Based on Ant Colony System. En: Australian Journal of Basic and Applied Sciences. 2011. vol, 5. no, 12. p. 2729-2737.

YOUSEFI KHOSHBAKHt, Majid y KHORRAM, Esmaile. Solving the vehicle routing problem by a hybrid meta-heuristic algorithm. En: Journal of Industrial Engineering International. Diciembre, 2012.vol, 8. no, 1. 11,p.

ZHANG, Xiaoxiay LANG, Lixin. A new hybrid ant colony optimization algorithm for the vehicle routing problem. En: Pattern Recognition Letters. 2009. vol. 30. P, 848–855.

ANEXOS

ANEXO A. DESCRIPCIÓN DE VARIABLES PARA CADA FUNCIÓN DESARROLLADA DEL MÉTODO EACO

A continuación se describe de manera breve las variables de cada una de las funciones desarrolladas en el software Matlab® para el método EACO.

FUNCIÓN: FACTORES_Y_NIVELES

Descripción: Definición de factores y niveles (alto-bajo) para 100 nodos clientes y 200 nodos clientes

Los factores y sus respectivos niveles son:

- *Dprom_N_bajo_100:* Demandas promedio, nivel bajo para 100 clientes
- *Dprom_N_alto_100:* Demandas promedio, nivel alto para 100 clientes
- *Dprom_N_bajo_200:* Demandas promedio, nivel bajo para 200 clientes
- *Dprom_N_alto_200:* Demandas promedio, nivel alto para 200 clientes
- *Desviacion_N_bajo_100:* Desviación de la demanda, nivel bajo para 100 clientes
- *Desviacion_N_alto_100:* Desviación de la demanda, nivel alto para 100 clientes
- *Desviacion_N_bajo_200:* Desviación de la demanda, nivel bajo para 200 clientes
- *Desviacion_N_alto_200:* Desviación de la demanda, nivel alto para 200 clientes
- *Coordenadas_N_bajo_100:* Coordenadas (x,y), nivel bajo, para 100 clientes
- *Coordenadas_N_alto_100:* Coordenadas (x,y), nivel alto, para 100 clientes

- *Coordenadas_N_bajo_200:* Coordenadas (x,y), nivel bajo, para 200 clientes
- *Coordenadas_N_alto_200:* Coordenadas (x,y), nivel alto, para 200 clientes
- *Coordenadas_N_bajo_100:* Coordenadas (x,y), nivel bajo, para 100 clientes
- *r:* Número promedio de clientes atendidos antes de la descarga preventiva

FUNCIÓN: FACTORES

Descripción: Definición de factores dependiendo del tipo de corrida experimental a evaluar.

Variables de entrada:

- *Experimento:* Corrida experimental a evaluar
- *Dprom_N_bajo_100:* Demandas promedio, nivel bajo para 100 clientes
- *Dprom_N_alto_100:* Demandas promedio, nivel alto para 100 clientes
- *Dprom_N_bajo_200:* Demandas promedio, nivel bajo para 200 clientes
- *Dprom_N_alto_200:* Demandas promedio, nivel alto para 200 clientes
- *Desviacion_N_bajo_100:* Desviación de la demanda, nivel bajo para 100 clientes
- *Desviacion_N_alto_100:* Desviación de la demanda, nivel alto para 100 clientes
- *Desviacion_N_bajo_200:* Desviación de la demanda, nivel bajo para 200 clientes
- *Desviacion_N_alto_200:* Desviación de la demanda, nivel alto para 200 clientes
- *Coordenadas_N_bajo_100:* Coordenadas (x,y), nivel bajo, para 100 clientes
- *Coordenadas_N_alto_100:* Coordenadas (x,y), nivel alto, para 100 clientes
- *Coordenadas_N_bajo_200:* Coordenadas (x,y), nivel bajo, para 200 clientes
- *Coordenadas_N_alto_200:* Coordenadas (x,y), nivel alto, para 200 clientes
- *Coordenadas_N_bajo_100:* Coordenadas (x,y), nivel bajo, para 100 clientes
- *r:* Número promedio de clientes atendidos antes de la descarga preventiva

Variables de salida:

- *Dprom:* Demandas promedio
- *S:* Desviación de la demanda
- *X:* Coordenadas (x,y) de los clientes y el depósito
- *r:* Desviación de la demanda Número promedio de clientes atendidos antes de la descarga preventiva
- *h:* Nivel del umbral de capacidad para cada cliente(1- Nivel bajo, 2 - Nivel alto)

FUNCIÓN: DEMANDAS

Descripción: Matriz de demandas definidas aleatoriamente entre [Dprom-S, Dprom+S]. Definición del umbral de capacidad dependiendo el nivel del factor.

Variables de entrada

- *Dprom:* Demandas promedio
- *S:* Desviación de la demanda
- *e:* Número de escenarios de demanda
- *h:* Nivel del umbral de capacidad para cada cliente(1- Nivel bajo, 2 - Nivel alto)

Variables dentro del proceso:

- *n:* Número de clientes más el depósito

Variables de salida:

- *h:* Umbral de capacidad para cada cliente
- *D:* Matriz de demandas

FUNCIÓN: MATRIZ_DISTANCIAS

Descripción: Cálculo de la matriz de distancias euclidianas a partir de una matriz de coordenadas.

Variables de entrada:

- *X:* Matriz de coordenadas (x,y) de los clientes y el depósito

Variables dentro del proceso:

- *n:* Número de clientes más el depósito
- *dim:* Número de columnas de la matriz de coordenadas
- *v:* Vector que guarda la diferencia entre las coordenadas del cliente j y los demás clientes, incluido el depósito

Variables de salida:

- *Dist:* Matriz de distancias euclidianas

FUNCIÓN: VECINO_MAS_CERCANO

Descripción: Aplicación del método del vecino más cercano para construir una ruta y evaluar su costo como un problema determinística, es decir, teniendo en cuenta solamente las distancias.

Variables de entrada:

- *Dist:* Matriz de distancias euclidianas

Variables dentro del proceso:

- *n:* Número de clientes más el depósito
- *s:* Indica el siguiente cliente más cercano al cliente actual

- *L:* Guarda el valor de la distancia de ir del cliente actual al cliente más cercano

Variables de salida:

- *Ruta_vecino_mas_cercano:* Vector que contiene la ruta construida a partir de la heurística del vecino más cercano. El nodo 1 indica el depósito
- *Costo_ruta_vecino_mas_cercano:* Costo de la ruta obtenida a partir de método del vecino más cercano, evaluada como un problema determinístico

FUNCIÓN: RUTAS_HORMIGAS

Descripción: Cálculo de una matriz de rutas a partir del método Ant System.

Variables de entrada:

- *m:* Número de agentes (hormigas)
- *alpha:* Parámetro de influencia de feromona
- *beta:* Parámetro de influencia de la heurística
- *Feromona:* Matriz de feromona
- *Heuristica:* Matriz de información heurística
- *Dist:* Matriz de distancias euclidianas

Variables dentro del proceso:

- *n:* Número de clientes más el depósito
- *Producto_heuristico_feromona:* Producto entre matriz de feromona y matriz de información heurística. Cada valor se eleva al correspondiente parámetro de influencia.
- *Producto_heuristico_feromona_privada:* Contiene la lista tabú. Con esta matriz se evita que las hormigas seleccionen nuevamente los clientes ya visitados.
- *Curnode:* Variable de control que indica el siguiente cliente a visitar

- *Resultados_del_nodo_actual:* Indica el valor de *Producto_heuristico_feromona_privada* del nodo actual con respecto a los otros nodos, teniendo en cuenta que los nodos visitados no pueden ser elegidos de nuevo.
- *Probabilidades_de_paso:* Indica la probabilidad de ir del nodo actual a los demás nodos.
- *Probabiliades_de_paso_acumuladas:* Vector de probabilidades acumuladas.
- *r:* Número aleatorio distribuido uniformemente entre [0,1], que permite elegir aleatoriamente el siguiente nodo a visitar, teniendo en cuenta las probabilidades de paso.

Variables de salida:

- *Rutas:* Matriz que contiene las m rutas construidas por los agentes en la iteración actual.

FUNCIÓN: EVALUAR_RUTA_CON_DEMANDAS

Descripción: Evaluación del costo de una ruta en función de la distancia total recorrida, de acuerdo a la política de reabastecimiento preventivo del VRPSD.

Variables de entrada:

- *Q:* Capacidad del vehículo
- *h:* Umbral de capacidad para cada cliente
- *Ruta:* Vector que indica el orden secuencial en que serán visitados los clientes, iniciando en el depósito
- *D:* Matriz de demandas
- *Dist:* Matriz de distancias euclidianas

Variables dentro del proceso:

- *n:* Número de clientes más el depósito
- *q:* Carga residual después de visitar al cliente i

Variables de salida:

- *Costo:* Costo de una ruta en función de la distancia total recorrida.

FUNCIÓN: INTERCAMBIO_2OPT

Descripción: El algoritmo examina los posibles cambios de parejas de bordes en la ruta y aplica el intercambio cuando encuentra un menor costo. Se intercambia el nodo i con el nodo j-1.

Variables de entrada:

- *Ruta:* Ruta a mejorar
- *e:* Número de escenarios de demanda
- *D:* Matriz de demandas
- *Costo_promedio:* Costo promedio de la ruta a mejorar
- *Q:* Capacidad del vehículo
- *h:* Umbral de capacidad para cada cliente
- *Dist:* Matriz de distancias euclidianas

Variables dentro del proceso:

- *n:* Número de clientes más el depósito
- *i:* Nodo a intercambiar con el nodo j-1
- *j-1:* Nodo a intercambiar con el nodo i
- *Costo:* Costo de la ruta intercambiando el nodo i con el nodo j-1
- *z:* Diferencia entre el costo de la ruta inicial y el costo de la ruta con el intercambio

- *zmin:* Guarda la diferencia entre el costo de la ruta inicial y el costo de la ruta generada con la aplicación del algoritmo
- *imin:* Guarda el cambio del nodo i
- *jmin:* Guarda el cambio del nodo j

Variables de salida:

- *Ruta_mejorada:* Ruta mejorada encontrada por el algoritmo al intercambiar dos nodos.
- *Costo_ruta_mejorada:* Costo promedio de la ruta mejorada. Evaluada con los diferentes escenarios de demanda.

FUNCIÓN: OPERADOR_MUTACION

Descripción: Construcción de nuevas rutas aplicando operadores de mutación. Si las nuevas rutas son mejores que las rutas iniciales, estas son reemplazadas.

Variables de entrada:

- *Rutas:* Rutas a las cuales se les aplicará el operador de mutación
- *Costo_promedio:* Costo promedio de la rutas iniciales
- *e:* Número de escenarios de demanda
- *D:* Matriz de demandas
- *Q:* Capacidad del vehículo
- *h:* Umbral de capacidad para cada cliente
- *Dist:* Matriz de distancias euclidianas

Variables dentro del proceso:

- *n:* Número de clientes más el depósito.
- *s:* Número de rutas a las cuales se les aplicará el operador de mutación.
- *m:* Equivale al doble del número de rutas iniciales, es decir, m=2*s .

- *Rutas_operador_mutacion*: Matriz que almacena las rutas construidas a partir de los operadores de mutación.
- *Puntos_de_insercion:* Vector de dos números aleatorios enteros entre [2,n] organizados de menor a mayor.
- *I,J:* Puntos de inserción que indican los nodos en donde se aplicará el operador de mutación. I,J deben ser dos valores diferentes, con I mayor que J.
- *k:* Número aleatorio entero entre [1,3], que permite seleccionar el tipo de mutación que se aplicará. Si k=1, se aplica el operador de mutación de vuelta. Si k=2, el operador de mutación de intercambio. Si k=3, el operador de mutación de deslizamiento.
- *Costo:* Vector que almacena los costos de evaluar una ruta construida a partir de los operadores de mutación con los diferentes escenarios de demanda.
- *Costo_promedio:* Vector que almacena el costo promedio de las rutas construidas a partir de los operadores de mutación.
- *Nuevas_rutas:* Matriz que agrupa las rutas construidas por las hormigas y las generadas por el operador de mutación.
- *Costo_nuevas_rutas:* Vector que agrupa los costos de las rutas construidas por las hormigas y las generadas por el operador de mutación.
- *Distancias_menor_a_mayor:* Vector que almacena los costos de las rutas construidas por las hormigas y las halladas con el operador organizados de menor a mayor.
- *Index:* Índices de los costos al ser organizados de mayor a menor. Permite elegir las rutas con menores costos.

Variables de salida:

- *Rutas_operador:* Matriz que guarda las mejores rutas entre las rutas iniciales y las obtenidas con el operador

- *Costo_rutas_operador:* Vector que guarda el costo de las mejores rutas entre las rutas iniciales y las obtenidas con el operador

FUNCIÓN: EACO

Descripción: Método evolutivo de colonia de hormigas para resolver el VRPSD. Aplica el algoritmo del vecino más cercano para la inicialización de feromona. A partir del algoritmo Ant System se construyen rutas a priori, que son evaluadas con la política de reabastecimiento preventivo del VRPSD. Cada 2 iteraciones, se construyen nuevas rutas aplicando operadores de mutación. Si las nuevas rutas son mejores que las iniciales, éstas son reemplazadas. Seguidamente se selecciona el 50% de las rutas con menor costo, que son mejoradas a partir de la heurística de búsqueda local 2-opt. La actualización de feromona local se realiza con las rutas obtenidas de la búsqueda local 2-opt, y con las mejores rutas entre las del operador de mutación y las construidas por las hormigas. La actualización de feromona global se hace con la mejor ruta en la iteración actual. Cada cinco iteraciones se tiene en cuenta la ruta Global_best. Esto se realiza en el 50% del total de iteraciones. La ruta Global_best se emplea para la actualización cada dos iteraciones hasta el 80% del total de iteraciones. En el 20% restante la actualización global sólo se realiza con la ruta Global_best.

Variables de entrada:

- *m:* Número de agentes (hormigas)
- *e:* Número de escenarios de demanda
- *D:* Matriz de demandas
- *Q:* Capacidad del vehículo
- *Dprom:* Demanda promedio de la corrida experimental
- *S:* Desviación de la demanda de la corrida experimental
- *h:* Nivel umbral de capacidad de la corrida experimental
- *Dist:* Matriz de distancias euclidianas

- *Iter_EACO:* Número de iteraciones para la metaheurística EACO
- *rho_local:* Tasa de evaporación local. Toma valores entre 0-1
- *rho_global:* Tasa de evaporación global. Toma valores entre 0-1
- *alpha:* Parámetro de influencia de feromona
- *beta:* Parámetro de influencia de la heurística
- *Costo_ruta_vecino_mas_cercano:* Costo de la ruta obtenida a partir de método del vecino más cercano, evaluada como un problema determinístico

Variables dentro del proceso:

- *n:* Número de clientes más el depósito
- *Feromona:* Matriz de feromona
- *Heuristica:* Matriz de información heurística
- *Iter_actual:* Iteración actual
- *Rutas:* Matriz que contiene las m rutas construidas por los agentes en la iteración actual. Cada dos iteraciones almacena las m rutas con menor costo entre las construidas por las hormigas y las generadas con el operador de mutación
- *h:* Umbral de capacidad para cada cliente
- *D:* Matriz de demandas
- *Costo:* Vector que guarda el costo de las rutas construidas por las hormigas
- *Costo_promedio:* Vector que almacena el costo promedio de las rutas construidas por las hormigas
- *Costo_rutas:* Vector que contiene el costo de las mejores rutas entre las construidas por las hormigas y las generadas con el operador de mutación
- *s:* Equivale al 50% del número de agentes.
- *Costo_rutas_menor_a_mayor:* Vector que almacena los costos de las rutas organizados de menor a mayor.
- *Indice:* Índices de los costos al ser organizados de mayor a menor. Permite elegir las rutas con menores costos.

- *Rutas_mejoradas:* Rutas mejoradas encontradas por el algoritmo 2-opt, con el intercambio de dos nodos. También es usada para almacenar las rutas con las que se hará la actualización de feromona
- *Costo_rutas_mejoradas:* Costo promedio de la rutas mejoradas. Evaluadas con los diferentes escenarios. También es usada para almacenar los costos con los que se hará la actualización de feromona
- *Global_best:* Vector que contiene la mejor ruta hasta la iteración actual.
- *Costo_global_best:* Costo de la mejor ruta hasta la iteración actual.
- *Iteration_best:* Vector que contiene la ruta con el menor costo en la iteración actual.
- *Costo_iteration_best:* Costo de la mejor ruta en la iteración actual.
- *Nueva_feromona:* Vector utilizado para la actualizar el valor de feromona local.
- *Nueva_feromona_global:* Vector utilizado para actualizar el valor de feromona global a partir de la mejor ruta hasta la iteración actual.
- *Nueva_fermona_iteration_best:* Vector utilizado para actualizar el valor de feromona global a partir de la mejor ruta de la iteración actual.

Variables de salida:

- *Rutas:* Matriz que guarda las rutas construidas aplicando operadores de mutación y las mejores rutas obtenidas con el algoritmo 2-opt
- *Costo:* Vector que guarda el costo de las rutas construidas aplicando operadores de mutación y las mejores rutas obtenida
- *Ruta_ECAO:* Ruta con el mínimo costo hallada con la metaheurística EACO
- *Costo_ruta_EACO:* Costo de la ruta hallada con la metaheurística EACO
- *Mejor_ruta_cada_iteracion:* Ruta con el mínimo costo hallada de la iteración actual
- *Costo_minimo_cada_iteracion:* Costo de la mejor ruta hallada en la iteración actual

ANEXO B. CÓDIGO DE PROGRAMACIÓN MÉTODO EACO

En el siguiente anexo se presenta el código de las funciones creadas y utilizadas para el desarrollo del método EACO.

FUNCIÓN: FACTORES

```
function [Dprom, S, X, r,h]= Factores(Experimento)

[Dprom_N_bajo_100,         Dprom_N_alto_100,         Desviacion_N_bajo_100,
Desviacion_N_alto_100,                              Coordenadas_N_bajo_100,
Coordenadas_N_alto_100,Dprom_N_bajo_200,                  Dprom_N_alto_200,
Desviacion_N_bajo_200,    Desviacion_N_alto_200,    Coordenadas_N_bajo_200,
Coordenadas_N_alto_200, r]= Factores_y_niveles();

switch Experimento
    case 1
        Dprom = Dprom_N_bajo_100;
        S = Desviacion_N_bajo_100;
        X = Coordenadas_N_bajo_100;
        r = r(1);
        h = 1;
    case 2
        Dprom = Dprom_N_alto_100;
        S = Desviacion_N_alto_100;
        X = Coordenadas_N_bajo_100;
        r = r(2);
        h = 2;
    case 3
        Dprom = Dprom_N_bajo_100;
        S = Desviacion_N_alto_100;
        X = Coordenadas_N_alto_100;
        r = r(2);
        h = 1;
    case 4
        Dprom = Dprom_N_alto_100;
        S = Desviacion_N_bajo_100;
        X = Coordenadas_N_alto_100;
        r = r(1);
        h = 2;
    case 5
        Dprom = Dprom_N_bajo_200;
        S = Desviacion_N_alto_200;
        X = Coordenadas_N_bajo_200;
        r = r(1);
        h = 2;
    case 6
        Dprom = Dprom_N_alto_200;
        S = Desviacion_N_bajo_200;
```

```
        X = Coordenadas_N_bajo_200;
        r = r(2);
        h = 1;
    case 7
        Dprom = Dprom_N_bajo_200;
        S = Desviacion_N_bajo_200;
        X = Coordenadas_N_alto_200;
        r = r(2);
        h = 2;
    case 8
        Dprom = Dprom_N_alto_200;
        S = Desviacion_N_alto_200;
        X = Coordenadas_N_alto_200;
        r = r(1);
        h = 1;
    otherwise
end

end
```

FUNCIÓN: DEMANDAS

```
function [D,h]= Demandas(Dprom, S, e, h)

% Número de clientes
n=numel(Dprom);

% Matriz de demandas.
D= zeros(n,e);

for i=1:n
    for j=1:e
        D(i,j)=fix((Dprom(i)-S(i))+( 2*S(i)+1)*rand());
    end
end

% Umbral de capacidad de cada cliente.
if h==1
    h = Dprom;
else
    h = zeros(1,n);
    for i=1:n
      y =[Dprom(i)-S(i):Dprom(i)+S(i)];
      h(i) =  median(y(find(y<median(y))));
    end
end

end
```

FUNCIÓN: MATRIZ_DISTANCIAS

```
function Dist= Matriz_distancias(X)
```

```
% Definición del número de clientes más el depósito (n) y número de
columnas (dim)
[n,dim] = size(X);

% Matriz de distancias
Dist = zeros(n);

for j = 1:n
    for k = 1:dim
        v = X(:,k) - X(j,k);
        Dist(:,j) = Dist(:,j) + v.*v;
    end
end
Dist = sqrt(Dist);
end
```

FUNCIÓN: VECINO_MAS_CERCANO

```
function [Ruta_vecino_mas_cercano, Costo_ruta_vecino_mas_cercano] =
Vecino_mas_cercano(Dist)

% Definición del número de clientes más el depósito (n)
n = size(Dist,1);

% Vector de ruta, a partir del método del vecino más cercano
Ruta_vecino_mas_cercano = zeros(1,n);

% La ruta inicializa en 1, es decir, en el depósito
Ruta_vecino_mas_cercano(1) = 1;

s = 1;
Costo_ruta_vecino_mas_cercano = 0;
for k = 2:n
    Dist(s,:) = inf;
    [L,s] = min(Dist(:,s));
    Ruta_vecino_mas_cercano(k) = s;
    Costo_ruta_vecino_mas_cercano = Costo_ruta_vecino_mas_cercano + L;
end
Costo_ruta_vecino_mas_cercano = Costo_ruta_vecino_mas_cercano +
Dist(s,1);

end
```

FUNCIÓN: RUTAS_HORMIGAS

```
function Rutas = Rutas_hormigas(m, alpha, beta, Feromona, Heuristica,
Dist)


% Definición del número de clientes más el depósito
```

```
n=size(Dist,1);

% Restablecimiento de los caminos de las hormigas
Rutas = NaN(m,n);

% Pre-calculo del producto de las feromonas y heuristicas (para la eficiencia)
Producto_heuristico_feromona = Feromona.^alpha .* Heuristica.^beta;

% Construcción de nuevos caminos para las hormigas
for k = 1:m

    % Inicio de la ruta en el nodo 1, es decir, el depósito
    Rutas(k,1) = 1;

     % Mantener una matriz de heuristicas y feromonas privadas (Lista tabú)
    Producto_heuristico_feromona_privada = Producto_heuristico_feromona;
    Producto_heuristico_feromona_privada(:,1) = 0;

    % Cálculo aleatorio del primer cliente a visitar
    curnode = ceil(1+(n-1)*rand);
    Rutas(k,2) = curnode;

    % Mantener una matriz de heuristicas y feromonas privadas(Actualizar la lista tabú)
    Producto_heuristico_feromona_privada(:,curnode) = 0;

    for j = 3:n

        % Cálculo la probabilidad de paso
        Resultados_del_nodo_actual = Producto_heuristico_feromona_privada(curnode,:);
        Probabilidades_de_paso = Resultados_del_nodo_actual ./ sum(Resultados_del_nodo_actual);

        % Siguiente nodo a visitar
        Probabiliades_de_paso_acumuladas = cumsum(Probabilidades_de_paso);

        r = rand() * Probabiliades_de_paso_acumuladas(n);

        curnode = 1;

        while (Probabiliades_de_paso_acumuladas(curnode) < r)
            curnode = curnode + 1;
        end

        Rutas(k,j) = curnode;

        % Actualización de la lista tabú.
```

```
        Producto_heuristico_feromona_privada(:,curnode) = 0;
    end
end

end
```

FUNCIÓN: EVALUAR_RUTA_CON_DEMANDAS

```
function Costo= Evaluar_ruta_con_demandas(Q,h,Ruta,D,Dist)

% Definición del número de clientes más el depósito
n = size(Dist,1);

% Cálculo de la distancia de ir del depósito al primer cliente visitado
i = 2;
Costo = Dist(1,Ruta(i));

% Carga residual después de visitar al cliente i
q = Q - D(Ruta(i)-1);

for k=1:(n-2)

     % Si la carga residual después de servir al cliente i es menor que
     su umbral de capacidad se realiza el
     % reabastecimiento preventivo: El vehículo regresa al depósito se
     reabastece y retorna al próximo cliente de la ruta
    if q < h(Ruta(i)-1)
       Costo = Costo + Dist(Ruta(i),1) + Dist(1,Ruta(i+1));
       q = Q;

    % Si la carga residual después de servir al cliente i es mayor o
    igual que su umbral de capacidad, el vehículo
    % vista al próximo cliente de la ruta.
    else
        Costo = Costo + Dist(Ruta(i),Ruta(i+1));
    end

    i=i+1;

    % Si la demanda del próximo cliente (i+1) es menor o igual que la
     carga residual, habrá suficiente capacidad para atender su demanda
    if  D(Ruta(i)-1) <= q
        q = q - D(Ruta(i)-1);

   % Si la demanda del próximo cliente (i+1) es mayor que la carga
   residual, se presentará un fallo de ruta. El vehículo entregará la
   carga
   % residual y regresa al depósito para el reabastecimiento, retornando
   al cliente para finalizar el servicio.
    else
        Costo = Costo + 2*Dist(1,Ruta(i));
        q = Q + q - D(Ruta(i)-1);
```

```
    end

end

% Distancia total, teniendo en cuenta la distancia de ir del último
cliente al depósito.
Costo = Costo + Dist(Ruta(i),1);

end
```

FUNCIÓN: INTERCAMBIO_2OPT

```
function [Ruta_mejorada,Costo_ruta_mejorada] =
Intercambio_2opt(Ruta,e,D,Costo_promedio,Q,h,Dist)

% Definición del número de clientes más el depósito
n = numel(Ruta);

zmin = 0;
i = 1;

while i < n-1
    i = i+1;
    j = i+1;

    while j < n+1

        if i==2 && j==n
            j = j+1;
        else
            j=j+1;
            Ruta_mejorada = Ruta;
            Ruta_mejorada(i:j-1) = Ruta_mejorada(j-1:-1:i);

            % Evaluar la ruta intercambiando el nodo i con j-1, con
            diferentes escenarios de demandas
            Costo_ruta_mejorada = 0;
            for k=1:e
                Costo =
                 Evaluar_ruta_con_demandas(Q,h,Ruta_mejorada,D(:,k),Dist);
                Costo_ruta_mejorada = Costo_ruta_mejorada + Costo;
            end
            Costo_ruta_mejorada = Costo_ruta_mejorada/e;

            % Diferencia entre el costo de la ruta inicial y el costo de
            la ruta con el intercambio
            z = Costo_ruta_mejorada - Costo_promedio;

            % Si el intercambio de los nodos i y j-1 genera una ruta con
            mejor costo, el proceso de intercambio de nodos se finaliza.
            if z < zmin
                zmin = z;
```

```
                imin = i;
                jmin = j;
                i= n-1;
                j= n+1;
            end
        end
    end
end

% Aplicar cambio
if zmin < 0
    Ruta(imin:jmin-1) = Ruta(jmin-1:-1:imin);
end
Ruta_mejorada = Ruta;
Costo_ruta_mejorada = zmin + Costo_promedio;

end
```

FUNCIÓN: OPERADOR_MUTACION

```
function [Rutas_operador, Costo_rutas_operador]= Operador_mutacion(Rutas,Costo_rutas_hormigas_menor_a_mayor,e,D,Q,h,Dist)

% Definición del número de rutas a las cuales se les aplicará el operador de mutación (s) y el número de clientes más el depósito (n)
[s,n] =size(Rutas);


% Inicializar vectores en los que se guardaran las rutas y sus respectivos costos
m=2*s;

Nuevas_rutas = zeros (m,n);
Costo_nuevas_rutas = zeros (1,m);

Nuevas_rutas (1:s,:) = Rutas;
Costo_nuevas_rutas(1:s) = Costo_rutas_hormigas_menor_a_mayor;

%Aplicación de operadores de mutación para la construcción de nuevas rutas
Rutas_operador_mutacion=zeros(s,n);

for p = 1:s
    Puntos_de_insercion = sort(ceil(1+ (n-1)*rand(1,2)));

    % Si los puntos de inserción asumen el mismo valor, se vuelven y generan otros valores hasta que estos sean diferentes
    while Puntos_de_insercion(1)==Puntos_de_insercion(2)
        Puntos_de_insercion = sort(ceil(1+ (n-1)*rand(1,2)));
    end
    I = Puntos_de_insercion(1);
    J = Puntos_de_insercion(2);
```

```
    % Construcción de nuevas rutas. Se define un número aleatorio k para
    elegir el tipo de operador de mutación que se aplicará
    Rutas_operador_mutacion(p,:)= Nuevas_rutas(p,:);
    k=ceil(3*rand);
    switch k
        case 1 % Vuelta
            Rutas_operador_mutacion(p,I:J)                                  =
            Rutas_operador_mutacion(p,J:-1:I);
        case 2 % Intercambio
            Rutas_operador_mutacion(p,[I                J])                 =
            Rutas_operador_mutacion(p,[J I]);
        case 3 % Deslizamiento
            Rutas_operador_mutacion(p,I:J)                                  =
            Rutas_operador_mutacion(p,[I+1:J I]);
        otherwise
    end
end

% Evaluar las nuevas rutas obtenidas con el operador de mutación con los
escenarios de demandas. Hallar un costo promedio
Costo = zeros(1,e);
Costo_promedio = zeros(1,s);
for k=1:s
    for i=1:e
        Costo(i)                                                            =
        Evaluar_ruta_con_demandas(Q,h,Rutas_operador_mutacion(k,:),D(:,i)
        ,Dist);
    end
    Costo_promedio(k) = sum(Costo)/e;
end

% Guardar las nuevas rutas y sus costos en la matriz de rutas y el vector
costos, respectivamente
Nuevas_rutas((s+1):m,:) = Rutas_operador_mutacion;
Costo_nuevas_rutas(s+1:m) = Costo_promedio;

% Escoger las mejores rutas entre las construidas por las hormigas y el
operador de mutación
[Distancias_menor_a_mayor, index]= sort(Costo_nuevas_rutas);
Rutas_operador = Nuevas_rutas(index(1:s),:);
Costo_rutas_operador = Distancias_menor_a_mayor(1:s);

end
```

FUNCIÓN: EACO

```
function
[Ruta_EACO,Costo_ruta_EACO,Costo_minimo_cada_iteracion,Mejor_ruta_cada_it
eracion]=
EACO(m,e,Q,Dprom,S,h,Dist,Iter_EACO,rho_local,rho_global,alpha,beta,Costo
_ruta_vecino_mas_cercano)
```

```
% Definición del número de clientes más un depósito
n= size(Dist,1);

% Incialización de los valores de retorno
Costo_minimo_cada_iteracion = zeros(1,Iter_EACO);
Mejor_ruta_cada_iteracion = zeros(Iter_EACO,n);

% Inicializar feromonas y heuristicas
Feromona = ones(n,n)./((n-1)*Costo_ruta_vecino_mas_cercano);
Heuristica = ones(n,n) ./ (1e-10 * eye(n) + Dist);
Heuristica(1:n+1:n*n) = 0;

% Administración de estadísticas
Iter_actual = 1;
Costo_global_best = inf;

% Método evolutivo de colonia de hormigas para la solución del VRPSD
while Iter_actual <= Iter_EACO
    Rutas = Rutas_hormigas(m, alpha, beta, Feromona, Heuristica, Dist);

    % Definición de la demanda, capacidad del vehículo y umbral de
     capacidad para cada cliente
    [D,h]= Demandas(Dprom, S, e, h);

    %Evaluar rutas de la hormigas
    Costo = zeros(1,e);
    Costo_promedio = zeros(1,m);
    for k=1:m
        for i=1:e
            Costo(i)                                                    =
            Evaluar_ruta_con_demandas(Q,h,Rutas(k,:),D(:,i),Dist);
        end
        Costo_promedio(k)=sum(Costo)/e;
    end

    Costo_rutas = Costo_promedio;

    % Operador de mutación
    if mod(Iter_actual,2)==0
         [Rutas_operador,                          Costo_rutas_operador]=
        Operador_mutacion(Rutas,Costo_rutas,e,D,Q,h,Dist);
        Rutas = Rutas_operador;
        Costo_rutas = Costo_rutas_operador;
    end

    %Organizar rutas de menor a mayor
    [Costo_rutas_menor_a_mayor, indice]=sort(Costo_rutas);
    s=m/2;

    % Mejorar rutas con intercambio 2-opt
    Rutas_mejoradas = zeros(m,n);
    Costo_rutas_mejoradas = zeros (1,m);
```

```
for k=1:s
    [Rutas_mejoradas(k,:),         Costo_rutas_mejoradas(k)]         =
    Intercambio_2opt(Rutas(indice(k),:),e,D,
    Costo_rutas_menor_a_mayor(k), Q, h,Dist);
end

Rutas_mejoradas((s+1):m,:) = Rutas(indice(1:s),:);
Costo_rutas_mejoradas((s+1):m) = Costo_rutas_menor_a_mayor(1:s);

% Encontrar la mejor ruta de cada iteración
[Costo_minimo_cada_iteracion(Iter_actual),Indice]                    =
min(Costo_rutas_mejoradas);
Mejor_ruta_cada_iteracion(Iter_actual,:) = Rutas_mejoradas(Indice,:);

Costo_iteration_best = Costo_minimo_cada_iteracion(Iter_actual);
Iteration_best = Mejor_ruta_cada_iteracion(Iter_actual,:);

% Encontrar la mejor ruta global
if  Costo_iteration_best < Costo_global_best
    Costo_global_best = Costo_iteration_best;
    Global_best = Iteration_best;
end

% Administración de estadísticas
Iter_actual = Iter_actual + 1;

% Actualizar feromona local
Nueva_feromona = zeros(n,n);
for k = 1:m
    Nueva_feromona(Rutas_mejoradas(k,n),        Rutas_mejoradas(k,1))=
     Nueva_feromona(Rutas_mejoradas(k,n),    Rutas_mejoradas(k,1))    +
    1/((n-1)*Costo_ruta_vecino_mas_cercano);
    for j = 1:n-1
        Nueva_feromona(Rutas_mejoradas(k,j),Rutas_mejoradas(k,j+1))  =
        Nueva_feromona(Rutas_mejoradas(k,j),   Rutas_mejoradas(k,j+1))
        + 1/((n-1)*Costo_ruta_vecino_mas_cercano);
    end
end
Feromona = (1 - rho_local) * Feromona + rho_local * Nueva_feromona;

% Actualizar Feromona Global

% Si el número de la iteración actual es menor que el 50% del total
de iteraciones, la actualización global se realiza con la mejor ruta
de cada iteración,y cada 5 iteraciones se tiene en cuenta la ruta
Global_best

if  Iter_actual<= Iter_EACO*0.5
    if mod(Iter_actual,5)==0

        Nueva_feromona_global = zeros(n,n);
```

```
            Nueva_feromona_global(Global_best(n),Global_best(1))   =   (1/
            Costo_global_best);
            for j = 1:n-1
                Nueva_feromona_global(Global_best(j),Global_best(j+1))   =
                (1/ Costo_global_best);
            end

            Nueva_feromona_iteration_best = zeros(n,n);

            Nueva_feromona_iteration_best(Iteration_best(n),Iteration_bes
            t(1)) = (1/ Costo_iteration_best);
            for j = 1:n-1

               Nueva_feromona_iteration_best(Iteration_best(j),Iteration_
               best(j+1)) = (1/ Costo_iteration_best);
            end

        Feromona   =   (1   -   rho_global)   *   Feromona   +   rho_global   *
        Nueva_feromona_global           +           rho_global           *
        Nueva_feromona_iteration_best;

        else
            Nueva_feromona_iteration_best = zeros(n,n);

            Nueva_feromona_iteration_best(Iteration_best(n),Iteration_bes
            t(1)) = (1/ Costo_iteration_best);
            for j = 1:n-1

                 Nueva_feromona_iteration_best(Iteration_best(j),Iteration
                 _best(j+1)) = (1/ Costo_iteration_best);
            end

            Feromona   =   (1   -   rho_global)   *   Feromona   +   rho_global   *
            Nueva_feromona_iteration_best;
        end
    else

        % La ruta Global_best se emplea para la actualización cada dos
        iteraciones hasta el 80% del total de iteraciones
        if Iter_actual > Iter_EACO*0.5 && Iter_actual <= Iter_EACO*0.8
            if mod(Iter_actual,2)==0

                Nueva_feromona_global = zeros(n,n);

               Nueva_feromona_global(Global_best(n),Global_best(1)) = (1/
               Costo_global_best);
                for j = 1:n-1

                   Nueva_feromona_global(Global_best(j),Global_best(j+1))
                   = (1/ Costo_global_best);
                end

                Nueva_feromona_iteration_best = zeros(n,n);
```

```
                Nueva_feromona_iteration_best(Iteration_best(n),Iteration_
                best(1)) = (1/ Costo_iteration_best);
                for j = 1:n-1

                   Nueva_feromona_iteration_best(Iteration_best(j),Iterat
                   ion_best(j+1)) = (1/ Costo_iteration_best);
                end

                Feromona = (1 - rho_global) * Feromona + rho_global *
                 Nueva_feromona_global        +        rho_global        *
                 Nueva_feromona_iteration_best;

            else
                Nueva_feromona_iteration_best = zeros(n,n);

                Nueva_feromona_iteration_best(Iteration_best(n),Iteration_
                best(1)) = (1/ Costo_iteration_best);
                for j = 1:n-1

                   Nueva_feromona_iteration_best(Iteration_best(j),Iterat
                   ion_best(j+1)) = (1/ Costo_iteration_best);
                end

                 Feromona = (1 - rho_global) * Feromona + rho_global *
                 Nueva_feromona_iteration_best;
            end
        else
            % En el 20% restante del total de iteraciones la
            actualización global sólo se realiza con la ruta Global_best
            if Iter_actual > Iter_EACO*0.8
                Nueva_feromona_global = zeros(n,n);
                Nueva_feromona_global(Global_best(n),Global_best(1))     =
                (1/ Costo_global_best);
                for j = 1:n-1
                   Nueva_feromona_global(Global_best(j),Global_best(j+1))
                   = (1/ Costo_global_best);
                end
                Feromona = (1 - rho_global) * Feromona + rho_global *
                 Nueva_feromona_global;
            end
        end
    end
end

Ruta_EACO = Global_best;
Costo_ruta_EACO = Costo_global_best;

end
```

FUNCIÓN: CÓDIGO_GENERAL

```
close all
clc

Experimento= input('Ingrese la instancia a evaluar: ');
m = input('Ingrese el número de agentes: ');
e = input('Ingrese el número de escenarios a evaluar: ');
Iter_EACO = input('Ingrese el número de iteraciones: ');
rho_local = input('Ingrese el valor de la tasa de evaporación local: ');
rho_global = input('Ingrese el valor de la tasa de evaporación global :
');
alpha = input('Ingrese el valor de la importancia de fermona: ');
beta = input('Ingrese el valor de la importancia de la información
heurística: ');

% Factores
    [Dprom, S, X, r,h] = Factores(Experimento);

% Capacidad del vehículo, en función de la demanda promedio y el número
promedio de clientes atendidos antes de la descarga preventiva (r).
    n = numel(Dprom);
    Q = (sum(Dprom)*r)/n;

% Matriz de distancias
    Dist = Matriz_distancias(X);

% Costo de la ruta construido con la heurística del vecino más cercano
    [Ruta_vecino_mas_cercano,       Costo_ruta_vecino_mas_cercano]       =
     Vecino_mas_cercano(Dist);

% Método AS

[Ruta_EACO,Costo_ruta_EACO,Costo_minimo_cada_iteracion,Mejor_ruta_cada_it
eracion]=
EACO(m,e,Q,Dprom,S,h,Dist,Iter_EACO,rho_local,rho_global,alpha,beta,Costo
_ruta_vecino_mas_cercano);
```