UNIVERSIDAD AUTONOMA DE NUEVO LEON
FACULTAD DE INGENIERIA MECANICA Y ELECTRICA
DIVISION DE ESTUDIOS DE POSTGRADO

BUSQUEDA TABU PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS

TESIS
EN OPCION AL GRADO DE MAESTRO EN CIENCIAS
EN INGENIERIA DE SISTEMAS

PRESENTA
ING. NADIA COBOS ZALETA

CD. UNIVERSITARIA JUNIO 2004

BUSQUEDA TABU PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS

UNIVERSIDAD AUTONOMA DE NUEVO LEON
FACULTAD DE INGENIERIA MECANICA Y ELECTRICA
DIVISION DE ESTUDIOS DE POSTGRADO

BUSQUEDA TABU PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS

T E S I S
EN OPCION AL GRADO DE MAESTRO EN CIENCIAS
EN INGENIERIA DE SISTEMAS

P R E S E N T A
ING. NADIA COBOS ZALETA

CD. UNIVERSITARIA JUNIO 2004

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUDIOS DE POSGRADO

BÚSQUEDA TABÚ PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS

POR:

ING. NADIA COBOS ZALETA

TESIS

EN OPCIÓN AL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA DE SISTEMAS

CD. UNIVERSITARIA **JUNIO 2004**

UNIVERSIDAD AUTÓNOMA DE NUEVO LEÓN

FACULTAD DE INGENIERÍA MECÁNICA Y ELÉCTRICA

DIVISIÓN DE ESTUDIOS DE POSGRADO

BÚSQUEDA TABÚ PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS

POR:

ING. NADIA COBOS ZALETA

TESIS

EN OPCIÓN AL GRADO DE MAESTRO EN CIENCIAS EN INGENIERÍA DE SISTEMAS

CD. UNIVERSITARIA **JUNIO 2004**

Universidad Autónoma de Nuevo León
Facultad de Ingeniería Mecánica y Eléctrica
División de Estudios de Posgrado

Los miembros del Comité de Tesis recomendamos que la tesis **"BÚSQUEDA TABÚ PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS"**, realizada por la alumna Nadia Cobos Zaleta, matrícula 791757, sea aceptada para su defensa como opción al grado de Maestro en Ciencias en Ingeniería de Sistemas.

El Comité de Tesis

Dra. Ada M. Álvarez Socarrás
Asesor

Dr. César E. Villarreal Rodríguez
Revisor

Dr. Igor S. Litvinchev
Revisor

Vo. Bo.

Dr. Guadalupe Alan Castillo Rodríguez
División de Estudios de Posgrado

Ciudad Universitaria, a Junio de 2004

DEDICATORIA

Quiero dedicar este trabajo principalmente a mi madre que me apoyado en todo incondicionalmente, quien siempre ha confiado en mí y a quienes siempre me han apoyado brindándome su cariño.

Gracias Dios mío por que me has permitido culminar una etapa más en mi vida.

También quiero agradecer a mi Tía Judith y a mi hermana que siempre estuvieron a mi lado apoyándome y dándome ánimos.

AGRADECIMIENTOS

A mis profesores, mis compañeros y a todas las personas que me ayudaron y me dieron su tiempo para poder concluir mis estudios de maestría.

Principalmente quiero agradecer a la Dra. Ada Álvarez, mi asesora, quien ha tenido mucha paciencia para lograr concluir este proyecto.

Gracias a mis revisores y profesores, por darse el tiempo y brindarme sus valiosos comentarios.

Agradezco la oportunidad al PISIS y CONACYT por haberme permitido realizar mis estudios de maestría, mediante la beca del proyecto 36669-A y por permitirme desempeñarme como Asistente de Investigación.

RESUMEN

Nadia Cobos Zaleta

Candidato para el Grado de Maestro en Ciencias en Ingeniería de Sistemas

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Título del Estudio:

BÚSQUEDA TABÚ PARA UN PROBLEMA DE DISEÑO DE RED MULTIPRODUCTO CON CAPACIDAD FINITA EN LAS ARISTAS

Número de páginas: 76

Objetivo y Método de Estudio

En esta tesis abordamos un problema de diseño de red capacitada multiproducto, que tiene como objetivo determinar qué aristas deben considerarse en el diseño de forma que se garantice la operación de la red y que el costo total en que se incurra (considerando costos de diseño y de operación) sea el menor posible.

En este problema se deben transportar varios productos a través de una red con capacidades finitas en las aristas desde su origen hasta su respectivo destino. Al hacer esto, se incurre en un costo por transportar flujo llamado "costo variable", así como un costo de construcción llamado "costo fijo", por el uso de cada arista. Asociadas a cada arista está la capacidad, que será compartida por todos los productos que la usen sin importar la dirección del flujo. El objetivo es decidir cuáles aristas deben ser incluidas en el diseño de red de forma que el costo total que se incurre por construcción y operación de la misma sea el menor posible. Es importante destacar que una vez que una arista ha sido incluida en el diseño de la red, se permitirá flujo en ambas direcciones, por lo que los costos variables de transportación asociados a cada arista dependerán no solamente del producto sino también del sentido en que esté circulando.

El problema descrito se formulará como un programa lineal entero mixto (compuesto por variables continuas y discretas) y desde el punto de vista de optimización, aún en los casos más sencillos, es difícil de resolver ya que se ha demostrado que pertenece a la clase NP-completo (Garey, Johnson, 1979; Hochbaum, Segev, 1989). Esto justifica la necesidad de diseñar algoritmos aproximados de solución (heurísticas).

Los objetivos centrales de esta tesis son:

- Desarrollar e implementar un sistema de optimización basado en técnicas inteligentes de búsqueda local aplicados a resolver el problema de diseño de red capacitada.
- Evaluar computacionalmente el algoritmo, es decir, encontrar los valores de los parámetros algorítmicos que provean soluciones de mayor calidad.
- Comparar los resultados obtenidos con un algoritmo evolutivo (Búsqueda Dispersa).

Para resolver nuestro problema empleamos la metaheurística búsqueda tabú (tabú search, TS, en inglés) basado en un algoritmo de búsqueda local. Lo que se pretende con la metodología propuesta, es encontrar soluciones de calidad para tamaño reales de instancias en un tiempo razonable.

Para lograr estos objetivos se siguió la siguiente metodología. En primera instancia se llevó a cabo un estudio de la estructura del problema para establecer el modelo matemático. Posteriormente, conociendo que la búsqueda tabú había resultado efectiva en un problema similar para redes orientadas, se hicieron las modificaciones necesarias para adecuarla al problema aquí abordado.

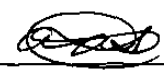
Posteriormente se utilizó el lenguaje de programación C y el optimizador CPLEX versión 7.1 (1999) en una estación de trabajo SUN Ultra 10 con sistema operativo Solaris versión 7, para la implementación del algoritmo de resolución del problema.

Se usaron diferentes tipos de instancias, 120 en total, para validar la metodología de solución. Las instancias se clasificaron por número de nodos, número de productos, tipo de costo, capacidad y tipo de red con la finalidad de probar el método y evaluar su

comportamiento.

Contribuciones y Conclusiones

- Se logra un entendimiento profundo de la metaheurística búsqueda tabú y una comprensión de las relaciones entre sus estrategias, los pivoteos simplex y la generación de columnas. El procedimiento representa un esfuerzo exitoso que propone una eficiente aproximación para obtener buenas soluciones factibles al problema abordado y se muestra robusto con respecto al tipo de problema, en términos de la importancia relativa de los costos fijos, capacidades, tamaño y especialmente en el número de productos.
- Se provee un método de solución rápido y eficiente para un problema difícil de optimización de redes, ya que hasta el momento no se conocen algoritmos exactos que puedan resolver los problemas de tamaño real, aquí abordados en un período razonable de tiempo.
- Se evalúa el método búsqueda tabú y se determina bajo que condiciones de sus parámetros se obtienen los mejores resultados.
- Se entrega un sistema computacional que aplica los procedimientos propuestos a un problema de diseño de red multiproducto con capacidad finita en las aristas y cargos fijos. Este sistema hace uso de archivos de formato específico para su ejecución y obtiene como resultado un buen diseño con soluciones de alta calidad, especificando los costos incurridos (fijos y variables), los volúmenes de flujo de cada producto que circularán por cada una de las aristas incluidas en la red para satisfacer las demandas.
- Los resultados obtenidos se comparan con los obtenidos con un algoritmo evolutivo basado en la técnica de búsqueda dispersa y puede observarse que el método aquí propuesto se desempeña mejor que el basado en búsqueda dispersa.

FIRMA DEL ASESOR: 

Dra. Ada M. Álvarez Socarrás

ÍNDICE

CAPÍTULO 1

INTRODUCCIÓN

1.1 Descripción y relevancia del problema

Los problemas de diseño de red óptima surgen en numerosas y variadas aplicaciones en el área de Investigación de Operaciones. Estos problemas se caracterizan por la búsqueda de la mejor configuración de red que satisfaga un conjunto dado de requerimientos. Aquí se pueden considerar problemas de expansión de red, problemas de ubicación de plantas, problemas de selección de procesos, así como una amplia variedad de problemas relacionados con distribución e inversión. Este tipo de problemas se presentan en muy diversos campos, por ejemplo en diseño o rediseño de redes de computadoras o telecomunicaciones, sistemas eléctricos de potencia, redes de transporte, etc. Una amplia compilación de modelos y aplicaciones puede ser encontrada en la obra de Magnanti y Wong (1984).

En cada uno de los problemas arriba mencionados, la decisión central es "invertir o no invertir", "construir o no construir" y puede ser modelada imponiendo "cargas fijas" en arcos o nodos apropiados de la red.

En los problemas de ubicación de instalaciones, por ejemplo, el diseño consiste en seleccionar un subconjunto de nodos de la red en los cuales se ubicarán instalaciones y asignar clientes a esas instalaciones con el objetivo de minimizar los costos totales (Daskin, 1995). En el problema de expansión de red, el diseño consiste en la selección de un subconjunto de arcos de forma que se garantice cierto desempeño en la red y además que se minimicen los costos totales por concepto de construcción (o utilización) de arcos y por transportación de bienes por los mismos. Estos tipos de problemas pueden considerar o no capacidades finitas en los arcos de la red, incluirlas hace el problema más complejo, pero también más realista.

En el presente trabajo se aborda un problema de diseño de redes capacitadas multiproducto con un solo nodo origen y destino por producto. Aquí varios productos son transportados a través de las conexiones o aristas de la red desde sus orígenes hasta sus destinos. A cada arista se le asignará un costo fijo por su utilización o construcción, así como un costo variable por transportar una unidad de producto por la arista, el cual depende del producto y del sentido en que circule. Así mismo cada arista tiene asociada una capacidad finita, la cual deberá ser compartida por todos los productos que la usen sin importar la dirección del flujo.

En este trabajo de diseño de red se dará respuesta a la siguiente pregunta de toma de decisiones: ¿Qué configuración de red minimiza la suma de los costos fijos de las aristas seleccionadas y el costo por transportación de bienes a través de la red definida por tales aristas, considerando que estas últimas poseen capacidades finitas?

Si bien esta clase de problemas puede formularse como un programa lineal entero mixto, aún en los casos más sencillos, es difícil de resolver de forma exacta, e inclusive ni los paquetes comerciales de optimización disponibles pueden obtener soluciones para tamaños de instancias reales. La mayoría de los métodos de solución propuestos en la literatura se basan en el hecho de que el problema de diseño de red óptima es *NP-completo* (Johnson, Garey, 1979; Hochbaum, Segev, 1989). Esto justifica la necesidad de diseñar algoritmos aproximados de solución llamados heurísticas, que si bien no garantizan el óptimo global, ofrecen una solución suficientemente buena (calidad aceptable) en un tiempo razonable de forma relativamente sencilla y rápida.

Si bien problemas similares han sido tratados sobre redes orientadas, el problema sobre redes no orientadas solo ha sido estudiado previamente por Herrmann et al. (1996), De Alba, Álvarez y Gónzalez-Velarde (2003) y De Alba (2004). Herrmann considera el problema de diseño de redes capacitadas en grafos no dirigidos, como se hace en el presente trabajo, y presenta una técnica ascendente dual para encontrar buenas cotas inferiores y soluciones cercanas a la óptima. Sin embargo, por una parte sólo es aplicable a redes muy pequeñas (60 aristas y 35 productos) tipo malla y por otra, fue demostrado posteriormente (Gendron y Crainic, 1996) que las cotas inferiores no mejoraban la relajación lineal. De Alba, por su parte, desarrolla un algoritmo heurístico

basado en la técnica de Búsqueda Dispersa. Sin embargo, la calidad de las soluciones entregadas por este algoritmo comienzan a degradarse para redes de 50 nodos y 100 productos o más, obteniendo soluciones de peor calidad en comparación con la mejor solución entera entregada por el optimizador CPLEX versión 7.1 (1999).

Por ello se ha considerado importante la realización del presente trabajo, dada la necesidad de desarrollar métodos específicos de solución que encuentren soluciones de alta calidad al problema de diseño de red multiproducto con capacidades en las aristas para instancias de tamaño real. En particular el problema será abordado utilizando la metodología Búsqueda Tabú, la cual ha resultado exitosa en numerosas aplicaciones a problemas combinatorios como por ejemplo el problema de ruteo de vehículos, el problema de localización de plantas y el problema del agente viajero entre otros.

1.2 Antecedentes Científicos

En la literatura se han reportado numerosos trabajos de diseño de redes: uniproducto ó multiproducto, con capacidades finitas ó infinitas en las conexiones y sobre redes dirigidas ó no dirigidas. A continuación se mencionarán algunos trabajos que se han desarrollado sobre el diseño de redes así como las técnicas de solución propuestas.

En los problemas de diseño de redes no capacitadas, toda la demanda de los productos puede ser transportada a través de una única ruta que conecte los nodos origen y destino para cada producto, debido a que en estas redes los arcos poseen capacidades infinitas.

Uno de los trabajos desarrollados para problemas de redes no capacitadas es el de Cruz, MacGregor y Mateus (1998). En este trabajo se aborda el diseño de una red para un problema de distribución a costo mínimo de la demanda de un producto sobre un conjunto de nodos destino en una red dirigida y no capacitada con costos fijos y costos de transportación. Este trabajo extiende el uso de la técnica de *ramificación y acotamiento* a problemas de dimensiones mayores. Se consideraron demandas unitarias para cada nodo destino. Los problemas se dividieron en dos tipos de acuerdo a una razón de costo fijo entre costo variable: problemas cercanos a un diseño de redes no capacitadas, que pueden resolverse polinomialmente, y cercanos a problemas *Steiner*, los

cuales son *NP-Completos*.

Balakrishnan y Magnanti (1989), presentan un procedimiento para problemas de diseño de redes a gran escala y se hace sobre redes no dirigidas, generalizando el algoritmo de ascenso dual usado para resolver los problemas de rutas más cortas, ubicación de plantas, redes de *Steiner* y árboles de expansión dirigida. Debido a que la red es no capacitada, para un diseño determinado de red, este problema se descompone en problemas de ruta más corta, un problema para cada producto que fluye en la red.

Un trabajo que trata el diseño para problemas de flujo en redes capacitadas dirigidas con costos fijos de un solo producto es el llevado a cabo por Khang y Fujiwara (1991). Aquí se utiliza la técnica de relajación lagrangeana para encontrar el árbol de expansión mínima, lo que proporciona una cota inferior al problema original. Además se emplea una heurística de escalamiento para encontrar una solución factible que servirá como cota superior. Los experimentos computacionales se llevaron a cabo en redes completas dirigidas con n nodos y $n(n\text{-}1)/2$ arcos. El número de nodos varía entre 5 y 40, con incrementos de 5 y de acuerdo a la forma de calcular los arcos, éstos varían de 10 hasta 780.

El trabajo realizado por Holmberg y Yuan (1998), propone una técnica para encontrar la solución óptima a un conjunto de modelos de diseño de redes con costos fijos. Estos modelos pueden ser capacitados o no capacitados, dirigidos o no dirigidos, incluyendo costos variables escalonados y requerimientos de supervivencia. En este trabajo se propone una heurística lagrangeana compuesta por una relajación lagrangeana, un procedimiento de optimización por subgradiente, y una heurística primal. Posteriormente, estos mismos autores ampliaron su investigación (Holmberg y Yuan, 2000) con las mismas instancias modificadas a fin de transformarlas para casos capacitados, con costos escalonados, etc.

Uno de los trabajos desarrollados sobre el diseño de redes capacitadas, no orientadas, multiproducto es el de Herrmann at al. (1996), de hecho, este problema es muy similar al problema que se aborda en el presente trabajo, y prácticamente poseen la misma formulación, con flujos que representan proporciones de demanda total para cada producto. La principal diferencia estriba en que la red propuesta por ellos está basada en

redes tipo malla. Se usa en este artículo la técnica propuesta por Balakrishnan (1989), para el problema no capacitado y se adecua al problema capacitado. En este trabajo se pretendía obtener mejores cotas inferiores que la relajación lineal, pero no se logra como lo demostró posteriormente Gendron (1999), sin embargo, obtiene una solución factible al problema. El experimento a pesar de tratar con problemas relativamente pequeños, no reporta óptimos para las retículas clasificadas con 30, 40, y 50 nodos. No presenta tiempos de ejecución para ningún caso.

Otro trabajo desarrollado para resolver redes con capacidades finitas multiproducto y costos fijos es el propuesto por Sridhar y Park (2000), para una aplicación específica de redes de telecomunicación. Las características del problema abordado son las siguientes: la red es completa (cada nodo se encuentra comunicado con todos los demás nodos) y hay oferta y demanda en todos los nodos, no existen costos variables por transportar la demanda en los arcos, los nodos tienen capacidad y todos los arcos tienen una misma capacidad. La metodología propuesta para resolver este problema usa el algoritmo de ramificación y acotamiento y un método de Benders-and-cut para resolver cada subproblema en cada uno de los nodos del árbol. Las instancias diseñadas para probar el método tienen las siguientes características: 6, 10, 15, y 20 nodos; 15, 45, 105, y 190 arcos.

Holmberg y Yuan (2000), implementan un método para el diseño de redes capacitadas multiproducto donde la red es dirigida. El método de solución empleado se basa en una heurística lagrangeana insertada dentro de un contexto de *ramificación y acotamiento*. La contribución de este trabajo es poder obtener soluciones exactas, o bien soluciones aproximadas dependiendo del tiempo disponible y de la calidad de solución buscada. Se trabajó con problemas de tamaños diferentes variando el número de parámetros de la siguiente manera: Nodos de 17 a 150, Arcos de 272 a 1000, y Productos de 16 a 282, subdivididos en 7 grupos de problemas, con un total de 65 problemas probados.

El trabajo realizado por Crainic, Gendreau y Farvolden (2000), también sobre redes dirigidas y capacitadas hace uso de la meta-heurística Búsqueda Tabú y el método *simplex* para encontrar soluciones relativamente buenas al problema de diseño de redes

multiproducto. La contribución del trabajo es que se obtienen soluciones buenas para problemas reales de hasta 400 productos con 20 nodos y 230 arcos. Sin embargo en los problemas grandes tiene variaciones desde un 17% hasta un 32% con respecto al optimizador.

En el trabajo desarrollado por De Alba, Álvarez y Gónzalez-Velarde (2001) y De Alba (2004), se aborda el mismo problema de diseño de red que será estudiado en el presente trabajo. El problema se modela como un programa entero-mixto, pero para su solución se diseña una heurística basada en un marco de Búsqueda Dispersa (Scatter Search, SS, en inglés), que trabaja con una población obtenida mediante un *grasp*, diseñado por De Alba, Alvarez y González-Velarde (2001). Se diseñó un experimento computacional para el cual fueron generadas instancias aleatorias de 30 nodos y 750 aristas con 10, 50 y 100 productos, 50 nodos y 1800 aristas con 10, 50 y 100 productos. El algoritmo funciona pero se le dificulta trabajar para instancias hasta de 50 nodos y 100 productos, obteniendo soluciones de peor calidad respecto a la mejor solución entera entregada por el optimizador CPLEX.

1.3 Heurísticas

El término heurística proviene de la palabra griega *heuriskein* relacionado con el concepto de encontrar. Se califica de heurístico a un procedimiento que encuentra soluciones de alta calidad con un costo computacional razonable, aunque no se garantice su optimalidad.

Las heurísticas son métodos para resolver problemas complejos en forma aproximada, además una ventaja importante es que su complejidad es reducida en comparación con los métodos exactos, por lo que suelen ser más fáciles de entender (por parte de los directivos de las empresas y gente no experta), son flexibles y tiene como objetivo, encontrar soluciones de buena calidad en un tiempo computacional razonable sin mencionar que generalmente ofrecen más de una solución permitiendo así ampliar las posibilidades de elección. A pesar de sus ventajas, no cabe duda de que cuando una técnica exacta está disponible debe ser preferida a cualquier tipo de heurística, sobre todo cuando los valores económicos manejados sean importantes y el tiempo para resolverlos no está limitado.

En los últimos años ha habido un crecimiento en el desarrollo de procedimientos heurísticos para resolver problemas combinatorios, debido a la necesidad de disponer de herramientas que permitan ofrecer soluciones rápidas a problemas reales. Los problemas de optimización combinatoria en particular son de gran dificultad debido a su complejidad, ya que crecen exponencialmente con el tamaño del problema, por lo cual se pretende que los métodos heurísticos se acercan a la solución óptima en un tiempo razonable.

Existen diferentes tipos de heurísticas, las cuales pueden clasificarse como:

a) Métodos constructivos: Son aquellos que añaden componentes individuales a una solución parcial hasta que se obtiene una solución factible. El más popular de estos métodos lo constituye un algoritmo goloso o voraz "greedy", el cual construye la solución buscando el máximo beneficio en cada paso, un ejemplo es el GRASP.

b) Métodos de descomposición: Consisten en dividir el problema en subproblemas más pequeños, siendo la salida de uno la entrada de otro, de forma que al resolver ambos subproblemas obtengamos una solución para el problema global. Un ejemplo de aplicación en un problema de programación lineal mixta, consistiría en decidir de alguna forma la solución para las variables enteras para luego resolver el problema como un LP.

c) Métodos de reducción: Identifican alguna característica que deba poseer la solución óptima y de este modo simplifican el problema. Por ejemplo la detección de alguna variable con ciertos valores o correlación, etc.

d) Manipulación del modelo: Modifican las estructuras del modelo con el fin de hacerlo más sencillo de resolver, deduciendo, a partir de la solución del problema modificado, la solución del problema original. Como por ejemplo, se puede reducir el espacio de soluciones o eliminado restricciones del problema.

e) Métodos de búsqueda por entornos: Parten de una solución factible inicial (probablemente obtenida de otra heurística) y mediante alteraciones de la solución, van iterando a otras factibles de su entorno, almacenado la mejor solución encontrada hasta que se cumpla un determinado criterio de parada.

1.4 Metaheurísticas

Las *metaheurísticas* son estrategias maestras inteligentes para diseñar o mejorar procedimientos heurísticos muy generales con un alto rendimiento. El término metaheurísticas se obtiene de anteponer a heurística el prefijo *meta* que significa "más allá" o "a un nivel superior".

Hoy día, el interés primordial de los investigadores es el de diseñar métodos generales que sirvan para resolver clases o categorías de problemas. Dado que estos métodos generales sirven para construir o guiar el diseño de métodos que resuelvan problemas específicos se les ha dado el nombre de Metaheurísticos. La siguiente definición fue introducida por Osman y Kelly (1995): "Los procedimientos metaheurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos clásicos no son ni efectivos ni eficientes. Los metaheurísticos proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de: inteligencia artificial, evolución biológica y mecanismos estadísticos"

Los metaheurísticos más utilizados y reconocidos en la Optimización Combinatoria son: Recocido Simulado, Búsqueda Tabú, GRASP y Algoritmos Genéticos, los cuales se explicarán brevemente a continuación.

1.4.1 Recocido Simulado

Desde que Kirkpatrick, Gelatt y Vecchi (1983) introdujeron el concepto de Recocido Simulado, RS, ("Simulated Annealing", SA, en inglés), esta metaheurística ha demostrado ser una herramienta muy exitosa para resolver una amplia gama de problemas de optimización combinatoria. El recocido simulado es una variante de la búsqueda local que permite movimientos ascendentes para evitar quedar atrapado en un óptimo local. El nombre está basado en un algoritmo diseñado para simular el enfriamiento de material (un proceso denominado "recocido").

Kirkpatrick et al. consideraron aplicar el algoritmo de Metrópolis del campo de la termodinámica estadística. Básicamente se modeló el proceso simulando iterativamente cambios energéticos en un sistema de partículas de una sustancia conforme decrece la temperatura hasta llegar a un equilibrio térmico, utilizando la distribución de Boltzman.

En el algoritmo de Metropolis se genera una perturbación aleatoria o cambios en el sistema y se calculan cambios de energía resultantes basado en una búsqueda local, donde si hay una caída energética el cambio se acepta (movimiento de mejora), por el contrario, si se produce un incremento energético (movimiento de no mejora), el cambio será aceptado con una probabilidad proporcional al factor de Boltzman. Dichas probabilidades están basadas en la analogía con el proceso de enfriamiento y se obtiene como función de la temperatura del sistema.

Algunos de los problemas que han sido tratados con esta técnica son: diseño de componentes electrónicos, procesamiento de imágenes, simulación física y distribución de recursos. Una amplia compilación de trabajos relacionados puede ser encontrada en Díaz et al. (1996).

1.4.2 Algoritmos genéticos

Los Algoritmos Genéticos (AG) son "técnicas de búsqueda basadas en la mecánica de la selección natural y la genética" y fueron introducidos por John Holland en 1970.

Debido en parte a la selección natural, cada especie gana una cierta cantidad de "conocimiento" o información hereditaria la cual es pasada a través de los cromosomas que contienen la información de todos esos factores, es decir, los genes, los cuales a su vez están compuestos por un determinado número de valores (alelos). Los organismos se agrupan formando una población y aquellos que mejor se adaptan son los que más probabilidades tienen de sobrevivir y reproducirse. Algunos de estos son seleccionados para ser cruzados y así producir una nueva generación, donde esporádicamente los genes de un cromosoma pueden sufrir ligeros cambios llamados mutaciones, las cuales consisten en alterar un bit de un cromosoma, con una probabilidad relativamente pequeña. El sobrecruzamiento es el intercambio de genes entre parejas, obteniendo así dos nuevos cromosomas de una nueva generación (Díaz et al, 1996).

Un AG empleará iterativamente las operaciones de reproducción, sobrecruzamiento y mutación de cromosomas (soluciones) de una población y generará nuevos organismos. Algunas de las aplicaciones de los algoritmos genéticos son: distribución de planta, partición de nodos, asignación de procesos, entre otras, las cuales

pueden ser encontradas en Díaz et al. (1996).

1.4.3 GRASP

Es un procedimiento de búsqueda miope aleatorizado y adaptativo ("greedy randomized adaptive search procedure", GRASP, en inglés) es una metaheurística para encontrar soluciones aproximadas a problemas de optimización combinatoria. El GRASP es un método multi-arranque y el término fue introducido por Feo y Resende (1995).

El GRASP es un procedimiento iterativo en donde cada paso consiste de una fase de construcción y otra de mejora. En la fase de construcción se aplica un procedimiento heurístico constructivo para obtener una buena solución inicial (solución miope aleatorizada). Esta solución se mejora en la segunda fase mediante un algoritmo de búsqueda local y finalmente la mejor de todas las soluciones examinadas se guarda como resultado final. Algunas de las principales aplicaciones del GRASP son de localización, asignación, ruteo, transportación, telecomunicaciones, entre otras. Una amplia compilación de trabajos relacionados puede ser encontrada en Glover y Kochenberger (2003).

1.4.4 Búsqueda Dispersa

La Búsqueda Dispersa, (Scatter Search, SS, en inglés), es un procedimiento metaheurístico basado en estrategias para combinar reglas de decisión, así como en la combinación de restricciones. El método SS opera sobre un conjunto de soluciones, llamado conjunto de referencia, combinando éstas para crear nuevas soluciones de modo que mejoren a las que las originaron. En este sentido decimos que es un método evolutivo.

Sin embargo, a diferencia de otros métodos evolutivos, como los algoritmos genéticos, SS no está fundamentado en la aleatorización sobre un conjunto relativamente grande de soluciones sino en elecciones sistemáticas y estratégicas sobre un conjunto pequeño, pues SS se basa en el principio de que la información sobre la calidad o el atractivo de un conjunto de reglas, restricciones o soluciones puede ser utilizado mediante la combinación de éstas en lugar de aisladamente.

Algunas de sus aplicaciones son: problemas de coloración de grafo, problemas de

ciclas máximas, secuenciamiento de tareas, diseño de redes capacitadas multiproducto, problemas de ruteo de arcos, entre otros. Una amplia compilación de trabajos relacionados puede ser encontrada en Martí y Laguna (2003).

1.4.5 Búsqueda Tabú

La Búsqueda Tabú, (Tabu Search, TS, en inglés) es una técnica para resolver problemas combinatorios de gran dificultad que está basada en principios generales de Inteligencia Artificial (IA). Tiene sus antecedentes en métodos diseñados para cruzar fronteras de factibilidad u optimalidad local tratadas como barreras en métodos clásicos para permitir la exploración de regiones no consideradas en otro caso. El nombre y la metodología fueron introducidos por Fred Glover (1989), quién la define como la guía de un procedimiento de búsqueda local para explorar el espacio de soluciones más allá del óptimo local.

TS es un procedimiento metaheurístico cuya característica es el uso de memoria adaptativa y el uso de estrategias inteligentes para la resolución de problemas, basadas en procedimientos implícitos y explícitos de aprendizaje. La memoria adaptativa hace referencia a cuatro dimensiones principales, consistentes en la propiedad de hechos recientes, frecuencia, calidad e influencia, las cuales serán tratadas con más detalles en los capítulos posteriores. Algunas aplicaciones de búsqueda tabú son: localización de plantas, diseño de redes de transporte, ruteo de vehículos, por mencionar algunos. Una amplia compilación de trabajos pueden ser encontradas en Díaz et al. (1996).

1.5 Objetivo de la Tesis

El problema en estudio es un problema de optimización, en el cual se determina qué aristas deben considerarse en el diseño de forma que se garantice la operación de la red y que el costo total en que se incurra (considerando costos de diseño y de operación) sea el menor posible. Nuestro objetivo es presentar un procedimiento eficiente que encuentre buenas soluciones a instancias de tamaños reales para dicho problema.

Los objetivos de la tesis se resumen en los siguientes puntos:

- Realizar un estudio de la estructura matemática del problema de diseño de red multiproducto con capacidades en las aristas.

- Investigar y desarrollar técnicas inteligentes de búsqueda local aplicadas a resolver el problema de diseño de red capacitada.
- Desarrollo e implementación de un sistema de optimización basado en técnicas inteligentes de búsqueda local para resolver problemas de diseño de red capacitada.
- Comparar los resultados obtenidos en este trabajo con los obtenidos con un algoritmo evolutivo (Búsqueda Dispersa).

1.6 Alcance del Trabajo

Con la realización del presente trabajo se entrega un sistema computacional basado en la metodología propuesta para encontrar soluciones al problema de diseño de red multiproducto con capacidades en las aristas y costos fijos. Los programas que constituyen el sistema proporcionan soluciones de calidad, fueron desarrollados en lenguaje C de programación y fueron ejecutados en una terminal SUN™ Ultra 10, con sistema operativo Solaris™ versión 7.

Para su ejecución se requiere de archivos de formato específico para la lectura de las instancias de red, lo cual se explicará en los capítulos relacionado a los resultados computacionales, así como la implementación del GRASP, para la obtención de soluciones iniciales.

1.7 Organización de la Tesis

Este trabajo está organizado de la siguiente forma: En el Capítulo 2, se presenta la descripción del problema, así como el modelo matemático y sus características. En el Capítulo 3, se ve la descripción de la metaheurística Búsqueda Tabú, su evolución, sus elementos, características y algunas aplicaciones. En el Capítulo 4, se describe la metodología de solución. En el Capítulo 5, se presenta el diseño del experimento computacional, los parámetros y su calibración, así como la implementación computacional. En el Capítulo 6, se evalúa el desempeño del método y se analizan sus resultados computacionales. Finalmente, en el Capítulo 7, se concluye con las aportaciones científicas de nuestro trabajo y las recomendaciones para trabajos posteriores.

CAPÍTULO 2

DESCRIPCIÓN DEL PROBLEMA

Como ha sido mencionado, el problema que nos ocupa consiste en determinar qué aristas deben considerarse en el diseño de la red de forma que se garantice la operación de la misma sin exceder la capacidad de cada arista y minimizando el costo total en que se incurra (considerando costos de diseño y de operación). En este capítulo se describirá matemáticamente dicho problema.

2.1 Formulación matemática

Existen dos formulaciones para este problema, una basada en aristas y otra basada en caminos y se describen enseguida. Ambas formulaciones son equivalentes, sin embargo en el resto del trabajo decidimos utilizar la formulación basada en caminos, ya que es más natural en problemas con estructura específica por producto.

A continuación describimos los parámetros utilizados en ambas formulaciones.

Sea $G = (N, A)$ un grafo que representa una red no orientada con un conjunto N de nodos, un conjunto A de aristas potenciales y el conjunto A' de arcos potenciales asociados a esas aristas. Sea K el conjunto de productos con demanda d^k para el k-ésimo producto. Del conjunto de nodos se distinguirán varias parejas origen-destino, asociadas cada una de ellas a un producto que circulará por la red. Sean O(k) y D(k) el nodo origen y nodo destino respectivamente del k-ésimo producto.

Por otra parte, a cada arista potencial $\{i, j\}$ se le asigna un costo fijo F_{ij} por su utilización o construcción, así como costos c_{ij}^k y c_{ji}^k por unidad de producto transportado, el cual depende del tipo de producto de que se trate y del sentido en que circule por la arista. Cada arista $\{i, j\}$ tiene asociada una capacidad finita u_{ij}, la cual deberá ser compartida por todos los productos que circulen en cualquier dirección de la

misma.

2.1.1 Formulación basada en aristas

El modelo tiene dos tipos de variables de decisión. El primer tipo es una variable binaria que modela las elecciones de diseño y se define como $y_{ij} = 1$, si la arista {i, j} se incluye en el diseño de la red, o bien, $y_{ij} = 0$, en caso contrario. El segundo tipo es una variable continua que modela las decisiones de flujo del producto k que circula por el arco orientado (i, j) y que denotaremos como x_{ij}^k.

Para cada producto $k \in K$ y cada nodo $i \in N$ impondremos las restricciones usuales de conservación de flujo en redes.

$$\sum_{\{j:(i,j)\in A'\}} x_{ij}^k - \sum_{\{j:(j,i)\in A'\}} x_{ji}^k = \begin{cases} d^k & \text{si } i = O(k) \\ -d^k & \text{si } i = D(k) \\ 0 & \text{en otro caso} \end{cases} \qquad \forall\ k \in K, \forall\, i \in N \tag{1}$$

También consideraremos que el flujo de todos los productos que circulan en cualquier dirección por cada arista $\{i, j\}$ no debe exceder la capacidad de dicha arista.

$$\sum_{k\in K} \left(x_{ij}^k + x_{ji}^k\right) \le u_{ij} y_{ij} \qquad \forall\ \{i, j\} \in A \tag{2}$$

Estas restricciones no solo aseguran que sean respetadas las capacidades de las aristas, sino también fuerzan a que el flujo de cualquier producto x_{ij}^k sea cero si la arista $\{i, j\}$ no ha sido seleccionada en el diseño.

Como ya habíamos dicho, exigimos que las variables continuas sean no negativas y que las variables de diseño sean binarias.

$$\begin{aligned} x_{ij}^k &\ge 0 && \forall\, k \in K;\ \forall\, (i, j) \in A' \\ y_{ij} &\in \{0,1\} && \forall\, \{i, j\} \in A \end{aligned} \tag{3}$$

Nuestro objetivo será minimizar el costo total en que se incurre por diseño y operación de la red, esto es,

$$\min\left[\sum_{k\in K}\sum_{(i,j)\in A'} c_{ij}^k x_{ij}^k + \sum_{\{i,j\}\in A} F_{ij} y_{ij}\right] \tag{4}$$

Las decisiones a tomar consisten en la selección de las aristas que deben incluirse

en el diseño final de la red esto es, los valores de las variables y_{ij} y los volúmenes de flujo de cada producto que circularán por cada una de las aristas incluidas en sus dos direcciones (x_{ij}^k, x_{ji}^k) para satisfacer las demandas.

2.1.2 Formulación basada en caminos

Un camino $\bar{u}$ de un grafo no orientado es una sucesión de nodos de la forma $\bar{u}$ = (ni_1, ni_2,..., ni_p,..., ni_q), donde el par (ni_p, ni_{p+1}) pertenece al conjunto de arcos A' para todo p=1, 2,1 ..., q-1, de tal manera

Además de las notaciones introducidas previamente, representamos por L^k el conjunto de caminos que pueden ser utilizados para transportar la demanda del producto $k \in K$. Asimismo δ_{ijl}^k será igual a 1 si el arco (i,j) pertenece al camino l del producto k y será 0 en caso contrario, mientras que C_l^k denotará el costo por transportar una unidad de producto k en el camino $l \in L^k$ y se obtiene mediante: $C_l^k = \sum_{(ij)\in A'} c_{ij}^k \delta_{ijl}^k$.

Las decisiones consisten en la selección de aristas (variable binaria) para el diseño final de la red (y_{ij} = 1, si la arista $\{i,j\}$ se incluye en el diseño de la red, o bien, y_{ij} = 0, en caso contrario) así como los volúmenes de flujo por los caminos para satisfacer la demanda: h_l^k denotará el flujo del producto k en el camino $l \in L^k$. El flujo correspondiente del producto k en el arco (i,j) se denota como x_{ij}^k y se define mediante: $x_{ij}^k = \sum_{l\in L^k} h_l^k \delta_{ijl}^k$.

Con las notaciones introducidas, la formulación basada en caminos para el problema de diseño de red capacitada multiproducto es la siguiente:

$$\min z(h,y) = \sum_{\{i,j\}\in A} F_{ij} y_{ij} + \sum_{k\in K}\sum_{l\in L^k} C_l^k h_l^k \qquad (5)$$

$$\sum_{l\in L^k} h_l^k = d^k \quad \forall k \in K \qquad (6)$$

$$\sum_{k\in K}\sum_{l\in L^k} h_l^k (\delta_{ijl}^k + \delta_{jil}^k) \le u_{ij} y_{ij} \quad \forall \{i,j\} \in A \qquad (7)$$

$$h_l^k \ge 0 \qquad \forall k \in K, l \in L^k \qquad (8)$$

$$y_{ij} \in \{0,1\} \; \forall \{i,j\} \in A \qquad (9)$$

Al problema (5) - (9) lo referiremos como diseño de redes capacitadas multiproducto (DRCM).

El objetivo (5) captura la relación entre el costo de transporte del producto k en el camino l y el costo fijo de las aristas de la red. La restricción (6) es una ecuación de conservación de flujo impuesta para cada producto. La restricción (7) indica que no puede excederse la capacidad en las aristas y prohíbe el flujo en las aristas al no incluirse en el diseño ($y_{ij} = 0$). La restricción (8) asegura la no negatividad de las variables continuas h_l^k y la restricción (9) fuerza a las variables discretas y_{ij} a asumir valores binarios.

2.2 Características del problema

Es bueno destacar aquí algunos aspectos esenciales del problema:

- Es importante distinguir en el modelo que, una vez que se haya decidido conectar dos nodos i, j, entonces, la arista $\{i, j\}$ formará parte del diseño de la red, por lo que se permitirá flujo en ambos sentidos, o sea, se considerarán en la red ambos arcos (i, j) y (j, i).
- Una vez que una arista ha sido incluida en el diseño de la red, los costos variables de transportación asociados a cada arista dependerán no solamente del producto sino también del sentido en que esté circulando.
- Cada arista posee una capacidad finita que será compartida por todos los productos que la usen sin importar la dirección del flujo.
- El problema dado es un programa lineal entero mixto y es bueno señalar que para un conjunto de valores de las variables de diseño (aristas que se van a usar para un diseño dado) la formulación del problema de diseño se convierte en un problema de flujos a costo mínimo en redes multiproducto con capacidades en las aristas.

2.3 Retos y Dificultades

La gran dificultad de este problema estriba en la interacción entre costos fijos y variables al construir una solución, así como la relación entre la capacidad finita de las aristas (compartida por todos los productos que tratan de circular en la red) y los costos

fijos de esas aristas. Estos aspectos elevan enormemente la dificultad cuando se intenta resolver instancias de tamaño real.

La mayoría de las técnicas de optimización parten de una solución inicial ó conjunto de soluciones iniciales que van mejorando iterativamente, pero hay que señalar que para este problema, identificar una solución inicial factible es en sí un reto. Por ejemplo, para algunas instancias de mediano tamaño (30 nodos, 350 aristas y 100 productos), luego de 6 horas, el CPLEX no pudo encontrar siquiera una solución factible.

De allí la necesidad de utilizar algoritmos de aproximación llamados heurísticas, que si bien no garantizan encontrar una solución exacta u óptima, ofrezcan una solución suficientemente buena (calidad aceptable) para tamaño reales de instancias en un tiempo razonable de forma relativamente sencilla y rápida.

CAPÍTULO 3

DESCRIPCIÓN DE LA METAHEURÍSTICA BÚSQUEDA TABÚ

3.1 Introducción

Actualmente no existen algoritmos exactos que puedan resolver problemas similares al aquí abordado en un período de tiempo razonable, especialmente para instancias grandes. De ahí que otro tipo de técnicas, tales como las heurísticas, deban emplearse, si bien no para encontrar la solución óptima, lo cual es ciertamente difícil, al menos para encontrar soluciones de calidad aceptable.

Como ha sido mencionado, la tendencia actual es desarrollar métodos generales para resolver clases o categorías de problemas, conocidos como procedimiento metaheurísticos. Estos algoritmos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos clásicos no son ni efectivos ni eficientes. Los metaheurísticos proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de la inteligencia artificial, evolución biológica y mecanismos estadísticos (Osman y Kelly, 1996).

Para dar solución a nuestro problema diseñaremos un procedimiento que combina las estrategias de Búsqueda Tabú con movimientos tipo pivoteo del método simplex, por lo que este capítulo se dedicará a describir las principales características de dicha metaheurística.

Búsqueda Tabú (tabú search, TS, en inglés) ha sido utilizada extensamente para resolver exitosamente o encontrar mejores soluciones a una diversidad de problemas de toma de decisiones y optimización, como por ejemplo en ruteo y distribución (Rochat y Semet, 1994; Gendreau, Laporte y Potvin, 1995) y telecomunicaciones (Skorin-Kapov y Skorin-Kapov,1994; Costomagna, Fanni y Giacinto, 1995), entre otras. El éxito de esta

metodología en diversos problemas de toma de decisiones se debe a la exploración inteligente de sus estructuras de memoria y al uso de estrategias de intensificación y diversificación.

Dado que se empleó con éxito en problemas similares de diseño de redes pero sobre redes orientadas (Crainic, Gendreau, Farvolden, 2000), es que se decidió utilizar y adaptar algunas de estas ideas al problema que nos ocupa.

3.2 Metaheurística Búsqueda Tabú

3.2.1 Evolución Histórica

Los orígenes de Búsqueda Tabú pueden situarse en diversos trabajos publicados hace alrededor de 20 años. Oficialmente, el nombre y la metodología fueron introducidos posteriormente por Fred Glover (1989). Numerosas aplicaciones han aparecido en la literatura, así como artículos y libros para difundir el conocimiento teórico del procedimiento (Glover y Laguna, 1997).

Las etapas de Búsqueda Tabú están marcadas por los siguientes cuatro principales desarrollos:

- Estrategias que combinen reglas de decisión basadas en búsquedas de re-estructuración lógica y no-monótona (variable a profundidad)
- Violación sistemática y restauración de factibilidad
- Memoria flexible basada en recencia y frecuencia
- Proceso selectivo de combinación de soluciones, aplicado en población mantenida sistemáticamente.

El primero de estos desarrollos proviene de un estudio de reglas de decisión para problemas de secuenciamiento de trabajos. Fisher y Thompson (1963) introducen una innovación de alternar entre múltiples reglas en cada nodo de decisión mediante una estrategia probabilística, las cuales utilizaron aprendizaje reforzado para corregir las probabilidades de elección de las reglas, de acuerdo a la calidad de secuenciamientos producidos mediante múltiples ejecuciones. Esta metodología de buscar beneficio a partir de múltiples reglas mediante una alternancia inteligente de ellas motivó la consideración de una estrategia de contraste (Glover, 1963) la cual explota una colección

de reglas de decisión estableciendo una forma de combinarlas para crear otras nuevas.

El segundo desarrollo, cuyas ideas han quedado estampadas en las estrategias asociadas a TS, fue enmarcado en un método para resolver problemas de programación entera. El método dota a cada variable con su propia memoria como una base para crear restricciones improductivas. Siguiendo la terminología popularizada por Papadimitriou y Steiglitz (1982), la metodología constituye un caso de lo que recientemente se ha dado en llamar "método de variable en profundidad".

El tercer desarrollo igualmente involucra una metodología exacta de problemas de programación entera. En este caso el procedimiento fue basado en una extensión del método simplex para programación lineal (PL), usando procesos de planos cortantes para introducir nuevas restricciones implicadas por los requerimientos enteros de las variables. El diseño típico de PL subyacente en las metodologías de planos cortantes de este tipo, era comenzar a partir de un punto inicial factible (en sentido primal o dual) y después mantener esta condición de factibilidad mientras progresa monótonamente a una solución óptima. Sin embargo, la optimización combinatoria presenta diferentes retos topológicos que la optimización lineal, dando lugares a regiones primales y duales que son no-convexas e inclusive discontinuas. Este hecho motivó la creación de un método pseudo primal-dual (Glover, 1968) que propone violar y luego restaurar las condiciones de factibilidad, haciendo posible atravesar varias regiones del espacio de solución. La estrategia de visitar tanto regiones factibles como infactibles en "ondas" sucesivas se ha convertido en un rasgo distintivo de una de las principales estrategias del TS.

El último de los desarrollos involucra la introducción de métodos de agregación de restricciones para programación entera (Glover, 1997). Tales métodos fueron basados en la estrategia de combinación de restricciones que producen nuevas, con el objetivo de producir información que no se obtiene con las restricciones originales. La función que captura la información rápidamente condujo al uso de memoria basada en hechos recientes y frecuencia, como un medio para extraer la información y como una manera de explotarla dentro de la heurística y los procesos algorítmicos.

3.2.2 Generalidades

Búsqueda Tabú es una técnica o procedimiento metaheurístico usado para resolver problemas de optimización combinatoria, utilizado para guiar cualquier procedimiento de búsqueda local para explorar el espacio de soluciones más allá de la simple optimalidad local.

El procedimiento local es una búsqueda cuya operación (llamada movimiento, usada para definir la vecindad de cualquier solución dada) pueda caracterizarse como una sucesión de movimientos que llevan al procedimiento de un estado (solución) a otro. El procedimiento examina soluciones "vecinas" y se mueve o avanza de forma inteligente, hacia uno de estos vecinos que provea una mejora en la solución objetivo. Este procedimiento se repite hasta que ya no sea posible mejorar una solución con respecto a las soluciones vecinas, entonces decimos que el procedimiento ha encontrado un óptimo local.

TS está basada en principios generales de Inteligencia Artificial (IA) y puede ser utilizado para guiar cualquier procedimiento de búsqueda local en la búsqueda agresiva del óptimo del problema. Por agresiva nos referimos a la estrategia de evitar que la búsqueda quede "atrapada" en un óptimo local que no sea global. A tal efecto, TS toma de la IA el concepto de memoria y lo implementa mediante estructuras simples con el objetivo de dirigir la búsqueda teniendo en cuenta la historia de ésta. Es decir, el procedimiento trata de extraer información de lo sucedido y actuar en consecuencia. En este sentido puede decirse que hay un cierto aprendizaje y que la búsqueda es inteligente.

El éxito de esta metodología en diversos problemas se debe a sus estructuras de memoria y al uso de estrategias de intensificación y diversificación. Las estructuras de memoria evitan retornar a soluciones visitadas anteriormente, permiten guardar atributos de buenas soluciones contribuyendo a identificar regiones de interés y más generalmente guiar la exploración del espacio de solución. Las estrategias de intensificación y diversificación permiten avanzar a una solución vecina que es peor que la solución actual, pero que proporciona la posibilidad de penetrar a un espacio del conjunto de soluciones factibles que de otro modo no habría sido visitado y que podría contener una

solución óptima global al problema.

TS considera dos tipos de memoria que interactúan entre sí, aunque en horizontes diferentes, una a corto y otra a largo plazo, pudiendo ser éstas del tipo de frecuencia o recencia. La memoria a largo plazo tiene dos componentes que son las estrategias de intensificación y diversificación, aplicadas dentro de un ámbito global o local. El objetivo es lograr el óptimo global, evitando con ello que el algoritmo se estanque en un óptimo local.

TS se basa en la premisa de que para poder calificar de inteligente la resolución de un problema, debe incorporar memoria adaptativa (es decir, vecindades no estáticas) y exploración sensible o responsiva, que son las características principales de la búsqueda tabú.

La memoria adaptativa (MA) nos permite recordar selectivamente elementos fundamentales del camino atravesado, lo cual crea un comportamiento más flexible de la búsqueda y permite la implementación de procedimientos que sean capaces de explorar el espacio de soluciones efectiva y económicamente.

La exploración sensible (ES) nos permite hacer elecciones estratégicas a través del camino. El énfasis se deriva de la suposición de que una mala elección estratégica puede producir más información que una buena elección al azar. ES integra principios básicos de búsqueda inteligente, por ejemplo la exploración de características de buenas soluciones mientras se exploran nuevas regiones prometedoras.

Para guiar el procedimiento de búsqueda, TS se enfoca en imponer restricciones para negociar regiones que serían difíciles de explorar de otra manera. Estas restricciones operan en dos formas, mediante la exclusión directa de alternativas de búsqueda y mediante traslación a evaluaciones modificadas y probabilidad de selección. A continuación se describirán más detalladamente los elementos básicos de la metaheurística Búsqueda Tabú.

3.2.3 Descripción del Método de Búsqueda Tabú

El método de TS comienza de la misma forma que cualquier procedimiento de búsqueda local. Utilicemos S para denotar el conjunto de soluciones posibles del problema, donde cada solución $s \in S$ tiene un conjunto de soluciones asociadas, que

denominaremos entorno de s y denotaremos N(s). El algoritmo de búsqueda local empieza con alguna solución inicial y va ejecutando movimientos iterativos de una solución $s \in S$ a otra en el entorno de la primera, mientras se decrementa el valor de la función objetivo. (Por convención se consideran problemas cuya función objetivo esta en dirección de minimizar).

Sin embargo, en lugar de considerar todo el entorno de una solución $s \in S$, TS define el entorno reducido N*(s) como aquellas soluciones disponibles del entorno de s (entorno usual eliminando en la k-ésima iteración a los elementos de un conjunto tabú denotado como T(s, k)). Así, se considera que a partir de s sólo las soluciones del entorno reducido son alcanzables y se obtienen a partir de s mediante una operación llamada movimiento.

Estas soluciones s' obtenidas mediante el entorno reducido N*(s) o soluciones modificadas son llamadas vecinos de s. Dado que el objetivo central de la estrategia es escapar del óptimo local, se permite que la búsqueda explore soluciones que no decrementen el valor de la función objetivo, pero solamente en aquellos casos donde las soluciones no sean prohibidas. Esto usualmente se obtiene guardando vestigios de las últimas soluciones en términos del mecanismo usado para transformar una solución a la siguiente. Cuando la operación o el movimiento es desarrollado, esta es considerada tabú para las siguientes K iteraciones, donde K es una distancia o duración de tiempo de estado tabú.

Así, el mejor movimiento se ejecuta y la estructura de datos tabú se actualiza y el procedimiento se repite hasta alcanzar el criterio de terminación.

A continuación se muestra un pseudo-código que refleja lo anteriormente descrito.

```
k = 1
Genera solución inicial
WHILE criterio de terminación no se alcance DO
   Identifique N(s). (Vecindad)
   Identifique T(s, k). (Conj. Tabú)
   Escoja la mejor s' ∈ N(s, k) = N(s) \ T(s, k)
   Memorice s' si mejora la anterior mejor solución conocida
   s: = s'
   k: = k+1
END WHILE
```

En los siguientes párrafos se irán explicando en detalle los elementos aquí mencionados.

3.2.4 Elementos Básicos de Búsqueda Tabú

Las estructuras de memoria operan basándose en 4 dimensiones principales o elementos básicos: hechos recientes, frecuencia, calidad e influencia.

Las estructuras de memoria basadas en hechos recientes son usadas regularmente para identificar atributos activos, y para determinar el status tabú de las soluciones que contienen esos atributos.

La calidad se refiere a la habilidad de diferenciar el mérito de las soluciones visitadas durante la búsqueda, es decir es usada para identificar elementos que son comunes a las buenas soluciones o caminos que conducen a tales soluciones.

La influencia considera el impacto de las elecciones hechas durante la búsqueda, no solo en cuanto a calidad sino también en estructura para su aprendizaje.

La frecuencia mantiene un registro de los atributos de las soluciones visitadas, tratando de identificar o diferenciar regiones.

3.2.4.1 Uso de Memoria

La memoria usada en TS puede ser explícita o basada en atributos.

La memoria explícita guarda soluciones completas y consiste en una élite de soluciones visitadas durante la búsqueda o entornos altamente atractivos pero inexplorados para tales soluciones.

La memoria atributiva, con el propósito de guiar, guarda información sobre atributos de las soluciones que cambian al moverse de una solución a otra.

La memoria explícita y la atributiva son complementarias: mientras la memoria explícita expande la vecindad durante la búsqueda local (incluyendo soluciones élites), la memoria atributiva la reduce, mediante la prohibición de ciertos movimientos mediante la selectividad.

Una distinción importante en TS resulta al diferenciar entre la memoria de corto y largo plazo. Ambos tipos de memoria llevan asociadas sus propias estrategias y atributos, y actúan en ámbitos diferentes.

El efecto de ambos tipos de memoria es la modificación de la vecindad de la solución actual. La vecindad modificada es el resultado de mantener una historia selectiva de los estados encontrados durante la búsqueda

3.2.4.2 Memoria de Corto Plazo y sus Elementos

La memoria de corto plazo está *basada en atributos*, es decir almacena atributos o lleva la cuenta de soluciones recientemente visitadas (que han sido cambiadas en el pasado reciente) y su objetivo es explorar a fondo una región dada del espacio de soluciones. Esta es llamada memoria basada en hechos recientes.

Para explotar esta memoria los atributos seleccionados que se presentan en soluciones recientemente visitadas son designados como tabú activos, y las soluciones que contienen estos elementos o combinaciones particulares de estos atributos, son las que se convierten en tabú.

El objetivo principal de etiquetar las soluciones visitadas como tabú es evitar que la búsqueda se cicle o sea, que soluciones recientes o ciertas soluciones del pasado reciente se vuelvan a visitar. Por ello se considera que tras un cierto número de iteraciones la búsqueda está en una región distinta y puede liberarse del estatus tabú a las soluciones antiguas. De esta forma se reduce el esfuerzo computacional de calcular el entorno reducido en cada iteración. La clasificación tabú sirve para identificar elementos de una vecindad excluidos de una vecindad modificada.

El proceso se maneja creando una o más listas tabú, las cuales registran los atributos tabú activos, y explícita e implícitamente identifican su estado actual. Se denomina tenencia tabú (TT) a la duración que un atributo permanece tabú-activo (medido en números de iteraciones T). La TT puede variar para diferentes tipos o combinaciones de atributos y para intervalos de tiempo o etapas de búsqueda. En general las TT cortas permiten la exploración de soluciones cercanas a un óptimo local, mientras que las TT largas ayudan a salirse de la vecindad de un óptimo local.

En los orígenes de la búsqueda tabú se sugería tenencias tabú de tamaño pequeño, actualmente se considera que estas pueden ajustarse dinámicamente según la estrategia que se esté utilizando.

3.2.4.3 Niveles de Aspiración

Los niveles de aspiración son introducidos para determinar cuándo pueden ser ignoradas las reglas de activación tabú, de esta forma se introduce cierta flexibilidad en la búsqueda y se mantiene su carácter agresivo.

En situaciones donde la vecindad es muy pequeña o la tenencia tabú es muy grande es posible que ocurran iteraciones donde todos los movimientos disponibles están clasificados tabú. En este caso se usa el criterio de aspiración por default, para permitir un movimiento con el menor status tabú. Otro criterio utilizado es eliminar el estatus tabú a una solución si su valor objetivo es mejor que cualquiera de las encontradas anteriormente.

3.2.4.4 Estrategias para la Lista Candidata (ELC)

Búsqueda Tabú elige reglas que busquen el mejor movimiento disponible. Para situaciones donde la vecindad modificada es grande o sus elementos son costosos de evaluar, son esenciales estrategias de lista candidata para restringir el número de soluciones examinadas en una iteración dada.

Una consideración importante para TS es determinar una estrategia para la lista candidata apropiada para estrechar el examen de elementos de la vecindad, con el objetivo de alcanzar un acuerdo efectivo entre la calidad de una solución y el esfuerzo consumado en encontrarla. Algunas de las clases generales de estas estrategias son:

- *Aspiración Plus* establece el umbral para la calidad de un movimiento. Una vez que encuentra un movimiento que satisface este umbral, examina "plus" movimientos adicionales.
- *Lista candidata élite* construye una lista maestra (LM) examinando todos los movimientos o un número relativamente grande de ellos y eligiendo los k mejores encontrados. Entonces en cada iteración subsiguiente, el mejor movimiento de la LM se ejecuta, continuando así hasta que el mejor movimiento caiga por debajo de un umbral de calidad o hasta un número de iteraciones dado.

3.2.4.5 Memoria de Largo Plazo y sus Elementos

El uso de memoria de largo plazo no requiere secuencias de larga duración, sus mejoras comienzan a manifestarse en un lapso de tiempo relativamente corto, debido a

que se encuentran soluciones de muy alta calidad dentro de un rango corto de tiempo.

La memoria a largo plazo almacena las frecuencias u ocurrencias de atributos en las soluciones visitadas tratando de identificar o diferenciar regiones.

La memoria de largo plazo y sus estrategias asociadas hacen significativamente más potente a TS. Tipos especiales de *memoria basada en frecuencia* son fundamentales en consideraciones de largo plazo y operan introduciendo penalizaciones o incentivos determinados por el rango relativo de tiempo en el que los atributos han permanecido en soluciones visitadas durante la búsqueda. Las *frecuencias de transición* mantienen un registro de con qué frecuencia cambian los atributos, mientras que las *frecuencias de residencia* mantienen el registro de las duraciones relativas de los atributos en las soluciones generadas.

La memoria a largo plazo tiene dos componentes importantes o estrategias asociadas: intensificar y diversificar la búsqueda.

Las estrategias de intensificación están basadas en modificar las reglas de selección de tal manera que favorezcan: combinación de movimientos y características de soluciones que han sido históricamente buenas. Estas estrategias pueden iniciar un retorno a regiones atractivas para buscar en ellas más detenidamente, es decir, consiste en regresar a regiones ya exploradas para estudiarlas más a fondo. Para ello se favorece la aparición de aquellos atributos asociados a buenas soluciones encontradas. Ya que las soluciones élites tienen que ser almacenadas con el objetivo de examinar sus vecindades inmediatas, la memoria explícita está estrechamente relacionada con la implementación de estrategias de intensificación.

La estrategia de diversificación, por su parte, estimula el proceso de búsqueda a examinar regiones no visitadas y generar soluciones que difieran en diversas formas significativas de las que fueron visitadas anteriormente.

Las estrategias de diversificación están diseñadas para conducir la búsqueda hacia nuevas regiones, es decir, nuevas áreas no exploradas del espacio de soluciones. Con frecuencia están basadas en modificar las reglas de elección para llevar a la solución atributos que no hayan sido usados frecuentemente. Las estrategias de diversificación crean reinicios parciales o completos de la búsqueda.

3.2.4.6 Oscilación Estratégica

La oscilación estratégica opera orientando los movimientos en relación a un cierto nivel crítico. Tal nivel crítico (frontera de oscilación) representa a menudo un punto donde el método se detendría normalmente. Sin embargo, en vez de detenerse al alcanzar este nivel, las reglas para elegir los movimientos se modifican para permitir que la región al otro lado de la frontera sea alcanzada. Se prosigue en una profundidad especificada más allá de la frontera de oscilación y se regresa. La búsqueda ahora procederá a alcanzar nuevamente la frontera de oscilación y traspasarlo, sólo que esta vez se hará en dirección opuesta a la que se hizo anteriormente, y el método se dirige a un nuevo punto de retorno.

El proceso de aproximarse, traspasar y volver sobre una determinada frontera crea un comportamiento oscilatorio, el cual da nombre a esta técnica. La oscilación estratégica proporciona un medio adicional para lograr una interacción muy efectiva entre intensificación y diversificación.

3.2.4.7 Reencadenamineto de Trayectorias

El reencadenamiento de trayectorias ha sido concebido como una forma de integrar las estrategias de intensificación y diversificación. Genera nuevas soluciones mediante la exploración de trayectorias que conectan soluciones élites, iniciando a partir de una de ellas, llamada solución iniciadora y generando un camino en el espacio de vecindad que conduzca a las otras soluciones o soluciones guías. Esto se logra mediante la selección de movimientos que introduzcan atributos contenidos en las soluciones guías. La motivación de esto es el hecho de que una trayectoria entre soluciones en un espacio de vecindad produce nuevas soluciones que comparten un subconjunto significativo de atributos contenidos en la solución iniciadora y la solución guía.

3.2.5 Algunas aplicaciones de Búsqueda Tabú

A continuación se mencionan algunos campos y sus respectivas aplicaciones (Díaz et al., 1996).

Optimización combinatoria en general	**Secuenciamiento**
Optimización de carga fija	Planeación
Optimización Entera Mixta	Horarios escolares

Lógica e Inteligencia Artificial
Lógica Probabilística
Diseño y Entrenamiento de redes neuronales

Ruteo
Ruteo con ventanas de tiempo
Problema del agente viajero

Diseño
Diseño de redes de transporte
Distribución en planta

Tecnología
Distribución de energía eléctrica
Construcción de estaciones espaciales

Telecomunicaciones
Ruteo de llamadas
Asignación de caminos

Localización
Multiproducto
Asignación cuadrática

CAPÍTULO 4

METODOLOGÍA DE SOLUCIÓN PARA EL PROBLEMA DRCM

4.1 Introducción

La metodología de solución propuesta combina la metaheurística Búsqueda Tabú, con el método del simplex revisado, para producir una búsqueda que explore el espacio de soluciones (*variables de flujo camino*) mediante movimientos que consisten en pivoteos del simplex revisado.

Esta metodología está basada en el algoritmo propuesto por Crainic Gendreau y Farvolden (2000) para un problema de diseño similar sobre redes orientadas.

4.2 Descripción de la metodología

La idea fundamental de la metodología propuesta por Crainic es utilizar una búsqueda tabú para explorar el espacio de solución (conjunto de puntos extremos del polihedro definido por las restricciones (6) – (8)) donde el algoritmo simplex revisado ofrece la estructura dentro de la cual se construye y explora la vecindad, así como se evalúan, seleccionan e implementan los movimientos.

La búsqueda tabú procede de la siguiente forma: luego de una fase de inicialización se prosigue con una secuencia de búsquedas locales y fases de diversificación hasta que se satisfaga un criterio de parada previamente determinado. Por su parte, cada búsqueda local está compuesta por varias series alternadas de movimientos de pivoteo y fases de generación de caminos. Esquemáticamente el procedimiento puede resumirse en los siguientes pasos, los cuales serán explicados a continuación.

1. Obtener una solución inicial.
2. Realizar una búsqueda local.

3. Ejecutar movimientos de diversificación.
4. Repetir pasos 2 y 3 un número predefinido de veces.

4.2.1 Obtención de la solución inicial

La solución inicial (X, Y) donde $X = \left(x_{ij}^{k}\right)_{k \in K,(i,j) \in A}$, $Y = \left(y_{ij}\right)_{(i,j) \in A}$, se obtiene mediante un procedimiento especifico desarrollado a tal efecto. El procedimiento que se usa está basado en la técnica heurística del GRASP (De Alba, Álvarez, González-Velarde, 2003). Por consiguiente, esta solución inicial ya es relativamente buena.

4.2.2 Búsqueda local

Dada una solución factible (X, Y) es necesario definir la vecindad donde se realizará la búsqueda local. Esta vecindad, la cual denominaremos *vecindad continua*, consta de todas las soluciones (puntos extremos) que pueden alcanzarse a partir de la actual mediante un pivoteo simplex.

Consecuentemente, un *movimiento* local corresponde a una transición de una base del sistema (6) - (8) a una adyacente, o sea, un camino básico es sustituido por uno de los actualmente no básicos. Nótese que una vez que han sido fijadas las variables de diseño, el problema que queda es un problema de flujo multiproducto.

Sin embargo, dado que nuestro problema no se enmarca en un verdadero contexto de programación lineal, se definieron dos formulaciones auxiliares de problema de flujo multiproducto (Minimum Cost Network Flow, MCNF, en inglés), las cuales son: la componente lineal (esta es la componente continua, es decir la parte lineal de la verdadera función objetivo del problema de diseño) y la formulación linealizada (donde se incluye la razón de costo fijo por arista dividido entre la capacidad de dicha arista). Estas dos formulaciones son usadas en la fase de búsqueda local que explorarán el espacio de soluciones en una iteración dada, es decir, el espacio de las variables de flujo-camino de acuerdo a principios del simplex revisado.

- *Formulación Lineal*

$$\min z(h) = \sum_{k \in K} \sum_{l \in L^k} c_l^k h_l^k$$

- *Formulación Linealizada*

$$\min z(\tilde{h}) = \sum_{k \in K} \sum_{l \in L_l^k} \tilde{c}_l^k h_l^k$$

donde :

$$\tilde{c}_l^k = \sum_{\{i,j\} \in A} \delta_{ijl}^k (c_{ij}^k + f_{ij} / u_{ij})$$

La fase de búsqueda local del procedimiento de TS se describe a continuación en los siguientes pasos:

0. Inicializar listas tabú de corto plazo. Definir un conjunto de caminos candidatos (CC), Zmejor como el valor de la mejor solución encontrada hasta el momento, Zlocal= ∞ (valor de la solución actual), Zprevia = ∞ (valor de la solución anterior) y g = 0 (valor del contador de la fase de generación de caminos).
1. Repetir los pasos del 2 al 7 hasta un número máximo de movimientos (max_move) consecutivos sin mejora en el valor de la solución local (Zlocal).
2. Identificar la base actual y calcular las variables duales correspondientes (según función lineal). Pivotear variable de holgura con costo marginal negativo.
3. Calcular costos reducidos (CR) para los caminos no básicos.
4. Para cada variable flujo camino no-básico (camino de entrada potencial)
   a) Determinar el camino a salir (camino saliente potencial) y el valor asociado de la variable a entrar.
   b) Determinar la correspondiente modificación al vector de variables de diseño, si la hay, y calcular la variación asociada en el costo fijo total de la red.
   c) Calcular el valor del movimiento potencial como la variación en la función objetivo original del problema de diseño.
5. Identificar el "camino de entrada" como aquel para el cual se obtuvo el menor incremento en la función objetivo original.
6. Si el movimiento seleccionado es no-tabú, implementarlo (si es tabú implementarlo sólo si mejora Zmejor) y si no seleccionar el siguiente mejor movimiento candidato.
7. Establecer y actualizar las tenencias tabú; actualizar memoria a largo plazo; si

mejora Zlocal actualizarla y si se requiere Zmejor.

8. Si Zlocal < Zprevia entonces Zprevia = Zlocal y g = 0; Si no g = g+1 y si g >= max_cam_gen entonces SALIR. Donde max_cam_gen se refiere al número máximo de caminos generados.
9. Calcular los valores de las variables duales para la formulación linealizada. Generar nuevos caminos. Ir al paso 1.

Para decidir qué movimiento implementar, se determina, para cada posible camino a entrar a la base, el correspondiente camino básico a salir y se evalúa el "valor" de este movimiento potencial (mejoría de la función objetivo). Para calcular el valor de un movimiento potencial el procedimiento suma las variaciones en el valor de la función objetivo del problema de diseño inducidas por una parte, por la variación del flujo ocasionada por el pivoteo simplex, y por otra parte, por las correspondientes modificaciones al vector de diseño debido a cambios en la utilización de la red. Entonces, cualquier movimiento con "valor" negativo es estrictamente mejorador con respecto a la solución actual. Cuando no hay disponible ningún movimiento estrictamente mejorador, el procedimiento selecciona el movimiento con el mayor "mejoramiento", aún cuando esto signifique degradar la solución.

Utilizar la función objetivo original para evaluar los movimientos implica un costo extra en tiempo de evaluación, pero ofrece un ordenamiento preciso de los movimientos potenciales y un criterio de aspiración directo.

Como puede observarse en la descripción anterior, un elemento fundamental en el proceso son los caminos de la red. Recuerde que estamos trabajando con la formulación del modelo basada en caminos (5) - (9). Allí se denotó por L^k el conjunto de caminos potenciales para transportar la demanda del producto k.

Estos caminos serán calculados en la red que incluye todas las aristas potenciales. Como el objetivo final es un diseño de red con un costo total por construcción y operación mínima, se desea disponer de caminos atractivos en ese sentido. Por tal motivo se seleccionarán los q caminos más cortos para cada producto, pero para ello es necesario definir una longitud de arco que considere tanto el costo fijo de la arista correspondiente, como el costo de transportación.

Se consideraron las siguientes 3 longitudes de arco y para cada una de ellas se buscan *q/3* caminos. Cuando el parámetro q no es divisible por estas 3 tipos de longitudes, el residuo se agregara a la longitud del tercer tipo (ejm: si q=10, $long_1$=3, $long_2$=3 y $long_3$=4).

1. $$long_{ij}^{k} = \left(c_{ij}^{k} + F_{ij}\right)\left(1 + \left|\frac{u_{ij} - d^{k}}{d^{k}}\right|\right)$$

2. $$long_{ij}^{k} = \left(c_{ij}^{k} + F_{ij}\right)\left(1 + \left|\frac{u_{ij} - \sum_{h \in H_{ij}} d^{h}}{\sum_{h \in H_{ij}} d^{h}}\right|\right)$$

3. $$long_{ij}^{k} = \left(c_{ij}^{k} + F_{ij}\right)\left(1 + \frac{f_{ij}}{f'}\right)$$

La longitud de tipo 1, penaliza la desviación de la demanda del producto k respecto a la capacidad de la arista correspondiente a ese arco. La finalidad de esta longitud es hacer atractivos aquellos arcos cuya capacidad es empleada efectivamente, es decir sin desperdicio, al intentar transportar la demanda de un producto a través de él. Debido a esto, un arco cuya capacidad sea igual a la demanda de un producto no es penalizado.

En la longitud de tipo 2, H_{ij} denota el subconjunto de productos que comparten la arista $\{i, j\}$ (esto es el arco (i, j) o (j, i)) en aquellas rutas encontradas con la longitud tipo 1. En este caso, la penalización se hace sobre la desviación de la capacidad de la arista respecto a la suma de las demandas de aquellos productos que potencialmente pudieran ser transportados a través del arco (i, j). La finalidad de esta longitud es hacer atractivas aquellas aristas cuyo costo fijo sea distribuido lo mejor posible entre todos los productos que la usen.

Finalmente, en la longitud tipo 3, se ha considerado un factor de penalización para introducir cierta diversificación en el conjunto de rutas ya generadas. En este caso, f_{ij} representa la frecuencia de la arista $\{i, j\}$ en las rutas previamente encontradas, esto es, el número de veces que la arista $\{i, j\}$ ha aparecido en algunas otras rutas, mientras que f' representa la frecuencia máxima de aparición de una arista en las rutas ya encontradas.

Es importante hacer notar que la longitud de un arco puede variar de producto a producto. Las rutas obtenidas con las longitudes de tipo 1 y 2 se pueden considerar como rutas atractivas, mientras que las rutas encontradas con la longitud del tipo 3 pueden considerarse diversas respecto a las anteriores. Es conveniente hacer notar que para cada producto que se transportará en la red, q rutas cortas deberán ser generadas.

Es bueno señalar aquí que no todos los caminos están disponibles en cada iteración, sino que se van incorporando cada vez que se llega al paso 8.

Una secuencia de exploraciones locales de vecindad (mediante pivoteo) seguida por una fase de generación de caminos (paso 8) se denomina ciclo de generación de caminos. La búsqueda local termina cuando no se obtiene ninguna mejoría en la mejor solución local actual después de cierto número de ciclos consecutivos de generación de caminos. El método entonces para o procede a diversificar la búsqueda.

4.2.3 Fase de diversificación.

El objetivo de esta fase es sacar la búsqueda del aparente óptimo local hacia una región prometedora. Para ello se define la *vecindad discreta*, con relación a las variables de diseño y se usa para modificar drásticamente la configuración de la red y diversificar la búsqueda.

La estrategia de diversificación implementada está basada en la observación que un número de "buenas" aristas aparecen una y otra vez en los caminos usados para satisfacer la demanda. Por lo tanto se implementa una estructura de memoria de largo plazo basada en frecuencia, que registra durante cuántas iteraciones ha estado una arista en la base, esto es, por cuanto tiempo pertenece al menos a un camino básico.

Las aristas que tengan puntaje alto en esta memoria usualmente han sido utilizadas en las soluciones ya exploradas anteriormente. Por ello, para diversificar, uno selecciona un número pequeño de estas aristas y quita de la base cualquier camino que contenga al menos a uno de los arcos asociados a estas aristas. Durante la tenencia tabú de estas aristas, ningún camino que las contenga podrá entrar a la base, a menos que el criterio de aspiración ignore el estatus tabú. Además, las aristas cerradas no están disponibles durante la fase de generación de caminos.

4.3 Pseudo-código del algoritmo

A continuación presentaremos el pseudo-código de nuestro algoritmo:

- Lee los datos de la red y sus caminos
- Poblar el problema con los datos obtenidos
- Obtiene solución inicial proporcionada por el GRASP

```
/* PASO I. INICIALIZACION: */
cont = 0, num_iter = 0,
num_move = 0, g = 0, diver = 0, max_diver_val = 0;
Hacer{ /* CICLO DE DIVERSIFICACIÓN */
   Paso 1. INICIALIZA LISTA TABU DE CORTO PLAZO
   Caminos candidatos (CC), Zlocal = INF, Zprevia = INF, Zmejor,
   g = 0
   Hacer{ /* CICLO DE GENERACIÓN DE CAMINOS */
       num_move = 0;
       Hacer{  /* CICLO DE BUSQUEDA LOCAL */
         /* PASO II. BUSQUEDA LOCAL */
           Paso 2. PIVOTEA VARIABLE SLACK CON COSTO REDUCIDO NEG
                Si (Zactual < Zlocal) Zlocal = Zactual;
                Si (Zlocal < Zmejor) Zmejor = Zactual; num_move = 0;
           Paso 3. CALCULA COSTOS REDUCIDOS PARA TODOS LOS CAMINOS
                   NO BASICOS
           Paso 4. PARA CADA VAR CAMINO NO BASICO, DETERMINAR SU
                   VAR A SALIR Y SU VALOR ASOCIADO
           Paso 5. IDENTIFICA EL MOV POTENCIAL
           Paso 6. IMPLEMETA MOV. SI NO-TABU, SI NO, IMPLEMENTAR
                   SOLO SI MEJORA ZMEJOR
           Paso 7: ESTABLECE Y ACTUALIZA TENENCIAS TABU, ZLOCAL Y
                        ZMEJOR SI MEJORAN
                  Si ( Zactual < Zlocal)
                        Zlocal = Zactual;
                        num_iter++; num_move++;
                  Si (Zlocal < Zmejor)
                         Zmejor = Zactual;
                         num_move = 0;
          REPETIR Paso 2 AL 7 HASTA EJECUTAR MAX_MOVE MOVS
          CONSECUTIVOS SIN MEJORA EN EL VALOR DE ZLOCAL
       }Mientras(num_move <> max_move);
           Paso 8. SI ZLOCAL < ZPREV, ZPREV=ZLOCAL, g=0
                   SINO g= g+1,  Si ( g >= max_cam_gen) SALIR
           Paso 9: GENERA NUEVOS CAMINOS Y SE AGREGAN A CC Y
                   VUELVE AL Paso 1
   }Mientras(g <= max_cam_gen);
   Si ( diver <= max_diver){
   /* PASO III. FASE DE DIVERIFICACION */
   Si (factible = TRUE){
   Si ( Zactual < Zlocal) Zlocal = Zactual;
   Si (Zlocal < Zmejor)    Zmejor = Zactual; best_iter = num_iter; }
   diver++;
}Mientras(diver < max_diver);
```

4.4 Ejemplo ilustrativo del procedimiento de solución

Consideremos el ejemplo de la figura 1, para encontrar el mejor diseño de red tal que los costos de transportación y construcción sean mínimos. Esta red tiene 6 nodos numerados del 1 al 6 y 10 aristas potenciales. Se tienen dos productos con demanda de 20 unidades del nodo 1 al nodo 4 y 10 unidades del nodo 2 al nodo 6 respectivamente. Cada arista tiene asociado 4 números separados por coma: los dos primeros representan los costos por transportar (CV) una unidad del producto1 y del producto 2 respectivamente, el tercero representa el costo fijo (CF) que se pagaría si la arista es incluida en el diseño final de la red y el último número representa la capacidad (U) de la arista.

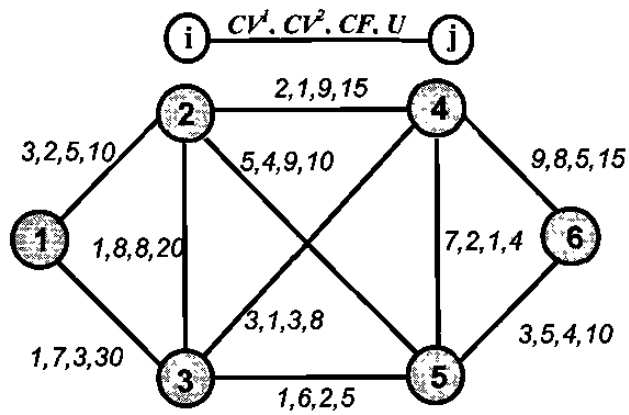


Figura 1. Ejemplo de un Diseño de Red

Con estos datos, se obtiene una solución ó diseño inicial mediante el GRASP, el cual se describe a continuación.

Prod	Camino	Nom Camino	Ruta	Flujo	CV
1					
	2	h12	1-3-4	8	32
	3	h13	1-2-4	10	50
	1	h11	1-3-2-4	2	8
2					
	3	h23	2-5-6	10	90

Costo de Solución: CV + CF = 180 + (3+8+9+3+5+9+4) = 221

La nomenclatura de los nombres de los caminos, se encuentra en el Apéndice B. La figura 2 muestra el diseño inicial apartir del cual va a iniciar nuestro algoritmo.

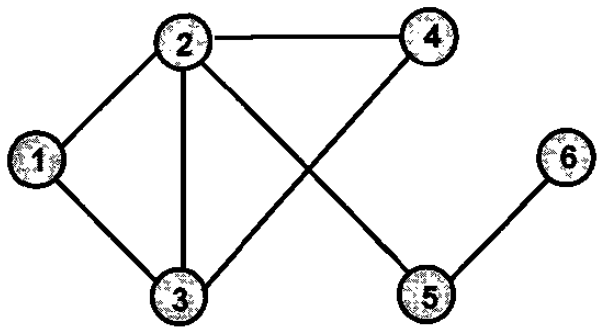


Figura 2. Diseño de red de la solución inicial

Una vez que ya se cuenta con una solución inicial factible, el siguiente paso consiste de una secuencia de búsquedas locales, compuesta por varias series alternadas de movimientos de pivoteo y fases de generación de caminos.

Parámetros predeterminados: max_move=2, max_cam_gen=1, max_div=1, tabu_val=3, el significado de estos parámetros se explican en el Apéndice D.

Zmejor = Infinito; div = 0;

Iter = 1, move = 1, g = 0, diver = 0;

II Fase. Búsqueda Local

(Move = 1, Iter = 1)

Paso 1: Zlocal = INF, Zprev = INF, g = 0, Zmejor = 221, CC (h11, h12, h13, h23)

Paso 2: Base (h11, h12, h13, h23)

	Cj	4	4	5	7	8	9	9	13	0	0	0	0	0	0	0	0	0	0
Cb	**Xb**	**B**	**h11**	**h12**	**h13**	**h14**	**h21**	**h22**	**h23**	**h24**	**S12**	**S13**	**S23**	**S24**	**S25**	**S34**	**S35**	**S45**	**S46**	**S56**
4	h11	2	1	0	0	-1	0	0	0	-1	-1	0	0	0	0	-1	0	0	0	0
4	h12	8	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0
5	**h13**	10	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
9	h23	10	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	S13	20	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	S23	18	0	0	0	2	0	0	0	1	1	1	0	0	0	1	0	0	0	0
0	S24	3	0	0	0	0	1	1	0	2	0	0	1	1	0	1	0	0	0	0
0	S25	0	0	0	0	0	-1	-1	0	-1	0	0	0	1	1	0	0	0	0	0
0	S35	5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	S45	4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	S46	15	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
0	S56	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	1
	Zj	180	4	4	5	5	9	9	9	9	1	0	0	0	0	0	0	0	0	0
	Cj - Zj		0	0	0	2	-1	0	0	4	-1	0	0	0	0	0	0	0	0	0

Zactual: CV + CF: 180 + (3+8+9+3+5+9+4) = 221

Pivotear variable de holgura con costo marginal negativo (Entra S12 y Sale h13)

	Cj	4	4	5	7	8	9	9	13	0	0	0	0	0	0	0	0	0	0
Cb	Xb	B	h11	h12	h13	h14	h21	h22	h23	h24	S12	S13	S23	S24	S25	S34	S35	S45	S46	S56
4	h11	12	1	0	1	0	0	0	0	-1	0	0	0	0	0	-1	0	0	0	0
4	h12	8	0	1	0	1	0	0	0	1	0	0	0	0	0	1	0	0	0	0
0	S12	10	0	0	1	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0
9	h23	10	0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0
0	S13	10	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	S23	8	0	0	-1	1	0	0	0	1	0	1	0	0	0	1	0	0	0	0
0	S24	3	0	0	0	0	1	1	0	2	0	0	1	1	0	1	0	0	0	0
0	S25	0	0	0	0	0	-1	-1	0	-1	0	0	0	1	1	0	0	0	0	0
0	S35	5	0	0	0	0	0	0	0	1	0	0	0	0	0	0	1	0	0	0
0	S45	4	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0	0
0	S46	15	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1	0
0	S56	0	0	0	0	0	0	-1	0	0	0	0	0	0	0	0	0	0	0	1
	Zj	170	4	4	4	4	9	9	9	9	0	0	0	0	0	0	0	0	0	0
	Cj - Zj		0	0	1	3	-1	0	0	4	0	0	0	0	0	0	0	0	0	0

Zactual: CV + CF: 170+(3+8+9+3+9+4) = 206

Si Zactual < Zlocal? (206<INF): Zlocal = 206, Zlocal < Zmejor? (206<221): Zmejor = 206, move = 1

Paso 3: Costos Reducidos de los caminos no básicos (h13 = 1, h21 = -1, h22 = 0)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

Entra: h13	Sale: S12	Valor con que entraria a la Base: 10
Hay modif en costo fijo	Entra 1-2	CF = 5
Var del valor del mov potencial:		(CR* b) + CF = 1*10+5 =15
Entra: h22	Sale: S24	Valor con que entraria a la Base: 3
Hay modif en costo fijo	Entra 4-6	CF = 5
Var del valor del mov potencial:		(CR* b) + CF = 0*3+5 =5
Entra: h21	Sale: S24	Valor con que entraria a la Base: 3
Hay modif en costo fijo	Entra 1-2	CF = 1
Var del valor del mov potencial:		(CR* b) + CF = -1*3+1 =-2

Lo anterior se resume de la siguiente forma:

hin	hout	Valor asociado	Δz
h21	S24	3.0	-2.0
h22	S24	3.0	5.0
h13	S12	10.0	15.0

Paso 5: Identifique el mov que produce el menor valor de Δz que no sea tabu o mejore Zmejor: (Entra: h21 y Sale s24)

Paso 6: Implementar movimiento: Zactual= 204.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (204< 206): Zlocal = 204,

Si Zlocal < Zmejor? (204 < 206): Zmejor = 204, move = 1

Si move≠max_move? (1≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENURES			
Iter	1	2	3
1	-	-	S24

(Move = 1, Iter = 2)

Paso 2: Base (h11, h12, h21, h23)

Paso 3: Costos Reducidos de los caminos no básicos (h13, h22)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

hin	hout	Valor asociado	Δz
h22	h21	3.0	7.0
h13	S12	10.0	15.0

Paso 5: Identifique el mov que produce el menor valor de Dz Δz que no sea tabu o mejore Zmejor:

(Entra: h22 y Sale h21)

Paso 6: Implementar movimiento: Zactual= 211.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (211< 204): Zlocal = 204

Si Zlocal < Zmejor? (204 < 204): Zmejor = 204, move = 2

Si move≠max_move? (2≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENENCIA			
Iter	1	2	3
2	-	S24	h21

Paso 8: Si Zlocal < Zprev? (204<INF): Zprev = Zlocal = 204 y g = 0; Sino g = g+1 ∴ g = 0

Si g > max_cam_gen? (0 >= 2) SALIR a Diversificar

Paso 9: Genera caminos h14 (1-2-3-4), h24 (2-4-3-5-6) e ir al Paso 2

(Move = 1, Iter = 3)

Paso 2: Base (h11, h12, h22, h23)

Paso 3: Costos Reducidos de los caminos no básicos (h13, h14, h21, h24)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

hin	hout	Valor asociado	Δz	
h21	h22	3.0	-7.0	*tabu y no mejora Zmejor
h24	h22	3.0	9.0	
h13	S12	10.0	15.0	
h14	h12	8.0	29.0	

Paso 5: Identifique el mov que produce el menor valor de Δz que no sea tabu o mejore Zmejor: (Entra:

h24 y Sale h22)

Paso 6: Implementar movimiento: Zactual= 214.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (214 < 204): Zlocal = 204

Si Zlocal < Zmejor? (204 < 204): Zmejor = 204, move = 2

Si move≠max_move? (2≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENENCIA			
Iter	1	2	3
3	S24	h21	h22

Paso 8: Si Zlocal < Zprev? (204<204): Zprev = Zlocal y g =0; Sino g = g+1 = 1 ∴ g =1

Si g > max_cam_gen? (1 >= 2) SALIR a Diversificar

Paso 9: Genera caminos h15 (1-3-5-2-4), h25 (2-5-4-6) e ir al Paso 2

(Move = 1, Iter = 4)

Paso 2: Base (h11, h12, h23, h24)

Pivotear variable de holgura con costo marginal negativo (Entra S34 y Sale s25): Zactual = 206

Si Zactual < Zlocal? (206<204): Zlocal = 204, Zlocal < Zmejor? (204<204): Zmejor = 204

Paso 3: Costos Reducidos de los caminos no básicos (h13, h14, h15, h22, h24, h25)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

hin	hout	Valor asociado	Δz	
H21	h24	0.0	1.0	*tabu y no mejora Zmejor
H22	h24	0.0	5.0	*tabu y no mejora Zmejor
H13	S12	10.0	15.0	
H14	h12	5.0	20.0	
H25	S45	4.0	26.0	
H15	S34	3.0	29.0	

Paso 5: Identifique el mov que produce el menor valor de Δz que no sea tabu o mejore Zmejor: (Entra: h13 Sale s12)

Paso 6: Implementar movimiento: Zactual= 221.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (221 < 204): Zlocal = 204

Si Zlocal < Zmejor? (204 < 204): Zmejor = 204, move = 2

Si move ≠ max_move? (2≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENENCIA			
Iter	1	2	3
4	h21	h22	S12

Paso 8: Si Zlocal < Zprev? (204 <204) y g =0; Sino g = g+1 ∴ g = 2

Si g > max_cam_gen? (2 >= 2) SALIR a Diversificar

III Fase. Diversificación

Si div < max_div?(0<1)entonces div = div +1 ∴div = 1 e Ir a Búsqueda Local

Sino PARAR

Buscando Caminos a cerrar con la arista 2 – 3 (h11), frec = 5, Zactual = 228.5

II Fase. Búsqueda Local

Paso 1: Zlocal = INF, Zprev = INF, g = 0, Zmejor = 204, CC (h11, h12, h13, h14, h15, h21, h22, h23, h24, h25)

(Move = 1, Iter = 5)

Paso 2: Base (h12, h13,h21,h23, h24)

Pivotear variable de holgura con costo marginal negativo (Entra S24 y Sale h24): Zactual = 224

Si Zactual < Zlocal? (224<INF): Zlocal = 224, Zlocal < Zmejor? (224<204): Zmejor = 204

Pivotear variable de holgura con costo marginal negativo (Entra S25 y Sale s24): Zactual = 223

Si Zactual < Zlocal? (223<224): Zlocal = 223, Zlocal < Zmejor? (223<204): Zmejor = 204

Paso 3: Costos Reducidos de los caminos no básicos (h11, h14, h22, h24, h25)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

hin	hout	Valor asociado	Δz
H11	h15	2.0	-4.0
H22	h21	3.0	7.0
H25	S45	1.0	10.0
H24	S25	1.0	11.0
H14	S25	1.0	15.0

Paso 5: Identifique el mov que produce el menor valor de Δz que no sea tabu o mejore Zmejor: (Entra: h11 y Sale h15)

Paso 6: Implementar movimiento: Zactual= 219.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (219 < 223): Zlocal = 219

Si Zlocal < Zmejor? (219 < 204): Zmejor = 204, move = 2

Si move ≠ max_move? (2≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENENCIA			
Iter	1	2	3
5	h22	S12	h15

Paso 8: Si Zlocal < Zprev? (219<INF) Zprev = Zlocal = 219 y g = 0; Sino g = g+1 ∴ g = 0

Si g > max_cam_gen? (0 >= 2) SALIR a Diversificar

Paso 9: Genera caminos h16 (1-3-5-4), h26 (2-1-3-4-5-6) e ir al Paso 2

(Move = 1, Iter = 6)

Paso 2: Base (h11, h12, h13, h21, h23)

Paso 3: Costos Reducidos de los caminos no básicos (h14, h15, h16, h22, h24, h25, h26)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

hin	hout	Valor asociado	Δz	
H15	h11	2.0	4.0	*tabu y no mejora Zmejor
H16	S45	1.0	6.0	
H22	h21	3.0	7.0	
H25	S45	1.0	10.0	
H14	h12	8.0	16.0	
H24	h21	3.0	19.0	
H26	h21	3.0	24.0	

Paso 5: Identifique el mov que produce el menor valor de Δz que no sea tabu o mejore Zmejor: (Entra: h16 y Sale S45)

Paso 6: Implementar movimiento: Zactual= 223.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (223 < 219): Zlocal = 219

Si Zlocal < Zmejor? (219 < 204): Zmejor = 204, move = 2

Si move ≠ max_move? (1≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENENCIA			
Iter	1	2	3
6	S12	h15	S45

Paso 8: Si Zlocal < Zprev? (219<223): Zprev = Zlocal = 223 y g = 0; Sino g = g+1 ∴ g = 1

Si g > max_cam_gen? (0 >= 2) SALIR a Diversificar

Paso 9: Genera caminos h17 (1-2-5-3-4), h27 (2-1-3-4-6) e ir al Paso 2

(Move = 1, Iter = 7)

Paso 2: Base (h11, h12, h13, h21, h23, h16)

Paso 3: Costos Reducidos de los caminos no básicos (h14, h15, h17, h22, h24, h25, h26, h27)

Paso 4: Calcular el valor del movimiento potencial para cada camino no-básico

hin	hout	Valor asociado	Δz	
H15	h11	1.5	-0.5	*tabu y no mejora Zmejor
H22	h11	1.5	1.5	
H25	h16	0.5	4.5	
H14	h12	8.0	16.0	
H17	s25	3.5	24.5	
H26	h21	3.5	28.0	
H27	h21	3.5	43.5	
H24	S35	4.5	45.0	

Paso 5: Identifique el mov que produce el menor valor de Δz que no sea tabu o mejore Zmejor: (Entra: h22 y Sale h11)

Paso 6: Implementar movimiento: Zactual= 229.0

Paso 7: Establece y Actualiza tenencias tabu

Si Zactual < Zlocal? (229 < 219): Zlocal = 219

Si Zlocal < Zmejor? (219 < 204): Zmejor = 204, move = 2

Si move ≠ max_move? (2≠2) ir paso 2 hasta realizar max_move consecutivos sin mejora en Zlocal

TENENCIA			
Iter	1	2	3
7	h15	S45	h11

Paso 8: Si Zlocal < Zprev? (219<219): Zprev = Zlocal = 219 y g = 0; Sino g = g+1 ∴ g = 2

Si g > max_cam_gen? (2 >= 2) SALIR a Diversificar

III Fase. Diversificación

Si div < max_div?(1<1)entonces div = div +1

Sino PARAR

El resultado al problema de diseño de red, empleando nuestro algoritmo se muestra a continuación y puede verse gráficamente en la figura 3.

Prod	Camino	Nom Camino	Ruta	Flujo	CV
1					
	1	h11	1-3-2-4	12	48
	2	h12	1-3-4	8	32
2					
	1	h21	2-4-5-6	3	24
	3	h23	2-5-6	7	63

Costo de Solución: CV + CF = 167 + (3+8+9+3+1+4+9) = 204

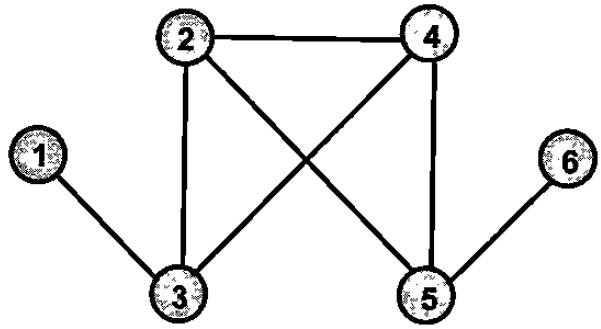


Figura 3. Resultado del Diseño de red

CAPÍTULO 5

DISEÑO DEL EXPERIMENTO COMPUTACIONAL

5.1 Establecimiento de los parámetros

La efectividad de cualquier metaheurística puede ser medida mediante una evaluación del desempeño en los experimentos (es decir, efectuar una evaluación de la influencia de uno o más parámetros). Un parámetro es cualquier variable controlable de un experimento que influye en el resultado del mismo. El desempeño de la metodología puede medirse en cuanto su calidad de solución, esfuerzo computacional y robustez.

Es importante remarcar que el desempeño de los experimentos computacionales puede ser afectado por las condiciones de experimentación y que los mejores valores de los parámetros dependen del problema a tratar en particular.

A continuación se describirán los parámetros que fueron utilizados a fin de obtener los mejores resultados posibles. Como las soluciones iniciales a las cuales se les aplicó el algoritmo propuesto aquí fueron obtenidas a través de un GRASP también se incluirán los parámetros usados por este.

5.1.1 Parámetros del GRASP

Algunos parámetros usados en el GRASP para la obtención de la solución inicial fueron la *longitud de la Lista Restringida Candidata* (len_LRC) la cual es dinámica e inicialmente trabaja con un 20% de rutas y aumenta gradualmente a medida que avanza la generación de soluciones o la cantidad de iteraciones del GRASP, *número máximo de iteraciones del GRASP* (GRASP_iter = 1000) el cual es mayor que las soluciones requeridas puesto que no siempre obtiene una solución factible, *número máximo de soluciones en la población* (MAX_SOL = 100), etc. Más detalles en De Alba (2004).

5.1.2 Parámetros del algoritmo de solución

Se hicieron diferentes consideraciones en nuestro algoritmo de solución a lo largo de su desarrollo, tales como:

- Decidir a que variable asignar el estatus tabú en la búsqueda local, ya sea a caminos que salieran de la base, entraran a la base o ambos, así como la longitud de esta tenencia tabú.
- Favorecer el pivoteo de cualquier variable de holgura con costo marginal negativo, o bien con un costo reducido negativo
- Decidir si para cada camino no-básico a salir entrarán solo caminos o también variables de holgura.

Se observó que los mejores resultados se obtuvieron donde el estatus tabú fue asignado solamente a caminos que salieran de la base, se pivoteara en primer lugar cualquier variable de holgura con costo reducido negativo y para cada camino a salir considerar a entrar tanto caminos como variables de holgura.

Otros parámetros a evaluar son la cantidad de movimientos en la fase de la búsqueda local: *número de movimientos pivotes* (max_move) y *número de fases de generación de caminos* (max_col_gen); *número de iteraciones en la fase de diversificación* (max_diver), así como el *valor de la tenencia tabú* para los caminos (tabu_h) y para las aristas (tabu_a). Para más detalles ver Apéndice D.

Se evaluó cada uno de los mencionados parámetros con diferentes valores en un rango de problemas significativos, observándose su comportamiento para después comparar los resultados obtenidos por cada una de estos. Todo esto nos ayudó a decidir que parámetros mostraron un mejor comportamiento, para implementarlos en nuestro algoritmo aquí presentado.

5.1.3 Calibración de los parámetros del algoritmo de solución

Es sabido que para el buen funcionamiento o desarrollo de una metaheurística es muy importante la apropiada calibración de los parámetros, pues constituye una parte integral en el desarrollo de cualquier procedimiento metaheurístico.

El proceso de calibración tiene por objetivo determinar un rango de valores que

serán usados en los parámetros de la búsqueda, de tal modo que el desarrollo de la metaheurística trabaje bien sobre un amplio rango de tipos de problemas.

Para lograr esto, se busca un conjunto de parámetros que sean robustos para un pequeño conjunto de problemas representativos y después se emplearán en los diferentes conjuntos o instancias de problemas.

Para la calibración de los parámetros, nosotros seleccionamos un pequeño número de instancias con diferentes atributos que abarcara un rango de diferentes tipos de redes, y posteriormente se fijaron los valores de estos parámetros. Nosotros probamos las siguientes combinaciones de valores para ellos [min, max].

- Intervalo de tenencia tabú medido en iteraciones, *tabu_h* y *tabu_a* =[1, 3]
- Número de movimientos o iteraciones para la evaluación de vecindades y elección de movimiento en la búsqueda local, *max_move* = [1, 5]
- Número de iteraciones para la generación de caminos, *max_cam_gen* = [1, 5]
- Número de movimientos en la fase de diversificación, *max_div* = [1, 5]

Los parámetros anteriores se evaluaron mediante la comparación obtenida de la diferencia relativa entre la mejor solución entera entregada por un algoritmo de solución exacta contra nuestro algoritmo de solución. A continuación sus respectivos valores:

Max_move	*max_cam_gen*	*max_div*	*tabu_h*	*tabu_a*	*Diferencia Relativa%*
2	1	3	3	3	0.38
2	1	4	3	3	0.43
2	2	3	3	3	-0.21

Se seleccionaron aquellos parámetros que mostraron el mejor comportamiento: *max_move* = 2, *max_col_gen* = 3, *tabu_h* = 3, *tabu_a* = 3, *max_div* = 3. El signo negativo indica una mejora en favor de nuestro algoritmo. Cabe recalcar que esta calibración no fue realizada a todas las instancias de los problemas.

5.2 Criterio de Comparación

La importancia en el desarrollo de cualquier método heurístico es la validación de sus resultados, es decir conocer la calidad de la solución. La comparación de los resultados contra el óptimo global del problema es regularmente una tarea difícil debido a los problemas de optimización combinatoria en particular son de gran dificultad

debido a su complejidad, ya que crecen exponencialmente con el tamaño del problema esto justifica el uso de métodos heurísticos, ya que se acercan a la solución óptima en un tiempo razonable. Por lo que se considera el siguiente criterio de evaluación para medir el desempeño del método que se implementará: *comparación contra la mejor solución entera.*

La evaluación en el presente trabajo consiste en la comparación contra la mejor solución entera encontrada por un optimizador durante un período de tiempo predeterminado, la cual obtiene una solución de cierta calidad en un período de tiempo razonable debido a que es difícil la obtención de un óptimo global para problemas de instancias reales, donde esta se comparará contra la mejor solución obtenida en un corto tiempo de ejecución del método heurístico.

5.3 Diseño del experimento

Con el objetivo de evaluar la eficiencia del procedimiento de solución propuesto se diseño un experimento computacional. Este experimento consistió en comparar el desempeño del algoritmo contra un método de solución exacta y contra un algoritmo evolutivo que está basado en la metaheurística Búsqueda Dispersa (Scatter Search, SS, en inglés).

CPLEX es un optimizador de alto desempeño para problemas de programación lineal, entero, entero mixta, etc. Uno de los métodos que tiene implementado es el método simplex para resolver los problemas de programación lineal (CPLEX, 1999).

CPLEX es flexible ya que provee unas librerías que autorizan a los desarrolladores a incrustarlas en sus programas de modo transparente y eficiente en sus aplicaciones, proporcionando transportabilidad. Esto nos da la facilidad para poder comenzar a partir de una solución básica factible (SBF) inicial, identificar las SBF y agregar columnas al problema tratado como LP. También nos permite modificar el valor de los parámetros default del optimizador, lo cual facilita nuestro desarrollo. Las opciones del optimizador se verán en el Apéndice C.

5.3.1 Generador aleatorio de instancias

Teniendo en cuenta que el método propuesto aquí va a ser comparado contra un método evolutivo implementado por De Alba (2004) para este mismo problema, se

utilizó el mismo generador aleatorio diseñado por él en su trabajo.

El generador crea un archivo texto (ASCII) con diferentes registros que contendrá información y características propias de cada instancia de red. El nombre de dicho archivo se definió según las características que la red representa (número de nodos, número de productos, tipo de red y tipo de costos), ver Apéndice B.

El archivo generado será usado como entrada para los programas de obtención de las rutas más cortas, GRASP (obtiene una solución inicial factible) y optimizador CPLEX.

A continuación se muestra el formato del archivo generador de instancias.

5.3.2 Archivo de instancia para DRCM

El formato del archivo que contiene las instancias de red posee 4 diferentes tipos de registro.

Registro tipo 1. Está compuesto por las características generales de la red (N, M, K)

Registro tipo 2. Características generales para cada arista ($\{i, j\}$, CF, Cap)

Registro tipo 3. Costos variables para cada producto en el arco ((i, j), k, CV)

Registro tipo 4. Información para cada producto (Or, De, Dem)

La Tabla 1 muestra la descripción de los parámetros que son de tipo entero. En la columna 1, se muestran tales parámetros usados en el archivo que contiene las instancias de red y en la columna 2 se describen los mismos.

Parámetro	Descripción
N	Cantidad de nodos
K	Cantidad de productos
M	Cantidad de aristas
Cap	Capacidad de la arista
CF	Costo fijo de la arista
CV	Costo variable del arco
Or	Nodo origen
De	Nodo destino
Dem	Demanda

Tabla 1. Descripción de parámetros de archivo de instancias

En cada archivo existirá un sólo registro tipo 1. Habrá tantos registros tipo 2 como aristas posea la red. Del registro tipo 3 existirán en función de las aristas y los productos

que circulen en la red, con un total de número de productos por número de aristas. Finalmente del registro tipo 4 habrán tantos registros como productos tenga la instancia. Para una descripción más detallada consúltese el Apéndice B.

5.3.3 Clasificación de instancias

Las instancias generadas aleatoriamente fueron clasificadas de acuerdo a sus características: según la predominancia de costos fijos o variables y de acuerdo a qué tan ajustadas son las capacidades en las aristas. Esta clasificación se muestra en la tabla 2.

Clase	Características	
	Costos	Tipo de red
I	Fijos	Holgada
II	Fijos	Restringida
III	Variables	Holgada
IV	Variables	Restringida

Tabla 2. Clasificación de clases de instancias

Para cada una de las anteriores clasificaciones, se generaron 5 instancias con diferentes cantidades de nodos (30, 50), aristas (350, 750) y productos (10, 50 y 100), dando un total de 120 instancias generadas. (Ver Apéndice A)

En la tabla 3 se muestran las instancias generadas y sus características. En la primera columna se indica el número de nodos, la columna 2 muestra el número de aristas, la columna 3 muestra la cantidad de productos, la columna 4 indica el tipo de costo predominante, es decir costos fijos (F) o variables (V), la quinta columna indica el tipo de red en relación con la capacidad de las aristas, es decir holgada (H) o restringida (R) y en la última columna se indica la cantidad de instancias elaboradas de cada tipo.

Nodos	Aristas	Prod.	Tipo de Costos	Tipo de Red	Probs.
30	350	10	F	H	5
				R	5
			V	H	5
				R	5
		50	F	H	5
				R	5
			V	H	5
				R	5
		100	F	H	5
				R	5
			V	H	5
				R	5

Nodos	Aristas	Prod.	Tipo de Costos	Tipo de Red	Probs
50	750	10	F	H	5
				R	5
			V	H	5
				R	5
		50	F	H	5
				R	5
			V	H	5
				R	5
		100	F	H	5
				R	5
			V	H	5
				R	5

Tabla 3. Clasificación de instancias

5.3.4 Implementación computacional

Para la programación de todas las rutinas necesarias se utilizó lenguaje C de programación y fueron ejecutadas en una terminal SUN™ Ultra 10, con sistema operativo Solaris™ versión 7 que permitió la realización de las pruebas. Para encontrar la solución óptima se utilizó el optimizador CPLEX versión 7.1 (1999). Los programas que se utilizaron en nuestro algoritmo pueden verse en el Apéndice D.

CAPÍTULO 6

RESULTADOS COMPUTACIONALES

6.1 Introducción

En este capítulo se mostrarán los diferentes resultados que se obtuvieron al aplicar el algoritmo de solución propuesto al conjunto de instancias generadas.

En el optimizador CPLEX se fijo un tiempo de 6 horas para resolver las instancias del problema entero mixto, sin embargo, fueron excepcionales los casos en que la solución óptima fue encontrada, y esto ocurrió exclusivamente en instancias pequeñas, aquéllas con 30 nodos, 700 arcos y 10 productos. Sin embargo no en todos los casos de instancias mayores le fue posible al optimizador entregar siquiera una solución entera factible.

Para evaluar el desempeño, las instancias fueron corridas con el optimizador y con el algoritmo propuesto en este trabajo, además de ello los resultados se compararon con los obtenidos mediante un algoritmo evolutivo basado en la técnica de Búsqueda Dispersa.

El GRASP utilizado para generar soluciones factibles iniciales entregó 100 soluciones para cada instancia. El resultado que se está reportando es el mejor obtenido al aplicar el método propuesto en este trabajo a cada una de las soluciones iniciales. Esto se hizo así para que sea válida la comparación que posteriormente se realizó con los resultados obtenidos de aplicar el SS a esa población de 100 soluciones (De Alba, 2004).

6.1.1 Evaluación del desempeño del método

La Tabla 4 muestra la clasificación de las instancias según las clases propuestas. La columna 1 muestra la clasificación de red de acuerdo a la predominancia de costo y a qué tan ajustadas son las capacidades en las aristas. La columna 2 muestra el promedio

de las diferencias relativas entre la solución entregada por el algoritmo propuesto y la mejor solución entera entregada por el optimizador CPLEX. Los valores con signo negativo indican una mejora a favor del algoritmo desarrollado en este trabajo. Si observamos la columna del promedio de las diferencias relativas, las instancias que mejor desempeño obtuvieron son las de redes holgadas con costo fijos altos (clase I) mientras que las de peor desempeño son las redes restringidas con costo fijos altos (clase II). En relación a la comparación entre los costos, se observa en general que se obtienen mejores resultados los costos variables altos, (clases III y IV) que los costos fijos altos (clases I y II). Respecto a las capacidades, se desempeñan mejor las instancias con redes holgadas (clases I y III) que las redes restringidas (clases II y IV).

Clase	TSvsCPLEX
I	-2.59%
II	5.33%
III	-2.23%
IV	-2.30%
General	0.44%

Tabla 4. Comparación cuando las instancias son agrupadas por su clase

Con el objetivo de ver que característica de la red influía más en el desempeño del algoritmo se agruparon las instancias por nodo-producto, costos fijo o variables y por cantidad de productos. En cada caso los resultados se promediaron sobre las cuatro clases anteriormente definidas.

La Tabla 5 muestra las instancias agrupadas por número de nodos y números de productos. La primera columna muestra el número de nodos y productos de las instancias de red. La columna 2 muestra el promedio de la diferencia relativa del procedimiento de TS comparado contra la mejor solución entera entregada por el optimizador CPLEX. Se designó un máximo de 6 horas para el optimizador y el resultado después de ese tiempo fue anotado, los guiones indican que no pudieron hacerse comparaciones ya que el optimizador no pudo encontrar una solución factible al cabo de este lapso de tiempo. Se observó que a medida que el número de productos crece, también crece la diferencia entre los resultados, concluyendo con esto que el número de productos afecta el desempeño del método de solución.

Nodos-Prod	TSvsCPLEX
30 -10	1.35%
30 - 50	-1.85%
30 - 100	6.88%
50 - 10	-2.97%
50 - 50	-
50 - 100	-

Tabla 5. Comparación cuando las instancias son agrupadas por nodos-productos

La Tabla 6 muestra los resultados obtenidos agrupando las instancias según la importancia de los costos fijos respecto a los costos variables. La columna 1 muestra la clasificación considerando la predominancia de los costos, ya sean costos fijos altos o costos variables altos. La columna 2 muestra el promedio de la diferencia relativa del procedimiento de TS comparado contra la mejor solución entera entregada por CPLEX. Se puede observar que las soluciones del procedimiento de TS pierden calidad cuando los costos fijos son altos, lo cual se explica por el hecho de que incluir en el diseño una arista no conveniente repercute más desfavorablemente que si el costo fijo que se paga por ella fuera bajo.

Tipo Costo	TSvsCPLEX
Fijos	1.84%
Variables	-2.26%
General	-0.21%

Tabla 6. Comparación cuando las instancias son agrupadas por su relevancia de costos

La Tabla 7 muestra las instancias agrupadas por cantidad de productos. La primera columna muestra los productos y la columna 2 muestra los porcentajes que representan el promedio de la diferencia relativa entre el algoritmo de solución propuesto en esta tesis y la mejor solución entera entregada por CPLEX. Como era de esperar el algoritmo propuesto en esta tesis se desempeña mejor en la red con pocos productos, mientras que cuando hay muchos productos para compartir las capacidades de las aristas, hay que incluir más aristas en el diseño y el costo se eleva.

Productos	TSvsCPLEX
10	-0.81%
50	-1.85%
100	6.88%
General	1.41%

Tabla 7. Comparación cuando las instancias son agrupadas por productos

6.1.2 Comparación contra un método de solución basado en Búsqueda Dispersa

A continuación se muestra una comparación entre los resultados obtenidos por el algoritmo aquí propuesto y un algoritmo basado en Búsqueda Dispersa.

El SS es un algoritmo basado en poblaciones que construye soluciones mediante la combinación de otras. En De Alba (2004) se probaron diferentes estrategias para cada uno de los métodos componentes de la búsqueda dispersa. Para los propósitos de nuestra comparación se está usando SS^{rda} donde el superíndice indica:

r: Utiliza un conjunto de referencia que incluye un subconjunto de soluciones de calidad y otro subconjunto con soluciones diversas.

d: Utiliza actualización dinámica del conjunto de referencia.

a: Utiliza una función de distancia medida en arcos para medir la "distancia" entre dos soluciones.

La versión SS^{rda} fue la que arrojó los mejores resultados reportados en De Alba (2004).

Las instancias en las tablas de la 8 a la 11 que aparecen a continuación fueron agrupadas en forma similar a las de las tablas 4 a la 7. En cada caso en la primera columna se indica esta agrupación, en la segunda columna se muestra el promedio de la diferencia relativa entre la solución entregada por el algoritmo propuesto en este trabajo y el método basado en SS, la tercera columna muestra el tiempo en segundos utilizado por nuestro algoritmo y la cuarta columna el tiempo en segundos utilizado por el método basado en SS. Al igual que en las anteriores tablas los valores negativos indican una mejoría contra lo que se esta comparando.

El resultado que se reporta para el método SS es el mejor luego de aplicarlo a la población de 100 soluciones entregada por el GRASP a cada instancia.

A continuación en la Tabla 8, se observa que nuestro método también se desempeña mejor en todos los casos.

Clase	TSvsSS	t_TS	t_SS
I	-11.75%	72.37	66.70
II	-8.32%	69.30	54.27
III	-8.60%	75.81	70.53
IV	-8.90%	66.05	56.27
General	-9.39%	70.88	61.94

Tabla 8. Comparación cuando las instancias son agrupadas por su clase

En la Tabla 9 puede observarse que el comportamiento del método basado en TS es muy similar al basado en SS para redes de 30, 50 nodos y pocos productos. Sin embargo para redes grandes, con muchos nodos y productos, el método aquí propuesto presenta mejores resultados.

Nodos-Prod	TSvsSS	t_TS	t_SS
30 - 10	-4.41%	1.97	16.60
30 - 50	-7.54%	18.93	58.85
30 - 100	-23.74%	64.04	88.90
50 - 10	-3.93%	7.76	27.40
50 - 50	-5.71%	78.14	62.10
50 - 100	-11.03%	254.45	117.80
General	-9.39%	70.88	61.94

Tabla 9. Comparación cuando las instancias son agrupadas por número de nodos-productos

En la Tabla 10 se observa que las instancias se desempeñaron mejor tanto para costos variables altos como para costos fijos altos.

Tipo Costo	TSvsSS	t_TS	t_SS
Fijos	-10.03%	70.83	60.48
Variables	-8.75%	70.93	63.40
General	-9.39%	70.88	61.94

Tabla 10. Comparación cuando las instancias son agrupadas por su relevancia de costos

De la Tabla 11 se puede observar que las redes con pocos productos presentan una mejoría, y esto se nota en los porcentajes. Sin embargo se puede notar que dicha mejora es más significativa cuando aumenta el número de productos.

Productos	TSvsSS	t_TS	t_SS
10	-4.17%	4.86	22.00
50	-6.63%	48.53	60.48
100	-17.39%	159.24	103.35
General	-9.39%	70.88	61.94

Tabla 11. Comparación cuando las instancias son agrupadas por productos

CAPÍTULO 7

CONCLUSIONES Y RECOMENDACIONES

7.1 Conclusiones

En este capítulo se mencionarán los aspectos relevantes del trabajo aquí presentado y se darán pautas para investigaciones futuras relacionadas con el tema.

Un aspecto importante en toda heurística es la definición de los parámetros que se emplean, ya que una mala selección conducirá a un mal desempeño de la metodología de solución. Se realizaron diversas pruebas con diferentes valores de parámetros para seleccionar el mejor de todos ellos, entregándose un conjunto de valores que se comporta en forma robusta.

En relación con el desempeño del método de solución propuesto se puede notar, que si bien los resultados son buenos en general, se desempeñó mejor en redes con costos variables predominantes sobre costos fijos. Esto se debe a que se seleccionan aristas caras lo cual con lleva a soluciones de mala calidad.

Otro factor de importancia relacionado con la calidad de las soluciones obtenidas es que a mayor cantidad de productos que circulen por la red, mayor es la complejidad del problema, lo cual se refleja en soluciones de peor calidad.

Además, puede verse que la capacidad de las aristas de la red es un factor determinante en el costo de la misma. Esto se explica por que al crecer el número de productos, crece la necesidad de compartir arcos y si la capacidad de ellos es restringida habría que añadir aristas al diseño lo cual lo encarecería.

Un aspecto interesante que vale la pena notar es el tiempo asignado para resolver las instancias por el optimizador. Se asignó un tiempo fijo de 6 horas, sin embargo, fueron excepcionales los casos en que la solución óptima fue encontrada, y esto ocurrió

exclusivamente en instancias pequeñas, aquéllas con 30 nodos, 700 arcos y 10 productos. En instancias mayores, el optimizador pudo entregar cotas inferiores, sin embargo, no en todos los casos le fue posible entregar una solución entera. Esto nos permite concluir la importancia de disponer de un método como el aquí presentado que si bien no garantiza entregar soluciones óptimas, entrega soluciones de muy buena calidad en un tiempo muy rápido.

Como puede observarse el comportamiento del método basado en TS es muy similar al basado en SS para las redes con 30 nodos y para las de 50 nodos y pocos productos. Sin embargo para las redes más grandes, donde se degrada la efectividad de SS, el método aquí propuesto presenta mejores resultados.

7.2 Aportaciones Científicas

En este trabajo, se hizo un estudio de la estructura matemática del problema particular de diseño abordado y se presentó para él la formulación basada en caminos que resulta en un programa entero mixto.

Hasta el momento no se conocen algoritmos exactos que puedan resolver los problemas abordados por nosotros en un período razonable de tiempo, por ello se considera de gran importancia la realización del presente trabajo, dada la necesidad de desarrollar métodos específicos de solución a problemas de tamaño real.

El procedimiento presentado representa un esfuerzo exitoso que propone una eficiente aproximación para obtener buenas soluciones factibles al problema de diseño de red multiproducto con capacidades en las aristas y cargos fijos.

El algoritmo propuesto, basado en la metaheurística búsqueda tabú, identifica muy buenas soluciones dentro de esfuerzos computacionales razonables. Además, el procedimiento se muestra robusto con respecto al tipo de problema, en términos de la importancia relativa de los costos fijos, capacidades, tamaño y especialmente en el número de productos.

Los resultados del presente trabajo han sido publicados en:

1. Cobos-Zaleta, N. (2004). Tabu Search-Based Algorithm for Capacitated Multicommodity Network Design Problem. *En Proceedings of the XIV*

International Conference on Electronics, Communicatios and Computers, Veracruz, México. Febrero.

2. Cobos-Zaleta, N. Búsqueda Tabú para un Problema de Diseño de Red Multiproducto con Capacidad Finita. Sometido a la *Revista Mexicana de Ingeniería Electrónica y Eléctrica*, Puebla, México.

y presentados en:

1. Cobos-Zaleta, N. (2002). Búsqueda Tabú para el Problema de Diseño de Red Capacitada. Facultad de Ingeniería Mecánica y Eléctrica. Ciclo de Seminarios de PISIS. FIME, UANL, San Nicolás de los Garza, N. L, Septiembre.
2. Cobos-Zaleta, N. (2004). Tabu Search-Based Algorithm for Capacitated Multicommodity Network Design Problem. XIV Congreso Internacional de Electrónica, Comunicaciones y Computadoras 2004. Veracruz, México. Febrero.

7.3 Recomendaciones para trabajos posteriores

Una interesante línea de investigación es ahora posible, con respecto a la aplicación de la metodología aquí presentada para otros tipos de problemas de diseño de redes donde se incluyan otras restricciones, como por ejemplo restricciones de capacidad parcial para cada producto en las aristas o restricciones de presupuesto. Con ello, no solamente se estaría haciendo una contribución al conocimiento científico y avance de la ciencia, sino también se proveerían herramientas útiles para problemas de toma de decisiones que aparecen frecuentemente en la actualidad donde los márgenes de planeación son cada vez mayores y la globalización origina tareas de decisión más complejas.

Por otra parte, consideramos que es posible extender y mejorar el trabajo desarrollado aquí mejorando la solución inicial de partida. Para ello podría utilizarse un GRASP con memoria adaptativa, en lugar del tradicional implementado por De Alba (2004). Otra extensión sería incorporar a TS otras estrategias de diversificación u otras técnicas, tales como oscilación estratégica que permitan obtener resultados aún mejores.

Por último, y quizás sea lo más importante, consideramos que se podría investigar

la posibilidad de realizar la generación de caminos en diferentes momentos de la búsqueda aprovechando así la información que reporte la historia.

BIBLIOGRAFÍA

Álvarez, A; González-Velarde, J.L. y De Alba, K. (2001). Un algoritmo de búsqueda para un problema de red capacitada multiproducto. *Memorias del 3er Encuentro Internacional de Computación.* Aguascalientes, Ags. Septiembre.

Balakrishnan, A. y Magnanti, T.L. (1989). A dual ascent procedure for large-scale uncapacitated network design. *Operations Research,* Vol. *37*, No. 5, pp. 716-740

CPLEX Optimimization, Inc. (1999). *ILOG CPLEX 7.1 Reference Manual.* Incline Village, NV, EUA.

Costamagna, F.; Fanni, A y Giacinto, G. (1995). Tabu search for the optimization of b-isdn telecommunication networks, Tech. Report No. 60, Dept. of Electrical Eng., University of Cagliari.

Crainic, T.G.; Gendreau, M. y Farvolden, J.M. (2000). A simplex-based tabu search method for capacitated network design. *INFORMS Journal on Computing*, Vol. 12, No. 3, pp. 223-236.

Cruz, F.R.B.; MacGregor, J. y Mateus, G.R. (1998). Solving to optimality the uncapacitated fixed-charge network flow problem. *Computers Operations Research.* Vol. 25, No.1, pp. 67-81.

Daskin, M.S. (1995). Network and discrete location. Models, Algorithms and Applications. Wiley. New York, EUA.

De Alba, K.; Álvarez, A. y González-Velarde, J.L. (2001). Scatter search for the multicommodity capacited network design problem. *Proceedings of 6th Annual International Conference On Industrial Engineering–Theory, Aplications and Practice.* San Francisco, USA, Noviembre.

De Alba, K; Álvarez, A. y Gónzalez-Velarde J.L. (2003). A memory-based GRASP for for finding good starting solutions to the multicommodity capacitated network design problem, *Proceedings, INFORMS Computing Society's Conference.* Chandler, Arizona, EUA. Enero.

De Alba. (2004). Un procedimiento Heurístico para un Problema de Diseño de Redes Multiproducto con Capacidad Finita y Cargos Fijos", Tesis Doctoral, FIME, UANL, San Nicolás de los Garza, NL, Febrero.

Díaz, A.; Glover F.; Ghaziri, H. M.; González-Velarde, J.L.; Laguna, M.; Moscato, P. y Tseng, F.T. (1996). *Optimización Heurística y Redes Neuronales*, Editorial Paraninfo. Madrid, España.

Dowsland, Kathryn A. y Adenso Díaz, Belarmino. (2003). Diseño de Heurísticas y Fundamentos del Recocido Simulado. *Inteligencia Artificial, Revista Iberoamericana Artificial,* No. 19, pp. 93-102.

Feo, T.A. y Resende, M.G.C. (1995). Greedy randomized adaptive search procedures. *Journal of Global Optimization*, Vol. 6, No.2, pp. 109-133.

Fisher, H. y Thompson, G.L. (1963). Probabilistic learning combinations of local job-shop scheduling rules, *Industrial Scheduling*, J.F. Muth and G.L. Thompson (editores), Prentice-Hall, pp. 225-251.

Gendreau, M.; Laporte, G. y J-Y, Potvin. (1995). Metaheuristics for the vehicle routing problem, J.K. Lenstra and E.H.L. Aarts (editores), *Local Search Algorithms*, John Wiley & Sons, Chichester.

Gendron, B. (2002). A note on "A dual-ascent approach to the fixed-charge capacitated network design problems". *European Journal of Operational Research*, Vol. 138, No. 3, pp. 671-675.

Gendron, B. y Crainic, T.G. (1996). Bounding procedures for multicommodity capacitated fixed charge network design problems. Publication CRT-96-06, Centro de Investigación del Transporte, Universidad de Montreal, Canadá, Enero.

Gendron, B. y Crainic, T.G. (1994b). Relaxations for multicommodity capacitated network design problems. Publication CRT-965, Centro de Investigación del Transporte, Universidad de Montreal, Canadá, Febrero.

Glover, Fred y Melián, Belén. (2003). Búsqueda Tabú. *Inteligencia Artifical, Revista Iberoamericana Artificial,* No. 19, pp. 29-48.

Glover, F. y Laguna, M. (1997). *Tabu Search.* Editorial Klumer. Academic

Publisher, Norwell, MA.

Glover, F. y Laguna, M. (1997). General purpose heuristics for integer programming. Part I, *Journal of Heuristics*, Vol 2, No. 4, pp. 343-358.

Glover, F. (1989). Tabu search - Part I, *ORSA Journal on Computing*, Vol. 1, pp. 190-206.

Glover, F. (1968). Surrogate constraints, *Operations Research*, Vol. 16, pp. 741-749.

Glover, F. (1963). Parametric combinations of local job shop rules, Chapter IV, ONR Research Memorandum, No. 117, GSIA, Carnegie-Mellon University, Pittsburgh, PA.

Glover, F. y Kochenberger, Gary A. (2003). *Handbook of Metaheuristics*. Editorial Klumer. Academic Publisher, Norwell, MA.

Herrmann, J.W.; Ioannou, G.; Minis, I. y Proth, J.M. (1996). A dual ascent approach to the fixed-charge capacitated network design problem. *European Journal of Operational Research,* Vol. 95, No. 4, pp. 476-490.

Holmberg, K. y Hellstrand, J. (1998). Solving the uncapacitated network design problem by a lagrangean heuristic and branch-and-bound. *Operations Research*, Vol. 46, No.2, pp. 247-258.

Holmberg, K. y Yuan, D. (2000). A Lagrangian heuristic based branch-and-bound approach for the capacitated network design problem. *Operations Research*, Vol. 48, No.3, pp. 461-481.

Holmberg, K. y Yuan, D. (1998). A Lagrangean approach to network design problems. *Int. Trans. Operational Research*. Vol. 5, No.6, pp. 529-539.

Johnson, D. y Garey, M. (1979). Computers and Intractability. A guide to the theory of np-completeness. W.H. Freeman and Company, New York.

Khang, D.B. y Fujiwara, O. (1991). Approximate solutions of capacitated fixed-charge minimum cost network flow problems. *Networks*, Vol. 21. pp. 689-704.

Kirkpatrick, S; Gelatt, C.D. y Vecchi, M..P. (1983). Optimization by Simulated

Annealing. *Science*, No. 220, pp. 671-680.

Magnanti, T. y Wong, R. (1984). Network design and transportation planning: models and algorithms. *Transportation Science,* Vol. 18, No. 1, pp. 1 – 55.

Martí, Rafael y Laguna, Manuel. (2003). Scatter Search: Diseño Básico y Estrategias Advanzadas. *Inteligencia Artificial, Revista Iberoamericana Artificial,* No. 19, pp. 123-130.

Melián, Belén; Moreno Pérez, José A.; Moreno Vega, Marcos J. (2003). Metaheuristics: A global view. *Inteligencia Artifical, Revista Iberoamericana Artificial,* No. 19, pp. 7-28.

Osman, I. H. y Kelly, J.P. (1996). Metaheuristics: Theory and applications, Kluwer Academic Publishers, Norwell, MA.

Papadimitriou, C,H. y Steiglitz. (1982). Combinatorial optimization: Algorithms and complexity, Prentice Hall, New York.

Resende, Mauricio y Gónzales Velarde, José Luis. (2003). GRASP: Procedimientos de búsqueda miopes aleatorizados y adaptativos. *Inteligencia Artificial, Revista Iberoamericana Artificial*, No. 19, pp. 61-79.

Rochat, Y y Semet, F. (1994). A tabu search approach for delivering pet food and flour, *Journal of the Operational Research Society*, Vol. 45, No. 11, pp. 1233-1246.

Sridhar, V. y Park, J.S. (2000). Benders-and-cut algorithm for fixed-charge capacitated network design problem. *European Journal of Operational Research*, Vol. 125, No. 3, pp. 622-632, 2000

Skorin-Kapov, D. y Skorin-Kapov, J. (1994). On tabu search for the location of interacting hub facilities, *European Journal of Operational Research*, Vol. 73, No. 3, pp. 501-508.

LISTA DE TABLAS

LISTA DE FIGURAS

APÉNDICE A

INSTANCIAS

El nombre de las instancias se compone de la siguiente forma. El primer componente es la letra o, enseguida se indica la cantidad de nodos (30 ó 50), después se indica el número de productos (10, 50 ó 100), a continuación se especifica si el tipo de costo predominante es fijo (F) ó variable (V), finalmente se indica si la red es holgada (L) ó restringida (T). Por ejemplo el archivo o3010fl1 indica que la red está constituida por 30 nodos, 10 productos, con costo fijo alto y red tipo holgada. A continuación se muestran las instancias que utilizamos en esta tesis.

N	P	C	R	Prob
30	10	F	L	o3010fl1
30	10	F	L	o3010fl2
30	10	F	L	o3010fl3
30	10	F	L	o3010fl4
30	10	F	L	o3010fl5
30	10	F	T	o3010ft1
30	10	F	T	o3010ft2
30	10	F	T	o3010ft3
30	10	F	T	o3010ft4
30	10	F	T	o3010ft5
30	10	V	L	o3010vl1
30	10	V	L	o3010vl2
30	10	V	L	o3010vl3
30	10	V	L	o3010vl4
30	10	V	L	o3010vl5
30	10	V	T	o3010vt1
30	10	V	T	o3010vt2
30	10	V	T	o3010vt3
30	10	V	T	o3010vt4
30	10	V	T	o3010vt5

N	P	C	R	Prob
30	50	F	L	o3050fl1
30	50	F	L	o3050fl2
30	50	F	L	o3050fl3
30	50	F	L	o3050fl4
30	50	F	L	o3050fl5
30	50	F	T	o3050ft1
30	50	F	T	o3050ft2
30	50	F	T	o3050ft3
30	50	F	T	o3050ft4
30	50	F	T	o3050ft5
30	50	V	L	o3050vl1
30	50	V	L	o3050vl2
30	50	V	L	o3050vl3
30	50	V	L	o3050vl4
30	50	V	L	o3050vl5
30	50	V	T	o3050vt1
30	50	V	T	o3050vt2
30	50	V	T	o3050vt3
30	50	V	T	o3050vt4
30	50	V	T	o3050vt5

N	P	C	R	Prob
30	100	F	L	o30100fl1
30	100	F	L	o30100fl2
30	100	F	L	o30100fl3
30	100	F	L	o30100fl4
30	100	F	L	o30100fl5
30	100	F	T	o30100ft1
30	100	F	T	o30100ft2
30	100	F	T	o30100ft3
30	100	F	T	o30100ft4
30	100	F	T	o30100ft5
30	100	V	L	o30100vl1
30	100	V	L	o30100vl2
30	100	V	L	o30100vl3
30	100	V	L	o30100vl4
30	100	V	L	o30100vl5
30	100	V	T	o30100vt1
30	100	V	T	o30100vt2
30	100	V	T	o30100vt3
30	100	V	T	o30100vt4
30	100	V	T	o30100vt5

N	P	C	R	Prob
50	10	F	L	o5010fl1
50	10	F	L	o5010fl2
50	10	F	L	o5010fl3
50	10	F	L	o5010fl4
50	10	F	L	o5010fl5
50	10	F	T	o5010ft1
50	10	F	T	o5010ft2
50	10	F	T	o5010ft3
50	10	F	T	o5010ft4
50	10	F	T	o5010ft5
50	10	V	L	o5010vl1
50	10	V	L	o5010vl2
50	10	V	L	o5010vl3
50	10	V	L	o5010vl4
50	10	V	L	o5010vl5
50	10	V	T	o5010vt1
50	10	V	T	o5010vt2
50	10	V	T	o5010vt3
50	10	V	T	o5010vt4
50	10	V	T	o5010vt5

N	P	C	R	Prob
50	50	F	L	o5050fl1
50	50	F	L	o5050fl2
50	50	F	L	o5050fl3
50	50	F	L	o5050fl4
50	50	F	L	o5050fl5
50	50	F	T	o5050ft1
50	50	F	T	o5050ft2
50	50	F	T	o5050ft3
50	50	F	T	o5050ft4
50	50	F	T	o5050ft5
50	50	V	L	o5050vl1
50	50	V	L	o5050vl2
50	50	V	L	o5050vl3
50	50	V	L	o5050vl4
50	50	V	L	o5050vl5
50	50	V	T	o5050vt1
50	50	V	T	o5050vt2
50	50	V	T	o5050vt3
50	50	V	T	o5050vt4
50	50	V	T	o5050vt5

N	P	C	R	Prob
50	10	F	L	o5010fl1
50	10	F	L	o5010fl2
50	100	F	L	o50100fl1
50	100	F	L	o50100fl2
50	100	F	L	o50100fl3
50	100	F	L	o50100fl4
50	100	F	L	o50100fl5
50	100	F	T	o50100ft1
50	100	F	T	o50100ft2
50	100	F	T	o50100ft3
50	100	F	T	o50100ft4
50	100	F	T	o50100ft5
50	100	V	L	o50100vl1
50	100	V	L	o50100vl2
50	100	V	L	o50100vl3
50	100	V	L	o50100vl4
50	100	V	L	o50100vl5
50	100	V	T	o50100vt1
50	100	V	T	o50100vt2
50	100	V	T	o50100vt3

APÉNDICE B

DESCRIPCIÓN DE ARCHIVOS DE INSTANCIAS

A continuación se muestra un ejemplo del formato de un archivo que contiene una red, la cual esta constituida por 4 diferentes tipos de registros.

```
 6 10 2          →  Registro tipo 1     (nodos, aristas, productos)
1 2 5 10
   0  0 3
   1  0 2
1 3 3 30         →  Registro tipo 2     Para cada arista: (origen, destino, CF, capacidad)
   0  0 1                               arista {1, 3}, costo fijo = 3, capacidad = 30
   1  0 7
2 3 8 20
   0  0 1
   1  0 8                              Para cada producto que circule en la arista:
2 4 9 15                                (número de producto, CV)
   0  0 2         →  Registro tipo 3     1er producto, costo variable en {2, 4} = 2
   1  0 1                               2do producto, costo variable en {2, 4} = 1
2 5 9 10
   0  0 5
   1  0 4
3 4 3 8
   0  0 3
   1  0 1
3 5 2 5
   0  0 1
   1  0 6
4 5 1 4
   0  0 7
   1  0 2
4 6 5 15
   0  0 9
   1  0 8
5 6 4 10
   0  0 3
   1  0 5                              Para cada producto: (origen, destino, demanda)
 1 4 20          →  Registro tipo 4     1er producto, origen = 1, destino = 4, demanda = 20
 2 6 10                                 2do producto, origen = 2, destino = 6, demanda = 10
```

Enseguida se muestra el archivo que contiene los caminos de la red anterior, donde el primer número es el producto de su respectivo camino (el cual comienza en 0, refiriéndose al primer producto), el segundo número esta relacionado con la cantidad de nodos que contiene el camino y los números restantes son las aritas que constituyen dicho camino.

La asignación del nombre de los caminos esta constituido por una letra h al inicio seguida de 2 números. Donde la letra h la nos indica que se trata de un camino, el primero número se refiere al producto y el segundo se refiere al número del camino.

										Nombre del Camino
0	4	1	3	2	4					h11
0	3	1	3	4						h12
0	3	1	2	4						h13
0	4	1	2	3	4					h14
0	5	1	3	5	2	4	→	Prod 1, 5 nodos, camino: 1 - 3 - 5 - 2 - 4		h15
0	4	1	3	5	4					h16
0	5	1	2	3	5	4				h16
0	5	1	2	5	3	4				h18
0	5	1	3	2	5	4				h19
-1	-1						→	Fin del producto 1		
1	4	2	4	5	6					h21
1	3	2	4	6						h22
1	3	2	5	6						h23
1	5	2	4	3	5	6				h24
1	4	2	5	4	6		→	Prod 2, 4 nodos, camino: 2 - 5 - 4 - 6		h25
1	5	2	3	4	5	6				h26
1	6	2	1	3	4	5	6			h26
1	4	2	3	4	6					h28
1	5	2	1	3	4	6				h29
-1	-1						→	Fin del producto 2		

APÉNDICE C

CPLEX

CPLEX provee un optimizador de alto desempeño para programación lineal. Su nombre proviene de la combinación de la letra "C" del lenguaje de programación y la palabra "simplex" del método simplex de programación lineal. Además es robusto, y flexible para resolver problemas de optimización como programación lineal (LP), problemas enteros mixtos (MIP) y cuadráticos (QP).

Además es una herramienta diseñada para facilitar el desarrollo de aplicaciones debido a sus librerías que permiten a los desarrolladores a incrustarlas en sus programas de modo transparente y eficientemente en sus aplicaciones, para resolver, modificar e interpretar los resultados de programas de optimización para problemas de optimización, como modificar la configuración de los parámetros del optimizador, los cuales influyen en el desempeño y velocidad de la solución de un problema. También permite ser accesada desde la mayoría de los ambientes de programación y usada por una gran variedad de plataformas, tales como UNIX y WINDOWS, proporcionando transportabilidad.

Opciones del Optimizador

Mostramos a continuación los parámetros usados por el optimizador y sus valores implementados para la realización de nuestro algoritmo:

Configuración de parámetros

1. `CPXlpopt`: Especifica que el tipo de problema a optimizar es un LP. (`CPXlpopt`).
2. `LPMETHOD`: Nos permite seleccionar un optimizador LP, por lo que se especifica el método primal simplex que se usará para Optimizar (`LPMETHOD = ALG_PRIMAL`).

3. PREIND: Parámetro que habilita el preoptimizador del preprocesamiento en el problema (PREIND = OFF), la cual incrementa la velocidad total de solución.
4. DATACHECK: Habilita el indicador que nos permite verificar que los datos sean consistentes (DATACHECK = ON).
5. CPX_MIN: Especifica el sentido es de minimización para la optimización del problema (CPX_MIN).
6. CPXcopybase y CPXreadcopybase: Nos ayuda a comenzar a partir de una base inicial nuestro problema.
7. ITLIM: Especifica el máximo número de iteraciones de simplex, esto nos ayuda a que itere paso a paso (ITLIM = 0).

APÉNDICE D

PARÁMETROS

Para ejecutar nuestro algoritmo se deben especificar los siguientes parámetros en la línea de comando:

nom_programa	Nombre de la aplicación que contiene nuestro algoritmo.
nom_problema	Nombre del archivo que contiene el problema de red.
num_col_gen	Número de caminos generados en la fase de generación de caminos.
max_move	Número de iteraciones que se realizarán en la fase de búsqueda local de nuestro algoritmo.
max_colgen	Número de iteraciones que se realizarán en la fase de generación de caminos de nuestro algoritmo.
max_div	Número de iteraciones que se realizarán en la fase de diversificaión de nuestro algoritmo.
tabu_h	Número de tabu tenures para los caminos.
tabu_ari	Número de tabu tenures para las aristas.

Ejemplo: gts_v1 o3010fl1 1 1 2 3 3 3 3

Una vez especificado los parámetros antes mencionados, nuestra aplicación comienza la lectura del nombre del archivo que contiene la información de la red especificada, después se manda llamar al GRASP, el cual genera una solución inicial (caminos que contendrá nuestro diseño de red), la cual es guardada en un archivo tipo texto, posteriormente prosigue con la lectura del archivo que contiene los caminos o rutas (nombre del archivo que contiene la red pero con terminación _k1) y después manda a llamar al optimizador CPLEX para realizar pivoteos y obtener información mediante estos.

APÉNDICE E

PROGRAMAS USADOS POR NUESTRA BÚSQUEDA TABÚ

El algoritmo propuesto está compuesto de los siguientes programas o funciones:

- Red2.c : Función que lee el archivo que contiene información de la red (instancia del problema).
- Grasp1.c: Programa que genera una solución inicial factible de un problema en particular.
- Ruta31.c: Función que lee el archivo que contiene las rutas más cortas.
- Problema1.c: Función que genera el problema para que sea leído por el optimizador CPLEX.
- Kshort.c: Función que genera un número predefinido de rutas para un producto en particular, mediante la asignación de longitud de cada arista que se transporta en la red. Estas son generadas cuando las rutas leídas son tabú activas (no pueden emplearse) o ya no puedan usarse por estar a su máxima capacidad.
- Gelim.c: Función que elimina duplicación de rutas repetidas por el algoritmo.

APÉNDICE F

COMPILACIÓN DEL CÓDIGO

Para la creación de nuestro programa ejecutable haremos uso de la herramienta **make**, la cual nos permite recompilar aquellos módulos o programas que hayan sido modificados. Para esto se debe proporcionar un archivo de comandos (**Makefile**) que contiene las dependencias entre los archivos y las órdenes necesarias para actualizar los programas ejecutables. Un archivo **Makefile** es un archivo ordinario de texto que contiene dependencias y comandos en un formato especificado por **make**, el cual esta constituido por un número de archivos con extensiones *.h* y *.c*, donde si alguno de ellos sufre alguna modificación, sólo será recompilado aquel que haya sido modificado y los programas que dependan de él.

Para crear un archivo ejecutable se debe escribir en la linea de comando lo siguiente:

make [objetivos]

donde **objetivos** serán los nombres de los archivos que se buscarán en **Makefile** y se ejecutarán las operaciones para generar cada uno de ellos. Para mayor información ver (Kelley y Pohl, 1998).

A continuación mostramos los archivos que utilizamos para la realización de nuestro algoritmo:

- Archivos.c: gts_b.c, ts_b.c, memoria_b.c, fct31_b.c, red2_b.c, ruta31_b.c, problema1_b.c, memory_b.c, kshort_b.c, shortest_b.c, gelim1_b.c, grasp2_b.c,
- Archivos.h: grasp2_b.h, shortest_b.h, prueba2_b.h

```
# Makefile: make gts_b
# ------------------------------------------------------------

# Compiler selection
# ------------------------------------------------------------
GCC = /usr/local/bin/gcc

# Compiler options
# ------------------------------------------------------------
COPT  = -xtarget=ultra -xarch=v9 -KPIC

# PATHS
# ------------------------------------------------------------
MICPLEXDIR    = /u/nadia/cplex_ejm/inst/kshort/modif/temp2
CPLEXDIR2     = /usr/local/ilog/cplex71
CONCERTDIR    = /usr/local/ilog/concert11
SYSTEM2       = ultrasparc_5_5.0
LIBFORMAT     = static_pic_mt

# Link options and libraries (contiene ilm.h y cplex.h)
# ------------------------------------------------------------
CONCERTINCDIR = $(CONCERTDIR)/include
CPLEXINCDIR   = $(MICPLEXDIR)
CPLEXLIBDIR3  = $(CPLEXDIR2)/lib/$(SYSTEM2)/$(LIBFORMAT)
CLNFLAGS3 = -L$(CPLEXLIBDIR3) -lcplex -lm -xarch=v9 -lsocket -lnsl

#  make all        : to compile the examples.
#  make execute  : to compile and execute the examples.
#-------------------------------------------------------
C_EX  = gts_b

all: $(C_EX)
execute: all
        make $(C_EX)
clean :
        /bin/rm -rf *.o *~

# Programs
# ------------------------------------------------------------
gts_b: gts_b.c ts_b.c memoria_b.c fct31_b.c red2_b.c ruta31_b.c problema1_b.c memory_b.c
kshort_b.c shortest_b.c gelim1_b.c grasp2_b.c
$(GCC) -o gts_b gts_b.c ts_b.c memoria_b.c fct31_b.c red2_b.c ruta31_b.c problema1_b.c
memory_b.c kshort_b.c shortest_b.c gelim1_b.c grasp2_b.c $(CLNFLAGS3)
```

FICHA AUTOBIOGRÁFICA

Nadia Cobos Zaleta

Candidato para el grado de Maestro en Ciencias en Ingeniería de Sistemas

Universidad Autónoma de Nuevo León

Facultad de Ingeniería Mecánica y Eléctrica

Tesis

Búsqueda Tabú para un Problema de Diseño de Red Multiproducto con Capacidad Finita en las Aristas

Nacida en Poza Rica de Hidalgo, Veracruz. Hija de Sr. Macario Cobos Ríos y la Sra. María Magdalena Zaleta Morgado, tercera de tres hermanos. Graduada en la Facultad de Ingeniería Mecánica y Eléctrica en la Universidad Autónoma de Nuevo León (1995-2000) como Ingeniero Administrador de Sistemas con titulación honorífica. Se desempeñó en el Área de Ingeniería en GRUPO SCAN como Analista y Desarrollador de Sistemas (1999-2001) e ingresó a la Maestría en Ciencias en Ingeniería de Sistemas en Febrero del 2001. Obtuvo una beca de CONACYT de manutención para realizar sus estudios de maestría y al mismo tiempo se desempeñó como Asistente de Investigación en el PISIS.