



ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Instituto de Ciencias Matemáticas

Ingeniería en Estadística Informática

“Uso e Implementación de Métodos Meta-Heurísticas de tipo Tabú para problemas tipo duros”

TESIS DE GRADO

Previa a la obtención del Título de:

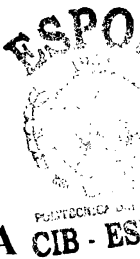
INGENIERO EN ESTADÍSTICA E INFORMÁTICA

Presentada por:

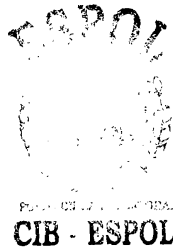
Jeffersson Saúl Reyes Lasso

GUAYAQUIL – ECUADOR

Año
2003



AGRADECIMIENTO



A todas las personas que de una u otra manera colaboraron en la realización de este trabajo, especialmente a la familia Jarrín Torres y a la gran ayuda de mi Director Mat. Fernando Sandoya

DEDICATORIA



A Dios

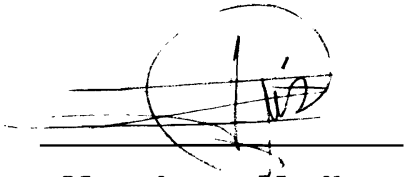
A mis padres

A mis hermanos

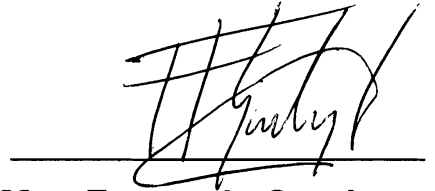
Y a todos mis

familiares

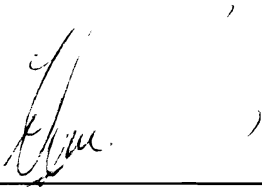
TRIBUNAL DE GRADUACIÓN



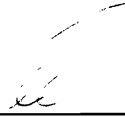
Mat. Jorge Medina
DIRECTOR DEL ICM



Mat. Fernando Sandoya
DIRECTOR DE TESIS



Ing. Washington Armas
VOCAL



Dr. Cristóbal Mera
VOCAL

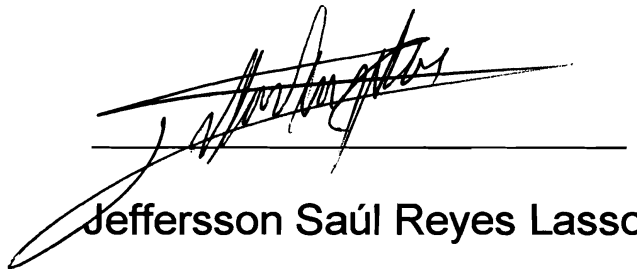


DECLARACIÓN EXPRESA

“ La responsabilidad del contenido de esta tesis de grado, me corresponde exclusivamente; y el patrimonio intelectual de la misma a la ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL”



(Reglamento de graduación de la ESPOL)



Jeffersson Saúl Reyes Lasso

RESUMEN

El presente trabajo desarrolla una herramienta de ayuda para la toma de decisiones, en cualquier empresa que tenga vehículos de transporte para satisfacer algún tipo de servicio a ciertos clientes determinados, el nombre del proyecto es: "Uso e Implementación de Métodos Meta heurísticos de tipo TABU para resolución de problemas de optimización Duros".

En la primera parte se da una breve introducción acerca de los cambios y evolución que ha tenido la optimización y su beneficios para la sociedad.

El segundo capítulo, hace referencia a los métodos más básicos existentes para optimizar.

La tercera parte nos presenta los nuevos métodos para resolver problemas más complejos.

El cuarto capítulo, nos explica acerca del método TABÚ, un poco de historia y además la estructura de funcionamiento.

La quinta parte, se ve la aplicación del método TABÚ en el problema de ruteo de vehículo.

Finalmente en el último capítulo se encuentran principales conclusiones y recomendaciones obtenidas luego de concluir el proyecto.

INDICE GENERAL

	Pág.
RESUMEN	II
ÍNDICE GENERAL	III
ÍNDICE DE FIGURAS	VI
ÍNDICE TABLAS	VII
ABREVIATURAS	VIII
SIMBOLOGÍA	IX

I.	INTRODUCCIÓN	
II.	MÉTODOS DE OPTIMIZACIÓN EXACTA EN NÚMEROS ENTEROS	
	2.1. PROGRAMACIÓN LINEAL	8
	2.1.2. PROGRAMACIÓN LINEAL ENTERA.	13
	2.1.2.1. BRANCH & BOUND (B&B)	15
III.	TÉCNICAS DE OPTIMIZACIÓN METAHEURÍSTICA	
	3.1 RESEÑA DE TÉCNICAS DE OPTIMIZACIÓN META HEURÍSTICA	22
	3.1.1 MÉTODOS DE BÚSQUEDA LOCAL	23
	3.1.2 RECOCIDO SIMULADO	26
	3.2. TÉCNICA DE BÚSQUEDA DE HORMIGAS	27
	3.3. TÉCNICA DE ALGORITMOS GENÉTICOS	32
IV.	RESEÑA DEL MÉTODO TABÚ	
	4.1.1 RESTRCCIONES TABÚ	38

4.1.2 ESTRATEGIAS DE APRENDIZAJE DE MEDIO Y

LARGO PLAZO 39

4.1.3 ESTRATEGIA A CORTO PLAZO 39

4.1.4 ESTRATEGIA DE INTENSIFICACIÓN 40

4.1.5 ESTRATEGIA DE DIVERSIFICACIÓN 40

4.1.6 ASPIRACIÓN PRO DEFECTO 42

4.1.7 ASPIRACIÓN POR OBJETIVO GLOBAL 43

4.1.8 ASPIRACIÓN PRO OBJETIVO REGIONAL 43

4.1.9 ASPIRACIÓN POR DIRECCIÓN DE BÚSQUEDA 43

4.1.10 ASPIRACIÓN DE INFLUENCIA 43

4.1.11 MOVIMIENTOS DE INFLUENCIA 46

4.1.12 OSCILACIÓN ESTRATÉGICA 46

4.1.13 ELECCIONES PROBABILÍSTICAS 47

4.1.14 UMBRALES TABÚ 47

4.1.15 REENCADENAMIENTO DE TRAYECTORIAS 48

V. DISEÑO E IMPLEMENTACIÓN DEL MODELO**5.1 DISEÑO DEL MODELO META-HEURÍSTICO****PARA EL PROBLEMA DE RUTEO DE VEHÍCULOS****51****5.1.1 ESTRATEGIA PROHIBIDA****54****5.1.2 ESTRATEGIA DE LIBERACIÓN****55****5.1.3 ESTRATEGIA A CORTO PLAZO****56****5.2 SOLUCIÓN DE PROBLEMAS DE PRUEBA PARA TABÚ****58****VI. CONCLUSIONES Y RECOMENDACIONES****64****BIBLIOGRAFÍA****INDICE DE FIGURAS**

	Pág.
Figura 5.1 Rutas de Vehículos	51
Figura 5.2 Recorrido del vehículo 1	59
Figura 5.3 Recorrido del vehículo 2	60

Figura 5.4 Recorrido del vehículo 3	60
Figura 5.5 Recorrido del vehículo 4	61
Figura 5.6 Recorrido del vehículo 2	62
Figura 5.7 Recorrido del vehículo 1	62
Figura 5.8 Recorrido del vehículo 3	63

INDICE DE TABLAS

	Pág.
Tabla I. Solución Inicial	59
Tabla II. Solución Encontrada	61

ABREVIATURAS

MT..- Método Tabú

NP .- No polinomial

PL .- Programación lineal

PE .- Programación entera

PEM .- Programación entera mixta

PEB .- Programación entera binaria

B&B .- Ramificación y acotamiento

CI .- Cota inferior

AG .- Algoritmo Genético

Capítulo 1

Introducción

La ciencia y la tecnología han crecido de una manera muy significativa en las últimas décadas, con lo cual el ser humano ha obtenido así respuesta a una gran diversidad de problemas. Todos estos avances que se han logrado nos han ayudado a conocer y comprender de una manera más eficiente el entorno que nos rodea. Además, dentro de estos pasos gigantescos en la evolución del ser humano, se han descubierto productos sustitutos como la luz, energía como el combustible, así como otras invenciones y descubrimientos que han ayudado a satisfacer de alguna manera las necesidades que tenemos. Con este afán, se han constituido miles de empresas alrededor del mundo, las cuales ofrecen productos y servicios varios de acuerdo a los requerimientos de su mercado objetivo, que cambia constantemente, por lo que siempre debe estar en continuo mejoramiento para la elaboración de nuevos productos y/o servicios de mejor calidad.

Enfoquémonos en una empresa que produce cierto producto, cualquiera que sea este, a lo largo de varios años la principal preocupación era cumplir una meta fijada, que generalmente tenía que ver con incrementar sus ventas de cualquier manera, para obtener así una utilidad mayor y en la mayoría de las

ocasiones lo lograban, pero sin saber si utilizaban sus recursos de la mejor manera. Como cumplían sus metas se podía decir que eran eficaces en la venta y elaboración de su producto. Al pasar el tiempo notaron que podían de mejorar el uso de la materia prima, debido que en muchos casos no la aprovechaban de mejor manera, entonces comenzó a surgir en la empresa las preguntas ¿Cómo utilizar óptimamente sus recursos humanos, financieros, de capital?, ¿Cuántos productos tenían que elaborar de cada tipo para maximizar sus utilidades?. Con estas preguntas claramente podemos considerar a la situación que tienen las empresas, como un problema de toma de decisiones, que para poder llegar a una solución se requiere la identificación de tres componentes principales:

- Las alternativas
- Las Restricciones
- Un criterio objetivo para la evaluación de las alternativas

Estos componentes, son los principales en la estructura que servirán para formar un modelo típico de investigación de operaciones (IO), donde el resultado final es un modelo matemático que relaciona las variables, las restricciones y la función objetivo. La solución del modelo produce entonces los valores de las variables de la decisión que optimizan (maximizan o minimizan) el valor de la función objetivo y al mismo tiempo satisfacen las

restricciones. A la solución resultante se la conoce como solución factible óptima. Estos modelos regularmente se organizan así:

- Maximizar o minimizar Función Objetivo
- Sujeto a restricciones

Una de las técnicas más simples es la programación lineal, donde todas las funciones objetivo y las restricciones son lineales, además todas las variables son continuas, de hecho estos supuestos difícilmente son satisfechos en la realidad. Otras técnicas conocidas son: programación dinámica, programación entera, programación no lineal, programación de metas, programación de redes, optimización combinatoria, etc.

La técnica de la programación Lineales bastante útil y a la vez básica, por lo que para poder utilizarla debe existir una relación lineal entre las variables y estas deberán tener un carácter continuo. Esto significa que el cambio de una variable estará acompañada por un cambio proporcional en otra. Entonces conociendo esto la pregunta en las empresas fue replanteada así: ¿Cómo minimizar los costos y a la vez maximizar las utilidades?; o simplemente: ¿Cómo optimizar un proceso?. Aquí la palabra optimizar tomó un significado importante, debido a cuando es utilizada estamos refiriéndonos a un valor óptimo describimos un valor en el cual se cumplen todas las

restricciones y Maximiza o Minimiza el objetivo. Con esto las empresas no sólo podrían contentarse con ser eficaces sino también eficientes en la producción de sus productos.

Algunos modelos matemáticos de optimización son tan complejos como la elaboración de horarios de clase, recolección de basura, ruteo de vehículos, entre otros; que es imposible resolverlos mediante cualquiera de los métodos exactos de optimización disponibles. Enfrentando el desafío de resolver problemas de ese tipo, que abundan en el mundo real, los métodos clásicos encuentran a menudo gran dificultad, por lo que es necesario abandonar la búsqueda de la solución óptima exacta y sólo buscar una “buena” solución, utilizando los denominados métodos heurísticos. Estos métodos no garantizan la optimalidad de la solución que ha sido encontrada. En los últimos años ha habido un crecimiento espectacular en el desarrollo de procedimientos heurísticos, y esto es, debido a la necesidad de ofrecer rápidas y buenas soluciones a los problemas reales.

Se los denomina heurísticos también, debido a que estaban desarrollados para resolver sólo un tipo de problemas y para ningún otro más. A estos tipos de problemas se los conoce como NP-duros o NP-difíciles (de las siglas Non Polinomial), donde cualquier procedimiento para hallar la solución óptima exacta tiene complejidad exponencial. Existen diversos métodos heurísticos

tales como: Métodos constructivos, de descomposición, de reducción, de manipulación del modelo y de búsqueda local. Estos métodos son utilizados para resolver problemas de optimización combinatoria.

En la actualidad, el objetivo principal de los investigadores en esta área es la de diseñar métodos generales para poder resolver clases o categorías de problemas. A estos nuevos métodos se los denomina Meta-heurísticos.

Los procedimientos Meta-heurísticos son una clase de métodos aproximados que están diseñados para resolver problemas difíciles de optimización combinatoria, en los que los heurísticos clásicos no son ni efectivos ni eficientes. Los Meta-heurísticos proporcionan un marco general para crear nuevos algoritmos híbridos combinando diferentes conceptos derivados de: inteligencia artificial, evolución biológica y mecanismos estadísticos.

Entre los procedimientos Meta-Heurísticos tenemos:

- Búsqueda Tabú
- Recocido Simulado
- Algoritmos Genéticos
- GRASP

En el capítulo III conoceremos más sobre los métodos anteriormente mencionados, pero intimaremos principalmente con el Método de Búsqueda Tabú.

Los orígenes del Método de Búsqueda Tabú (MT) puede situarse en diversos trabajos publicados hace alrededor de 20 años por Glover. Este método cambia nuestra habilidad de resolver problemas de importancia en la práctica. La aplicación en la actualidad puede ser diversas ramas, tales como: las telecomunicaciones, análisis financiero, ingeniería molecular, exploración mineral, análisis biomédico, conservación ambiental, entre otros.

El MT es una técnica para resolver problemas combinatorios de gran dificultad que está basada en principios generales de la Inteligencia Artificial. En esencia es un procedimiento meta-heurísticos que puede ser utilizado para guiar cualquier procedimiento de búsqueda local.

Una de las principales características del MT es la de poseer una memoria flexible de búsqueda. Básicamente, el empleo de dicha memoria flexible consiste en modificar (restringir y expandir) sobre la marcha el criterio de vecindad, de acuerdo a observaciones realizadas durante el proceso de búsqueda.

En el capítulo IV de la presente tesis se conocerá mucho más sobre el método Tabú, comprenderemos más sobre la resolución de una gran diversidad de problemas de tipo NP-duros, con el que obtendremos buenas soluciones y además se implementará el algoritmo de éste método con lo que se podrá demostrar su aplicabilidad.

Capítulo 2

2.1. METODOS DE OPTIMIZACION EXACTA EN NUMEROS ENTEROS

2.1.1 Programación Lineal

En los siglos XVII y XVIII, grandes matemáticos como **Newton**, **Leibnitz**, **Bernouilli** y, sobre todo, **Lagrange**, que tanto habían contribuido al desarrollo del cálculo infinitesimal, se ocuparon de obtener máximos y mínimos condicionados de determinadas funciones.

Posteriormente el matemático francés Jean Baptiste-Joseph **Fourier** (1768-1830) fue el primero en intuir, aunque de forma imprecisa, los métodos de lo que actualmente llamamos programación lineal y la potencialidad que de ellos se deriva.

Si exceptuamos al matemático **Gaspar Monge** (1746-1818), quien en 1776 se interesó por problemas de este género, debemos remontarnos al año 1939 para encontrar nuevos estudios relacionados con los métodos de la actual programación lineal. En este año, el matemático ruso Leonodas Vitalyevich **Kantarovitch** publica una

extensa monografía titulada *Métodos matemáticos de organización y planificación de la producción* en la que por primera vez se hace corresponder a una extensa gama de problemas una teoría matemática precisa y bien definida llamada, hoy en día, programación lineal .

En 1941-1942 se formula por primera vez el problema de transporte, estudiado independientemente por **Koopmans y Kantarovitch**, razón por la cual se suele conocer con el nombre de *problema de Koopmans-Kantarovitch*.

Tres años más tarde, **G. Stigler** plantea otro problema particular conocido con el nombre de régimen alimenticio optimal.

En estos años posteriores a la Segunda Guerra Mundial, en Estados Unidos se asumió que la eficaz coordinación de todas las energías y recursos de la nación era un problema de tal complejidad, que su resolución y simplificación pasaba necesariamente por los modelos de optimización que resuelve la programación lineal.

Paralelamente a los hechos descritos se desarrollan las técnicas de computación y los ordenadores, instrumentos que harían posible la resolución y simplificación de los problemas que se estaban gestando.

En 1947, **G.B. Dantzig** formula, en términos matemáticos muy precisos, el enunciado estándar al que cabe reducir todo problema de programación lineal. Dantzig, junto con una serie de investigadores del *United States Department of Air Force*, formarían el grupo que dio en denominarse *SCOOP (Scientific Computation of Optimum Programs)*.

Una de las primeras aplicaciones de los estudios del grupo SCOOP fue el puente aéreo de Berlín.

“En 1946 comienza el largo período de la guerra fría entre la antigua Unión Soviética (URSS) y las potencias aliadas (principalmente , Inglaterra y Estados Unidos). Uno de los episodios más llamativos de esa guerra fría se produjo a mediados de 1948, cuando la URSS bloqueó las comunicaciones terrestres desde las zonas alemanas en poder de los aliados con la ciudad de Berlín, iniciando el bloqueo de Berlín. A los aliados se les plantearon dos posibilidades: o romper el bloqueo terrestre por la fuerza, o llegar a Berlín por el aire. Se adoptó la decisión de programar una demostración técnica del poder aéreo norteamericano; a tal efecto, se organizó un gigantesco puente aéreo para abastecer la ciudad: en diciembre de 1948 se estaban transportando 4500 toneladas diarias; en marzo de 1949, se llegó a las 8000 toneladas, tanto como se transportaba por carretera y ferrocarril antes del corte de las comunicaciones. En la planificación de los



suministros se utilizó la programación lineal. (El 12 de mayo de 1949, los soviéticos levantaron el bloqueo).”

Luego de este acontecimiento se continuó con infinidad de aplicaciones de tipo preferentemente militar.

Hacia 1950 se constituyen, fundamentalmente en Estados Unidos, distintos grupos de estudio para ir desarrollando las diferentes ramificaciones de la programación lineal. Cabe citar, entre otros, Rand Corporation, con Dantzig, Orchard-Hays, Ford, Fulkerson y Gale, el departamento de Matemáticas de la Universidad de Princeton, con Tucker y Kuhn, así como la Escuela Graduada de Administración Industrial, dependiente del Carnegie Institute of Technology, con Charnes y Cooper.

Los fundamentos matemáticos de la programación lineal se deben al matemático norteamericano de origen húngaro **Janos von Neuman** (1903-1957), quien en 1928 publicó su famoso trabajo *Teoría de Juegos*. En 1947 conjetura la equivalencia de los problemas de programación lineal y la teoría de matrices desarrollada en sus trabajos. La influencia de este respetado matemático, discípulo de David Hilbert en Gotinga y, desde 1930, catedrático de la *Universidad de Princeton de Estados Unidos*, hace que otros investigadores se

interesaran paulatinamente por el desarrollo riguroso de esta disciplina.

Se ha estimado, de una manera general, que si un país subdesarrollado utilizase los métodos de la programación lineal, su producto interior bruto (PIB) aumentaría entre un 10 y un 15% en tan sólo un año.

La programación lineal (PL), que trata exclusivamente con funciones objetivos y restricciones lineales, es una parte de la programación matemática, y una de las áreas más importantes de la matemática aplicada. Se utiliza en campos como la ingeniería, la economía, la gestión, y muchas otras áreas de la ciencia, la técnica y la industria.

El propósito de la programación lineal es el de MAXIMIZAR o MINIMIZAR funciones lineales de la forma :

$$\text{Max / Min. } f(x) = C_1 X_1 + C_2 X_2 + \dots + C_n X_n = C^T X$$

s.a.r

$$a_{i1} X_1 + a_{i2} X_2 + a_{i3} X_3 + \dots + a_{in} X_n \geq 0$$

$$X_1 \geq 0, X_2 \geq 0 \dots X_n \geq 0$$

Escrito de manera matricial:



$$\text{Max / Min. } C^T X; X \in \mathfrak{R}^n$$

s.a.r

$$Ax \leq b; A \in \mathfrak{R}^{m \times n}$$

$$x \geq 0; C \in \mathfrak{R}^m$$

La solución que satisface todas las restricciones del modelo es una solución factible pero lo que en realidad nos interesa es la solución factible óptima que produce, la cual cumple con todas las restricciones y es el mejor resultado de los obtenidos.

2.1.2 Programación Lineal Entera.

Dentro de las muchas aplicaciones de la programación lineal nos podemos dar cuenta que una de las grandes limitaciones que impiden su empleo es la *suposición de divisibilidad*. Esta dice que las variables de decisión solo tienen un sentido real si su valor es entero. Por ejemplo, con frecuencia es necesario asignar personas, máquinas, objetos, animales o vehículos a las actividades en cantidades enteras. Si el hecho de exigir valores enteros es la única diferencia que tiene un problema con la formulación de programación lineal, entonces estamos hablando de un problema de *programación entera* (PE) (su nombre completo: *programación lineal entera*).

El modelo matemático para programación lineal entera es sencillamente el modelo de programación lineal con la restricción adicional de que las variables deben tener valores enteros (y la suposición de divisibilidad se cumple para el resto), el modelo se conoce como de *programación entera mixta (PEM)*. cuando se hace la distinción entre un problema con todas las variables enteras y el mixto, en el primer caso se llama de *programación entera pura*.

Se han desarrollado numerosas aplicaciones de programación entera que involucran una extensión directa de programación lineal en la que se debe eliminar la suposición de divisibilidad. Sin embargo, existe otra área de aplicación que puede ser mucho mas importante, como el problema que incluye cierto número de "decisiones si o no" interrelacionadas. En las decisiones de este tipo, las únicas dos elecciones posibles son si y no. Por ejemplo, ¿debe emprenderse un proyecto específico?, ¿debe hacerse una inversión fija específica?, ¿se debe localizar una instalación en un sitio en particular?.

Con solo dos posibilidades, este tipo de decisiones se puede representar mediante variables de decisión restringidas a solo dos valores, por ejemplo 0 y 1. Así, la j - ésima decisión si o no se puede representar por x_j , tal que

$$x_j = \begin{cases} 1, & \text{si la decisión } j \text{ es si} \\ 0, & \text{si la decisión } j \text{ es no} \end{cases}$$

Las variables de éste tipo se llaman variables binarias (o variables 0-1). De esta manera se pretende mostrar que los problemas de programación entera que contienen solo variables binarias son conocidos como de programación entera binaria (PEB) (o problemas 0 – 1 de programación entera).

El principal algoritmo para resolver un problema de programación entera es el algoritmo de Branch & Bound.

2.1.2.1 Branch & Bound (B&B)

Uno de los métodos de Programación Entera más usados es sin duda el método de Branch & Bound (B&B), al que se refieren a veces en español como método de Ramificación y Acotamiento. B&B es un método de optimización exacto el cual utiliza la programación lineal como herramienta para obtener una solución del problema sin tomar en cuenta la restricción de que las variables deben tomar solamente valores enteros. Aquí describiremos B&B brevemente para problemas de maximización (para minimización el proceso es similar).

El enfoque de B&B es resolver el problema como si se tratara de un problema de programación lineal, sin tomar en cuenta que las variables deben ser enteras. B&B empieza entonces utilizando alguna rutina de Programación Lineal (que llamaremos aquí **PL**). A esta iteración le llamaremos la iteración inicial. Si en la iteración inicial, la solución es entera, el problema se ha resuelto, de modo que la solución obtenida con **PL** es la solución óptima del problema entero. De otra manera, habrá que continuar buscando la solución del problema entero.

Si en la iteración inicial, la solución obtenida no es entera, se procede a seleccionar una variable no entera X_i . Se encuentran los dos enteros más próximos de X_i . Sean estos enteros X_{menor} y X_{mayor} los enteros más pequeño y más grande que X_i , respectivamente. B&B forma entonces dos nuevas restricciones que agrega una a una al problema original, de modo que se obtienen dos nuevos problemas. A la obtención de nuevos problemas se le llama ramificación. Acto seguido se procede a buscar una solución entera en cada uno de estos problemas, para lo cual se procede (para cada uno de ellos) como en la iteración inicial, usando **PL** para resolver los problemas de programación lineal correspondientes.

B&B utiliza para su análisis una cota inferior (para problemas de maximización), que denominaremos aquí **CI**. B&B solo examina aquellos problemas cuya solución proporcionada por **PL** estén por arriba de **CI**. De esta forma es necesario que el algoritmo de B&B determine una **CI** lo más pronto posible. Una forma común es empezar con una **CI** para los valores de las **X's** iguales a su cota inferior. No obstante existen rutinas de B&B que esperan a obtener una **CI** igual a la primera solución que solo tenga valores enteros. Nótese que en el caso de la iteración inicial el proceso termina si la solución tiene solo valores enteros. B&B cambia su **CI** cuando detecta que existe un problema con una solución entera cuyo valor de la función objetivo es mayor que la **CI** actual.

Una vez que B&B ha determinado una **CI**, si esta no corresponde a la iteración inicial, procede a obtener nuevos problemas mediante el proceso de partición. Solamente se analizan aquellos problemas que tengan como solución un valor de la función objetivo mayor que la última **CI** determinada. B&B termina cuando no existen más problemas por analizar.

El modelo matemático para programación lineal entera es sencillamente el modelo de programación lineal con la



restricción adicional de que las variables deben tener valores enteros (y la suposición de divisibilidad se cumple para el resto), el modelo se conoce como de programación entera mixta (PEM).

La forma general del problema que se va a estudiar es:

Paso inicial: se establece $Z^* = -\infty$. Se aplica el paso de acotamiento, el paso de sondeo y la prueba de optimalidad que se describen después del problema completo. Si no queda sondeado, se clasifica este problema como el único subproblema restante para realizar la primera iteración completa.

Pasos para cada iteración:

1. **Ramificación:** entre los subproblemas restantes (no sondeados), se selecciona el de más reciente creación. (Los empates se rompen con la cota más grande). Entre las variables restringidas a enteros, que tienen valores no enteros en la solución óptima de la soltura de PL del subproblema, se elige la primera en el orden natural, como la variable de ramificación. Sea x_i , esta variable y x_i^* , su valor en esta solución. Se ramifica desde el nodo del subproblema para

crear dos nuevos subproblemas agregando las restricciones respectivas $x_i \leq [x_i^*]$ y $x_i \geq [x_i^*] + 1$.

2. Acotamiento: para cada subproblema se obtiene su cota aplicando el método simplex (o el método simplex dual si se reoptimiza) a su soltura de PL y utilizando el valor de Z para la solución óptima resultante.

3. Sondeo: para cada nuevo subproblema se aplican las pruebas de sondeo que se dan en seguida y se descartan aquellos subproblemas que quedan sondeados por cualquiera de las pruebas.

prueba 1: su cota $\leq Z^*$, donde Z^* es el valor de Z en la solución de apoyo actual.

prueba 2: su soltura de PL no tiene soluciones factibles.

prueba 3: la solución óptima para su soltura de PL tiene valores enteros para todas sus variables restringidas a enteros. (Si esta solución es mejor que la de apoyo, se convierte en la nueva solución de apoyo y se vuelve a aplicar la prueba 1 con la nueva Z^* a todos los subproblemas no sondeados.)



Prueba de optimalidad: el proceso se detiene cuando no hay subproblemas restantes; la solución incumbente actual es óptima. De otra manera, se realiza otra iteración.

Ejemplo:

$$\text{Max} Z = 5X_1 + 4X_2$$

s.a.r

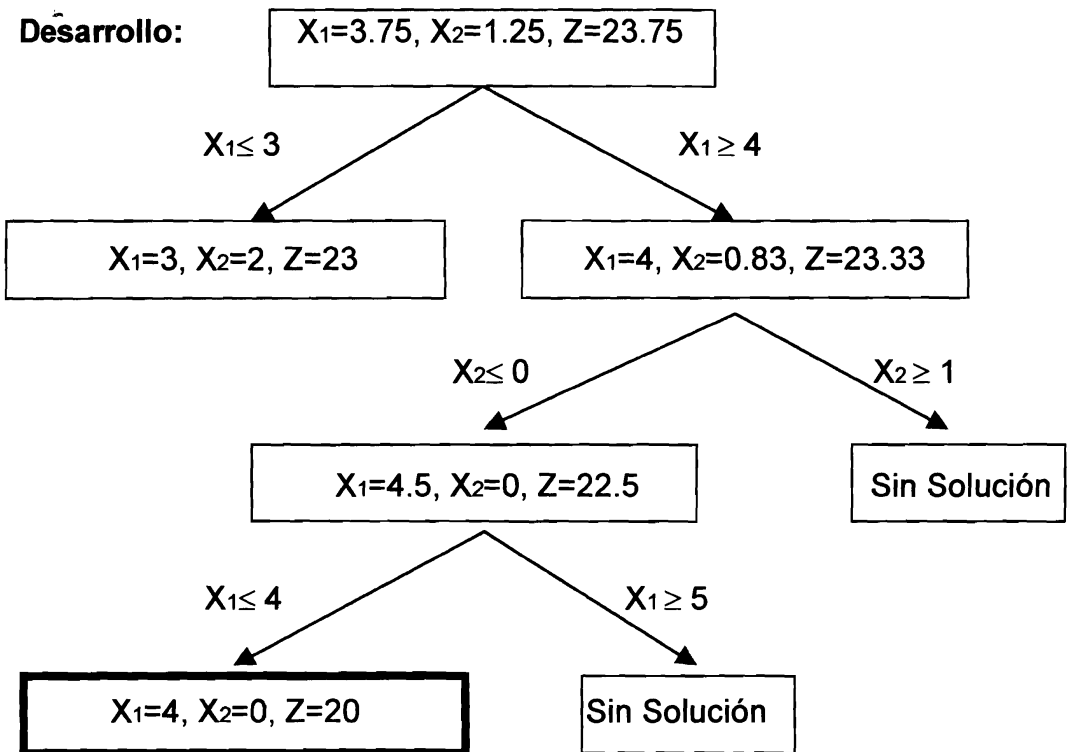
$$X_1 + X_2 \leq 5$$

$$10X_1 + 6X_2 \leq 45$$

$$X_1, X_2 \geq 0$$

$$X_1, X_2 \text{ entero}$$

Desarrollo:



ESPE

Podemos observar que en dos ramas de la resolución no se puede llegar a una solución pero luego al realizar otra iteración, por la otra rama obtuvimos una solución mejor.

Capítulo 3

TÉCNICAS DE OPTIMIZACION METAHEURISTICA

3.1 Reseña de técnicas de optimización meta heurística

El problema de encontrar el valor óptimo (máximo o mínimo) de una función definida sobre un conjunto finito (sea este muy grande o no) de puntos se conoce como problema de optimización combinatoria. Si tenemos una función f definida sobre un conjunto discreto \mathcal{S} ; la siguiente formulación:

$$\left\{ \begin{array}{l} \max/\min f(x) \\ \text{s.a.r} \\ x \in \mathcal{S} \end{array} \right.$$

describe el problema de optimización combinatoria de minimizar f sobre \mathcal{S} . Por lo general la notación más usada, f se conoce como función objetivo, y a \mathcal{S} se le denomina espacio de búsqueda. A los elementos del espacio de búsqueda se les llama soluciones factibles.

En la mayoría de los casos, la forma de f y la estructura discreta de \mathcal{S} impiden la aplicación de mecanismos de solución analítica comunes. La

conocida técnica básica de programación lineal, el algoritmo simplex, tendría el obstáculo de la falta de linealidad de f y la estructura no continua de S .

Al encontrarse en un caso como este, lo que se podría realizar es evaluar f en cada punto de S y buscar el óptimo. Este mecanismo de solución se le llama exploración exhaustiva, conduce a soluciones buenas en el caso del que el tamaño de S sea pequeño, teniendo en cuenta el beneficio de tener la tecnología de computadoras.

Pero para problemas de mayor tamaño, o cierto tipo de problemas donde la cardinalidad de S depende exponencialmente de uno más parámetros determinados (problemas NP-completos), la exploración exhaustiva no tiene aplicabilidad práctica debido al gran esfuerzo computacional que haría, lo que tomaría un tiempo de ejecución demasiado largo tanto así como siglos o incluso milenios. Aquí es donde las heurísticas de búsqueda local toman fuerza.

3.1.1 Métodos de Búsqueda local

Como se dijo anteriormente los métodos de búsqueda local son heurísticas para resolver problemas de optimización combinatoria en los cuales los espacios de búsqueda son demasiados grandes para utilizar la exploración exhaustiva. Debido a que son heurísticos, estos

métodos no siempre conducirán a una solución óptima, pero producen soluciones buenas en un tiempo razonable.

Todo método de búsqueda local parte de una solución factible inicial y obtiene, con esta soluciones que al evaluarlas en la función objetivo nos den un mejor valor. Cada solución tiene un conjunto de soluciones asociadas, que denominaremos entorno o vecindad de x . Cada solución del entorno, puede alcanzarse directamente a partir de x mediante un movimiento.

La definición de entorno / movimiento, depende en gran medida de la estructura del problema a resolver así como de la función objetivo. También se pueden definir diferentes criterios para seleccionar una nueva solución del entorno. Uno de los criterios más simples consiste en tomar la solución con mejor evaluación de la función objetivo, siempre que la nueva solución sea mejor que la actual; este criterio es conocido como Glotón. El algoritmo se detiene cuando la solución no puede ser mejorada en su vecindad, por lo que es un óptimo local del problema con respecto al entorno definido. Es bien probable que la solución encontrada no sea el óptimo global debido al método utilizado.

Esta limitación de los algoritmo glotonos es el punto de inicio de muchas de las técnicas meta heurísticas basadas en la búsqueda

local: evitar quedar atrapadas en un óptimo local lejano del global. Por lo que se permite que se realicen movimientos que empeoren la función objetivo. Esto plantea dos problemas. El primero, al permitir movimientos de mejora y de no mejora, el procedimiento se puede ciclar, revisando y volviendo a soluciones ya vistas. Segundo, hay que establecer un criterio de parada ya que podría iterar indefinidamente debido a las características del procedimiento.

Dado un $X_0 \in \mathcal{S}$ fijo, el conjunto de todas las soluciones factibles que pueden obtenerse a partir del mismo se conoce como la vecindad X_0 y se denotará de aquí en adelante como $V(X_0)$.

Algoritmo de búsqueda local

Generar una solución inicial $X_0 \in \mathcal{S}$

Hacer

Generar $V(X_0)$

Elegir $X_0 \in V(X_0)$

$X_0 := X$

Mientras X_0 no satisfaga un criterio de parada.



La diferencia entre cada uno de los métodos radica principalmente en la forma de seleccionar el elemento x de la vecindad.

3.1.2 Recocido Simulado

Este método es uno de los más conocidos de la familia de los de Monte Carlo en los cuales la aceptación de tales movimientos e hace dependiente de alguna distribución de probabilidad.

Este procedimiento se basa en una analogía con el comportamiento de un sistema físico al someterlo a un baño de agua caliente. S. Kirkpatrick y otros se inspiraron en una técnica experimental empleada en la metalurgia para obtener sólidos en estados bien ordenados, la misma que se conoce como recocido: en esta técnica el material es llevado al estado líquido y luego se enfría muy lentamente con posibles aumentos temporales de temperatura hasta solidificarlo. De esta manera se obtienen estructuras muy uniformes, con características de niveles energéticos globales bajos. La simulación produjo buenos resultados y este fue el punto de partida para el planteamiento de un método general para la solución de problemas de optimización combinatoria. El recocido simulado ha sido probado con

éxito en numerosos problemas de optimización, mostrando gran habilidad para evitar quedar atrapado en óptimos locales.

3.2 Técnica de Búsqueda de Hormigas

Para explicar de una mejor manera esta técnica nos valdremos de la siguiente analogía de las hormigas guerreras:

Era una vez en el reino de los insectos, que por desgracia se encontraban en guerra dos grupos de hormigas, las hormigas negras contra las hormigas rojas.

A poco tiempo de iniciar la guerra en el túnel principal del hormiguero de las hormigas rojas se sucita la siguiente situación: se encuentran de frente un par de hormigas, una sale hacia la guerra, la otra viene de ella; por la situación bélica en que viven en ese momento las hormigas rojas no han tenido tiempo de hacer el túnel de tal forma que puedan transitar las hormigas de manera adecuada y, en estos momentos en dicho túnel sólo cabe a lo ancho una hormiga a la vez, por lo que para que puedan pasar una hormiga es necesario que salte por encima de ella; además cuando se alcanzan a ver las dos hormigas sólo hay entre ellas un espacio para que quepa una de ellas, además para que se salten tienen que ponerse de acuerdo por lo que tienen que estar de frente a frente la hormiga que va a saltar como la hormiga que va a

esperar que la salten, pues de lo contrario no sabrían quién salta a quién. Como la organización de las hormigas es muy buena siempre los grupos que salen y llegan a combatir son del mismo número, por lo que con un par de hormigas no hay problema son pocos los movimientos que se tienen que hacer, pero cuando se encuentran frente a frente dos pares de hormigas frente a frente el número de movimientos crece, cuando son tres pares crece aún más, y así sucesivamente.

La reina observa esto e imposibilitada en estos momentos para mandar ensanchar el túnel por causa de la guerra, y viendo que es importante que cuando tienen que entrar las hormigas que vienen de la guerra en el mismo túnel con las hormigas que van a la guerra, ordenó a sus sabios que le dijeran la forma más rápida de que los grupos que se encuentren de frente sigan su camino unos hacia la guerra y otros a descansar de la guerra.

Entonces los sabios se pusieron a estudiar el problema y vieron que el menor número de hormigas que se podían encontrar en la situación planteada, era un par, o sea dos hormigas y para ese caso eran 3 movimientos (que la hormiga que sale avance hasta donde esta la hormiga que entra, la hormiga que sale le dice a la hormiga que entra "sáltame!" y después de esto ella puede avanzar hacia donde estaba

la hormiga que entra y puede salir a la guerra), partiendo de esa base pensaron que la hormiga que va hasta delante del grupo de hormigas que sale debía determinar el número de movimientos que debían hacer, para eso ella contaba con todo el apoyo de su grupo (digamos que era un general y no quería trabajar mucho), entonces como el único valor que se conoce es el de un par, al número de elementos del grupo le resto una unidad y a la hormiga que estaba atrás de ella le dijo que le preguntara ese número menos uno a la hormiga que estaba atrás de ella, y así sucesivamente hasta llegar a la última hormiga y que cuando ella les dijera el resultado le sumaran el producto de dos (pues son pares) por el número que se iban transmitiendo de una a otra hormiga y le sumara uno pues existe un hueco para realizar movimientos, de esta manera determinaría el número total de movimientos cuando se encontrarán de frente los dos grupos de hormigas en el túnel principal del hormiguero.

Después de saber el número de movimientos que tendrían que realizar los grupos que se encuentran en el túnel (pensaron los sabios), era determinar la forma en que se iban a realizar dichos movimientos.

Para eso vieron lo siguiente :

1. Se tienen que determinar los siguientes movimientos mover una hormiga determinada, en un sentido (izquierda o derecha), de un grupo específico (el que entra o el que sale).
2. Enumerar las hormigas de cada uno de los grupos de 1, hasta el último y del último hasta 1.

También tras múltiples ejercicios y observaciones vieron que el comportamiento era el siguiente :

1. Desde la primera hormiga del grupo hasta la última hormiga del grupo (a la hormiga que actualmente nos refiramos le llamaremos Ana) se tienen que hacer la siguiente pregunta :
2. Ana se tiene que preguntar: ¿soy un elemento par?, sí es así el sentido para el movimiento será a la derecha, y un individuo de mi grupo será el que se moverá, y el individuo que saltará será del grupo contrario al mío; en caso contrario el sentido para el movimiento será a la izquierda, un individuo del grupo contrario será el que se moverá y un individuo de mi grupo será el que saltará.
3. A continuación se llevará a cabo el movimiento en el sentido que se determinó anteriormente, por un individuo del grupo que también se determinó que saltará, el elemento será igual al número total de elementos del grupo - el número que le corresponda a Ana.

4. Después desde la primera hormiga hasta Ana (a esta hormiga la llamaremos Claudia) las hormigas deberán hacer lo siguiente : Saltar la hormiga que le corresponda el número total de hormigas en el grupo menos el número de hormiga que sea Claudia, esta hormiga deberá ser del grupo de salto que se determinó dos pasos antes, saltará a la hormiga cuyo número sea igual al número total de hormigas en el grupo menos el número de hormiga que sea Ana más el número de hormiga que sea Claudia, esta hormiga será del grupo contrario al que saltará.
5. Después de que se Ana sea igual al número total de hormigas en el grupo, se deberá hacer un movimiento en el sentido que se determinó la última vez en el paso 2; el individuo que se moverá será igual al número de hormigas en el grupo, y el grupo al que pertenece dicha hormiga será de igual forma el que se determinó la última vez en el paso 2.
6. Ahora desde la penúltima hormiga hasta la primera (a quién llamaremos una vez Ana) se deberán realizar la pregunta de la siguiente forma :
7. Ana se tiene que preguntar: ¿soy un elemento par?, sí es así el sentido para el movimiento será a la izquierda, y un individuo del grupo contrario será el que se moverá, y el individuo que saltará será de mi grupo; en caso contrario el sentido para el movimiento

será a la derecha, un individuo de mi grupo será el que se moverá y un individuo del grupo contrario será el que saltará.

8. Después desde la primera hormiga hasta Ana (a esta hormiga la llamaremos Claudia) las hormigas deberán hacer lo siguiente : Saltar la hormiga que le corresponda el número de hormiga que sea Ana menos el número de hormiga que sea Claudia, esta hormiga deberá ser del grupo de salto que se determinó dos pasos antes, saltará a la hormiga cuyo número sea igual al número de hormiga que sea Claudia más uno , esta hormiga será del grupo contrario al que saltará.
9. A continuación se llevará a cabo el movimiento en el sentido que se determinó en el paso 7, por un individuo del grupo que también se determinó que saltará en dicho paso , el elemento será igual al número de hormiga que sea Ana.

3.3 Técnica de Algoritmos Genéticos

Los Algoritmos Genéticos (AG) fueron introducidos por John Holland en 1970 inspirándose en el proceso observado en la evolución natural de los seres vivos.

Los biólogos han estudiado en profundidad los mecanismos de la evolución, y aunque quedan parcelas por entender, muchos aspectos están bastantes explicados. De forma muy general podemos decir que

en la evolución de los seres vivos el problema al que cada individuo se enfrenta cada día es la supervivencia. Para ello cuenta con las habilidades innatas provistas en su material genético. A nivel de los genes, el problema es el de buscar aquellas adaptaciones beneficiosas en un medio hostil y cambiante. Debido en parte a la selección natural, cada especie gana cierta cantidad de conocimiento, el cual es incorporado a la información de sus cromosomas.

En otras palabras, la evolución tiene lugar en los cromosomas, en donde está codificada la información del ser vivo. La información almacenada en el cromosoma varía de unas generaciones a otras. En el proceso de formación de un nuevo individuo, se combina la información cromosómica de los progenitores aunque la forma exacta en que se realiza es aún desconocida.

Lo más sorprendente y útil de los AG es que no es necesario disponer de un conocimiento profundo del problema a resolver, sino que se parte de estructuras simples que interactúan, dejando que sea la evolución quien haga el trabajo. Un AG es un método de programación opuesto al habitual: en vez de descomponer un problema en sub-problemas, creamos los sub-problemas más sencillos que podamos imaginar y dejamos que se combinen entre sí.

Otra gran ventaja de los AG es que son adaptativos: son capaces de solucionar problemas que cambian en el tiempo.

La gran sencillez de los AG los hace muy atractivos para los programadores, pero existen casos en los que puede ser difícil, imposible o poco práctico aplicarlos. Se podría hablar del problema de la variedad, el problema de la reproducción y el problema de la selección, para los que se van a proponer algunas soluciones.

Pasos de un Algoritmo Genético :

- 1.-Se genera un conjunto de 1-N soluciones al problema. Normalmente, estas entidades se generan al azar.
- 2.-Se evalúan las soluciones existentes, de manera que se eliminan unas y se mantienen otras (o se limita el tiempo de ejecución).
- 3.-Se permite la reproducción o combinación de genes (normalmente por parejas) de las entidades existentes.
- 4.-Se efectúan mutaciones (cambios al azar) de los nuevos patrones, según una tasa determinada.
- 5.-Se continúa en el paso 2

Los algoritmos genéticos tienen la ventaja de ofrecer un mecanismo adaptativo de resolución de problemas, de forma que aunque el problema cambie, éste se pueda seguir resolviendo. "Resolver un problema cambiante", llevado al límite, es igual a "Resolver cualquier problema". Según este planteamiento podríamos pensar en diseñar un esquema común que facilite a los agentes aprender en cualquier entorno de problema.

Capítulo 4

4.1 Reseña del Método Tabú

Los orígenes de la búsqueda tabú se extienden hasta la década de los 70. En 1986, *Fred Glover* lo desarrolló en su forma actual. Las ideas básicas fueron esbozadas también por *Hansen*. Muchas pruebas experimentales han mostrado que la búsqueda tabú puede ser considerada como una técnica de optimización imprescindible para el estudio de gran variedad de problemas. Gracias a su flexibilidad puede derrotar a muchas de las técnicas clásicas. Hasta ahora no hay una explicación formal de su buen comportamiento.

La búsqueda tabú intenta solventar el problema de que el algoritmo quede atrapado en un óptimo local. Se basa en la utilización de estructuras de memoria flexibles, junto con restricciones estratégicas y niveles de aspiración, como medios para explorar el espacio de soluciones posibles. Se llama "solución" a una configuración de las variables independientes que den un resultado. Para una correcta utilización se requiere un buen modelado del problema en el que se desea aplicar.

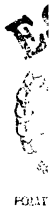
El Método Tabú (MT) pertenece a una familia de heurísticas de optimización local. Su memoria flexible permite que las decisiones tomadas en cada iteración se vean afectadas por aquellas que se tomaron en iteraciones

anteriores. Ha sido calificado a menudo como una meta - heurística o heurística de alto nivel, debido a que es independiente de un problema dado.

Este método toma de la Inteligencia Artificial (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 principio del MT podría resumirse como:

“Es mejor una mala decisión basada en información que una buena decisión al azar, ya que, en un sistema que emplea memoria, una mala elección basada en una estrategia proporcionará claves útiles para continuar la búsqueda. Una buena elección fruto del azar no proporcionará ninguna información para posteriores acciones”.

La idea básica consiste en permitir el paso de una solución a otra, que denominaremos “movimiento”, aún cuando esta última empeore el resultado. Para evitar el posible ciclado se introduce la **lista tabú**. En ella se guarda durante cierto tiempo, un atributo que permita identificar la solución o el movimiento realizado. Así, todo movimiento que tenga un atributo en la lista tabú se considera prohibido, y no se permite su realización.



Al estar constituida la lista tabú por atributos y no por el movimiento o solución, se puede prohibir pasar a soluciones óptimas. Para evitar esto, se utilizan los niveles de aspiración o promoción de un movimiento. Si un movimiento es tabú, y por tanto prohibido, y supera los criterios de aspiración, entonces se libera de su condición tabú y se permite su ejecución.

4.1.1 Restricciones Tabú

Controla lo que entra en la lista tabú. Con ella se crea la lista tabú, cuya única función es impedir que se repita la búsqueda en una solución de prueba que ya se ha explorado. Un movimiento puede tener atributos múltiples. Es importante la elección correcta de éstos. Si el atributo elegido es muy general puede que permita pocos movimientos; si es muy específico, puede impedir la salida de una región con un mínimo local. En cuanto al tamaño de la lista tabú, si es demasiado pequeña, la probabilidad de la entrada en una búsqueda cíclica es grande. En cambio, si la longitud es demasiado grande, entonces la búsqueda puede alejarse de posibles buenas soluciones antes de que esas regiones sean exploradas. La lista tabú funciona con el procedimiento FIFO, el primero en entrar en la lista, es el primero en abandonarla. La principales estrategias son las siguientes:



- ✓ **Recency-based:** Consiste en una lista formada por los atributos de las últimas soluciones visitadas.
 - **Reglas estáticas:** La longitud de la lista se mantiene constante.
 - **Reglas dinámicas:** La longitud varía.
- ✓ **Frequency-based:** Consiste en medir la frecuencia con que ocurren ciertos atributos, como por ejemplo, el número de veces que una variable ha tomado cierto valor.

4.1.2 Estrategias de aprendizaje a medio y largo plazo

Se implementan mediante funciones de memoria de medio y largo plazo. La función a medio plazo proporciona un elemento de intensificación. Opera guardando las mejores características de un número de movimientos generados durante la ejecución del algoritmo. Puede ser considerado una estrategia de aprendizaje que busca nuevas soluciones que presentan características parecidas a aquellas guardadas. Esto se consigue restringiendo aquellos movimientos que no poseen características favorables.

4.1.3 Estrategia a corto plazo (short term)

Dirige la interacción entre las estrategias anteriores. Aparte de estas estrategias, puede haber también una estrategia de aprendizaje que consiste en el uso de funciones de memoria a medio y largo plazo.



Esta estrategia recoge información durante la ejecución de la búsqueda tabú que será usada para dirigirla en siguientes ejecuciones.

4.1.4 Estrategia de intensificación

Después de haber visitado varias soluciones, en ocasiones puede observarse que las mejores soluciones tienen alguna propiedad en común. Conviene entonces guardar en memoria estas soluciones para someterlas a un análisis más intenso, por ejemplo, cuando la búsqueda no produzca mejora.

4.1.5 Estrategia de diversificación

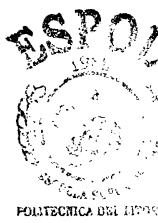
Es el procedimiento contrario a la intensificación. Es necesaria para permitir una búsqueda más efectiva en todo el espacio de soluciones. Con ellas se pretende permitir el paso a soluciones que estén alejadas de la actual, es decir, que permita una adecuada exploración. Un método sencillo para hacerlo consiste en modificar la función objetivo f , de modo que para soluciones alejadas de la actual decrezca y crezca para las más cercanas.

MT comienza de la misma forma que cualquier procedimiento de búsqueda local, procediendo iterativamente de una solución x a otra y en la vecindad de la primera: $N(x)$. Sin embargo, en lugar de

considerar todo la vecindad de una solución, MT define la vecindad reducida $N^*(x)$ como aquellas soluciones disponibles de la vecindad de x . Así se considera que a partir de x , sólo las soluciones de la vecindad reducida son alcanzables.

Existen muchas maneras de definir la vecindad reducida de una solución. La más sencilla consiste en etiquetar como tabú las soluciones previamente visitadas en un pasado cercano. Esta forma es la conocida a corto plazo (short term memory) y está basada en guardar en una **lista tabú** T las soluciones visitadas recientemente (Recency-based). Así en una iteración determinada, el entorno reducido de una solución se obtendría como el entorno usual eliminando las soluciones etiquetadas como tabú.

El objetivo principal de etiquetar las soluciones visitadas como tabú es el de evitar que la búsqueda se cicle. Por ello se considera que tras un cierto número de iteraciones la búsqueda está en una región distinta y puede liberarse del estatus tabú (pertenencia a T) a las soluciones antiguas. De esta forma se reduce el esfuerzo computacional de calcular la vecindad reducida en cada iteración. En los orígenes de MT se sugerían listas de tamaño pequeño, actualmente se considera que las listas pueden ajustarse dinámicamente según la estrategia que se esté utilizando.



Se define un **criterio de aspiración** como aquellas condiciones que, de satisfacerse, permitirían alcanzar una solución aunque tenga estatus tabú. Una implementación sencilla consiste en permitir alcanzar una solución siempre que mejore a la mejor almacenada, aunque esté etiquetada tabú. De esta forma se introduce cierta flexibilidad en la búsqueda y se mantiene su carácter agresivo.

Es importante considerar que los métodos basados en búsqueda local requieren de la exploración de un gran número de soluciones en poco tiempo, por ello es crítico el reducir al mínimo el esfuerzo computacional de las operaciones que se realizan a menudo. En ese sentido, la memoria a corto plazo de MT está basada en atributos en lugar de ser explícita; esto es, en lugar de almacenar las soluciones completas (como ocurre en los procedimientos de búsqueda exhaustiva) se almacenan únicamente algunas características de éstas.

Los principales criterios de aspiración son:

4.1.6 Aspiración por defecto

Si todos los movimientos posibles están clasificados como tabú y no aparece como admisible por otro criterio de aspiración, entonces se acepta el movimiento a la solución “menos tabú”. Esta puede ser una



solución que pierde su condición tabú por ser la que lleve más tiempo en la lista.

4.1.7 Aspiración por objetivo global

Una solución de prueba es eliminada de la lista tabú cuando muestra el mejor resultado de todos los encontrados hasta entonces.

4.1.8 Aspiración por objetivo regional

Una solución de prueba es eliminada de la lista tabú cuando muestra el mejor resultado encontrado en la región en la que se encuentra.

4.1.9 Aspiración por dirección de búsqueda

Un atributo puede ser añadido o eliminado de una solución (sin importar su condición tabú), si la dirección de búsqueda no ha cambiado (mejore o no los resultados).

4.1.10 Aspiración por influencia

El estado tabú de un movimiento poco influyente puede ser revocado en el caso que se haya efectuado un movimiento de alta influencia desde el momento en que se estableció la condición tabú de dicho movimiento poco influyente.



La **memoria mediante atributos** produce un efecto más sutil y beneficioso en la búsqueda, ya que un atributo o grupo de atributos identifica a un conjunto de soluciones, del mismo modo que los hiperplanos utilizados en los algoritmos Genéticos. Así, un atributo que fue etiquetado como tabú por pertenecer a una solución visitada hace n iteraciones, puede impedir en la iteración actual, el alcanzar una solución por contenerlo, aunque ésta sea muy diferente de la que provocó el que el atributo fuese etiquetado. Esto provoca, a largo plazo, el que se identifiquen y mantengan aquellos atributos que inducen una cierta estructura beneficiosa en las soluciones visitadas.

Con los elementos descritos puede diseñarse un algoritmo básico de MT para un problema de Optimización dado. Sin embargo, MT ofrece muchos más elementos para construir algoritmos realmente potentes y eficaces. A menudo, dichos elementos han sido ignorados en muchas aplicaciones y actualmente la introducción de estos en la comunidad científica constituye un reto para los investigadores del área.

Un algoritmo MT está basado en la interacción entre la memoria a corto y la memoria a largo plazo. Ambos tipos de memoria llevan asociadas sus propias estrategias y atributos, y actúan en ámbitos diferentes. Como ya hemos mencionado la memoria a corto plazo suele almacenar atributos de soluciones recientemente visitadas, y su

objetivo es explorar a fondo una región dada del espacio de soluciones. En ocasiones se utilizan estrategias de listas de candidatos para restringir el número de soluciones examinadas en una iteración dada o para mantener un carácter agresivo en la búsqueda.

La memoria a largo plazo almacena las frecuencias u ocurrencias de atributos en las soluciones visitadas tratando de identificar o diferenciar regiones. La memoria a largo plazo tiene dos estrategias asociadas: Intensificar y Diversificar la búsqueda. La intensificación 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. La Diversificación consiste en visitar nuevas áreas no exploradas del espacio de soluciones. Para ello se modifican las reglas de elección para incorporar a las soluciones atributos que no han sido usados frecuentemente.

Existen otros elementos más sofisticados dentro de MT que, aunque poco probados, han dado muy buenos resultados en algunos problemas. Entre ellos se pueden destacar:



4.1.11 Movimientos de Influencia

Son aquellos movimientos que producen un cambio importante en la estructura de las soluciones. Usualmente, en un procedimiento de búsqueda local, la búsqueda es dirigida mediante la evaluación de la función objetivo. Sin embargo, puede ser muy útil el encontrar o diseñar otros evaluadores que guíen a ésta en determinadas ocasiones. Los movimientos de influencia proporcionan una evaluación alternativa de la bondad de los movimientos al margen de la función objetivo. Su utilidad principal es la determinación de estructuras subyacentes en las soluciones. Esto permite que sean la base para procesos de Intensificación y Diversificación a largo plazo.

4.1.12 Oscilación Estratégica

La Oscilación Estratégica opera orientando los movimientos en relación a una cierta frontera en donde el método se detendría normalmente. Sin embargo, en vez de detenerse, las reglas para la elección de los movimientos se modifican para permitir que la región al otro lado de la frontera sea alcanzada. Posteriormente se fuerza al procedimiento a regresar a la zona inicial. El proceso de aproximarse, traspasar y volver sobre una determinada frontera crea un patrón de oscilación que da nombre a esta técnica. Una implementación sencilla consiste en considerar la barrera de la factibilidad / infactibilidad de un



problema dado. Implementaciones más complejas pueden crearse identificando determinadas estructuras de soluciones que no son visitadas por el algoritmo y considerando procesos de construcción / destrucción asociados a éstas. La oscilación estratégica proporciona un medio adicional para lograr una interacción muy efectiva entre intensificación y diversificación.

4.1.13 Elecciones Probabilísticas

Normalmente MT se basa en reglas sistemáticas en lugar de decisiones al azar. Sin embargo, en ocasiones se recomienda aleatorizar algunos procesos para facilitar la elección de buenos candidatos o cuando no está clara la estrategia a seguir (quizá por tener criterios de selección enfrentados). La selección aleatoria puede ser uniforme o seguir una distribución de probabilidad construida empíricamente a partir de la evaluación asociada a cada movimiento.

4.1.14 Umbrales Tabú

El procedimiento conocido como Tabú Thresholding (TT) se propone aunar ideas que provienen de la Oscilación Estratégica y de las Estrategias de Listas de Candidatos en un marco sencillo que facilite su implementación. El uso de la memoria es implícito en el sentido que no hay una lista tabú en donde anotar el status de los movimientos.



pero la estrategia de elección de los mismos previene el ciclado. TT utiliza elecciones probabilísticas y umbrales en las listas de candidatos para implementar los principios de MT.

4.1.15 Reencadenamiento de Trayectorias

Este método trata de volver a unir dos buenas soluciones mediante un nuevo camino. Así, si en el proceso de búsqueda hemos encontrado dos soluciones x e y con un buen valor de la función objetivo, podemos considerar el tomar x como solución inicial e y como solución final e iniciar un nuevo camino desde x hasta y . Para seleccionar los movimientos no consideraremos la función objetivo o el criterio que hayamos estado utilizando sino que iremos incorporando a x los atributos de y hasta llegar a ésta. Por eso esperamos que alguna de las soluciones intermedias que se visitan en este proceso de Entorno Constructivo sea muy buena. En algunas implementaciones se ha considerado el explorar el entorno de las soluciones intermedias para dar mas posibilidad al descubrimiento de buenas soluciones.

Es importante destacar el hecho de que muchas de las aplicaciones basadas en MT no utilizan los últimos elementos descritos, por lo que son susceptibles de ser mejoradas. Al mismo tiempo, los éxitos de las numerosas implementaciones del procedimiento han promovido la investigación hacia formas de explotar con mayor intensidad sus ideas

subyacentes. En este territorio podemos destacar los últimos trabajos de Modelos de entrenamiento y aprendizaje tabú, Maquinas tabú y Diseño tabú.

4.1.16 Algoritmo Método Tabú

Paso 0: Inicialización.

$X :=$ solución inicial factible

$t_{max} :=$ máximo número de iteraciones

Mejor solución := X

número de soluciones = $t := 0$

lista tabú := vacía

Paso 1: Parada

Si cualquier movimiento posible de la solución actual es tabú o si

$t = t_{max}$

entonces parar. Entregar *Mejor solución*.

Paso 2: Mover

Elegir el mejor movimiento no-tabú factible $\Delta x(t+1)$

Paso 3: Iteración.

Modificar $X(t+1) := X(t) + \Delta x(t+1)$



Paso 4: Reemplazar el mejor.

Si el valor de la función objetivo de $X(t+1)$ es superior a *Mejor solución* entonces a *Mejor solución* := $X(t+1)$

Paso 5: Actualizar Lista Tabu.

Eliminar desde la lista tabu cualquier movimiento que ha permanecido un suficiente número de iteraciones en la lista.

Agregar un conjunto de movimientos que involucran un retorno inmediato desde $X(t+1)$ a $X(t)$

Paso 6: Incrementar

$t := t+1$, Volver a Paso 1.



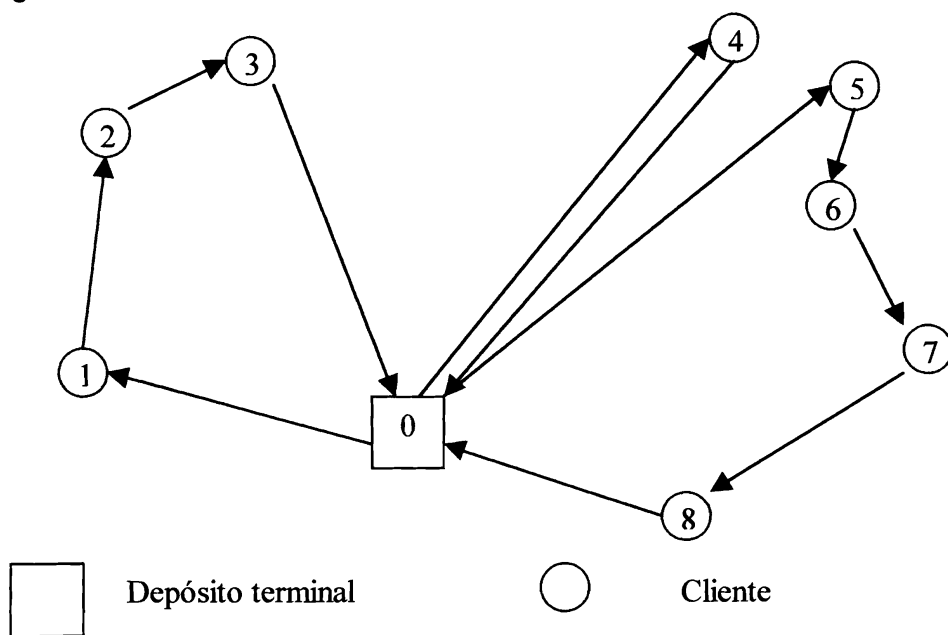
Capítulo 5

DISEÑO E IMPLEMENTACIÓN DEL MODELO

5.1 Diseño del Modelo Meta-heurístico para el problema de ruteo de vehículos (VRP por sus siglas en inglés)

En el problema de ruteo de vehículos bajo restricciones de capacidad y distancia, el diseño reduce el costo por ruta que cada uno de los vehículos debe recorrer para satisfacer a cierto número de clientes. En este problema todas las rutas inician y terminan en un depósito central o terminal, el cual brinda algún tipo de prestación a ciertos consumidores. Cada cliente es cubierto exactamente por una ruta de vehículo. La demanda de cada vehículo no debe exceder de la capacidad del transporte.

Figura 5.1: Rutas de Vehículos



Para el problema se utilizará la siguiente notación:

n = Número de clientes

N = Conjunto de Clientes, $N = \{1, \dots, n\}$

q_i = La demanda del cliente $i \in N$ ($i=0$ denota la terminal, $q_0=0$)

c_{ij} = El tiempo de viaje (distancia) entre los clientes i y j , $c_{ij} = c_{ji} \forall i, j \in N$ ($c_{ii} = \infty, \forall i \in N$)

v = el numero de vehículos, la cual es la variable de decisión de nuestro problema

V = Conjunto de vehículos, $V = \{1, \dots, v\}$

Q = capacidad del vehículo

R_p = Conjunto de cliente servido por el vehículo p .

$C(R_p)$ = El costo (distancia) del viaje óptimo.

L = El límite superior está predeterminado sobre longitud máxima del recorrido.

S = La solución factible está definida como $S = \{R_1, \dots, R_v\}$

$C(S)$ = La suma total de cada distancia recorrida, $C(R_p)$ para todo $p \in V$



Nuestra meta es encontrar una solución óptima que minimice la distancia total del viaje y satisfaga:

$$\text{Min: } C(s) = \sum_{p \in V} C(R_p)$$

s.a.r

$$\bigcup_{p=1}^v R_p = N \quad R_p \cap R_q = \phi, \forall p \neq q \in V$$

Esta restricción nos indica que todos los clientes debes ser al menos por una ruta de transporte.

$$C(R_p) = \sum_{i \in R_p \cup \{0\}} (c_{i\pi(i)}) \leq L \quad \forall p \in V$$

El costo de cada recorrido estará limitada por un valor L.

$$\sum_{i \in R_p} d_i \leq Q \quad \forall p \in V$$

La demanda de los clientes no podrá exceder de la capacidad de vehículo.



donde $\pi = \{\pi_1, \dots, \pi_p, \dots, \pi_v\}$ es un óptimo del problema del agente viajero (TSP por sus siglas en inglés) que minimiza la distancia del recorrido para cada $p \in V$.

5.1.1 Estrategia prohibida

Esta estrategia de búsqueda clasifica ciertos movimientos como prohibidos (o tabú) basado en condiciones tabú las cuales están identificadas por atributos de un movimiento. Para evitar que se cicle, el proceso revisa que soluciones que han sido visitadas anteriormente, no sean nuevamente visitadas, pero para poder realizar esta operación, se requiere un gran uso de memoria y esfuerzo computacional.

La estructura de datos de la lista tabú, TABL, toma la forma de una matriz de $(n+1) \times v$ (n filas para los clientes, una para el cliente null que está involucrado en el proceso $(0,1)$ o $(1,0)$ y v columnas una por cada ruta R_p . Un movimiento consiste en dos pares (i, R_p) y (j, R_q) la cual identifica que el cliente i del conjunto de R_p de los clientes de la ruta p ha sido cambiada con un cliente j de un conjunto R_q , y viceversa. Los atributos (i, R_p) y (j, R_p) luego del movimiento pasan a ser parte de las restricciones tabú, es decir que son movimientos prohibidos de realizar. Un movimiento es juzgado de tipo tabú, si i

retorna a R_p y j retorna a R_q . Esta es una aproximación para movimientos prohibidos y las ventajas es que las soluciones pueden ser representadas y revisadas rápidamente. $TABL(i,p)$ guarda el número de iteraciones en las cual un cliente i es removido de la ruta del conjunto R_p . Inicialmente, la matriz $TABL$ es inicializada con valores negativos altos para evitar falsa identificación de clientes como tabú durante las iteraciones iniciales.

5.1.2 Estrategia de Liberación

Cuando mencionamos la estrategia de liberación no referimos al manejo de los valores que salen de la lista tabú después de las TS iteraciones; donde TS es conocido como el tamaño de la lista tabú. El valor TS es determinado por una función que depende de las características del problema y de los movimientos estratégicos seleccionados.

El conjunto de movimientos prohibidos es gravado en la lista tabú por un periodo de TS iteraciones. Una simple y rápida validación del estado del tabú es de gran importancia, especialmente cuando el problema y el tamaño de la lista del tabú incrementa. En una iteración K un movimiento es clasificado como tabú sin ningún i debe retornar a R_p ni j debe retornar a R_q durante las siguientes TS iteraciones. Esto es,

$k\text{-TABL}(i, p) < |T_s|$ y

$k\text{-TABL}(j, q) < |T_s|$

En la lista denominada TABL, se guarda el número de iteración. El estado tabú de un movimiento potencial puede ser revisado usando las dos simples operaciones que vimos anteriormente. Con TABL el estatus tabú de los movimientos anteriores son actualizados automáticamente opuesta a la clásica lista circular tabú la cual necesita más controles de ingreso desde la estrategia de liberación.

5.1.3 Estrategia a corto plazo

La forma de estrategia a corto plazo es el centro del Tabú. Esta está diseñada para la búsqueda del mejor movimiento admitido en la vecindad basada en las restricciones tabú y criterios de aspiración. Un movimiento es considerado admisible si este no es un movimiento tabú o un movimiento tabú el cual a pasado un nivel de criterio de aspiración. Las restricciones tabú y los criterios de aspiración juegan un rol contrario obligando y guiando el proceso de búsqueda. En la lista tabú nosotros almacenamos algunos atributos de los movimientos que representan soluciones. Así, algunas soluciones no tabú pueden ser prevenidas por las restricciones tabú cumplidas para esta aproximación y criterios de aspiración son probadas para corregir tal

prevención. La siguiente función de espiración puede ser usada para permitir una búsqueda en una nueva dirección y garantiza que no existan ciclos. Dado un S_b sea una de las mejores soluciones encontradas durante la búsqueda. Dado un $S' \in N_1(S)$ sea una solución tabú y $\Delta = C(S') - C(S)$. La nueva solución S' es admitida si $C(S') < C(S_b)$.

Dos estrategias de selección pueden ser seleccionadas como movimientos admitidos de una lista de movimientos: la estrategia del mejor admitido (BA) y la estrategia del primer mejor admitido (FBA). La primera estrategia selecciona el movimiento mejor admitido de una vecindad, la cual mejora o no la función objetivo. El algoritmo tabú el cual usa el mejor admitido es denotado por TS+BA. La estrategia FBA combina la aproximación de glotón con la estrategia BA. Esta selecciona el primer movimiento admitido el cual provee una mejora en el valor objetivo sobre una solución actual, si todos los movimientos en la lista de los candidatos son probados sin ninguna mejora entonces FBA selecciona el mejor movimiento de no mejora gravado. El algoritmo tabú el cual utiliza la estrategia de selección FBA es denotado TS+FBA. La lista de candidatos para el algoritmo TS+FBA es toda la vecindad de $N_1(S)$ y su tamaño es dinámico y determinado automáticamente por la búsqueda de si mismo. Esta muestra dinámica es una vía deseada para buscar una vecindad grande. De cualquier

forma la lista de candidatos de movimientos para la estrategia BA es toda la vecindad $N_1(S)$ y si tamaño es fijo. Esta lista es bien demasiado costosa para la computadora para problemas de tamaño grande, debido a que $N_1(S)$ debe ser reevaluada para seleccionar el mejor movimiento después de cada iteración. Así nosotros proponemos una estructura de datos la cual solo nos permite un pequeño número de reevaluaciones en orden de identificar un nuevo mejor movimiento de una iteración a otra.

5.2 Solución de Problemas de prueba para tabú

A continuación está la mejor solución encontrada para 2 problemas usando el método Tabú.

N.- representará el número de clientes.

C(S).- Representa la solución.

Q.- Capacidad restante.

L.- Límite de la longitud del recorrido. **δ_i -** Límite del tiempo de servicio.

1) **N=30** **Q=100** **L=520** **$\delta_i=90$**

Solución Inicial C(S)= 2495.5

TABLA I

P	Q	R _p	Ruta
1	10	12	0-1-2-3-4-5-6-7-8-9-10-11-12-0
2	0	8	0-13-14-15-16-17-18-19-20-0
3	10	9	0-21-22-23-24-25-26-27-28-29-0
4	70	1	0-30-0

Al graficar la coordenadas de los clientes luego de obtener la solución inicial, las rutas quedan de la siguiente manera:

Figura 5.2

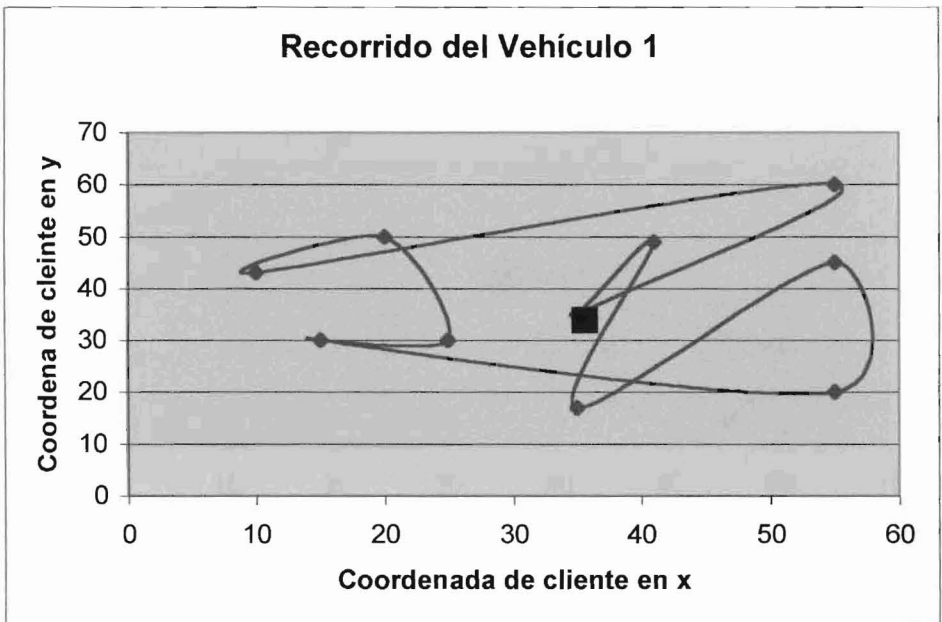


Figura 5.3

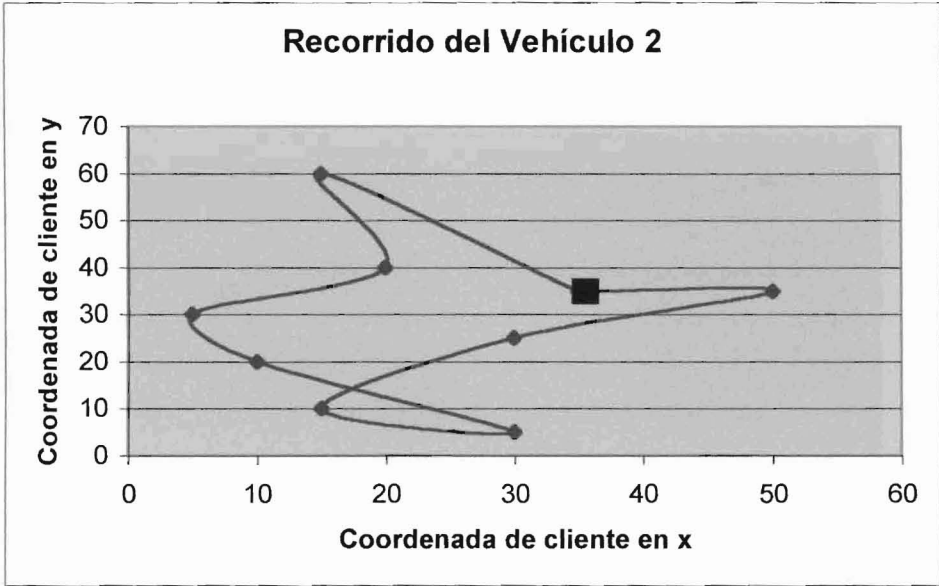


Figura 5.4

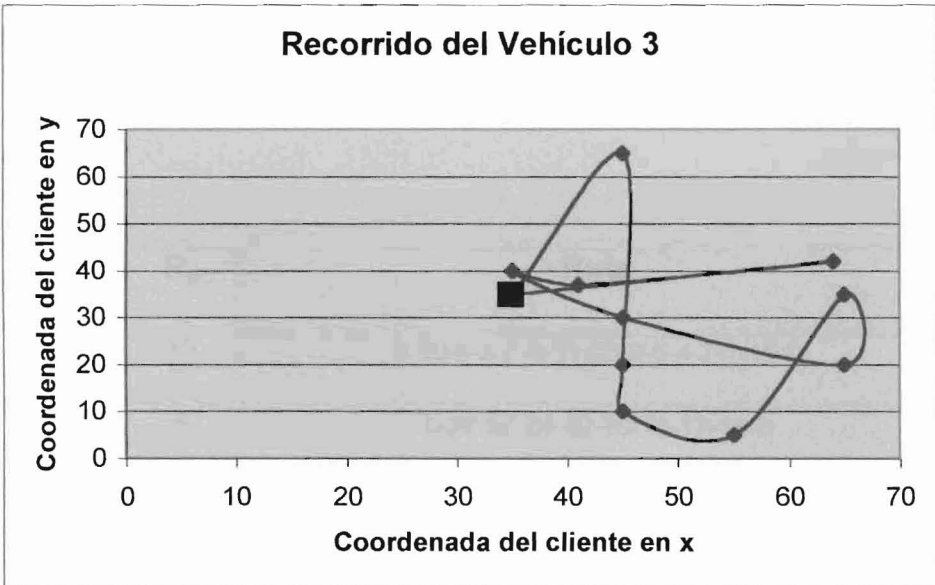
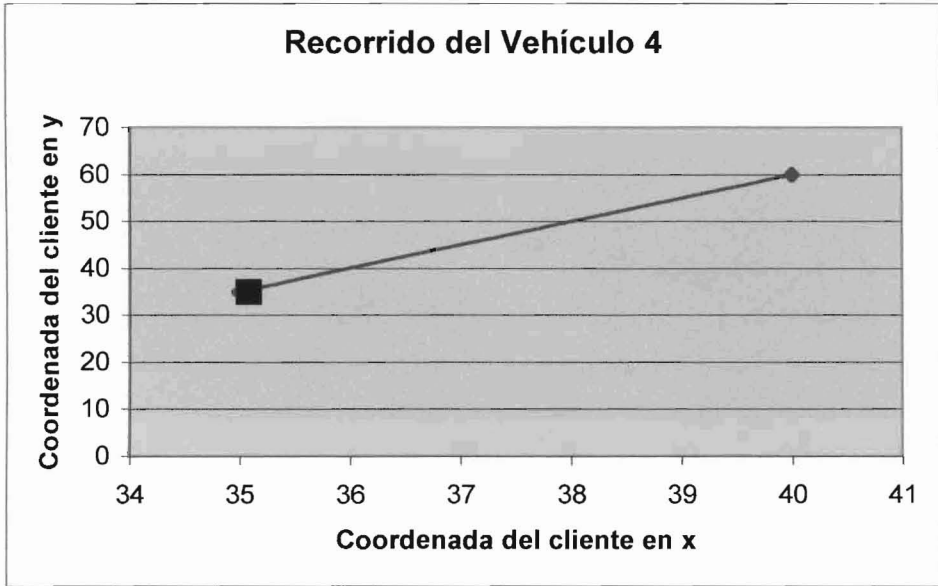


Figura 5.5



Solución Encontrada: 1308.11

TABLA II

P	Q	R _p	Ruta
1	0	13	0-30-5-3-7-8-11-26-9-6-4-2-1-16-0
2	0	8	0-27-25-24-22-20-13-17-18-0
3	10	9	0-29-15-21-23-14-12-10-28-19-0

Las nuevas rutas que se obtuvieron luego de utilizar el método tabú, resultan de la siguiente manera:

Figura 5.6

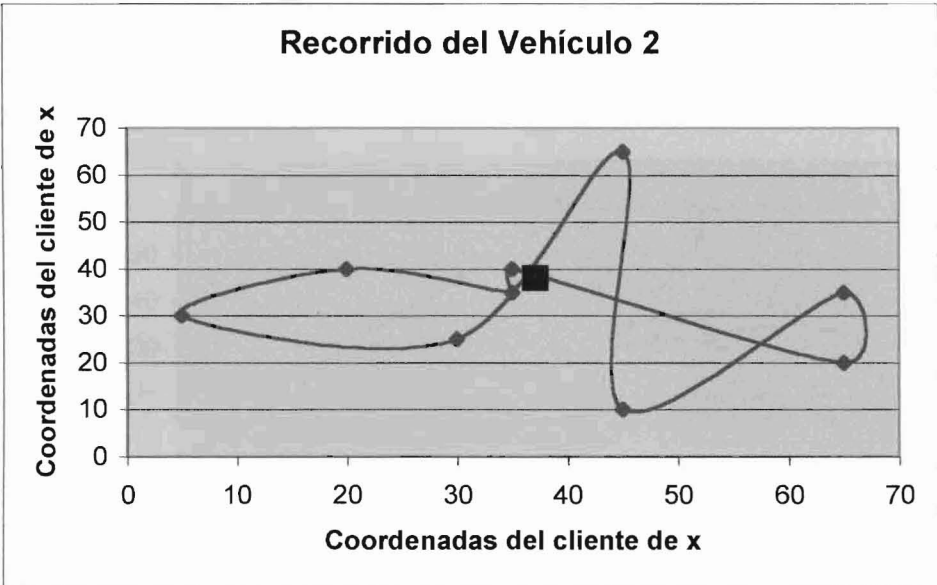


Figura 5.7

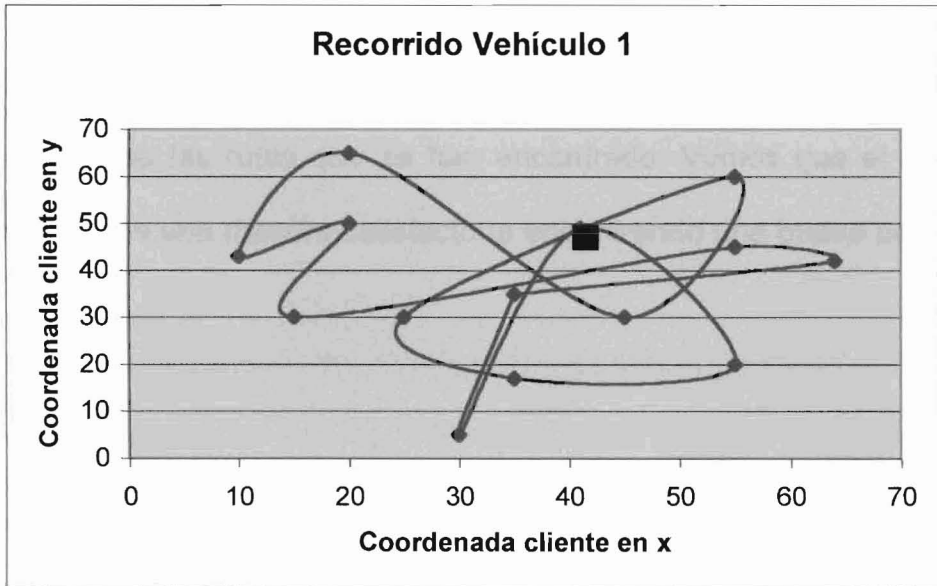
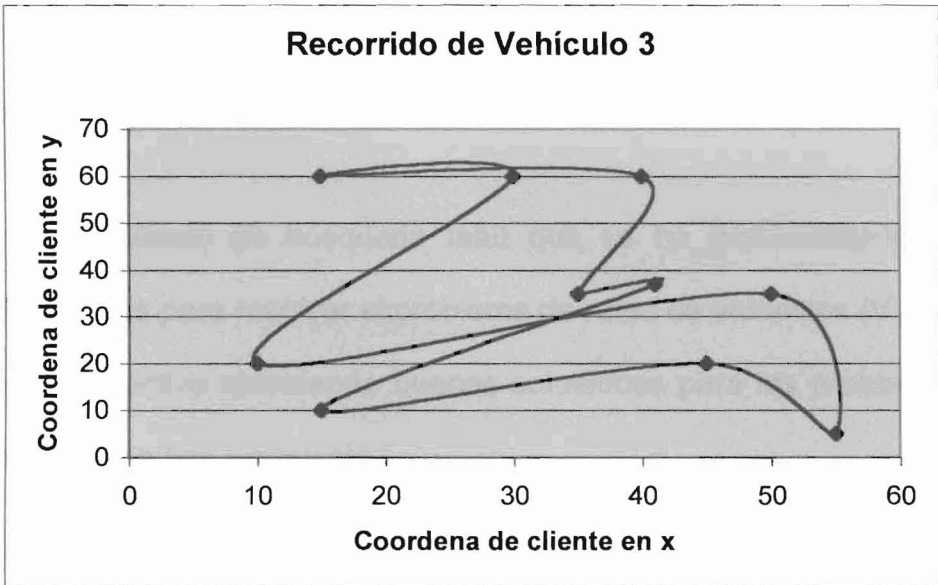


Figura 5.8



Como se ha podido observar luego de la ejecución del método tabú con la solución inicial del cuadro anterior, se redujo de 4 rutas a 3 rutas, es decir, se disminuyó un vehículo, pero hay que observar que la solución objetivo se ha reducido de 2495,5 a 1308,11 estos valores representan el costo que tiene el recorrer todas las rutas que se han encontrado. Vemos que el método ha funcionado de una manera satisfactoria encontrando una buena solución.

Capítulo 6

6.1 Conclusiones

6.1.1.- El método de búsqueda tabú que se ha implementado en la presente tesis para resolver el problema de ruteo de vehículos (V.R.P) ha resultado efectivo obteniendo buenas soluciones para los problemas de prueba que se han propuesto.

6.1.2.- El problema del ruteo de transportes presenta complejidad de orden exponencial. El tamaño de este está dado por el número de clientes y el número de vehículos que se utilizarán. Esta generará varias posibles soluciones. Con la implementación de la búsqueda Tabú encontramos una de las soluciones que mejora la solución objetivo.

6.1.3.- En la implementación realizada de la búsqueda Tabú ha utilizado la estrategia FBA el mejor accesible, con esto se reduce el esfuerzo computacional, debido a que abandona la búsqueda en cada vecindad luego de encontrar un solución que mejore la función objetivo.



6.1.4.- Existen algunos métodos que se emplean para obtener soluciones para el tipo de problemas de transportes. Una de las ventajas de la búsqueda Tabú es el carácter agresivo de la búsqueda. Con esto este método puede tomar las soluciones de otros algoritmos y partir de ahí su búsqueda para obtener una mejor solución.

6.2 Recomendaciones

6.2.1.- Sería bueno que en un futuro en el programa se realicen ciertos cambios en las estrategias de búsqueda, con las que se tendría una búsqueda más profunda y exhaustiva. Esto favorecería al usuario, debido a que se puede obtener una solución de mayor calidad al ejecutar el programa y en un tiempo prudencial.

6.2.2.- El Método de búsqueda Tabú puede ser una gran herramienta para la ayuda en la toma de decisiones, y teniendo el conocimiento de que es muy difícil encontrar una solución óptima a estos tipos de problemas, la buena solución que obtendremos es lo suficientemente confiable para la utilización de esta en los problemas cotidianos.



6.2.3.- El Método sería de gran utilidad en nuestra sociedad, debido a que en nuestro país muchas decisiones se toman sin tener algún respaldo de alguna técnica o estudio previo, para que los resultados posteriores sean un poco más previsibles y manejables.



BIBLIOGRAFÍA



1. Glover, F., "Tabu Search: A Tutorial",1990.
2. Glover, F., & Laguna "Tabu Search",1993.
3. Ibrahim Hassan Osman, "Meta-strategy Simulated and Annealing and Tabu Search Algorithms for the Vehicle Routing Problem", 1991
4. Manuel de la Herrán Gascón,"Algoritmos Genéticos Avanzados",1997
5. José Luis Orduña Centeno, "El Dilema de las hormigas guerreras",2000