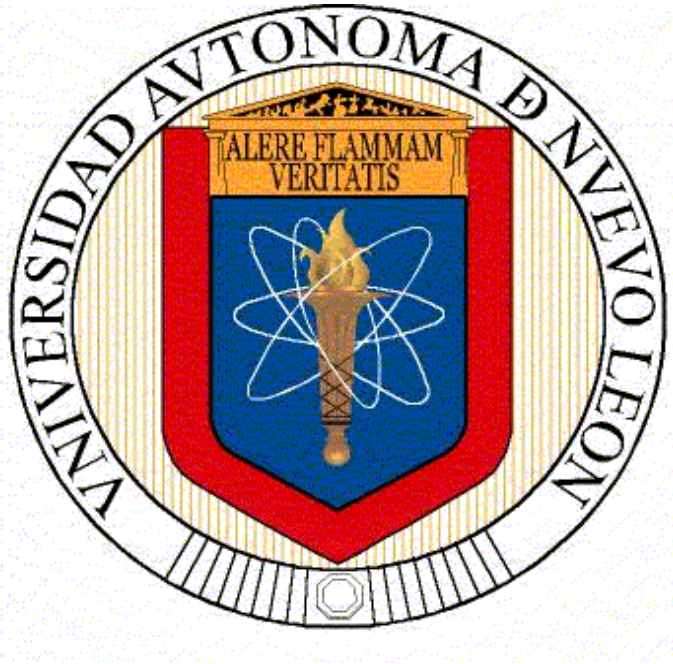
UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN
FACULTAD DE CIENCIAS FÍSICO-MATEMÁTICAS



UN ALGORITMO BASADO EN LA BUSQUEDA DISPERSA PARA RESOLVER EL PROBLEMA DE PRODUCCIÓN-DISTRIBUCIÓN DE UNA CADENA DE SUMINISTRO

PRESENTA

RAFAEL MUÑOZ SÁNCHEZ

EN OPCIÓN AL GRADO DE MAESTRÍA EN CIENCIAS CON ORIENTACIÓN EN MATEMÁTICAS

FEBRERO 2014

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE CIENCIAS FÍSICO-MATEMÁTICAS

CENTRO DE INVESTIGACIÓN EN CIENCIAS FÍSICO-MATEMÁTICAS

UN ALGORITMO BASADO EN LA BUSQUEDA DISPERSA PARA RESOLVER EL PROBLEMA DE PRODUCCIÓN-DISTRIBUCIÓN DE UNA CADENA DE SUMINISTRO

PRESENTA

RAFAEL MUÑOZ SÁNCHEZ

EN OPCIÓN AL GRADO DE:

MAESTRÍA EN CIENCIAS CON ORIENTACIÓN EN MATEMÁTICAS

SAN NICOLAS DE LOS GARZA, NUEVO LEÓN FEBRERO 2014

ÍNDICE GENERAL

RESUMEN

En este trabajo nosotros consideramos el problema de planeación de producción y distribución de una cadena de suministro en una red, que consiste de un conjunto de centros de distribución que buscan dar servicio a un conjunto de minoristas, y dichos centros de distribución abastecidos por un conjunto de plantas, buscando minimizar los costos de transportación en la red y de operación en las plantas, basado en el problema propuesto por Herminia y Calvete en 2011.

El problema es formulado como un programa matemático binivel donde el nivel superior (líder) consiste en fijar las rutas de distribución de productos enviados de los centros de distribución a los minoristas, satisfaciendo sus demandas sin exceder de un tiempo límite de duración de cada ruta. Por otro lado, en el nivel inferior (seguidor) se reciben las órdenes de cada centro de distribución y se deciden cuales plantas producirán estas órdenes satisfaciendo las demandas allí conjuntadas sin sobrepasar las capacidades de producción de las plantas. La función objetivo del nivel superior minimiza los costos incurridos en el envío de los productos desde los centros de distribución hacia los minoristas y los costos asociados al envío desde las plantas hasta los centros de distribución considerando un costo de descarga por artículo. En el nivel inferior se busca minimizar los costos de operación en las plantas.

En este trabajo proponemos un algoritmo heurístico basado en el equilibrio de *Stackelberg y la Búsqueda Dispersa.* El algoritmo propuesto consiste en aplicar la búsqueda dispersa en las variables del nivel superior encontrando la mejor respuesta del nivel inferior para cada solución obtenida por la búsqueda dispersa obteniendo así un equilibrio entre estos dos niveles. Nuestro algoritmo ha mostrado ser competitivo y brinda buenos resultados comparados con los publicados por Herminia y Calvete en 2011.

CAPÍTULO 1

INTRODUCCIÓN

En la actualidad las empresas han notado la importancia de la toma de decisiones en los procesos operativos y se han dado cuenta que decidir de manera empírica puede costar mucho trabajo, tiempo o esfuerzo para la compañía. Cuando esto ocurre, las compañías tienden a perder muchos recursos. Es por esto que es necesario implementar sistemas que busquen optimizar o mejorar la calidad de los procedimientos de operación dentro de una empresa y sustituyan las decisiones empíricas.

Lo anterior lo vemos especialmente reflejado en las cadenas de suministro (SC, por sus siglas en inglés). En este trabajo consideramos una cadena de suministro como un sistema que involucra desde el procesamiento de materias primas y se convierten en productos hasta llegar al consumidor. Los agentes que intervienen en estas cadenas son: proveedores, fabricantes, distribuidores y clientes.

Una de las clasificaciones del manejo de la cadena de suministro es la siguiente: 1.- La planeación de la producción y el control de inventarios. 2.- El proceso de distribución y logística. Estos dos son considerados como procesos básicos dentro de la SC. En Beamon (1998) podemos ver cómo el primer proceso se encuentra desde la fabricación y el almacenamiento de productos, así como sus subprocesos y sus interfaces. Por otra parte el segundo proceso determina cómo se transportan los productos desde el almacén hasta los minoristas.

Un factor importante para las firmas son los costos de transporte, estos juegan un papel muy importarte en la SC. Esto podemos verlo en la gran cantidad de producción científica en el área de transporte. Dentro de la planificación de la

distribución encontramos diferentes tipos problemas, uno de los más típicos es el llamado ruteo de vehículos que a continuación describimos.

En general; el problema de ruteo de vehículos (por sus siglas en inglés, VRP) consiste en encontrar la mejor configuración de rutas de una flota de vehículos, que dan servicio a un conjunto de clientes distribuidos a través de una red, buscando minimizar los costos de transporte o el número de vehículos utilizados. Se puede decir que el origen del VRP se presenta en las década de los 50´s y es introducido por Dantzig et al. (1954), quienes trabajaron el problema del agente viajero (Travelling Salesman Problem por sus siglas en inglés, TSP) a gran escala.

Dentro de los artículos pioneros del VRP destaca el de Dantzig and Ramser (1959) en el cual se introduce un procedimiento formulado como un problema lineal para encontrar las rutas óptimas para un sistema de distribución de gasolina. Una variante al trabajo anterior es el propuesto por Clarke and Wrigth (1964) en el cual consideran capacidades múltiples en los vehículos, ellos resuelven el problema con un procedimiento iterativo y de selección rápida.

El VRP es uno de los problemas más desafiantes dentro de la optimización combinatoria debido a su complejidad computacional. El aporte proporcionado por Lenstra and Rinnoy (1981) se convirtió en una de las características principales de este tipo de problemas al demostrar que los VRP y sus variantes, pertenecen al tipo NP-Hard.

A continuación mencionamos brevemente algunas de las diferentes variantes del VRP, teniendo en cuenta que cada una de ellas busca encontrar la ruta óptima en el menor costo y tiempo satisfaciendo las demandas de los clientes, de la misma manera:

VRP capacitado (CVRP): Se cuenta con una flota de vehículos homogénea con la misma capacidad.

VRP con múltiples depósitos (MDVRP). En esta versión del VRP se cuenta con la posibilidad de tener más de un depósito para servir a los minoristas.

VRP con entregas y devoluciones (VRPPD). Esta extensión del VRP se caracteriza principalmente por la posibilidad de que los minoristas puedan regresar algún tipo de mercancía, y por lo tanto se debe tomar en cuenta que los artículos puedan caber en el mismo vehículo de entrega.

VRP con ventanas de tiempo (VRPTW): A cada cliente se le asocia un intervalo $[a, b]$ llamada ventana de tiempo, así cada cliente debe ser atendido en un horario fijo previamente definido.

VRP con entregas divididas por diferentes vehículos (SDVRP): Se considera como una relajación del problema original. Difiere en que los minoristas pueden ser servidos por diferentes vehículos.

VRP estocástico (SVRP): Los componentes de esta variante son aleatorios, existen 3 casos para esta variante. 1.- Los minoristas estocásticos, significa que cada minorista está presente con una probabilidad *p* y ausente con probabilidad *1-p*. 2.-La demanda de cada minorista es una variable aleatoria. 3.- Los tiempos de traslado y de servicio son variables aleatorias.

Existen más variantes del VRP e inclusive combinaciones de los ya aquí mencionados. Podemos encontrar interesantes revisiones sobre el problema de ruteo de vehículos en Toth and Vigo (2002), Golden et al. (2008) y Choong et al. (2008). Una parte de nuestro trabajo se enfoca en el problema de múltiples depósitos asociados a un VRP (por sus siglas MDVRP).

Al hablar del problema de MDVRP tenemos que enfatizar en el primer trabajo de esta variante de VRP propuesto por Laporte and Norbert (1983) el cuál más tarde fue formalizado por Laporte et al. (1988). Así como también es importante mencionar a Carpaneto et al. (1989) quienes proponen un algoritmo basado en ramificación y acotamiento para dar solución al problema. Este problema ha tomado más relevancia para los investigadores en los últimos 5 años según las gráficas expuestas en López and Nieto (2012).

Los trabajos que involucran al VRP o alguna de sus variantes suelen tener un solo objetivo, estos son conocidos como mono-objetivos. Frecuentemente los procesos dentro de una SC se consideran como sistemas separados, es decir no involucran los dos procesos básicos dentro de un mismo sistema.

Sin embargo debemos mencionar que las últimas dos décadas se han incrementado la cantidad de trabajos en los que se involucran más de un elemento en la SC. Así los componentes más importantes en la cadena de suministro que tienen que integrarse en el mismo problema son los procesos de producción-distribución. Lo anterior lo podemos ver en Fahimnia et al. (2013).

Los trabajos que involucran a estas dos actividades dentro de una SC podemos ver que son modelados con programación multiobjetivo. Estos se caracterizan por tener más de un objetivo dentro de una misma organización. Este es el enfoque más común para el desarrollo de modelos que involucran diferentes etapas en una SC. Los problemas de programación multiobjetivo también tienden a ser interesantes y desafiantes para su análisis.

Una extensa revisión de literatura de los modelos que asocian la producción con la distribución se presentan en Vidal and Goetschalckx (1997); en Sarmiento and Nagi (1999) y Goetschalckx et al. (2002) se presenta una revisión de modelos que integran estos dos procesos. En Chen (2004) se reconoce la taxonomía de estos modelos basados en la estructura de integración, el nivel de las decisiones y los parámetros por los que están formados.

En Fahimnia et al. (2013), este problema integrado se clasifica de acuerdo a su complejidad, es decir, se basa en uno o múltiples productos, diversas plantas, el número de tiendas, vehículos individuales o múltiples, el número de períodos en los vehículos además de realizar una reclasificación sobre la base de técnicas de implementación se utilizaron para la resolución.

Los problema de producción-distribución han sido resueltos de diferentes maneras; una de las formas en que se han solucionado estos es de forma exacta

mediante el uso de software de optimización, sin embargo los optimizadores sólo han sido capaces de resolver con precisión casos de estudio de dimensión pequeña, la principal razón es debido a que contienen una gran cantidad de restricciones y variables [ver Dhaenens-Flipo and Finke (2001), Jayaraman and Pirkul (2001), Jang et al. (2002)].

Métodos exactos como relajación Lagrangeana, generación de columnas, métodos de puntos interiores, entre otros, se han utilizado para la solución o aproximación de estos problemas. Uno de los métodos más utilizados es la técnica de la relajación Lagrangeana, podemos encontrar algunas aplicaciones de esta técnica en problemas de producción y distribución en Barbarosoglu and Ozgur (1999) y Elhedhli and Goffin (2005).

Además algunos algoritmos heurísticos se han desarrollado para estos problemas. Ejemplos de este tipo de algoritmo son vistos en Cohen and Lee (1998) ellos resolvieron el problema en una forma jerárquica mediante la fijación de variables, Keskin and Uster (2007) propusieron diversas heurísticas constructivas con el fin de obtener buenas soluciones para una parte especifica del problema y la solución de los subproblemas resultantes.

En Williams (1981) se pueden encontrar siete heurísticas para resolver un problema de producción-distribución simultánea. Motivados por el reto de obtener una solución de buena calidad en un tiempo razonable, diversas metaheurísticas han sido desarrolladas para este tipo de problemas; Keskin and Uster (2007) presentan un algoritmo basado en Búsqueda Dispersa con trayectorias vinculadas, trayectorias base y un algoritmo de Búsqueda Tabú; Boudia et al. (2007) propusieron un GRASP con trayectorias vinculadas para un problema de producción-distribución; algoritmos genéticos pueden ser encontrados en Syarif et al. (2002), Chan et al. (2005) y Kazemi et al. (2009); y algoritmos meméticos son propuestos por Boudia and Prins (2009).

El problema que es analizado en este trabajo y el cuál describiremos más adelante en la sección 1.1 es propuesto por Calvete et al. (2011) y conjunta los dos procesos importantes de la cadena de suministro mencionados anteriormente. A

diferencia de los modelos ya existentes, en Calvete et al. (2011) conjuntan por primera vez estas dos actividades de una SC mediante un modelo de programación binivel.

La solución presentada en Calvete et al. (2011) es mediante un algoritmo de colonia de hormigas. Después, Legillon et al. (2011) presenta un algoritmo coevolutivo que difiere del algoritmo presentado por Calvete et al. (2011). Los resultados de Legillon et al. (2011) no son comparados con el trabajo original, después analizaremos más a detalle estos dos trabajos.

Nosotros proponemos un algoritmo heurístico que resuelve satisfactoriamente el problema de planeación de la producción-distribución de una cadena de suministro. Dicho algoritmo se fundamenta en la Búsqueda Dispersa y en el equilibrio de Stackelberg. Los resultados son presentados y a su vez comparados con los mostrados en Calvete et al. (2011).

Según Laguna and Martí (2003) la Búsqueda Dispersa es un método evolutivo que ha sido aplicado en problemas de optimización muy complejos. Los principios de este algoritmo fueron propuestos por primera vez en la década de los 70´s, y están basados en formulaciones realizadas en la década de los 60´s, las cuales se fundamentan en reglas de combinación y problemas con restricciones. Este método utiliza estrategias de diversificación e intensificación para la búsqueda, y tiene una amplia variedad de implementaciones en los diferentes problemas de optimización. En el Capítulo 4 detallaremos el algoritmo de la Búsqueda Dispersa.

Como ya mencionamos anteriormente este algoritmo también se fundamenta en el equilibrio de Stackelberg, recordemos que este equilibrio implica específicamente que un juego es repetitivo. Se tienen dos o más firmas que compiten a fin de dominar el mercado, una de estas firmas se denomina líder y las demás se conocen como seguidores. El líder realiza primero su movimiento y consecuentemente el seguidor reacciona ante la decisión del líder, una vez que el líder conoce la respuesta del seguidor este realizará su movimiento, así hasta tener un equilibrio jerárquico entre estos participantes.

1.1 PLANTEAMIENTO DEL PROBLEMA

En una cadena de suministro sabemos que existen varios procesos que interactúan, como los mencionados anteriormente. En la literatura dichos procesos son modelados dentro de la programación matemática como problemas en los cuales se tienen objetivos comunes de una sola firma.

Estos modelos propuestos en la literatura que intentan manejar estos dos o más procesos se pueden dividir en dos casos: 1.- La empresa puede controlar todos los procedimientos en una cadena de suministro o 2.- se asume que varias empresas pueden controlar dichas actividades pero todas ellas con un mismo fin.

Es interesante pensar qué pasaría si estos mismos procesos son controlados por diferentes compañías, por ejemplo, la producción (operación de artículos) manejada por una compañía y la distribución por otra diferente a la primera, donde cada uno busca su beneficio en particular, sin embargo están mutuamente relacionadas.

No es difícil ver que si planteamos este escenario no sería para nada viable modelar este problema con la programación multiobjetivo debido a que no son objetivos en común de una misma firma. Ahora supongamos que existe dentro de ellas una jerarquía, es decir hay una prioridad, es aquí donde este problema puede verse como un problema de programación binivel.

Ahora analicemos la situación, dentro de una cadena de suministro tenemos una empresa que se dedica exclusivamente a la distribución de algunos productos y otra que se dedica únicamente a la producción de estos mismos. Antes de distribuir la firma debe adquirir los bienes desde las plantas donde son producidos, estos bienes son almacenados antes de distribuirse, en lugares que se conocen como centros de distribución. Todo esto con el fin de minimizar los costos de distribución totales y los costos de adquirir los artículos.

La compañía destinada a la producción intenta minimizar sus costos de operación tomando la decisión en base a sus condiciones y a las de la firma distribuidora. Es evidente que la decisión que selecciona la firma productora afecta en la determinación que tome la firma distribuidora, puesto que la mejor decisión que tome la que se encarga de producir no siempre será la decisión óptima de la que se ocupa de distribuir.

En el análisis anterior podemos apreciar que la empresa que se encarga de suministrar debe tomar en consideración la decisión de la compañía productora, todo esto para su propio beneficio. No es absurdo pensar que esta cuestión se puede modelar con programación binivel, debido a que ambos tienen estrategias de optimización diferentes y existe una jerarquía entre ellas.

Recordemos que la programación binivel está formada por un nivel superior de mayor jerarquía y un nivel inferior de menor jerarquía, más adelante abordaremos este tema de manera extensa. En nuestro problema podemos apreciar que el nivel superior es la firma que se dedica a la distribución de los artículos, él se encuentra influenciado por el comportamiento de la segunda firma, la cual controla el proceso de producción y están limitados por la resolución del nivel superior.

1.2 MOTIVACIÓN

Diferentes disciplinas se han desarrollado dentro de la programación binivel, dentro de ellas destacan las áreas de ingeniería. En particular, el problema a tratar en este trabajo cae dentro de esta área. Se sabe que el problema de ruteo de vehículo con múltiples depósitos tiene una complejidad importante debido a sus propias características y es evidente que se vuelve un poco más complejo si añadimos otro problema a este mismo.

Dentro de la revisión de literatura vamos a ver más adelante que sólo existen dos trabajos relacionados que abordan esta problemática, por lo cual es muy

novedoso para el desarrollo de nuevas soluciones. Asimismo sabemos que los problemas combinatorios no tienen una metodología única para su solución, por lo cual es importante desarrollar técnicas nuevas y eficientes.

Si bien ya existe algún algoritmo que fue propuesto para su posible solución, en el primer trabajo desarrollado no garantizan tener los resultados óptimos, por este motivo principal nosotros decidimos proponer un algoritmo heurístico que tenga resultados de mejor calidad en la función objetivo y además cuente con tiempos eficientes en cuanto a su rendimiento.

1.3 OBJETIVO

Los objetivos de este trabajo de tesis consisten en proporcionar una alternativa diferente de solución al problema de planeación de la producción y distribución de una cadena de suministro ya recientemente planteado en la literatura. Nuestra variante consiste en desarrollar una metodología basada en la metaheurística de Búsqueda Dispersa y en el equilibrio de Stackelberg. Proporcionamos una metodología nueva para el método de combinación agregando cierta aleatoridad.

Este algoritmo es implementado en el lenguaje de programación C++ y proporciona resultados eficientes en cuanto a tiempos comparados con los presentados en la literatura.

1.4 ESTRUCTURA DE LA TESIS

En este capítulo describimos como está formado nuestro trabajo. En el capítulo 1 dimos una breve introducción acerca de las cadenas de suministro y sus procesos, como son normalmente manejados (separados o juntos) y como será modelado en este trabajo. Además de introducir brevemente la forma en que será resuelto el problema.

En el capítulo 2 describiremos a detalle la programación binivel y algunas aplicaciones detalladas que existen en esta rama. En el capítulo 3 describimos de manera explícita el problema para después formular el modelo matemático (en la literatura no se encuentra el modelo a detalle), y describir cada una de sus restricciones y variables.

En el capítulo 4 presentamos el algoritmo heurístico y describimos cada uno de sus pasos y por último en el capítulo 5 presentamos los resultados obtenidos por las pruebas computacionales y mostramos las conclusiones de nuestro trabajo.

CAPÍTULO 2

PROGRAMACIÓN BINIVEL

En nuestros días frecuentemente las personas encargadas de tomar decisiones en empresas privadas, instituciones de gobierno, institutos educativos entre otras, tienen ciertas prioridades en la realización de actividades, como importancia de costos, tiempos de traslado o existencia de riesgos dentro de estas organizaciones. Por lo cual es necesario analizarlas por niveles de jerarquía. El área de investigación que se dedica a un conjunto de estructuras de jerarquía en los procesos de toma de decisiones se conoce como programación multi-nivel.

Los modelos de programación multi-nivel se caracterizan por tener diferentes niveles de jerarquía dentro de sus restricciones, los cuales representan cada uno un problema de optimización diferente. Siendo los niveles jerárquicos los de mayor importancia para el tomador de decisiones, esto es conforme los niveles vayan descendiendo suelen tener menos relevancia al tomar la decisión.

En general las características de estos modelos son las siguientes:

- El modelo está compuesto de unidades de toma de decisiones que interactúan de una estructura jerárquica de niveles.
- Cada nivel inferior selecciona su política después de conocer las decisiones de los niveles superiores, esto es los niveles de decisión optimizan sus objetivos de forma secuencial.
- Cada nivel intenta optimizar su objetivo independientemente de los otros, y estos a su vez pueden alterarse por las acciones y reacciones de alguno de ellos.

- La influencia de un nivel superior en el problema correspondiente al nivel inferior se puede reflejar en su función objetivos como el conjunto de decisiones factibles.

Un caso particular de la programación multi-nivel es la que está compuesta únicamente por dos niveles, a ésta se denomina programación binivel. Al nivel de mayor jerarquía se le conoce como nivel superior (líder) y al otro como nivel inferior (seguidor).

2.1 CONCEPTOS GENERALES

A continuación presentamos un modelo de programación binivel donde se tienen los vectores x como variable del líder y y como variable del seguidor. Primero mostramos el nivel inferior:

$$\psi(y) \coloneqq min_x \{f(x,y): g(x,y) \leq 0\} \tag{2.1}$$

Sea $f: \Re^n \mathrm{X} \Re^m \longrightarrow \Re$ la función objetivo y sean $g: \Re^n \mathrm{X} \Re^m \longrightarrow \Re^p$, $[g_1(x,y), \dots, g_p(x,y)]^T$, el conjunto de restricciones. La solución del problema (2.1) se conoce como el conjunto $\psi(y)$ para algún $y \in \Re^m$.

Por otra parte, la función objetivo del líder $F(x,y)$, donde $F: \Re^n \mathrm{X} \Re^m \longrightarrow \Re$, y el conjunto de restricciones $G: \Re^m \longrightarrow \Re^w$. Así la formulación del problema del líder puede verse como:

$$"min"_y \{F(x,y): G(y) \leq 0, x(y) \in \psi(y)\} \tag{2.2}$$

Así el modelo de (2.1)-(2.2) es propuesto en Dempe (2002)

Las comillas son utilizadas debido a la incertidumbre que existe en el caso donde el nivel inferior no tenga solución única.

El siguiente conjunto se le conoce como como la región inducida:

$$\mathcal{IR} = \{(x, y) \in \Re^n \mathrm{X} \Re^m : G(y) \leq 0, x \in \psi(y)\} \tag{2.4}$$

Este conjunto reagrupa los puntos factibles del problema de programación binivel, y corresponde al conjunto factible del líder. Generalmente éste conjunto de soluciones factibles del líder es no convexo e inclusive poder ser vacío por algunas restricciones del nivel superior.

Podemos notar que en el problema (2.3) el líder no puede influir en la decisión del seguidor para su propio beneficio. Por lo tanto, el líder no puede predecir el verdadero valor de su función objetivo hasta conocer la respuesta del seguidor. Es aquí donde se introducen dos enfoques especiales para estos problemas. Es importante aclarar que estos casos ocurren cuando haya más de un único óptimo para el nivel inferior. A continuación los mencionamos:

El Primer caso se asume que el líder es capaz de tener influencia en la respuesta del seguidor, al seleccionar una $x(y) \in \psi(y)$ tal que esta solución es mejor para el líder. A este caso se le conoce como *Caso Optimista*, y se puede apreciar de la siguiente manera:

$$\rho_o(y) \coloneqq min_x \{F(x, y) : x \in \psi(y)\} \tag{2.5}$$

La ecuación (2.5) denota el valor de la función objetivo del líder dado que selecciona una $x \in \psi(y)$, así el problema binivel optimista se reduce a:

$$min_y \{\rho_o(y) : G(y) \leq 0\} \tag{2.6}$$

Así vemos cómo esta posición optimista no parece tener ningún problema, a excepción en los casos donde exista la no cooperación, es decir donde el seguidor no actúa a favor del líder. Esto se puede ver de la siguiente forma:

$$\rho_p(y) \coloneqq max_x \{F(x, y) : x \in \psi(y)\} \tag{2.7}$$

Así el problema pesimista se reduce a:

$$min_y \{\rho_p(y): G(y) \leq 0\} \tag{2.8}$$

Algunos de estos conceptos y definiciones presentados en este trabajo pueden verse en Dempe (2002) más detalladamente.

2.2 PROPIEDADES DE PROGRAMACIÓN BINIVEL

Existen diversas aportaciones teóricas, dentro de las más importantes para solucionar problemas de programación binivel podemos ver las siguientes: Jeroslow (1985) muestra que los problemas de programación bilineales son NP-Hard, y poco después Ben-Ayed and Blair (1990) y Bard (1991) demuestran que algunos problemas de programación binivel son de tipo NP-Hard.

En la búsqueda de condiciones de optimalidad necesarias y suficientes diversos investigadores se han esforzados por obtener algunas de ellas. Uno de los primeros en analizar este tipo de condiciones fue Bard (1984) haciendo un análisis similar con una programación matemática de un solo nivel, teniendo un conjunto de restricciones infinitas.

Poco tiempo después de la aportación anterior Clarke and Westerbeg (1988) muestran un contraejemplo a las condiciones propuestas anteriormente, así como Haurier et al. (1990) hace propiamente un contraejemplo. Por lo anterior podemos ver en Savard (1989) concluye que los algoritmos basados en estas condiciones no son convergentes. Por otra parte es importante resaltar los trabajos de Dempe (1992a), Dempe (1992b), Chen and Florian(1991) y Outrata (1993) quienes usan técnicas de análisis no suave como herramientas [éste puede verse en Clarke(1990)] para establecer condiciones de optimalidad.

Por último se puede ver en Savard and Gauvin (1994) y Vicente and Calamai (1995) desarrollan condiciones de optimalidad tomando en cuenta la geometría de la

región inducida: el primer trabajo se refiere a una adaptación de las nociones del método de mayor descenso (*steepest descent*) para el caso de la programación binivel no lineal, mientras que el segundo trabajo establecen las condiciones de primer y segundo orden para el caso de la programación binivel cuadrática con el nivel inferior estrictamente convexo. El principal resultado del trabajo de Vicente and Calamai (1995) que para cada punto de la región inducida, existe un número finito de conos convexos de direcciones en ésta.

Sin embargo las anteriores condiciones pueden no ser tan convenientes para el uso práctico y además no proporcionan criterios de paro para los algoritmos enumerativos. En la siguiente sección mostraremos más trabajos de aplicaciones de Programación Binivel y algunos algoritmos exactos que se han desarrollado.

2.3 REVISIÓN DE LITERATURA PROGRAMACIÓN BINIVEL

En esta sección haremos una breve descripción de algunos trabajos destacados dentro de la programación binivel, desde sus inicios hasta las actuales aplicaciones que existen. Su formulación original apareció en el trabajo de Bracken and McGill (1973), sin embargo fue en Candler and Norton (1977) donde se usó por primera vez la designación binivel y multinivel.

La mayor parte de los trabajos han sido motivados por la teoría de juegos de Stackelberg (1934), en éste podemos apreciar una investigación sobre situaciones reales de mercado y es considerado como el primero en realizar estudios basados en programación binivel. Así también podemos ver en Stackelberg (1952) la forma en que un problema binivel es analizado bajo el enfoque de teoría de juegos.

En las últimas tres décadas diversos investigadores han resuelto problemas binivel relacionados con áreas de transporte, administración, planeación y diseño de la ingeniería, entre otras. A continuación mencionamos algunos trabajos que se han desarrollado en estas áreas:

- *Problemas de Transporte*: LeBlanc and Boyce (1986) formulan un problema de diseño de redes de transporte con enfoque binivel y lo resuelven mediante un algoritmo exacto. Marcotte and Marquis (1992) implementan un heurístico a un problema de diseño de redes donde se asume que el usuario se comporta de acuerdo al principio del equilibrio de tráfico. Otro trabajo interesantes es el de Florian and Chen (1991) en el cual se proporciona una estimación para la demanda del problema de congestión en una red, la aportación más significativa de éste trabajo es el algoritmo exacto, el cual se basa en una función de penalización.

  Por otra parte podemos ver en Bianco et al. (2009) el problema de transportación de materiales peligrosos modelado con dos niveles, este modelo es reformulado con las condiciones de KKT y resuelto mediante un optimizador. Una variante al problema descrito anteriormente es el propuesto por Hongmei et al. (2010), en el cual se consideran restricciones de tiempo y calendarización y la función objetivo del líder depende del tiempo a diferencia de Bianco et al. (2009).

  También podemos ver en Sonia et al. (2008) otro problema de transporte con restricciones de suministro y demanda en el cual la función objetivo consiste de un min-max de tiempo. Kalashnikov et al. (2010) proponen una formulación estocástica para el problema de flujo de gas natural mediante el uso de programación binivel.

- *Administración*: En Miller et al. (1992) se propone un modelo para la localización de plantas de producción de una empresa, además se propone un algoritmo heurístico de solución. En Guo (2012) se considera un modelo de transferencia de agua, en el nivel superior un multi-deposito optimiza la asignación de recursos hidráulicos mientras en el seguidor se considera un depósito individual el cual busca el mejor suministro para él. En Lukac et al. (2008) podemos ver un problema de administración de la producción de unos productos, la acción del líder

es asignar los productos a las máquinas con el fin de minimizar el tiempo de producción mientras que el seguidor reduce los costos de producción.

- *Planeación*: Mientras que un problema típico de planeación de la agricultura (energía y medio ambiente) es visto con el enfoque binivel en Candler et al. (1981). Por otra parte se modela con un problema de interacción entre empresas de servicios públicos y co-generadores de energía eléctrica, este problema es resuelto con un algoritmo de aproximación y se resuelve un caso de estudio. Un problema poco usual en la literatura pero interesante es el presentado en Scaparra and Church (2008), que plantea la problemática de ataques de terroristas. El nivel superior decide que instalaciones proteger para disminuir las pérdidas de instalaciones sin protección, mientras el seguidor asume el problema de destrucción del terrorista.

  Un modelo innovador es el presentado en Jia et al. (2014) donde se modela la planeación de la producción y distribución, donde el líder es la firma encargada de la distribución y el seguidor es la de la producción, éste último tiene dos objetivos. Este problema es resuelto mediante un algoritmo genético.

- *Diseño de Ingeniería*: En Kocvara and Outrata (1995) se presenta un problema de sistemas de control los cuales son descritos por desigualdades de orbitas elípticas. Un trabajo interesante en cuestión de energía es el presentado por Zugno et al. (2013), en el cual se propone un modelo entre minoristas y consumidores en un entorno de precios dinámicos. El modelo determina la máxima ganancia para el minorista (líder) con el patrón de carga óptima para los consumidores el seguidor (seguidor).

Dentro de los algoritmos que son utilizados para solucionar los problemas binivel podemos destacar a los algoritmos enumerativos, principalmente cuando los

problemas tienen forma lineal, por su exactitud suelen dar buenas soluciones en tiempos no tan pequeños. Algunos de ellos podemos verlos en Bialas et al. (1984), Calvete et al. (1999). Este tipo de algoritmos también se han propuesto para problemas no lineales, un caso de estos se puede ver en Camacho and Muñoz (2013).

Algoritmos tradicionales basados en ramificación y acotamiento y algoritmos de pivoteo complementario han sido implementados para problemas con ciertas características, especialmente cuando el nivel inferior se reemplaza por las condiciones de Karush-Kuhn-Tucker, en las cuales se aprovecha la naturaleza de las condiciones de holgura complementaria. Algoritmos basados en esta idea puede verse en Judice et al. (1992) y Bard and Falk (1982). Un trabajo reciente sobre la adaptación de algoritmos de ramificación y acotamiento a problemas de programación binivel lineales puede verse en Shi et al. (2006).

Problemas donde las funciones no son lineales suelen solucionarse mediante la determinación de direcciones de descenso como en Savard (1994) proponen un método de descenso para problemas binivel cuadraticos convexos, donde la función objetivo es una función cuadrática y las restricciones son lineales. Otro ejemplo de este tipo de algoritmos podemos verlo en Falk and Liu (1995) en el cual presentan un método donde monitorean la disminución de la función objetivo de acuerdo a la información subdiferencial del nivel inferior, este método lo llaman *the leader predominate algorithm*. Una de las características de los algoritmos anteriores es que garantizan exclusivamente la optimalidad local de la solución encontrada, para más especificación véase Dempe (2002).

Por otra parte existen métodos de penalización importantes los cuales constituyen una clase de algoritmos para la solución de problemas no lineales, con éstos algoritmos suelen encontrarse puntos mínimos locales por lo cual son muy fáciles de estancarse, esto se puede ver en Colson et al. (2005).

Debido a que los problemas son no convexos diversos investigadores han decidido proponer algoritmos heurísticos para solucionar estos problemas. Los primeros en desarrollar algoritmos genéticos (GA, por sus siglas en inglés) para

problemas lineales fueron Mathieu et al. (1994), éstos argumentan que debido a las características de estos algoritmos como su sencillez, una perspectiva global y un paralelismo implícito pueden ser eficientes para estos problemas. También podemos apreciar en Wang et al. (2007), Calvete et al. (2008), Hejazi et al. (2002) y Wang et al. (2008) algoritmos genéticos aplicados a problemas binivel.

Para finalizar mencionaremos algunos algoritmos heurísticos o metaheurísticos usuales en este tipo de problemas son Tabu Search en Gendreau (1996) así como Rajesh J. (2003) aplicado a un problema de ingeniería química, algoritmos difusos los vemos en Oña (2011), así como optimización de colonias en Calvete (2011) y Scatter Search en González-Velarde et al. (2012). Entre otros trabajos que involucran algoritmos heurísticos se pueden ver en Wan et al. (2013), Camacho-Vallejo et al. (2013), Dussault (2006) y Marić et al. (2012). Así vemos cómo diferentes algoritmos heurísticos se han implementado a problemas binivel teniendo algunas ventajas importantes sobre los métodos exactos particularmente en estos problemas.

CAPÍTULO 3

DESCRIPCIÓN Y MODELACIÓN DEL PROBLEMA

En este capítulo describiremos detalladamente el planteamiento del problema de la planeación de la producción y distribución. Primero se presenta una revisión de literatura sobre los trabajos relacionados que abordan este problema y las metodologías propuestas para su solución. Después presentamos el modelo matemático, así como su descripción.

3.1 TRABAJOS RELACIONADOS

En la literatura nosotros encontramos dos trabajos relacionados con el mismo problema: el trabajo en el que es descrito por primera vez y un trabajo que propone otra alternativa para solucionar el problema, la cual consiste en una metaheurística para problemas de programación binivel sin el enfoque tradicional de solución. En esta sección describimos y analizamos los dos métodos de solución previos a nuestra propuesta planteada.

El primer enfoque de solución presentado en Calvete et al. (2011) se basa en la meta-heurística de colonia de hormigas con el enfoque de Stackelberg. Una parte de las instancias con las cuales fue probado su algoritmo fueron obtenidas de un caso de estudio real y otras son una adaptación de las instancias proporcionadas por Cordeau et al. (2008) utilizadas comúnmente para comparar algoritmos para problemas de MDVRP, añadiendo únicamente la parte de datos del nivel inferior.

Los algoritmos de colonia de hormigas (por sus siglas en ingles BACS) intentan imitar el comportamiento de éstos animales cuando están en busca de alimentos. En la

realidad se conoce que la mayor parte de estos animales son capaces de mandar información entre ellos para encontrar fuentes de alimentos. Mientras ellos suelen moverse ponen un rastro de feromonas químicas en el suelo, dicha cantidad de feromonas depende de la calidad y cantidad de alimento encontrado. Al elegir su camino las hormigas huelen las feromonas y tienden a elegir los caminos macados por fuertes concentraciones de alimento.

Los autores de este trabajo utilizan las ideas de algoritmos de colonia de hormigas para encontrar una solución del nivel superior. Al mismo tiempo ellos resuelven el problema del nivel inferior una vez que se conoce la información en cada centro de distribución. Esta información se actualiza dependiendo del rastro de la calidad de la feromona de las soluciones encontradas. A continuación se describe los pasos más importantes de éste algoritmo.

Como primer paso se genera una solución inicial al problema, posteriormente se inicializan los parámetros del BACS los cuales son seleccionadas de la forma típica que recomienda la literatura. Después se busca formar un conjunto de M soluciones factibles para el problema binivel y se actualiza cierta información en los arcos. Posteriormente se resuelve el nivel inferior. Así hasta completar la cantidad de M soluciones, manteniendo la información de la mejor solución, este proceso de construcción de soluciones se realiza hasta iterar una cierta cantidad de veces. Finalmente la solución de mejor valor de la función objetivo del líder es seleccionada.

Para la construcción de una solución el algoritmo trabaja de la siguiente forma: Cada hormiga representa un vehículo, la cual va construyendo rutas factibles, ésta va seleccionando minoristas sucesivamente los cuales no han sido visitados anteriormente. En el caso en que al añadir un minorista se produzca infactibilidad, entonces la hormiga regresa a su origen, es decir al centro de distribución de donde partió y comienza de nuevo otra ruta hasta terminar de visitar todos los minoristas.

La selección de minoristas y de centros de distribución por los cuales serán visitados las hormigas se basa aplicando la regla se pseudo-aleatoridad proporcional. Esta elección es influenciada mediante el uso de rastros de feromonas en los arcos

previamente utilizados por la misma hormiga mediante un parámetro τ_{ij}, el cuál vuelve más atractiva la selección de arcos o menos atractivos si es que éstos no son convenientes y así poder construir diferentes soluciones. Esta feromona o parámetro depende en cierta parte del valor de la función objetivo del líder, la cual solo se obtiene resolviendo el nivel inferior. Para más detalle de la selección de este parámetro véase Calvete et al. (2011).

Una vez creadas las M soluciones los rastros de feromonas se actualizan de acuerdo a la calidad de las soluciones y para cada iteración se prevalece el mejor rastro de feromona de acuerdo a la calidad del valor de la función objetivo. En ese trabajo se realizan aproximadamente un máximo de 4000 iteraciones.

Es interesante mencionar cómo en este trabajo se aplica un algoritmo en el nivel superior y a la vez se ve influenciado en la creación de nuevas soluciones mediante ciertos parámetros que dependen del nivel inferior. Es así como podemos ver el enfoque del equilibrio de Stackelberg en este trabajo.

A continuación describimos el trabajo propuesto por Legillon et al. (2011) el cual presenta un enfoque de solución diferente al utilizado comúnmente en los problemas de programación binivel. Para este artículo ellos diseñan un algoritmo coevolutivo, este algoritmo trabaja con dos poblaciones separadas, mejorando cada nivel de manera independiente e intercambiando información de forma periódica para mantener una visión sobre el conjunto de soluciones.

Los algoritmos coevolutivos son un subgrupo de metaheurísticas que extienden el esquema de los evolutivos. De manera general podemos verlos como procedimientos que asocian varios algoritmos evolutivos y aplicaciones de transformaciones, como la mutación y cruce de sus diferentes poblaciones. Oduguwa and Roy (2002) describen de manera más explícita cómo trabajan estos algoritmos para problemas binivel.

Los pasos de COBRA son los siguientes: Primero se crean una población inicial para el nivel superior y otra para el nivel inferior, después entran a un ciclo donde se

mejora cada población, posteriormente se guardan ambas poblaciones mejoradas y son seleccionados ciertos individuos los cuales pasan al proceso de coevolución entre ambos niveles, añadiendo algunos nuevos individuos del proceso de coevolución a la población original (no se seleccionan los de mejor calidad), lo anterior se realiza hasta completar un criterio de paro.

En este trabajo además de introducir el COBRA (*Coevolutionary Bi-level method using Repeated Algorihms*) proponen una nueva medida de desempeño la cual se basa en la proximidad del valor óptimo de las variables de nivel superior con el valor óptimo de las variables del nivel inferior. Así también añaden un objetivo al nivel inferior y resuelven mediante este algoritmo ese problema. Ellos aclaran que esta metaheurística funciona especialmente para problemas binivel de mayor complejidad (tamaños grandes) que podrían ser muy costosos computacionalmente para otros algoritmos.

Los resultados del algoritmo de COBRA no son comparados con el primer trabajo debido a que presentan enfoques diferentes de solución. Sin embargo en Legillon et al. (2011) resuelven el mismo problema con otro algoritmo denominado *repairing algorithm*, este último toma el enfoque clásico de solución para este tipo de problemas. Sus resultados son comparados por los 2 algoritmos y en ninguna instancia son capaces de mejorar el valor de la función objetivo del líder al algoritmo de colonias de hormigas, mostrando la importancia que tiene generar buenas soluciones iniciales debido a que es un problema de ruteo. Es por lo anterior que en nuestro algoritmo decidimos utilizar la metaheurística GRASP para generar buenas soluciones.

3.2 DESCRIPCIÓN DEL PROBLEMA

En la sección 1.2 pudimos apreciar un breve análisis del planteamiento del problema a resolver en este trabajo, especialmente se describió la motivación que orilló a modelar este problema con programación binivel. En esta sección detallaremos por qué se modela el nivel superior como un problema de ruteo con múltiples depósitos y como un problema de transporte el nivel inferior.

En el nivel superior existen ciertos requerimientos que deben satisfacerse por parte de la empresa distribuidora. Ésta se encarga de satisfacer las necesidades de los minoristas al suministrar artículos, contando con una cantidad de vehículos con flota homogénea, es decir todos ellos cuentan con una misma capacidad.

Los vehículos se encargan de hacer rutas para abastecer a los minoristas, estas rutas empiezan en los diferentes centros de distribución y terminan en el mismo centro de distribución donde comenzó la ruta. Es decir, se cuenta con un problema de rutear vehículos con múltiples depósitos o centros de distribución conocidos en la literatura como un MDVRP,(por sus siglas en inglés) los cuales están catalogados como NP-Hard.

Se conoce que cada ruta no debe exceder la capacidad de los vehículos, es decir la suma de las demandas de los minoristas en una ruta no debe ser mayor a la capacidad de éste, además tendrán un tiempo límite de manera similar a una jornada laboral incluyendo el tiempo que se toma un operador en descargar los artículos. Lo anterior tiene que realizarse con el objetivo de minimizar sus costos de distribución y adquisición.

Una vez que se conocen los requerimientos por parte de los centros de distribución las plantas tienen que producir estos artículos con el fin de minimizar sus costos de operación sujeto a la capacidad que existe en cada una de ellas. Esta parte es modelada como el nivel inferior y para su respuesta óptima toma en cuenta la decisión del nivel superior.

El nivel inferior puede considerarse como un problema usual de transporte en el que se involucra el nivel superior al satisfacer las demandas de los centros de distribución. Si alguno de los dos niveles no considera la decisión del otro podría perjudicar sus costos totales. Es así como un problema de producción-distribución puede ser visto con un enfoque diferente al tradicional.

A continuación describimos los supuestos del problema además de presentar la forma en que está modelando este problema.

3.3 MODELACIÓN DEL PROBLEMA

Los modelos matemáticos son representaciones de la realidad mediante expresiones matemáticas las cuales buscan tener la mayor aproximación de ésta. En particular este trabajo se enfoca en un modelo matemático dentro de la investigación de operaciones que busca encontrar decisiones óptimas para determinados fines dentro dos organizaciones.

En esta sección se presenta la modelación del problema planteado anteriormente, describiendo primero los supuestos del problema, es decir las consideraciones que se tomaron en cuenta para después definir los conjuntos, parámetros y variables. Posteriormente analizamos cada una de las expresiones del modelo.

3.3.1 SUPUESTOS DEL PROBLEMA

Para la formulación del modelo matemático se consideraron los siguientes supuestos:

- Las demandas de los minoristas son conocidas.
- Las rutas de los vehículos empiezan en un centro de distribución y terminan en ese mismo.

- Las demandas de los minoristas solo pueden ser satisfechas por un vehículo.
- No se tiene un número fijo de vehículos en cada centro de distribución.
- Se cuenta con una flota homogénea.
- Un vehículo no debe visitar a dos centros de distribución diferentes.
- Las plantas cuentan con una capacidad fija.
- El costo de ir de un origen i a un destino j es proporcional a su distancia.
- El costo de ir del origen i al destino j es igual a ir de j a i.

3.3.2 CONJUNTOS, PARÁMETROS Y VARIABLES

Los conjuntos para el problema son los siguientes:

E : Conjunto de arcos conectados entre centros de distribución y minoristas.

K : Conjunto de plantas (índice k).

L : Conjunto de centros de distribución (índice l).

R : Conjunto de minoristas (índice r).

S : Conjunto de minoristas servidos por el centro de distribución l.

R_l : Conjunto de minoristas servidos por el centro de distribución l.

R_s : Conjunto de minoristas servidos por el vehículo s.

S_l : Conjunto de vehículos utilizados por el centro de distribución l.

V : La unión de los conjuntos L, R.

Los respectivos parámetros son los que a continuación presentamos:

b_r : Demanda del minorista r.

A_k : Producción disponible en la planta k.

c_{ij}^{11} : Costo de enviar un artículo desde i a j siendo $(i,j) \in E$.

c_{kl}^{12} : Costo de adquirir un artículo más el costo de descargarlo desde k hacia l.

c_{kl}^{22} : Costo de operación de un artículo, que se produce en la planta k y es enviado al centro de distribución l.

t_{ij} : El tiempo de traslado de ir del origen i al destino j.

τ_r : El tiempo de descargar de un vehículo en el minorista r.

T_s : Tiempo límite del vehículo s.

Para nuestro problema se considera un grafo completo $G = (V, E)$ definido por el conjunto de nodos $V = L \cup R$, y el conjunto de arcos E. La variable de decisión en el nivel superior para el problema de múltiples depósitos para el ruteo de vehículos es la siguiente:

x_{ij}^{s} : $\begin{cases} 1, & \text{si un elemento del conjunto } s \text{ ocupa el arco } (i,j) \in E \\ 0, & \text{de otro modo} \end{cases}$

Por otra parte la variable de decisión en el nivel inferior es:

y_{kl} : La cantidad de artículos producidos por la planta k enviado al centro de distribución l.

Para nuestro problema se considera un grafo completo $G = (V, E)$ definido por el conjunto de nodos $V = L \cup R$, y el conjunto de arcos E

3.3.3 MODELO MATEMÁTICO

En esta sección se presenta el modelo matemático que describe el problema de producción y distribución de una cadena de suministro. Posteriormente describimos cada una de las expresiones.

$$\min \sum_{s\in S}\sum_{(i,j)\in E} c_{ij}^{11}\, x_{ij}^{s} \; + \sum_{k\in K}\sum_{l\in L} c_{kl}^{12} y_{kl} \tag{3.1}$$

$$s.t. \quad x_{ij}^{s} \in \{0,1\} \qquad (i,j) \in E, s \in S \tag{3.2}$$

$$\sum_{s\in S}\sum_{j\in R\cup L} x_{ij}^{s} = 1 \qquad i \in R \tag{3.3}$$

$$\sum_{s\in S}\sum_{i\in R\cup L} x_{ij}^{s} = 1 \qquad j \in R \tag{3.4}$$

$$\sum_{j\in R_l} x_{ij}^{s} \le 1 \qquad s \in S_l, i, l \in L \tag{3.5}$$

$$\sum_{i\in R_l} x_{ij}^{s} \le 1 \qquad s \in S_l, j, l \in L \tag{3.6}$$

$$\sum_{j\in L} x_{ij}^{s} = 0 \qquad s \in S_l, i, l \in L \tag{3.7}$$

$$\sum_{i\in L} x_{ij}^{s} = 0 \qquad s \in S_l, j, l \in L \tag{3.8}$$

$$\sum_{i\in W}\sum_{j\in W} x_{ij}^{s} \le |W| - 1 \qquad W \subseteq R, 2 \le |W| \le |R|, s \in S \tag{3.9}$$

$$\sum_{i\in R_s} b_i \sum_{j\in R_s} x_{ij}^{s} \; \le Q \qquad s \in S \tag{3.10}$$

$$\sum_{i\in R_s}\sum_{j\in R_s} (t_{ij} + \tau_i) x_{ij}^{s} \le T_s \qquad s \in S \tag{3.11}$$

$$y \in \text{Argmin} \sum_{k\in K}\sum_{l\in L} c_{kl}^{22} \tilde{y}_{kl} \tag{3.12}$$

$$s.t. \quad \sum_{l\in L} \tilde{y}_{kl} \le A_k \qquad k \in K \tag{3.13}$$

$$\sum_{k\in K} \tilde{y}_{kl} \; = \sum_{s\in S_l}\sum_{r\in R_s} b_r \qquad l \in L \tag{3.14}$$

$$\tilde{y}_{kl} \ge 0 \qquad k \in K, l \in L \tag{3.15}$$

En (3.1) se presenta la función objetivo del nivel superior, esto es, se busca minimizar el costo de transportar artículos de los centros de distribución a los minoristas más el costo de adquirir un artículo desde una planta hacia los centros de distribución y el costo de descargar un artículo. En (3.2) se describe la variable binaria del nivel superior. La ecuación (3.3) indica que del cliente sólo puede salir un vehículo mientras en (3.4) indica que sólo puede llegar al igual un vehículo, esto es, el minorista es visitado una sola vez. Después, en (3.5) y (3.6) se establece que los centros de distribución pueden o no ser utilizados, es decir pueden existir soluciones en las cuales no se ocupen vehículos de un centro de distribución. En la ecuación (3.7) indica que no puede salir de ningún centro de distribución un vehículo para conectar con otro centro de distribución, de la misma manera en (3.8) se indica que no puede llegar ningún vehículo procedente de un centro de distribución. Con las restricciones 3.7 y 3.8 se imposibilita la conexión entre centros de distribución. La restricción (3.9) es la clásica condición de no permitir *subtours* dentro de un vehículo, en este caso se considera la restricción para cada vehículo que pertenece a un centro de distribución. En (3.10) se indica que todos los vehículos tienen la misma capacidad y la suma de demandas de los minoristas que hay en una ruta no deben sobrepasar ésta. Por último la restricción (3.11) dentro del problema de múltiples depósitos de ruteo de vehículos se encuentra la restricción del tiempo para los vehículos, ésta indica que cada uno de los vehículos tiene una cantidad fija de tiempo que no deben exceder al realizar su rutas, dentro de los tiempos deberán considerar el tiempo de descarga además del tiempo de traslado de un origen a un destino. La restricción (3.12) es la que implica que este problema sea un modelo de programación binivel ya que un conjunto de variables deben ser la solución óptima de un problema de programación matemática, dicha restricción es la función objetivo del nivel inferior, donde se busca minimizar el costo total de la producción (y envío) de los artículos fabricados en cada planta y transportarlos a los centros de distribución. Por otra parte las restricciones de este problema dentro del nivel inferior son: la desigualdad (3.13) indica que cada planta tiene una capacidad de producción y esta no debe sobrepasarse al producir los artículos. Así mismo en la restricción (3.14) es donde se involucra el nivel inferior con el nivel superior, en dicha restricción se busca satisfacer las demandas de los

minoristas servidos por cada centro de distribución. Como ya se mencionó anteriormente, en (3.14) solo se conocerán las demandas de los centros de distribución hasta las rutas creadas por el líder. El lado derecho de esta ecuación se puede reescribir como:

$$\sum_{s\in S_l}\sum_{r\in R_s} b_r = \sum_{s\in S_l}\sum_{u=1}^{h-1} b_r x^s_{i^s_u i^s_{u+1}} \qquad l = i^s_0, i^s_1, i^s_2, \dots, i^s_{h-1_s}, l = i^s_{h_s} \quad l \in L \qquad (3.14)$$

E índica que para todos los vehículos que sean parte de una ruta debe sumarse sus demandas. Para finalizar la descripción de éste modelo se tiene la condición de no negatividad por parte de las variables del nivel inferior en la desigualdad (3.15), así mismo se asume que son continuas.

CAPÍTULO 4

DESCRIPCIÓN DE LA HEURISTICA

El algoritmo heurístico que presentamos a continuación se basa en la metaheurística Búsqueda Dispersa y el equilibrio de Stackelberg, el cual se obtiene en problemas con jerarquía como el que aquí se analiza. En el nivel superior se busca aplicar la búsqueda para las soluciones del líder, una vez encontrada una solución del líder se resuelve el nivel inferior de manera exacta mediante el optimizador CPLEX y es aquí donde el líder reacciona ante la respuesta del seguidor y así sucesivamente, logrando de esta manera un equilibrio de Stackelberg.

Primero mencionamos de manera general la búsqueda dispersa en la sección 4.1, para después describir cada uno de los pasos que seguimos en la metodología de éste algoritmo en las siguientes secciones.

Los pasos son los siguientes:

1.- Generador de soluciones diversas.

2.- Creación y actualización del conjunto de referencia.

3.- Generación de subconjuntos.

4.- Método de combinación.

5.- Método de mejora.

4.1 DESCRIPCIÓN DE LA METODOLOGÍA

Como primer paso el método se basa en generar un conjunto de soluciones diversas denominado P, para generar dicho conjunto decidimos emplear la técnica clúster-ruteo, detallamos lo anterior en la sección 4.2. Es importante aclarar que este método puede trabajar con soluciones infactibles y una vez aplicada la fase de mejora estas pueden obtener factibilidad. Ya creado este conjunto extraemos de él un subconjunto de menor cardinalidad con el cual procedemos a desarrollar combinaciones entre las soluciones. A continuación describimos brevemente éste subconjunto.

El conjunto de referencia está formado por b soluciones, ${}^{b}/_{2}$ de ellas son las de mejor calidad del conjunto P y las otras ${}^{b}/_{2}$ son las más diversas. La calidad en nuestro problema está dada por la función objetivo del líder, es decir, la suma de los costos de envío de los centros de distribución a los minoristas y de adquisición desde las plantas a los centros de distribución, por otra parte la diversidad la definimos mediante una métrica que describimos posteriormente en la sección 4.3 al igual que los pasos de la creación y actualización de éste subconjunto.

El método de combinación en nuestra búsqueda dispersa se enfoca en combinar los subconjuntos de cardinalidad 2 del conjunto de referencia que busquen mejorar la calidad de los elementos que lo conforman. Las soluciones nuevas encontradas por este método entran de manera dinámica al conjunto de referencia. Al combinar dichos pares de soluciones nosotros damos mayor peso a la solución de mejor calidad, además de introducir un elemento aleatorio a nuestro proceso de combinación. Antes de entrar al conjunto de referencia cada una de las nuevas soluciones entra a la fase de mejora, la cual explicaremos a continuación en qué consiste.

El procedimiento de mejora o fase de mejora consiste en una búsqueda local para mejorar cada una de las soluciones ya sea del conjunto P o de una solución combinada. La búsqueda local se basa en tres movimientos inter-rutas e intra-rutas, siendo la de más relevancia para nuestro problema las técnicas inter-rutas. Éstas son

de manera consecutiva, es decir para cada solución se aplicarán los tres métodos siguientes:

Remover los q peores minoristas de cada ruta de la solución e insertar los minoristas que se encuentran fuera de la solución, éste es un movimiento inter-rutas. El movimiento 2-Opt, se considera intra-rutas y por último el intercambio de clientes, debido a la naturaleza del problema éste movimiento puede ser clave ya que puede provocar en algunas ocasiones un cambio significativo en la respuesta del seguidor.

Posteriormente analizamos cada una de estas tres técnicas al igual que el tipo de mejora a utilizar en este trabajo en la sección 4.4. Una vez presentado cada uno de los pasos de la búsqueda dispersa especificamos cada uno de ellos. Para mayor apreciación ver imagen 4.1 (anexo 1) donde se muestra el pseudocódigo.

4.1.2 Representación de la solución

Para nuestro problema binivel de planeación de la producción-distribución tenemos como variable del nivel superior x, así como la variable del nivel inferior y. Siendo x representada en nuestra solución como un conjunto de vectores (arreglo), donde cada uno de ellos forma una ruta que inicia en un centro de distribución y va visitando cada uno de los minoristas. El orden que representan en nuestra solución indica que los primeros t vectores son los primeros vehículos de cada centro de distribución, posteriormente los segundos t vehículos y así sucesivamente hasta tener los vehículos necesarios.

A continuación mostramos un ejemplo de la representación de x, donde el cero y uno indican dos centros de distribución diferentes, los demás números indican los minoristas que son visitados.

Ejemplo 4.1 (a)

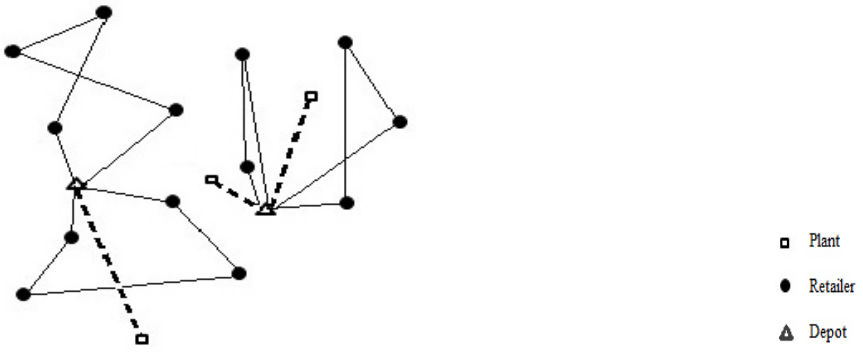
Vehículo 1	0	4	7	8	9	0		
Vehículo 1	1	2	18	20	21	14	13	1
Vehículo 2	0	19	3	10	16	6	0	
Vehículo 2	1	11	15	12	17	5	1	

Para la representación de nuestra variable *y* definimos un arreglo con número de filas igual a la cantidad de plantas, y número de columnas igual al número de centros de distribución. Cada posición en el arreglo representa la cantidad de envío de artículos de una planta a un centro de distribución. El siguiente arreglo muestra como se representa:

Ejemplo 4.1 (b)

	C. D. 0	*C. D. 1*
Planta 0	160	210
Planta 1	200	230

Imagen 4.2: Planificación Inicial



4.2 CONJUNTO DE SOLUCIONES DIVERSAS

El proceso de construcción de soluciones diversas lo dividimos en dos partes: pimero hacemos un clúster y después un ruteo, esto es para cada solución generada del conjunto P. Al realizar el clúster nosotros resolvemos un problema de asignación, en éste buscamos minimizar la suma de distancias (costos) de los minoristas a los centros de distribución, sujeto a que todos los clientes deben estar asignados únicamente a un centro de distribución, y en los centros de distribución no existe una capacidad, puesto que no tenemos un número fijo de vehículos.

Es evidente que si nosotros resolvemos este problema de asignación cada vez que deseamos encontrar una nueva solución para nuestro conjunto P siempre encontraremos la misma asignación, como los parámetros del problema son determinísticos la asignación realizada siempre será la misma.

Es aquí donde nosotros decidimos introducir una variante en la función objetivo del problema de asignación, esta consiste en añadir un ruido (ε) en cada distancia (costo) al ser sumada o restada en dicha función, es decir cada vez que vayamos a sumar la distancia de ir de un centro de distribución a un minorista d_{ij} generamos un número aleatorio entre 0 y 1 éste decidirá si añadimos un ε o si le disminuimos esa distancia el mismo ε.

Si él número generado es menor o igual que 0.5 la distancia prevalece de la siguiente forma $(d_{ij} + \varepsilon * d_{ij})x_{ij}$, en cambio sí es mayor que 0.5 cambia de la siguiente manera $(d_{ij} - \varepsilon * d_{ij})x_{ij}$; es así que nuestro modelo de asignación es:

Variables

$$x_{ij} = \begin{cases} 1, & \text{se asigna el minorista } j \text{ al centro de distribución } i \\ 0, & \text{no se asigna el minorista } j \text{ al centro de distribución} \end{cases}$$

Parámetros

b_j = Demanda del minorista j

Cap = Capacidad del centro de distribución (Infinita)

d_{ij} = Distancia del minorista j al centro de distribución i

ε = Valor entre 0 y 1

$$\min \sum_{ij} (d_{ij} \pm \varepsilon * d_{ij}) x_{ij}$$

s.a.

$$\sum_{i}^{l} x_{ij} = 1 \qquad j = 1,2,\dots,r$$

$$x_{ij} \epsilon \{0,1\}$$

De esta manera logramos evitar asignaciones deterministas, buscando siempre tener diferentes asignaciones, así buscamos de alguna forma no tener soluciones iguales en nuestro conjunto P. Este problema de asignación se resuelve mediante el optimizador CPLEX para cada solución generada.

Debido a la naturaleza de la Búsqueda Dispersa tenemos que cumplir una cantidad fija de soluciones, en este caso corresponde cardinalidad de P, es decir nosotros efectuamos nuestro clúster-ruteo mientras la cardinalidad de P sea menor o igual a la cantidad de soluciones totales.

Una de las características importantes en la Búsqueda Dispersa para la generación de P es la diversidad en las soluciones, es por ello que optamos por una adaptación de la heurística GRASP (Procedimiento de búsqueda voraz aleatorizado y adaptativo) a nuestro problema, en esta heurística solo utilizamos su fase de construcción y no la de búsqueda local.

A continuación describiremos la heurística GRASP para nuestro problema de ruteo con múltiples depósitos en el nivel superior.

Una vez asignados los minoristas a cada centro de distribución, para cada solución a generar hacemos lo siguiente: seleccionaremos un vehículo de cada un

depósito, este vehículo visitará un minorista a la vez, es decir hacemos un ruteo de forma paralela a los centros de distribución.

Es importante señalar que construimos las t rutas de forma paralela, esto es, una ruta en cada de centro de distribución, y sólo podremos empezar nuevas rutas hasta haber terminado las t rutas iniciales. Las rutas pueden terminar de alguna de las siguientes formas:

1.- Si la capacidad del vehículo se agotó: es decir no se puede introducir ningún cliente más a la ruta debido a que no cuenta con capacidad disponible para insertar algún minorista que todavía no ha sido visitado y su demanda es mayor a la capacidad que dispone en ese momento el vehículo.

2.- Si el tiempo de la ruta se terminó: de manera análoga a la capacidad de un vehículo, estos tienen un tiempo límite y no deben excederlo.

3.- Si no quedan clientes por visitar dentro de la lista de candidatos del centro de distribución al que pertenece el vehículo.

Ahora para la parte de inserción, supongamos que consideramos a un determinado vehículo, el procedimiento para insertar un minorista a la ruta será el siguiente: habiendo definido previamente los minoristas de cada centro de distribución, para cada uno de ellos evaluamos su costo, éste está definido por una función la cual conocemos como función adaptativa, ésta se define de la siguiente forma: $c(x_j) = d_{ij} + d_{jk} - d_{ki}$: e indica el costo de insertar el minorista j en una ruta, proveniente del minorista i, es igual a la distancia de ir de i a j más la distancia de ir de j a k, siendo k el centro de distribución en donde inicio.

Después de evaluar todos los candidatos en esta función encontramos el c^{max} y c_{min}, es decir el costo máximo y el costo mínimo de todos los candidatos que no han sido insertados, éstos nos servirán para poder construir el conjunto LRC (Lista restringida de candidatos).

Este conjunto está formado por todos aquellos minoristas a insertar x_j en la ruta, tales que cumplan la siguiente condición $c(x_j) \leq c_{min} + \alpha \left(c^{max} - c_{min}\right)$, esto es, aquellos minoristas que al ser evaluados en la función adaptativa sean menores que la suma del costo mínimo más un α multiplicado por la diferencia del máximo y mínimo de dicha función.

El valor α es un parámetro de la heurística y él cual tomará valores entre cero y uno. Debido a que solo contamos con una pequeña cantidad de instancias el valor de α fue seleccionado específicamente para cada para cada una de ellas, esto se detalla en el capítulo5. Es importante notar que si α toma el valor de 0 el único candidato a insertar sería determinado de forma voraz, en cambio si α toma el valor de 1 los candidatos serían insertados completamente aleatorios.

Cabe aclarar que nuestro conjunto LRC está formado no solo por aquellos elementos que cumplieron la condición señalada anteriormente, sino también cumplieron con la restricción de capacidad y disponibilidad de tiempo de ese vehículo, satisfaciendo ambas restricciones.

Una vez que se ha creado el conjunto LRC seleccionaremos un elemento de forma aleatoria, el cual será el candidato insertado en nuestra ruta. Este elemento que fue insertado se elimina de la lista de minoristas de ese centro de distribución.

Si el conjunto LRC es vacío para alguna ruta, entonces la ruta se cierra, es decir ya no existe ningún candidato para ser insertado y si es que aún existen candidatos, estos candidatos son visitados por algún otro vehículo del mismo centro de distribución.

Este procedimiento se realizará hasta haber visitado todos los minoritas de cada centro de distribución, es decir hasta que la cardinalidad de las listas de los minoristas de cada centro de distribución es vacía.

En la imagen 4.2 del anexo 1 muestra el pseudocódigo del conjunto de soluciones diversas.

4.3 CONJUNTO DE REFERENCIA

También conocido como $RefSet$, este conjunto está formado por b soluciones de las cuales $^{b}/_{2}$ soluciones son las de mejor calidad y las $^{b}/_{2}$ restantes serán las más diversas respecto a las demás soluciones que se encuentran en el conjunto P, como no existe una métrica general para todas las soluciones tenemos que definir una métrica para nuestro problema en particular.

Primero describimos la creación de dicho conjunto: Las primeras $^{b}/_{2}$ soluciones son formadas en base a la calidad, éstas son comparadas con las soluciones del conjunto P. La calidad se basa en la función objetivo del nivel superior, en nuestro caso son aquellas $^{b}/_{2}$ soluciones que se encuentren con el mínimo valor en dicha función. Para la creación de este conjunto recordemos que a cada solución generada por el líder, buscaremos la respuesta óptima por parte del nivel inferior, una vez encontradas ambas decisiones podremos evaluar la función objetivo del nivel superior.

Como se mencionó anteriormente se debe definir o seleccionar una métrica para las siguientes $^{b}/_{2}$ soluciones, para poder ser ingresadas al conjunto $RefSet$. La métrica seleccionada se basa en medir la cantidad de arcos en común entre dos soluciones. Así después podemos insertar en el conjunto aquellas que posean la menor cantidad de arcos iguales comparadas con las soluciones restantes en P.

A continuación se describe la métrica: sea x_1 una solución dentro de P y sea x_2 otra solución diferente de x_1 entonces medimos la distancia de estas dos soluciones de la siguiente forma: $d(x_1, x_2) = 1 - \frac{2e_c}{e_1+e_2}$

donde e_c es el número de arcos en común de estas dos soluciones y e_1, e_2 son la cantidad de arcos totales de cada una de ellas. Ésta métrica se puede ver en Fung et al. (2009).

Después obtenemos la métrica para todos los pares de soluciones y se llena una matriz de tamaño $(Psize - {}^{b}/_{2}) \times (Psize - {}^{b}/_{2})$, enseguida encontramos la máxima de

las mínimas distancias comparando los valores que fueron introducidos a la matriz, así hasta obtener un total de $^{b}/_{2}$ soluciones más diversas.

La actualización de este conjunto depende de la calidad de las nuevas soluciones generadas por el método de combinación. Si una de las nuevas soluciones formada por este método es factible y tiene un mejor valor con respecto a la función objetivo del líder que alguna de las soluciones ya existentes, entonces esta pasará a ser nueva integrante de $RefSet$, desplazando a la última solución de este conjunto, es decir a la de peor calidad, ordenando las soluciones en base a calidad. Esta actualización es conocida como dinámica.

4.4 MÉTODO DE MEJORA

El método de mejora consta de tres búsquedas locales las cuales se clasifican como *inter-rutas* e *intra-rutas*. Entre los movimientos *inter-rutas* que proponemos para nuestro algoritmo se encuentran *remove-insertion* e *interchange siendo* estos *inter-rutas* y el movimiento clásico llamado *2-opt* como *intra-ruta*.

Estos movimientos se implementan para cada solución del conjunto P antes de entrar a *RefSet*, y para cada solución que es generada por el método de combinación, estos tienen la siguiente secuencia:

1.- *Remove-insertion*
2.- *interchange*
3.- *2-opt*.

A continuación describimos a detalle en qué consisten las tres búsquedas locales en las siguientes secciones.

Esta mejora se basa en la heurística *regret*, dada una solución de cada ruta removemos los q minoristas que aporten los mayores costos por los arcos de sus respectivas rutas. Para cada minorista a remover seleccionaremos tres (valor encontrado por pruebas piloto) de los más costosos candidatos en cada la ruta y

alguno de ellos será seleccionado de manera aleatoria para ser removido. Cabe señalar que solo removeremos q minoritas si la ruta cuenta con más de esos q.

Sea el minorista i a remover y sus respectivos j, k antecesor y sucesor en la ruta correspondiente, entonces el costo de la diferencia asociada a la función objetivo original para remover dicho minorista es: $-c_{ij} - c_{jk} + c_{ik}$; esto es sin tomar en cuenta la respuesta del seguidor. Una vez encontradas todas las diferencias de cada uno de los posibles clientes a remover en cada ruta, seleccionaremos una al azar de las tres más altas diferencias (proporcionado por las pruebas piloto) y la dividiremos entre el total de la suma de todas las diferencias, y si este cociente es mayor que 0.5 será removido dicho minorista, si no pasaremos a otro posible candidato.

A continuación se muestra la técnica *remove*:

Antes de remover:

Vehículo 1 | 0 | 4 | 7 | 8 | 9 | 0 |

Vehículo 1 | 1 | 2 | 18 | 20 | 21 | 14 | 13 | 1 |

Vehículo 2 | 0 | 19 | 3 | 10 | 16 | 6 | 0 |

Vehículo 2 | 1 | 11 | 15 | 12 | 17 | 5 | 1 |

Después de remover:

Vehículo 1 | 0 | 4 | 7 | 8 | 0 | *Minorista Removido* | 9 |

Vehículo 1 | 1 | 2 | 18 | 21 | 14 | 13 | 1 | *Minorista Removido* | 20 |

Vehículo 2 | 0 | 19 | 3 | 10 | 6 | 0 | *Minorista Removido* | 16 |

Vehículo 2 | 1 | 11 | 12 | 17 | 5 | 1 | *Minorista Removido* | 15 |

Por otra parte después del movimiento *remove* pasemos a la descripción de *insertion*. La heurística *insertion* que proponemos enlista como primer paso los candidatos que no se encuentran en la solución, es decir esta heurística solo se efectúa para insertar aquellos minoristas que no se encuentren dentro una solución.

Para cada elemento de dicha lista se busca la posición en la cual se va a insertar con menor costo considerando sólo el ruteo, si el elemento en la posición al ser insertado produce alguna infactibilidad éste pasa a una segunda lista en la cual se encuentran todos aquellos minoristas los cuales en su mejor posición a insertar no son factibles y se eliminará de la primer lista.

Este procedimiento se realiza hasta no haber elementos en la primer lista, si existen elementos en la segunda lista se procede de la misma manera, buscando la posición de menor costo para cada uno de ellos a excepción de lo siguiente, si algún elemento produce infactibilidad por parte de la segunda lista este elemento pasará a forma parte de la primer lista, este intercambio de elementos entre estas dos listas sólo se efectúa máximo tres veces siempre y cuando haya elementos en alguna de estas listas, si no los hay el ciclo habrá terminado. La solución nueva reemplaza a la solución que ingresa al método si es factible y tiene un valor en la función objetivo menor que la ya existente.

Antes de insertar

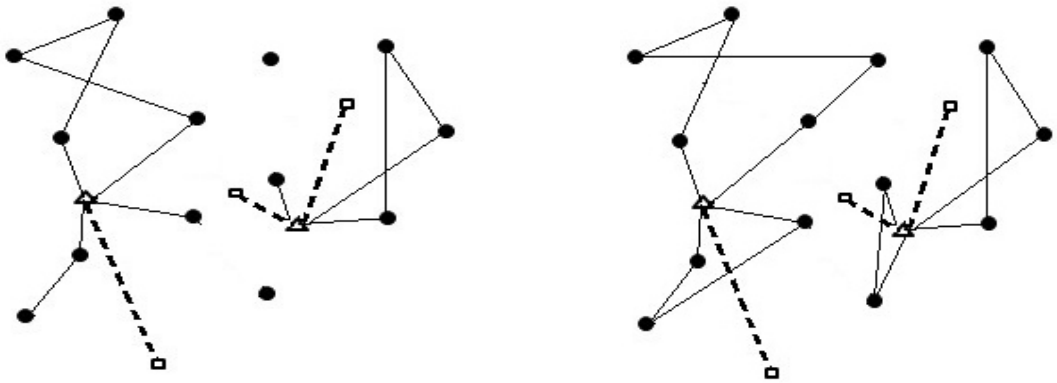
Vehículo 1	0	4	7	8	0		
Vehículo 1	1	2	18	21	14	13	1
Vehículo 2	0	19	3	10	6	0	
Vehículo 2	1	11	12	17	5	1	

Después de insertar

Vehículo 1	0	4	7	8	0				
Vehículo 1	1	2	15	18	21	14	13	1	*Minoristas insertados* 15
Vehículo 2	0	19	3	10	6	9	16	0	*Minoristas insertados* 9 16
Vehículo 2	1	11	12	20	17	5	1		*Minoristas insertados* 20

Imagen 4.3: Primer Mejora

Remove-Insertion



Después de *Remove* **Después de *Insertion***

La segunda búsqueda local es conocida como *interchange*, ésta se realiza entre minoristas de diferentes rutas. Sabemos que es muy probable que la respuesta del seguidor produzca un cambio en la decisión del líder acerca de las rutas que él emplea como solución al nivel superior. Es por ello que el intercambio de minoristas entre las rutas que son de diferente centro de distribución será una estrategia de suma importancia para encontrar el equilibrio entre el líder y seguidor.

Ésta mejora se realizará de dos formas:

1.- El intercambio de minoristas entre rutas del mismo centro de distribución: este intercambio se realiza si y sólo si al intercambiar elementos entre rutas la solución nueva mejora en la función objetivo, es decir la nueva solución minimiza los costos de distribución de ruteo de la siguiente forma:

Sea el minorista k en la posición i y el minorista l en la posición j los posibles candidatos al intercambio, entonces si

$$c_{i-1,j} + c_{j,i+1} + c_{j-1,i} + c_{i,j+1} < c_{i-1,i} + c_{i,i+1} + c_{j-1,j} + c_{j,j+1}$$

El minorista k pasa a la posición j y el minorista l a la posición i.

Es evidente que no es necesario considerar los costos de transportar los artículos de las plantas a los centros de distribución, debido a que no altera las demandas en los centros de distribución.

2.- El intercambio de minoristas entre vehículos pertenecientes a diferentes centros de distribución: no es descabellado pensar que este intercambio puede provocar una respuesta en el seguidor que afecte a la función objetivo del nivel superior. Es así que para cada intercambio de este tipo resolvemos el nivel inferior, después de encontrar la respuesta del seguidor evaluamos la función objetivo del líder y si ésta disminuye comparada con la solución anterior efectuamos el intercambio de clientes si y sólo si es factible a las restricciones, si no continuamos con la misma solución sin haber hecho el intercambio.

Para cada uno de estos movimientos fijamos un minorista, supongamos que se encuentra en la posición i este sólo es probado con los k minoristas más cercanos respecto a su costo de ruteo. Una vez probados estos k minoristas seguimos con el minorista que se encuentre en la posición $i + 1$, y así sucesivamente hasta terminar con todos los minoristas.

Para estas estrategias existen diferentes tipos de mejorar como las siguientes:

- *La primera mejora*: consiste en encontrar la disminución de la función objetivo por primera vez en un intercambio de minoristas ya sea del primer o segundo intercambio y únicamente se hace ese primer intercambio que logre la primer reducción en la función objetivo del nivel superior.

- *La mejor de todas las mejoras*: en este movimiento se prueban todas las posiciones de los minoristas que se encuentren en la ruta. Esto es, para cada minorista que se encuentre en la posición i será probado con los k minoristas más cercanos a este, así hasta termina con todas las posiciones en las que se encuentra un minorista en la ruta.

En este trabajo solo emplearemos la técnica de la mejor de todas las mejoras.

Como vemos el intercambio de minoristas solo corresponderá a las posiciones en la ruta, no considerando que tenemos una nueva solución si se llega a dar el intercambio y éste produzca un retroceso en el intercambio y empecemos desde cero cada vez que haya una nueva solución. Si hiciéramos lo anterior los tiempos computacionales aumentarían considerablemente además de que tendríamos problemas con la memoria de la máquina.

Antes de intercambio de clientes (Líder x)

Vehículo 1	0	4	7	8	0			
Vehículo 1	1	2	15	18	21	14	13	1
Vehículo 2	0	19	3	10	6	9	16	0
Vehículo 2	1	11	12	20	17	5	1	

Antes de intercambio de clientes (Seguidor y)

	C. D. 0	*C. D. 1*
Planta 0	160	210
Planta 1	200	230

Después del intercambio de clientes (Líder x)

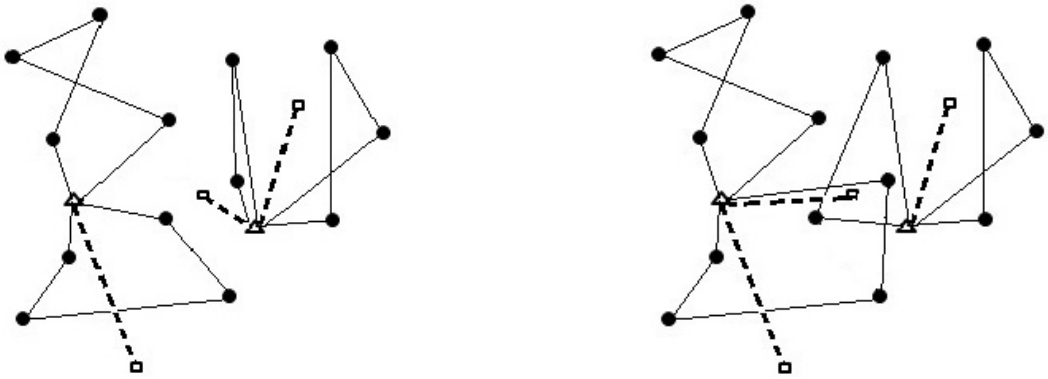
Vehículo 1	0	2	7	8	0				*Intercambio*	4 ⇔ 2
Vehículo 1	1	4	15	18	17	14	13	1	*Intercambio*	21 ⇔ 17
Vehículo 2	0	19	12	10	6	9	16	0	*Intercambio*	3 ⇔ 12
Vehículo 2	1	11	3	20	21	5	1			

Después del intercambio de clientes (Seguidor y)

	C. D. 0	*C. D. 1*
Planta 0	180	240
Planta 1	180	200

Imagen 4.4: Segunda Mejora

Interchange



Antes de *Interchange* **Después de *Interchange***

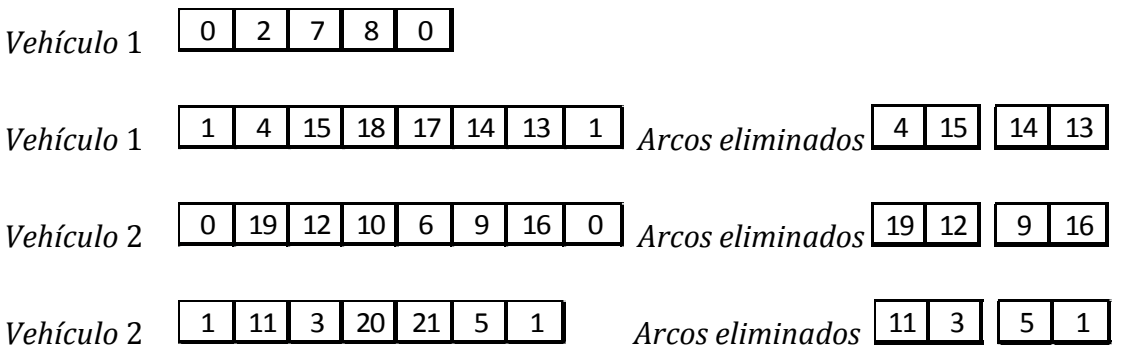
Por último describiremos la estrategia 2-*opt*. Esta estrategia solo se utilizará para mejorar el costo de distribución de cada ruta en una solución. Dada una ruta en

una solución procedemos de la siguiente manera: eliminamos dos arcos no adyacentes, para posteriormente conectar los 4 elementos de la ruta que quedaron desconectados de una forma distinta, obteniendo así una nueva ruta.

Este movimiento al igual que los anteriores se efectuará si y sólo si mejora la función objetivo del nivel superior y si es factible dicha solución, de lo contrario permanecerá la solución anterior. Al igual que en la mejora anterior sólo empleamos la *mejor de todas las mejoras*.

La *mejor de todas las mejoras* consiste en examinar todas las combinaciones que podemos hallar en una ruta al eliminar dos arcos de dicha ruta. En nuestro caso tenemos más de una ruta en cualquier solución, entonces hallamos para cada ruta todas las posibles combinaciones y nos quedamos con la mejor solución de cada ruta, es decir la ruta de cada vehículo con menor valor de la función objetivo del nivel superior. La siguiente figura muestra el movimiento:

Antes de 2-*Opt (Líder x)*



Después de 2-Opt

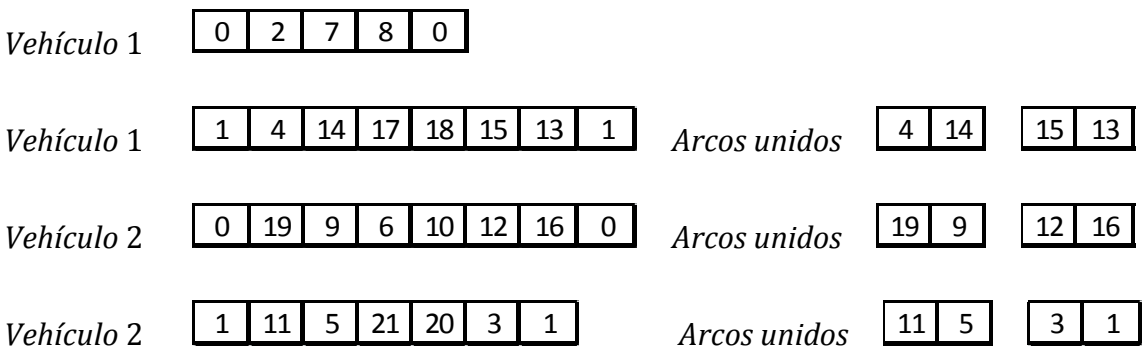
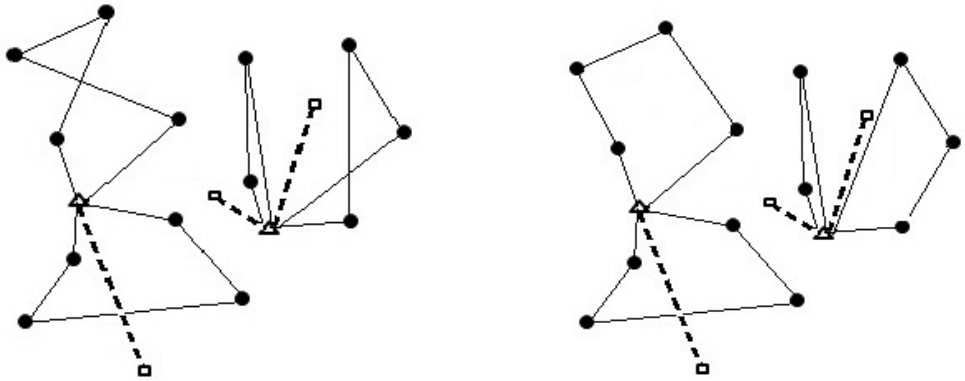


Imagen 4.5: Tercer Mejora

2-Opt



Antes de 2-Opt **Después de 2-Opt**

4.5 MÉTODO DE GENERACIÓN DE SUBCONJUNTOS

Para la creación de nuestra heurística hemos decidido generar subconjuntos de cardinalidad 2, es decir, dado el conjunto de referencia de tamaño *b* encontraremos todas las combinaciones de dos elementos de este conjunto. Generando para cada par una nueva solución, si esta solución es mejor en la función objetivo del nivel superior desplazará a la de peor valor de *RefSet*. Así el método considera todas las parejas que se pueden formar con los elementos del conjunto de referencia de manera exhaustiva y a todas ellas les aplica el método de combinación.

4.6 MÉTODO DE COMBINACIÓN

Para el método de combinación utilizamos una de las estrategias más usuales por parte de la búsqueda dispersa, la cual consiste en dar mayor peso a la solución que tiene mejor calidad formadas por el método de generación de subconjuntos.

Como primer paso, dadas dos soluciones reordenamos las rutas de la segunda solución en base a la primera. El reordenamiento consiste en lo siguiente para cada vehículo i de la primer solución encontramos un vehículo j en la segunda solución el cual coincida con la mayor cantidad de minoristas que tenga el vehículo i en la primer solución, no importa el orden. Así ahora el vehículo j de la segunda solución pasará a ser el vehículo i en la segunda solución. Esto lo hacemos para cada conjunto de vehículos de cada centro de distribución.

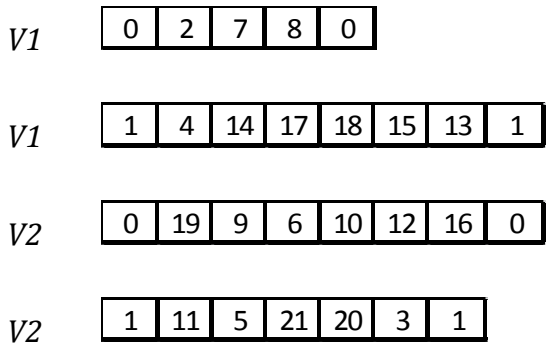
Después de reordenar el subconjunto describimos la combinación. Introducimos un elemento aleatorio en este procedimiento, es decir, supongamos que tenemos el minorista 1 de la solución uno (solución de mayor calidad) y de la solución 2 (solución de menor calidad) ambos pertenecientes al vehículo de la ruta 1, entonces generamos un número aleatorio ($rand$) entre cero y uno ($0 < rand < 1$), si $rand < Prob$ ($Prob = Probabilidad$) el minorista de la solución uno pasará a formar parte de la nueva solución, de lo contrario el minorista de la segunda solución es el que formará parte de esta solución. Esto es, para cada posición de la nueva solución competirán por esa posición cada uno de los candidatos respectivos a esa posición los candidatos de la solución uno y dos. El valor de $Prob$ tiene que ser mayor que 0.5 para que los minoristas de la solución de mejor calidad tienden a prevalecer en la solución combinada.

Ahora veremos cuándo algún minorista no podrá entrar a la nueva solución. Si el minorista que fue seleccionado por el número aleatorio provoca alguna infactibilidad, éste no es introducido, pero continuamos con los minoristas restantes de esa ruta, así hasta terminar con todos los minoristas de ella y posteriormente con los de cada ruta. Si algún minorista que ya entró a una ruta de la solución combinada es de nuevo seleccionado por el número aleatorio, no podrá volver a entrar debido a que produce infactibilidad, de esta manera continuamos con los siguiente candidatos.

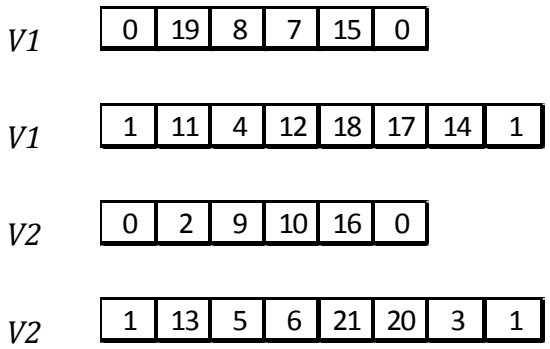
Como ya lo mencionamos anteriormente una vez generada la solución combinada ésta pasa al método de mejora y ésta a su vez si es mejor en cuanto a la calidad que alguna de las soluciones del conjunto de referencia es insertada a dicho

conjunto, la posición que ocupa depende del valor de la función objetivo, para ello reordenamos el conjunto de referencia con la nueva solución insertada, por consecuencia saldrá la solución de peor calidad.

Solución 1



Solución 2



Solución Combinada

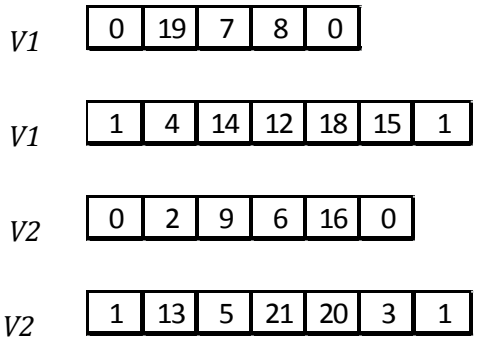
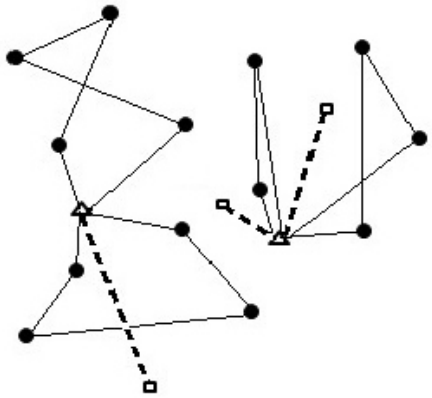
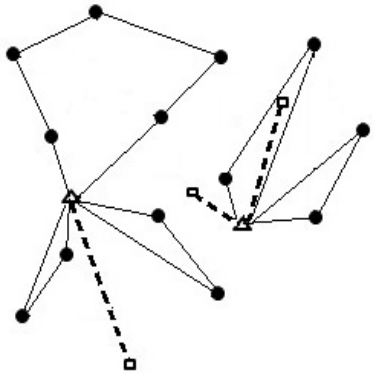


Imagen 4.6: Segunda Mejora

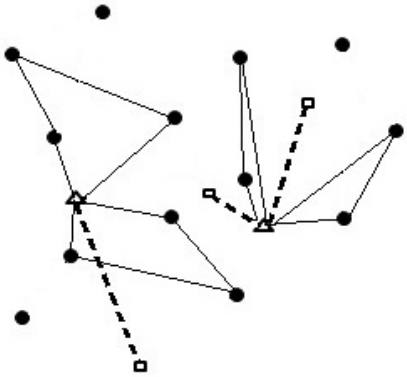
Método de Combinación



Solución 1



Solución 2



Solución Combinada

CAPÍTULO 5

EXPERIMENTACIÓN COMPUTACIONAL Y CONCLUSIONES

Debido a la naturaleza de los algoritmos heurísticos en los cuales no se garantiza optimalidad, es necesario medir el desempeño de estas metodologías para establecer la calidad de las soluciones. Para conocer su funcionamiento decidimos una evaluación de este algoritmo con experimentos computacionales utilizando instancias obtenidas de la literatura. Las condiciones de la implementación se proporcionan más adelante así como una comparación con el primer trabajo.

Este capítulo se estructura de la siguiente manera: en la sección 5.1 se muestra el ambiente computacional donde fue implementado el algoritmo. Por otra parte en la sección 5.2 se describe detalladamente cada una de las instancias que fueron tomadas del trabajo de la literatura. Mientras en la sección 5.3 se muestran los resultados computacionales así como su comparación y la afinación de parámetros para la metodología heurística. Por último se muestra las conclusiones y una posible extensión de este trabajo.

5.1 AMBIENTE COMPUTACIONAL

Un conjunto de condiciones experimentales pueden afectar el comportamiento de las implementaciones de las diferentes metodologías. Uno de los objetivos de este trabajo es el de proporcionar una metodología alternativa para la solución del problema planteado en el Capítulo 2, por lo cual debemos tener condiciones análogas al trabajo de Calvete et al. (2011) y poder hacer una comparación de los resultados. En

esta sección describimos el ambiente computacional en el cual fue implementado nuestro algoritmo heurístico.

La experimentación computacional fue desarrollada en un equipo personal con las siguientes características: Un Procesador (R) Dual-Core con una velocidad de 3.00GHz y con capacidad de memoria RAM de 2GB. Así también los códigos fuentes fueron implementados en Microsoft Visual Studio 2010 en el lenguaje de programación C++. Por otra parte se hizo uso del optimizador CPLEX versión 8.0 para resolver el nivel inferior de manera exacta.

5.2 DESCRIPCIÓN DE LAS INSTANCIAS

Para realizar la experimentación computacional de la metodología propuesta se utilizaron un conjunto de instancias las cuales se describen a continuación. La parte del nivel superior involucra datos de un MDVRP, estos fueron obtenidos del banco de instancias de Cordeau et al. (1997). Este conjunto de datos contiene las coordenadas de los centros de distribución, minoristas, la cantidad de vehículos, el tiempo límite de la duración de las rutas, el tiempo de descarga en cada minorista, la demanda de los minoristas y la capacidad de los vehículos. Se asume que los vehículos forman una flota homogénea.

Mientras que el conjunto de datos que son ocupados en el nivel inferior son la localización y capacidad de producción de las plantas. La localización de plantas es de forma aleatoria en el intervalo de [-200, 200] x [-200, 200] mientras que la capacidad de las plantas es obtenida con la razón de [*Total de la demanda/Cantidad de plantas*, *Total de la demanda*] de acuerdo a una distribución uniforme. Por otra parte los costos c_{kl}^{12} se asume que no depende de la planta y es generado por una distribución uniforme entre [0.5, 1] y c_{kl}^{22} es obtenido con el producto de dos términos, el primero es un aleatorio entre [2, 5] y el segundo es proporcional a la distancia entre plantas y centro de distribución.

La siguiente tabla muestra el tamaño de los conjuntos en las diferentes instancias utilizadas.

Instancia	$\|K\|$	$\|L\|$	$\|R\|$	T_s	Q
1	4	4	48	500	200
2	4	4	96	480	195
3	4	4	144	460	190
4	4	4	192	440	185
5	4	4	240	420	180
6	4	4	288	400	175
7	6	6	72	500	200
8	6	6	144	475	190
9	6	6	216	450	180
10	6	6	288	425	170

Tabla 5.2.1 Descripción de conjuntos de MDVRP con plantas

La Tabla 5.2.1 muestra las cardinalidades de los conjuntos de plantas, centros de distribución, plantas, tiempo límite de vehículos y capacidad de vehículos.

5.3 RESULTADOS COMPUTACIONALES

Los resultados de las pruebas computacionales son presentados en esta sección, previamente a estos resultados se hizo una afinación de los parámetros del algoritmo. A continuación describimos la influencia de los siguientes parámetros:

$|P|$: El tamaño de la población del conjunto generador de soluciones diversas.

α : Este parámetro es utilizado en la heurística GRASP para la función adaptativa y su valor es entre cero y uno.

q^1 : La cantidad de minoristas removidos en la generación de soluciones diversas.

q^2 : La cantidad de minoristas removidos en la Búsqueda Dispersa (después del método de combinación).

ρ : La probabilidad de sesgo del volado en el método de combinación.

Estos parámetros fueron encontrados mediante numerosas pruebas computacionales, para cada instancia fue encontrada la mejor combinación de éstos, debido a la insuficiente cantidad de instancias no fue posible calibrar estos parámetros en un software de calibración especial. En seguida presentamos la Tabla 5.3.1 donde se muestra la afinación de parámetros:

	Parámetros				
Instancia	P	α	q^1	q^2	ρ
1	30	0.25	2	2	0.7
2	30	0.20	2	2	0.7
3	30	0.15	2	2	0.7
4	30	0.25	2	2	0.7
5	30	0.15	3	1	0.8
6	30	0.15	3	2	0.7
7	100	0.15	3	1	0.8
8	100	0.15	3	1	0.7
9	100	0.15	3	2	0.7
10	100	0.15	3	2	0.7

Tabla 5.3.1. Descripción de los parámetros seleccionados para la experimentación.

Podemos observar como el parámetro P tiene los valores de 30 y 100, siendo el menor valor seleccionado para el conjunto de instancias 1-6. Por otra parte el valor $\alpha = 0.25$ predomina en la mayor cantidad de instancias, así como q^1 sobresalen los valores de 2 y 3. El valor de ρ seleccionado en la mayor parte de las instancias es 0.7.

Con los parámetros descritos anteriormente obtuvimos los resultados de la Tabla 2. En esta tabla son presentados también los resultados del algoritmo de colonia de hormigas. La columna $\overline{f}_1$ representa el promedio del valor de la función objetivo obtenido de 10 repeticiones efectuadas de cada instancia, mientras que la columna f_1^b representa el mejor valor de la función objetivo encontrado por nuestro algoritmo y

por último $\overline{T}$ representa el tiempo promedio requerido para cada la solución de cada instancia.

El parte final de la Tabla 2 se muestran los gaps calculados de la siguiente manera:

$$\text{Gap} = \left(\frac{\text{Valor Conocido} - \text{Valor encontrado}}{\text{Valor Conocido}}\right) \times 100\% \qquad (5.1)$$

Si el valor anterior es negativo indica que nuestro algoritmo es mejor que el valor existente en la literatura. A continuación mostramos la Tabla 2.

	Algoritmo basado en la Búsqueda Dispersa			**Algoritmo de Colonia de Hormigas**			**Gap(%)**	
Instancia	$\overline{f}_1$	f_1^b	$\overline{T}$	$\overline{f}_1$	f_1^b	$\overline{T}$	$\overline{f}_1$	f_1^b
1	1,408.36	1,395.64	475.3	1,475.39	1,439.97	250.8	**-4.54**	**-3.08**
2	2,173.25	2,110.91	682.2	2,282.11	2,236.25	441	**-4.76**	**-5.60**
3	3,204.08	3,121.02	1558.8	3,479.40	3,336.38	742.8	**-6.45**	**-7.91**
4	4,355.29	4,244.52	2066.9	4,836.10	4,750.10	1165.8	**-9.94**	**-10.64**
5	4,720.99	4,662.94	1899.1	5,371.51	5,178.45	1705	**-9.95**	**-12.11**
6	5,293.20	5,099.22	2961.8	6,147.06	6,037.67	2292.6	**-13.89**	**-15.54**
7	1,926.51	1,901.95	896.5	1,883.48	1,815.77	388.8	2.28	4.74
8	3,449.27	3,391.70	873.5	3,558.58	3,449.53	800.4	**-3.07**	**-1.67**
9	4,682.84	4,588.00	830.5	4,798.19	4,686.25	1454.4	**-2.40**	**-2.08**
10	7,308.60	7,175.59	621.3	7,459.10	7,338.37	2369.4	**-2.01**	**-2.21**

Tabla 5.3.2 Resultados numéricos de los experimentos computacionales

Ahora analicemos cada uno de los resultados de cada instancia de la Tabla 5.3.2. La metodología descrita en este trabajo mejoro los valores de las funciones objetivo de las instancias consideradas en Calvete et al. (2011). Los holguras mostradas en la columna GAP de los mejores valores de la función objetivo del líder se encuentran en un intervalo de 1.67% a 15.64%. Así también se puede apreciar que la instancia 6 no se obtuvo un mejor valor en función objetivo.

Podemos resaltar que nuestros resultados encontrados en el promedio del valor de la función objetivo del líder son muy cercanos al mejor valor de ésta, al igual que en el caso anterior fueron 9 de 10 instancias las mejoradas. Con esto queremos

decir que no fue coincidencia el obtener una mejoría de la misma calidad que el mejor valor del líder (2.01% a 13.89%) en la mayoría de las instancias, comparado con los resultados existentes en la literatura.

5.4 CONCLUSIONES

Después de haber realizado un estudio detallado sobre el problema de producción y distribución de una cadena de suministro y aportar una nueva metodología la cual fue implementada y realizar una experimentación análoga al primer trabajo es necesario hacer un análisis sobre el aporte de nuestro algoritmo heurístico y observar si se cumplieron los objetivos planteados en el Capítulo 1.

En este trabajo se desarrolló un algoritmo heurístico basado en la metodología de la Búsqueda Dispersa y considerando el equilibrio de Stackelberg para resolver el problema binivel de la producción y distribución de una cadena de suministro. No es difícil pensar en la dificultad de este problema binivel, debido al tipo de problema del nivel superior (MDVRP que es NP-Hard) se añade a éste el problema del nivel inferior (un problema de producción) por lo cual no es nada trivial encontrar una solución.

Una de las razones principales para el análisis de este problema y proponer el algoritmo que describimos en el Capítulo 4 fue la escasez de trabajos que abordan estos problemas los cuales integran más de un proceso de la cadena de suministro modelado con Programación Binivel.

La principal aportación de nuestro algoritmo es que proporciona un enfoque único para el método de combinación considerando un cierto grado de aleatoriedad que no se ha encontrado en la literatura relacionada con los problemas de ruteo. Además se propone una nueva adaptación de la heurística GRASP para el método generador de soluciones diversas y la integración de los tres métodos de mejora.

La experimentación computacional se llevó a cabo con algunos casos de referencia y nuestro algoritmo propuesto demostró ser eficiente mediante la mejora

de la calidad de las soluciones que se encuentran en otros trabajos. En la mayoría de los casos probados, nuestro algoritmo ha mejorado de manera significativa los valores de la función objetivo reportados en la literatura, lo cual muestra que puede ser una alternativa conveniente para resolver este problema.

Por último, teniendo en cuenta que este modelo integra dos procesos de planificación (producción y distribución) se puede suponer que en la vida real, las decisiones relacionadas con los dos procesos no se hacen muy a menudo durante el día. Por lo tanto, el tiempo necesario para la resolución de este importante problema pierde un poco de importancia y con esto nos podemos centrar en los costos de la función objetivo, por eso podemos concluir que nuestro algoritmo es competitivo.

5.5 TRABAJO A FUTURO

En esta sección se proponen trabajos que pueden ser líneas de investigación como extensiones de este trabajo. Debido a los resultados obtenidos con nuestra metodología propuesta se propone la implementación de diferentes métodos como Búsqueda por Entornos Variables, Búsqueda Tabú o Algoritmos Genéticos que busquen una mejor eficiencia en los tiempos obtenidos por nuestro algoritmo heurístico e inclusive una mejor calidad en la función objetivo.

Otra posible extensión a este trabajo es agregar un objetivo a nuestro problema en el nivel inferior, como aparece en Legillon et al. (2011) y adaptar nuestra metodología bajo este nuevo supuesto. De acuerdo a la cadena de suministro podemos ir encontrando diferentes variantes al problema de MDVRP del nivel superior. Una de estas variantes es el problema de entregas divididas con diferentes vehículos (por sus siglas, SDVRP), este problema tiende hacer muy desafiante para futuras investigaciones.

REFERENCIAS

- Barborosoglu G., Özgür D., Hierarchical design of an integrated production and 2-echelon distribution system, *European Journal of Operational Research*, 118,464-484, 1999.
- Bard J.F., Optimality conditions for the bilevel programming problem. *Naval Research Logistic Quartely*, 31, 13-26, 1984.
- Bard J.F., Practical Bilevel Optimization, Algorithms and Applications, *Kluwer Academic Publishers*, Dordrecht, 1998.
- Bard J.F., Some properties of the bilevel linear programming, *Journal Optimization Thoery and Application* 68, 371-378, 1991.
- Beamon B.M., Supply chain design and analysis: Models and methods, *Int. J. Production Economics*, 55, 281-294, 1998.
- Ben-Ayed O., Blair O., Computationl difficulty of bilevel linear programming, *Operation Reseach,* 38, 556-560, 1990.
- Bialas W. F., Karwan M.H., Two Level Linear Programming, *Management Science*, 30, 1004-1021, 1982.
- Bianco L., Caramia M., Giordani S., A bilevel flow model for hazmat transportation network design, *Transportation Research Part C*, 175-196, 2009.
- Boudia, M., Louly, M.A.O., Prins, C., A reactive GRASP and path relinking for a combined production-distribution problem, *Computers & Operations Research*, 34:11, 3402-3419, 2007.
- Boudia, M., Prins, C., A memetic algorithm with dynamic population management for an integrated production-distribution problem, *European Journal of Operational Research*, 195:3, 703-715, 2009.
- Braken J., McGill J.M., Mathematical programs with optimization problems in the constraints, *Operations Research*, 21, 37-44, 1973.
- Calvete H. I., Galé C., The bilevel linear/linear fractional programming problem, *European Journal of Operational Research*, 114:1, 188-197, 1999.

- Calvete H.I., Gale C., Mateo P.M., A new approach for solving linear bilevel problems using genetic algorithms, *European Journal of Operational Research,* 188, 14–28, 2008.
- Calvete H.I., Gale C., Oliveros M.J., Bilevel model for production-distribution planning solved by using colony optimization, *Computers & Operation Research*, 38, 320-327, 2011.
- Camacho-Vallejo J.F., Cordero-Franco A.E., González-Ramírez R.G., Solving the bilevel facility location problem under preferences by a Stackelberg-Evolutionary algorithm, *Mathematical Problems in Engineering*, in Press 2013.
- Camacho-Vallejo, J.F., Muñoz-Sánchez, R., A path based algorithm for solve the hazardous materials transportation bilevel problem, *Applied Mechanics and Materials*, Vols. 253-255 ,1082-1088, 2013.
- Candler W., Norton R., "Multilevel programming", Technical Report 20, *World Bank Development Research Center*, 1977.
- Carpaneto G., Dell'amico M., Fischetti M., Toht P., A branch and bound algorithm for the multiple depot vehicle scheduling problem*, Networks*, 19, 531-548, 1989.
- Chan F.T.S., Chung S.H., Wadhwa S., A hybrid genetic algorithm for production and distribution, *Omega*, 33,345–55, 2005.
- Chen Z.L., Integrated Production and Distribution Operations: Taxonomy, Models and Review, Handbook of Quantitative Supply Chain Analysis: Modeling in the E-Business Era, *Kluwer Academic Publishers*, 2004.
- Choong L., Rosmanira W., Omar K., Zirouri M., Vehicle Routing Problem: models and solutions*, Journal of Quality Measurement and Analysis*, 4(1), 205-218, 2008.
- Clarke G., Wright J. W., Schedulling of vehicles from a depot to a number of delivery points, *Operation Research*, 12, 568-581, 1964.
- Clarke P., Westerberg A., A note on the optimality conditions for the bilevel programming problem, *Naval Research Logistic*, 35,413-418, 1988.
- Cohen M.A., Lee H.L., Strategic analysis of integrated production-distribution systems: models and methods, *Operations Research Society of America*, 36,216-228, 1988.

- Colson B., Marcotte P., Savard G., An overview of bilevel optimization, *Annal of Operational Research*, 153, 235-256, 2007.
- Cordeau J.F., Gendreau M., Laporte G., A tabu search heuristic for periodic and multi-depot vehicle routing problem, *Networks,* 30, 105-119, 1997.
- Dantzig G., Fulkerson R., Johnson S. Solution of a large-scale travelling salesman problem, *Operations Research*, 2, 393-410, 1954.
- Dantzig G., Ramser J., The truck dispatching problem, *Management Science*, 6, 80-91, 1959.
- Dempe S., Foundations of Bilevel Programming, *Kluwer Academic Publisher*, 2002.
- Dhaenens-Flipo C., Finfe G., An integrated model for an industrial production-distribution problem, IIE Transactions, 33, 705-715, 2001.
- Dussault J.P., Marcotte P., Roch S., Savard G., A smoothing heuristic for a bilevel pricing problem*, European Journal of Operational Research*, 174, 1396-1413, 2006.
- Elhedhli S., Goffin J.L., Efficient Production-Distribution System Design, *Management Science*, 51:7, 1151-1164, 2005.
- Fahiminia B., Farahani R.Z., Marian R., Luon L., A review and critique on integrated production-distribution planning models and techniques, *Journal of Manufacturing Systems*, 32, 1-19, 2013.
- Feo T., Resende M.G.C., Greedy Randomized Adaptive Search Procedures, *Journal of Global Optimization*, 2, 1-27, 1995.
- Fung R. Y.K., Tang J., Zhang J., A Multi-Depot Vehicle Routing Problem with Weight-Related Costs*, IEEE*, 978-1-4244-4136, 2009.
- Gendreau M., Marcotte P., Savard G., A hybrid Tabu-ascent algorithm for the linear bilevel programming problem, *Journal of Global Optimization*, 8, 217–233, 1996.
- Glover F., Heuristics for Integer Programming Using Surrogate Constraints, *Decision Sciences,* 8, 156-166, 1977.
- Goetschalckx M., Vidal, C.J., Dogan, K., Modeling and design of global logistics systems: A review of integrated strategic and tactical models and design algorithms, *European Journal of Operational Research*, 143, 1-18, 2002.
- Golden B.L., Raghavan S., Wasil E.A., The Vehicle Routing Problem: Latest Advances and New Challenges, *Operations Research/Computer Science Interfaces Series*,

Springer, New York, USA, 2008.

- Gonzalez-Velarde J. L., Pinto G., Camacho F., Bilevel Optimization Model for Determining Highway Tolls*, Proceedings CLAIO SBPO*, 2012.
- Guo X., Hu T., Zhang T., Lv Y., Bilevel model for multi-reservoir operating policy in inter-basin water transfer-supply project, *Journal Hydrology*, 252-263, 2012.
- Haurie A., Savard G., Rhite D., A note on: an efficiente poin algorithm for a linear two-stage optimization problems, *Operations Research*, 38,553-555, 1990.
- Hejazi S.R., Memariani A., Jahanshahloo G., Sepehri M.M., Linear bilevel programming solution by genetic algorithm, *Computer & Operation Research*, 29, 1913-1925, 2002.
- Hongli G., Juntao L., Hong, G., A Survey of Bilevel Programming Model and Algorithm, *2011 Fourth International Symposium on Computational Intelligence and Design*, 2, 199-203, 2011.
- Hongmei J., Bin W., Lin Z., Yanping L., A bilevel time-dependenti scheduling model for Hazmat road transportation, Road Transport Information and Control Conference and the ITS United Kingdom Members' Conference (RTIC 2010),1-6, 2010.
- Júdice J. J., Faustino A. M., A sequential LCP method for bilevel linear programming, *Annals of Operations Research*, 34, 89-106, 1992.
- Jang Y.J., Jang S.Y., Chang B.M., Park J., A combined model of network design and production/distribution planning for a supply network, *Computers & Industrial Engineering*, 43:1-2, 263-281, 2002.
- Jayaraman V., Pirkul H., Planning and coordination of production and distribution facilities for multiple commodities, *European Journal of Operational Research*, 133:2, 394-408, 2001.
- Jeroslow R., The polynomial hierarchy and a simple model for competitive analysis, *Mathemaical Programming*, 32, 146-164, 1985.
- Jia L., Wang Y., Fan L., Multiobjective bilevel optimization for production-distribution planning problems using hybrid genetic algorithm, *Integred Computer-Aided Engineering*, 21, 77-90, 2014.
- Kalashnikov V. V., Pérez-Valdés G. A., Tomasgard A., Kalashnykova N. I., Natural gas

cash-out problem: Bilevel stochastic optimization approach, *European Journal of Operational Research*, 206, 18-33, 2010.

- Kazemi A., Fazel Zarandi, M.H., Moattar Husseini, S.M., A multi-agent system to solve the production-distribution planning problem for a supply chain: a genetic algorithm approach*, International Journal of Advanced Manufacturing Technology*, 44, 180-193, 2009.
- Keskin B.B., Üster H., Meta-heuristic approaches with memory and evolution for a multi-product production-distribution system design problem, *European Journal of Operational Research*, 182:2, 663-682, 2007.
- Kocvara M., Outrata J.V., On the solution of optimum design problems with variational inequalities. *Recent Advances in Nonsmooth Optimization*, 171-191, 1995.
- Laguna M., Martí R., Scatter Search: Methodology and Implementations in C, *Kluwer Academic Publishers*, 2003.
- Laporte G., Nobert Y., A branch and bound algorithm for the capacited vehicle routing problem, *OR Spektrum*, 5, 77-85, 1983.
- Laporte G., Nobert Y., Taillefer S., Solving a family of multi-depot vehicle routing and location-routing problems, *Transportation Science*, 22, 161-172, 1988.
- Legillon F., Liefooghe A., Talbi E.G., A coevolutionary meta-heuristic for bi-level optimization, *Rapport de Recherche*, 30, 320-327, 2011.
- Lenstra J. K., Rinnoy Kan A. H. G, Complexity of vehicle routing and scheduling problems, *Networks*, 11, 221-227, 1981.
- Lin S., Computer Solutions of the Traveling Salesman Problem*, Bell Systems Technical Journal*, 44, 2245-2269, 1965.
- Lopez J., Nieto S., Heuristic for the generation of a set reference that solve the vehicle routing problem with multiple deposits MDVRP, *Tenth LACCEI Latin American and Caribben Conference (LACCEI'2012)*, 2012.
- Lukac Z., Šoric K., Rosenzweig V., Production planning problem with sequence dependent setups as a bilevel programming problem, *European Journal of Operational Research*, 187, 1504-1512, 2008.
- Marcotte P., Marquis G., Efficient implementation of heuristic for the continuous

network design problem. *Annals of Operations Research,* 34, 142-162, 1986.

- Marić M., Stanimirovic Z., Milenkovic N., Metaheuristic methods for solving the bilevel uncapacitated facility location problem with clients preferences, *Electronic Notes in Discrete Mathematics*, 39, 43-50, 2012.
- Mathieu R., Pittard L., Anandalingam G., Genetic algorithm based approach to bi-level linear programming, *RAIRO-Operations Research,* 28, 1–21, 1994.
- Miller T., Friesz T., y Tobin R., "Heuristic algorithms for delivered price spatially competitive network facility location problems." *Annals of Operations Research,* 34, 155-202, 1992.
- Oduguwa V., Roy R., Bi-level optimization using genetic algorithm, *In IEEE International Conference on Artificial Intelligence Systems (ICAIS 2002)*, 322-327, 2002.
- Oña J.d., Gómez P., Mérida-Casermeiro E., Bilevel fuzzy optimization to preprocess traffic data to satisfy the law of flow conservation, *Transportation Research Part C: Emerging Technologies*, 19, 29–39,2011.
- Rajesh J., Gupta K., Kusumakar H.S., A tabu search based approach for solving a class of bilevel programming problems in chemical engineering, *Journal of Heuristics,* 9, 307–319, 2003.
- Sarmiento, A.M., Nagi, R., A review of integrated analysis of production-distribution systems*, IIE Transactions*, 31, 1061-1074, 1999.
- Savard G, *Contribution* à la *programmation mathématique* à *deux niveaux*. *PhD thesis, Ecole. Polytechnique de Montréal, Université de Montréal*, 1989.
- Scaparra M.P., Church R. L., A bilevel mixed-integer program for critical infrastructure protection planning, *Computers & Operation Research*, 35, 1905-1923, 2008.
- Shi Ch., Lu J., Zhang G., Zhou H., An extended branch and bound algorithm for linear bilevel programming, *Applied Mathematics and Computation*, 180, 529-537, 2006.
- Sonia*, Khandelwal A., Puri, Bilevel time minimizing transportation problem, *Discrete Optimization*, 714-723, 2008.
- Stackelberg H., The theory of Market Economy, Oxford University Press, 1952.

- Stackelberg H.V., Marktform und Gleichgewitcht, Vienna: Springer, 1934.
- Syarif, A., Yun, Y., Gen, M., Study on multi-stage logistic chain network: A spanning tree-based genetic algorithm approach, *Computers and Industrial Engineering*, 43, 299-314, 2002.
- Toth, P., Vigo, D., The Vehicle Routing Problem, *Monographs on Discrete Mathematics and Applications*, SIAM, Philadelpia, PA, 2002.
- Vicente L.N., Calamai P.H., Bilevel and multilevel programming: a bibliography review, *Journal of Global Optimization*, 5, 291–306, 1994.
- Vidal C.J., Goetschalckx M., Strategic production-distribution models: A critical review with emphasis on global supply chain models, *European Journal of Operational Research*, 98:1, 1-18, 1997.
- Wan S., Wang G., Sun B., A hybrid intelligent algorithm by combining particle swarm optimization with chaos searching technique for solving nonlinear bilevel programming problems, *Swarm and Evolutionary Computation*, 8, 26-32, 2013.
- Wang G., Wan Z., Wang X., Lv Y., Genetic algorithm based on simplex method for solving linear-quadratic bilevel programming problem, *Computer & Mathematics with Applications*, 56, 2550-2555, 2008.
- Wang G.M, Wang X.J., Wan Z.P., Jia S.H., An adaptive genetic algorithm for solving bilevel linear programming problem, *Applied Mathematics and Mechanics,* 28, 1605–1612, 2007.
- Williams, J.F., Heuristic Techniques for Simultaneous Scheduling of Production and Distribution in Multi-Echelon Structures: Theory and Empirical Comparison, *Management Science*, 27:3, 336-352, 1981.
- Zugno M., Morales J.M., Pinson P., Madsen H., A bilevel model for electricity retailers participation in a demand response market environment, *Energy Economics*, 36, 182-197, 2013.

ANEXO 1

Imagen 4.1: Pseudocódigo Búsqueda Dispersa

1) Método de Generación Diversa: Se basa en un conjunto *P* de soluciones diversas, basado en la metodología de Cluster (Asignación con épsilon) – Ruteo (GRASP).
2) Método de Mejora: Se aplica una mejora heurística a las soluciones en *P*
    2.1) *Remove-Insertion*
    2.2) *Interchange*
        2.2.1 Resuelve el Nivel Inferior para *P*
    2.3) 2-Opt
3) Crear Refset: Seleccionas b1 mejores soluciones en P (Función del Líder) y b2 más diversas en P
4) Nuevas Soluciones=Verdadero
While (Nuevas Soluciones=Verdadero) {

- 5) Nuevas Soluciones = Falso
- 6) Generación de Subconjuntos: Creas los subconjuntos de Refset para ser combinados
    - 6.1) Combinación de Soluciones: Aplicas el método de combinación a cada pareja
    - 6.2) Método de Mejora: Se aplica una mejora heurística a las soluciones en *P*
        6.2.1) *Remove-Insertion*
        6.2.2) *Interchange*
            6.2.2.1) Resuelve el Nivel Inferior para *Nuevo elemento*
        6.2.3) 2-Opt
    - 6.3) Actualizar Refset: Actualizas Refset con *b* soluciones en *Refset* U *Nuevo elemento*
- 7) if (Refset ha cambiado) Nuevas Soluciones=Verdadero

}

Imagen 4.2: Pseudocódigo conjunto de soluciones diversas

$P \leftarrow \emptyset$

$x \leftarrow \emptyset$

$while \mid P \mid \leq P_{size}$

R =*A.C.D.Asignación clientes a Depot*

$x_j \in A.C.D, \ \forall j \in L \ (j \in Candidatos)$

$x \leftarrow x_j$

$while \ |R \setminus x| \geq 0$

$c(x_j) = d_{ij} + d_{jl} - d_{li}\ ; \quad \forall\, x_j \in C = R \setminus x$

$c^{max} \leftarrow \max \{c(x_j) : x_j \in C\}$

$c_{min} \leftarrow \min \{c(x_j) : x_j \in C\}$

$RCL \leftarrow \{x_j \in C: c(x_j) \leq c_{min} + \alpha \left(c^{max} - c_{min}\right)\}$

$Seleccionar\ un\ x_j\ uniformemente\ al\ azar\ de\ RCL$

$x \leftarrow x \cup x_j$

$Calcular\ c(x_j) \ \forall\ x_j \in C$

end

$x\ entra\ Método\ de\ Mejora$

$X \leftarrow X \cup x$

$P \leftarrow P \cup \mathrm{X}$

$x \leftarrow \emptyset$

end