“FORMULAR LAS METAHEURÍSTICAS BÚSQUEDA TABÚ Y RECOCIDO SIMULADO PARA LA SOLUCIÓN DEL CVRP (CAPACITATED VEHICLE ROUTING PROBLEM)”

DAVID FERNANDO GÓMEZ ATUESTA

CARLOS EDUARDO RANGEL CARVAJAL

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2011

“FORMULAR LAS META HEURÍSTICAS BÚSQUEDA TABÚ Y RECOCIDO SIMULADO PARA LA SOLUCIÓN DEL CVRP (CAPACITATED VEHICLE ROUTING PROBLEM)”

DAVID FERNANDO GÓMEZ ATUESTA

CARLOS EDUARDO RANGEL CARVAJAL

Trabajo de investigación en la modalidad “Tesis de Grado” como requisito para optar al título de Ingeniero Industrial

Director:
HENRY LAMOS DÍAZ
Ph.D. En Física- Matemática

UNIVERSIDAD INDUSTRIAL DE SANTANDER
FACULTAD DE INGENIERÍAS FÍSICO-MECÁNICAS
ESCUELA DE ESTUDIOS INDUSTRIALES Y EMPRESARIALES
BUCARAMANGA
2011

AGRADECIMIENTOS

Los autores agradecemos al grupo OPALO de investigación por la oportunidad de realizar el proyecto para dicho grupo.

A los profesores integrantes del grupo de investigación, en especial al Doctor Henry Lamos Díaz por el apoyo brindado junto con los calificadores del proyecto Carlos Eduardo Díaz y Néstor Raúl Ortiz.

A la Universidad Industrial de Santander y Escuela de Estudios Industriales y Empresariales por la formación académica.

DEDICATORIA

Agradezco a Dios por permitirme alcanzar una meta más y por darme la fuerza necesaria para seguir adelante.

A mis padres por su comprensión, amor y paciencia.

A mis hermanos, en especial a Oscar Andrés por su gran apoyo.

A Diana Joya por ser la persona que siempre está a mi lado dándome todo su amor.

A mis amigos, por compartir conmigo sus sueños.

Carlos Eduardo Rangel Carvajal

A mis padres, a mis buenos profesores y a mis amigos.

David Fernando Gómez Atuesta

CONTENIDO

Pág.

LISTA DE TABLAS

Pág.

LISTA DE ILUSTRACIONES

Pág.

LISTADO DE ANEXOS

RESUMEN

TÍTULO

FORMULAR LAS METAHEURÍSTICAS BÚSQUEDA TABÚ Y RECOCIDO SIMULADO PARA LA SOLUCIÓN DEL CVRP (CAPACITATED VEHICLE ROUTING PROBLEM).[1]

AUTORES

DAVID FERNANDO GÓMEZ ATUESTA Y CARLOS EDUARDO RANGEL CARVAJAL. [2]

PALABRAS CLAVES

Ruteo de vehículos con capacidad (CVRP), metaheurísticas, Búsqueda Tabú, Recocido Simulado, Recocido Lineal, Recocido Factor, Clarke and Wrigth, Vecino más Cercano, MATLAB®.

El objetivo principal del problema de ruteo de vehículos con capacidad (CVRP) es encontrar una serie de rutas óptimas de entrega, que permitan satisfacer la demanda de los clientes. Desde la perspectiva de la optimización, el CVRP es un problema de optimización de tipo combinatorio de gran complejidad debido al gran número de posibles soluciones existentes.

En la literatura se encuentran diferentes métodos de solución a este tipo de problemas. La investigación se encamina hacia el uso de las metaheurísticas; específicamente en las metodologías de búsqueda tabú y recocido simulado. Estas metaheurísticas, permiten encontrar soluciones al CVRP cercanas al óptimo en tiempos computacionales relativamente pequeños. Para ello se desarrollaron dos algoritmos (búsqueda tabú y recocido simulado) que fueron programados en el software MATLAB®. y cuyo resultado fue una herramienta que permite solucionar el CVRP de forma rápida y eficiente.

Por último se realizó un diseño de experimentos con el fin de hallar cuáles fueron los parámetros que incidieron en la respuesta dada por los algoritmos y concluir cuál de las dos metodologías es más eficiente en la solución del CVRP.

[1] Proyecto de grado

[2] Facultad de Ingenierías Físico-mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Henry Lamos Diaz.

ABSTRACT

TITLE

ASK THE TABU SEARCH META HEURISTIC AND SIMULATED ANNEALING FOR THE SOLUTION OF CVRP (CAPACITATED VEHICLE ROUTING PROBLEM). [3]

AUTHORS

DAVID FERNANDO GOMEZ ATUESTA AND CARLOS EDUARDO RANGEL CARVAJAL.[4]

KEY WORDS

Capacitated Vehicle Routing Problem(CVRP), Metaheuristics, Tabu Search, Simulated Annealing, Linear Annealing, Annealing Factor, Clarke and Wright, Nearest Neighbor, MATLAB®.

Principal objective for Capacitated Vehicle Routing Problem (CVRP) is finding a set of optimal routes for delivering, to get customer demand. From optimization perspective, CVRP is a very complicated problem of combined type because so many solutions exist.

There are different methods to solve these kinds of problems in literature. Investigation drives to use of Meta heuristics, specifically “Tabu Search” and “Simulated Annealing” methods. These Meta heuristics, can find solutions to CVRP so closed to optimal in relatively small computer times. To do this we developed two algorithms (Tabu Search and Simulated Annealing), they were programmed in MATLAB® software, and result was used as tool to solve the CVRP quickly and efficiently.

Finally, we drived an experimental design to find how parameters influenced the response of the algorithms, and get a conclusions about which one of these two methods is more efficient than other in solving of the CVRP.

[3] Proyecto de grado

[4] Facultad de Ingenierías Físico-mecánicas. Escuela de Estudios Industriales y Empresariales. Director: Henry Lamos Diaz.

INTRODUCCIÓN

El problema de ruteo de vehículos, conocido como VRP (Vehicle Routing Problem), consiste en hallar una ruta o una serie de rutas óptimas que permitan satisfacer la demanda de un grupo de clientes distribuidos geográficamente, el problema de ruteo de vehículos presenta una gran cantidad de variaciones dadas por las restricciones que presenta el entorno.

Dentro de estas variaciones encontramos el problema de ruteo de vehículos con capacidad restringida CVRP (Capacitated Vehicle Routing Problem), este, se encuentra catalogado como un problema de optimización combinatoria perteneciente a la clase de Problemas NP-completos, es decir, la solución óptima se obtiene usando algoritmos que emplean un tiempo de computación que crece de forma superpolinomial (normalmente exponencial). Por consiguiente, en muchos casos, la solución óptima no puede ser obtenida en un tiempo razonable.

Existen dos enfoques para encontrar una solución del problema de ruteo de vehículos; El primer enfoque constituye la clase de algoritmos Óptimos o Exactos, entre los que destacan los métodos de enumeración implícita, como los algoritmos tipo Branch & Bound, plano de corte o las técnicas de programación dinámica[5]. El segundo enfoque constituyen la clase de algoritmos de aproximación: heurísticos o meta-heurísticos, como los de búsqueda local, algoritmos constructivos, búsqueda incompleta, etc. Para cada modelo concreto se pueden realizar una clasificación más exhaustiva de cada una de estas dos clases. Por lo general los algoritmos heurísticos tienen la ventaja añadida de que pueden ser más fácilmente adaptables para resolver modelos más complejos[6].

En el presente trabajo, se propone dar solución a una familia de problemas de ruteo de vehículos con capacidad restringida mediante el uso de las Metaheurísticas búsqueda tabú (Tabú Search - TS) y recocido simulado

[5] H. Papadimitriou, Kenneth Steiglitz "COMBINATORIAL OPTIMIZATION: ALGORITHMS AND COMPLEXITY" Englewood Cliff N.J 2 Editions (1982) pp. 3

[6] B. Nag, BL Golden, and A. Assad, "Vehicle Routing with Site Dependencies,in: Vehicle Routing; Methods and Studies" edited by BL Golden N.J 2 Editions (1988) pp. 35

(Simulated Annealing - SA). El software usado para la programación de los algoritmos es MATLAB®. versión R2009a, el software suministra herramientas suficientes que permiten dar solución al problema de ruteo de vehículos con capacidad, además de un lenguaje de programación sencillo y un excelente desempeño en cuanto a velocidad de procesamiento de datos.

Para conocer la eficiencia de la herramienta se corrieron instancias de la literatura para el CVRP. Se usaron librerías de Augerat et al., de Fisher, de Christofides, y de Gillet y Johnson.

1 OBJETIVOS

1.1 Objetivo General

- Desarrollar algoritmos en MATLAB®.con las metaheurísticas de Búsqueda Tabú y Recocido Simulado para la solución del problema CVRP.

1.2 Objetivos Específicos

- Estudiar las Metaheurísticas Búsqueda Tabú (Tabú Search-TS) y Recocido Simulado (Simulated Annealing)
- Presentar los métodos TS y SA como solución al CVRP.
- Diseñar la programación en MATLAB® de los algoritmos de TS y SA para la solución del CVRP.
- Contrastar las soluciones encontradas con TS y SA para definir cuál método de solución es mejor.

2 MARCO TEÓRICO

Existe en la literatura científica una gran cantidad de proyectos de investigación relacionados con el problema de ruteo de vehículos, tales como los algoritmos de búsqueda tabú y recocido simulado los cuáles son los temas de interés para el presente proyecto. A continuación se discutirán algunos conceptos necesarios para el desarrollo del presente trabajo.

2.1 Optimización Combinatoria

La optimización combinatoria es una rama de a optimización matemática; específicamente se relaciona con la modelación y métodos de solución de problemas de optimización definidos en un conjunto discreto.

Un problema de optimización combinatoria se puede expresar de la siguiente manera:

Minimizar o maximizar

$$\boldsymbol{F(s)};\ \forall\ \boldsymbol{s}\ \in \boldsymbol{S}\ \ en\ \ \boldsymbol{Z}^{\boldsymbol{E}}\ (\boldsymbol{Z}\ es\ el\ conjunto\ de\ números\ enteros)$$

Conjunto $\boldsymbol{E}: \{\boldsymbol{1}, \boldsymbol{2}, \boldsymbol{3}, \dots, \boldsymbol{n}\}$

El conjunto de soluciones factibles $\boldsymbol{S}$ es un conjunto del conjunto $\boldsymbol{Z^E}$**.**

Una función objetivo definida en $Z^E\ a\ R$.

Esto es $F: Z^E \rightarrow R$

El problema de optimización consiste en buscar una solución

- $s^* \in S\ tal\ que\ f(s^*) < f(s)$ para un problema de minimización.
- $s^* \in S\ tal\ que\ f(s^*) > f(s)$ para un problema de maximización.

$\boldsymbol{F(s)}$ Es la llamada función objetivo y representa o mide la calidad de las decisiones.

S Es el conjunto de soluciones factibles que satisfacen ciertas restricciones del problema.

La función objetivo puede ser lineal o no lineal. Es muy común encontrar los problemas de tipo combinatorio en la industria en áreas como la logística, ingenierías y administración de organizaciones con múltiples aplicaciones como por ejemplo: ruteo de vehículos en redes de distribución, planificación de la producción, planificación y secuenciación de trabajos etc.

2.2 Complejidad Computacional

La complejidad computacional de un algoritmo está dada por la dificultad inherente del problema a tratar. Los recursos requeridos durante el cómputo de un algoritmo comúnmente estudiado son el tiempo de ejecución, el espacio y parámetros como el número de procesadores entre otros.

El objetivo general de la complejidad computacional es clasificar los problemas de acuerdo a su tratabilidad, tomando el o los algoritmos más eficientes para resolverlos[7]. Los problemas que pueden ser resueltos en la teoría, pero no en la práctica, son intratables.

2.3 Clases de Complejidad

2.3.1 Complejidad P

Se llaman problemas de complejidad P a aquellos que pueden ser resueltos en un tiempo por un algoritmo en un tiempo polinomial [1].

Se puede demostrar que un problema es de tipo P si:

- Se puede crear un algoritmo que los resuelve en tiempo polinomial.

[7] Problemas de optimización combinatorial; UNIVERSIDAD NACIONAL MAYOR DE SAN MARCOS. Facultad ciencias Matemáticas. http://sisbib.unmsm.edu.pe/bibvirtualdata/monografias/basic/riojas_ca/cap1.pdf

- Usando una transformación polinomial a otro problema que ya está establecido como tipo P.

2.3.2 Complejidad NP (Non-Deterministic Polynomial Time)

Se llaman problemas de complejidad NP a los problemas que **no** pueden ser resueltos por un algoritmo en un tiempo polinomial [1].

Hay dos formas de establecer si un problema es de tipo NP:

- Si se puede encontrar una solución usando una máquina de Turing[8] no – deterministica en tiempo polinomial.
- Si su solución se puede verificar en una máquina de Turing deterministica en tiempo polinomial.

2.3.3 Complejidad NP-Completo (Non-Deterministic Polynomial Time-Complete)

Un problema de decisión es NP-Completo si:

- Es un problema NP
- Todo problema de NP se puede transformar polinomialmente en él.

Debido a esto si se tiene un algoritmo polinomial para el problema, se tendría una solución en P para todos los problemas NP[9].

2.3.4 Complejidad NP-Hard (Non-Deterministic Polynomial Time-Hard)

Son aquellos problemas que a lo sumo son tan difíciles como los NP-Complete, pero no necesariamente en NP.

[8] El matemático Alan Mathison Turing, ideó un dispositivo imaginario al que denominó Máquina de computación lógica LCM ("Logical Computing Machine"). En pocas palabras es un dispositivo que se mueve sobre una secuencia lineal de datos. En cada instante la máquina puede leer un solo dato de la secuencia (generalmente un carácter) y realiza ciertas acciones en base a una tabla que tiene en cuenta su "estado" actual (interno) y el último dato leído. Entre las acciones está la posibilidad de escribir nuevos datos en la secuencia; recorrer la secuencia en ambos sentidos y cambiar de "estado" dentro de un conjunto finito de estados posibles

[9] Stepehn Cook 1971. "The Complexity of Theorem Proving Procedures" ("La Complejidad de los Procedimientos de Prueba de Teoremas")

2.4 El Problema de Ruteo de Vehículos (Generalidades)

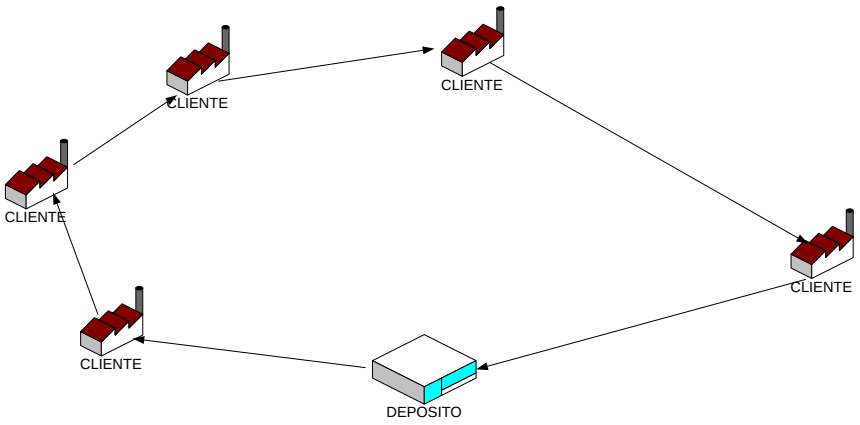
Para empezar a hablar de ruteo de vehículos podemos remontarnos al año 1956 cuando Merill M. Flood publicó “The Traveling Salesman Problem” donde describe algunos métodos heurísticos que nos dan buenos recorridos pero no el mejor. Entre ellos se incluían el nearest-neighbor y el 2-opt. Aquí se introduce la idea de no buscar el resultado óptimo, sino buenas aproximaciones a un menor costo computacional que es pauta que nos guía a lo largo de la investigación.

El problema se puede plantear de la siguiente manera:

Un vehículo (Agente Viajero) necesita visitar $\boldsymbol{n}$ puntos de la ciudad o $\boldsymbol{n}$ ciudades (clientes), sin importar el orden en que lo haga, cada cliente debe ser visitado solo una vez, con el menor costo posible. No hay restricciones de tiempo, no hay demanda asociada a los clientes y siempre se debe retornar al punto de origen.

Se tiene como función objetivo minimizar la distancia o el tiempo de viaje, para lo cual se debe hallar la mejor secuencia de la ruta. Una representación gráfica de la solución al problema del agente viajero (TSP) se presenta en la ilustración 1.

Ilustración 1. Solución gráfica del TSP.



Fuente autores

El problema del TSP es un ejemplo de problema matemático que aparentemente parece tener una solución fácil y en la práctica se presenta como un problema de gran complejidad. Se clasifica dentro de los problemas de tipo $\boldsymbol{NP-Hard}$, debido a que el número de posibles rutas para un número $\boldsymbol{N}$ de clientes es $\boldsymbol{N}!$. A mayor número de clientes el problema se vuelve exponencialmente más complejo. Actualmente, para el problema se conoce el método de solución teórico sin embargo en la práctica no es aplicable para instancias mayores a 10 clientes debido a la cantidad de tiempo computacional que requiere para la solución. La metodología se basa en la búsqueda exhaustiva de la ruta óptima, evaluando todas las posibles combinaciones de recorridos y escogiendo aquella cuyo trazado tiene la menor distancia. El efecto que tiene la combinatoria de las posibles rutas en relación a cantidad de clientes se muestra en la tabla 1.

Tabla 1. Combinatoria de rutas.

Número de Clientes	Número de Posibles Rutas
1	1
2	2
5	120
10	3.628.800
15	1.307.674.368.000
20	2,4329E+18
30	2,65253E+32
40	8,15915E+47
50	3,04141E+64

Fuente autores

El Problema de los $\boldsymbol{m}$ Agentes Viajeros (m-TSP) es una generalización del TSP, aquí se tiene un depósito y una flota de $\boldsymbol{m}$ vehículos. Se debe generar una ruta para cada uno de los agentes, teniendo en cuenta que cada cliente *i* debe ser

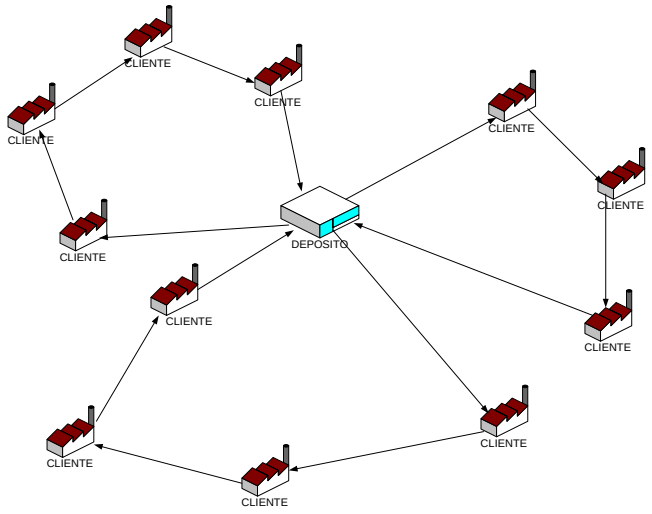
visitado solo una vez por un agente minimizando la distancia total recorrida por los $\boldsymbol{m}$ agentes.

En 1959, Dantzig y Ramser [2], fueron quienes formularon por primera vez el Problema de Ruteo de Vehículos para una aplicación de distribución de combustible. Luego, Clarke y Wright [3] propusieron el popular algoritmo de "Ahorros" que resultó efectivo para su resolución. A partir de estos trabajos, el Ruteo de Vehículos se ha convertido en un área de estudio para el desarrollo de algoritmos que permiten resolver los problemas de manera eficiente.

Como una extensión del m-TSP, se plantea el VRP como la generalización de un conjunto de problemas cuyo objetivo es hallar las rutas de para una flota de vehículos que van a servir a un conjunto de clientes ubicados en cierta zona geográfica, de manera que se satisfagan los requerimientos de los clientes, las restricciones operativas y se minimice el costo total de transporte. El VRP pertenece a la clase de los Problemas de Optimización Combinatoria (COP).

Una representación gráfica de la solución del ruteo para el VRP se presenta en la ilustración 2.

Ilustración 2. Representación gráfica VRP.

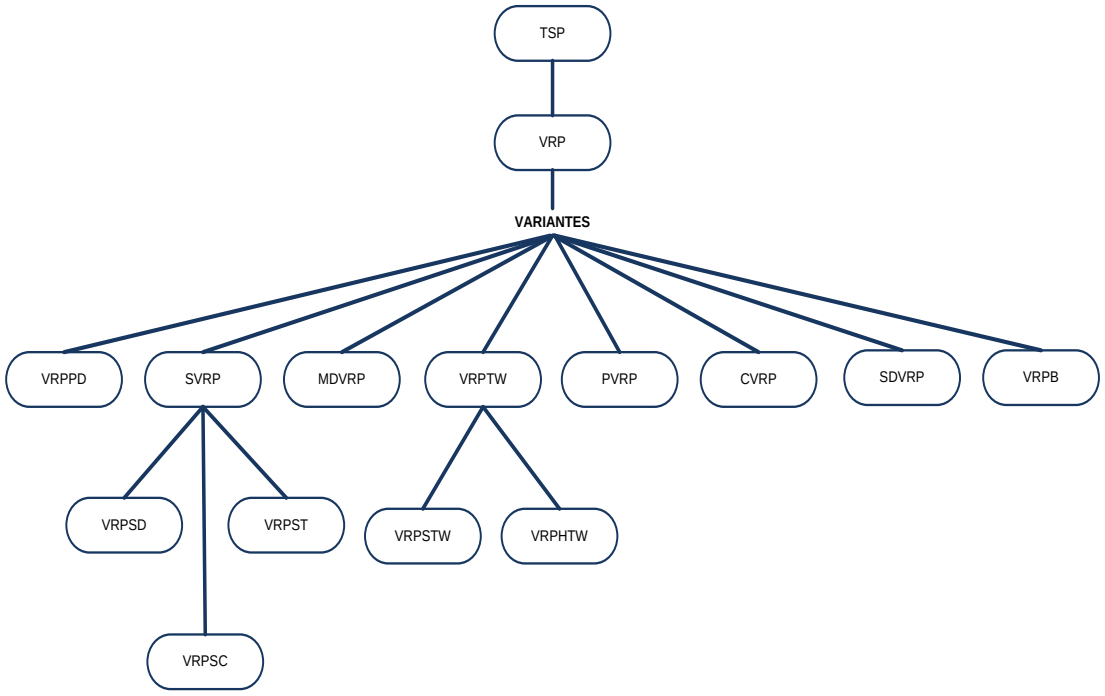


Fuente autores

Las características de los clientes, depósitos y vehículos, así como diferentes restricciones operativas sobre las rutas, dan lugar a diferentes variantes del problema.

Según [4], las variantes más importantes del VRP son las que aparecen en la ilustración 3.

Ilustración 3. Variaciones del VRP.



Fuente Autores

A continuación se explican cada una de las variantes del problema VRP:

2.4.1 Problema de Ruteo de Vehículos Con Envío y Recogida (VRPPD)

Es la variante del VRP con la posibilidad de que el producto recogido en el cliente i, pueda ser llevado al cliente j o que los clientes devuelvan productos a la empresa proveedora. Se debe asegurar que los vehículos que hacen las rutas, tengan capacidad disponible para embarcar los productos devueltos.

2.4.2 Problema de Ruteo de Vehículos Probabilístico (SVRP)

Es la variante del VRP en donde uno o más componentes del problema presentan cierta probabilidad de ocurrencia Pi.

2.4.3 Problema de Ruteo de Vehículos Con Demandas Estocásticas (VRPSD)

Es la variante del VRP en donde las demandas de los clientes son variables aleatorias con una distribución de probabilidad conocida.

2.4.4 Problema de Ruteo de Vehículos Con Clientes Estocásticos (VRPSC)

En este caso la variable aleatoria son los clientes, quienes se presentan con una determinada probabilidad.

2.4.5 Problema de Ruteo de Vehículos Con Tiempos Estocásticos (VRPST)

En esta variante los tiempos de viaje y de servicio son variables estocásticas.

2.4.6 Problema de Ruteo de Vehículos Con Múltiples Depósitos (MDVRP)

Es la variante del VRP con múltiples depósitos y flotas de vehículos. Todos los clientes deben ser asignados a un solo depósito con el fin de minimizar los costos de servicio.

2.4.7 Problema de Ruteo de Vehículos Con Ventanas De Tiempo (VRPTW)

Es la variante del VRP con la restricción adicional de que cada cliente tiene asociado un intervalo de tiempo fijo durante el cual pueden ser visitados.

2.4.8 Problema de Ruteo de Vehículos Con Ventanas De Tiempo Fuertes (VRPHTW)

Para este problema se pueden considerar ventanas de tiempo duras en las que no es posible realizar la entrega al cliente fuera de los intervalos de tiempo establecidos.

2.4.9 Problema de Ruteo de Vehículos Con Ventanas de Tiempo Suaves (VRPSTW)

En esta variante con ventanas de tiempo suaves se permite la entrega fuera de estos intervalos de tiempo pero con una penalización.

2.4.10 Problema de Ruteo de Vehículos Periódico (PVRP)

Es la variante del VRP con entregas realizadas sobre un periodo de M días, en lugar de un único día de servicio.

2.4.11 Problema de Ruteo de Vehículos Capacitado (CVRP)

Es la variante del VRP con la restricción adicional de que todos los vehículos que conforman la flota tienen capacidad de carga limitada y uniforme. La demanda total asignada a una ruta no debe exceder la capacidad del vehículo.

2.4.12 Problema de Ruteo de Vehículos Con Entregas Divididas (SDVRP)

Es la variante del VRP con clientes que pueden ser servidos por más de un vehículo.

2.4.13 Problema de Ruteo de Vehículos con Devoluciones (VRPB)

El VRPB es un VRP en que los clientes pueden demandar o devolver artículos. Se debe tener en cuenta que las devoluciones de los clientes caben en el vehículo. Pero además, se debe cumplir que todas las entregas se realizan antes de las recogidas. Esto se debe al hecho de que los vehículos se cargan por la parte trasera y que la recolocación de la carga en los vehículos se considera antieconómica o no factible.

2.5 Descripción del Problema

El Problema CVRP básico trata de determinar los recorridos de $\boldsymbol{k}$ vehículos de capacidad $\boldsymbol{Q_k}$ que partiendo de un origen común (depósito) deben pasar por un conjunto de lugares de interés (clientes) para recoger o distribuir mercancías según una demanda $\boldsymbol{d_i}$, y volver de nuevo al origen.

2.5.1 Características Generales

A continuación se presentan las características generales del CVRP, es importante aclarar que no son las únicas existentes y varían teniendo en cuenta la particularidad de cada problema.

1) Definir el costo a minimizar usando como función objetivo cualquiera de los siguientes criterios:
   a) Minimizar el costo de transporte total, teniendo en cuenta las distancias entre los clientes y los depósitos, además de los costos asociados a los vehículos usados.
   b) Minimizar la cantidad de vehículos usados para el abastecimiento de todos los clientes.
   c) Balancear las rutas con carga de los vehículos.

2) Los clientes deben ser visitados una sola vez y por un solo vehículo, es decir, un cliente solo puede existir en una sola ruta.
3) Existe un único depósito, del cual se reparten todas las mercancías a todos los clientes.
4) Todas las rutas inician y terminan en el depósito.
5) Las demandas de los clientes son fijas y se conocen.
6) Los vehículos usados son iguales en cuanto a capacidad y costo.
7) El problema es simétrico, es decir, es costo de ir al cliente *i* al cliente *j* es igual que el de ir del cliente *j* al cliente *i.*

8) El número de vehículos se puede obtener dividiendo los requerimientos totales sobre la capacidad de un vehículo.

$$Número\ de\ Vehículos = \frac{Requerimientos\ totales}{Capacidad\ del\ Vehículo}$$

2.5.2 Formulación Matemática

Índices

Los índices del modelo son:

$\boldsymbol{i}$ = nodo de partida $\boldsymbol{i}$ $(\mathbf{1}, \mathbf{2}, \ldots\ldots, \boldsymbol{n})$

$\boldsymbol{j}$ = nodo de llegada $\boldsymbol{j}$ $(\mathbf{1}, \mathbf{2}, \ldots\ldots, \boldsymbol{n})$

$\boldsymbol{n}$ = nodos totales

$\boldsymbol{k}$ = Vehículo $\boldsymbol{k}$ $(\mathbf{1}, \mathbf{2}, \ldots\ldots, \boldsymbol{K})$

Variables

Las variables que se definen son:

$\boldsymbol{X_{i,j}^{k} = 1}$ Si se asigna el vehículo $\boldsymbol{k}$ para recorrer el arco del nodo $\boldsymbol{i}$ al nodo $\boldsymbol{j}$, de lo contrario cero (0). Es una variable binaria.

$\boldsymbol{y_{i,j} = 1}$ Si se realiza el recorrido desde *i* hasta *j* o cero (0) de lo contrario. Es una variable binaria.

$\boldsymbol{k} =$ Número de vehículos a utilizar.

Parámetros

Los parámetros del problema son:

$\boldsymbol{C_{i,j}}$ = Costo de transporte del nodo i al nodo *j*

$\boldsymbol{d_i}$ = Demanda en el nodo *i*

$\boldsymbol{d_j}$ = Demanda en el nodo *j*

$\boldsymbol{Q}$ = Capacidad del vehículo

$\boldsymbol{n}$ = Número de clientes

Modelo

Minimizar

$$\sum_{(i,j)\in A} C_{i,j} y_{i,j} \ (1)$$

La anterior ecuación representa la función objetivo.

Sujeto a:

$$\sum_{1\leq k\leq K} x_{i,j}^{k} = y_{i,j}; \forall\, i,j \ (2)$$

La ecuación hace obligatoria la asignación de un vehículo a la ruta(i,j), 1 si es recorrida, y 0 en caso contrario.

$$\sum_{1\leq j\leq n} y_{i,j} = 1 \forall\, i \ (3)$$

$$\sum_{1\leq i\leq n} y_{i,j} = 1 \forall\, j \ (4)$$

$$\sum_{1\leq j\leq n} y_{0,j} = k \ (5)$$

$$\sum_{1 \le i \le n} y_{i,0} = k \quad (6)$$

Las restricciones (5) y (6) indican que k es la cantidad de vehículos utilizados en la solución y que todos los que parten del almacén deben regresar al mismo.

$$\sum_{1 \le i \le n} \sum_{1 \le j \le n} d_i x_{i,j}^k \le; Q \; \forall \; k \quad (7)$$

La restricción (7) garantiza que los vehículos no sobrepasen su capacidad.

$$\sum_{i \in S} \sum_{j \in S} y_{i,j} \le |S| \quad (8)$$

La restricción (8) vigila que la solución no contenga ciclos usando los nodos $1, 2, \ldots n$.

$\forall$ $subconjunto\ S\ del\ conjunto\ de\ elementos\ (1, 2, 3 \ldots . n)$;

$$k \le K \; (9)$$

La restricción (9) limita el número de vehículos a usar.

2.6 CONCEPTO DE HEURÍSTICA Y META-HEURÍSTICA.

Según Melián[10], se habla de heurística para referirse a una técnica, método o procedimiento inteligente que realiza una tarea que no es producto de un riguroso análisis formal, sino de conocimiento experto sobre la tarea, de manera que se tengan buenas soluciones a un problema determinado en cuanto a calidad y recursos empleados, sin garantizar la optimalidad.

Las metaheurísticas, son estrategias inteligentes que sirven para mejorar procedimientos heurísticos muy generales, es decir, muchas veces las

10 María Belén Melián Batista, Profesora titular de la Universidad de la Laguna España. Dpto. de Estadística, Investigación Operativa y Computación.

metaheurísticas se basan en heurísticas muy sencillas a las que se les aplica una estrategia adicional con el fin de no caer en óptimos locales y de esta manera encontrar una mejor solución al problema.

Las metaheurísticas se pueden clasificar en:

- Metaheurísticas de Relajación
- Metaheurísticas de Construcción
- Metaheurísticas de Búsqueda
- Metaheurísticas de Evolución

Las metaheurísticas de relajación son procedimientos de resolución que utilizan relajaciones del modelo original para facilitar la solución del problema planteado sin descuidar la calidad de las soluciones. Las relajaciones más típicas son la eliminación, debilitamiento o modificación de las restricciones. Algunas metaheurísticas de relajación son los métodos de relajación lagrangiana o de restricciones subordinadas [7].

Las metaheurísticas constructivas establecen estrategias para seleccionar de manera iterativa, partiendo de una estructura inicialmente vacía, las componentes con las que se construye una buena solución del problema. Las meta-heurísticas más conocidas son la estrategia voraz o greedy y la meta-heurística GRASP, la cual incorpora a la estrategia greedy pasos aleatorios con criterios adaptativos para la selección de los elementos a incluir en la solución [8].

Las metaheurísticas de búsqueda establecen estrategias para recorrer el espacio de soluciones del problema, transformando de forma iterativa soluciones iniciales. Las metaheurísticas pueden basarse en búsquedas monótonas, algoritmos escaladores o búsquedas locales. Las búsquedas locales se basan en el estudio de soluciones del vecindario o entorno de la solución que realiza el recorrido. La metaheurística establece como pauta elegir iterativamente la mejor de las soluciones mientras exista alguna mejora posible. Sin embargo, las búsquedas

locales pueden quedarse atrapadas en un óptimo local. Lo anterior puede ser solucionado mediante la aplicación de búsquedas globales que incorporan pautas para escapar de los óptimos locales. Estas pautas pueden ser: Volver a iniciar la búsqueda desde otra solución de arranque, modificar la estructura de entornos que se está aplicando y permitir movimientos o transformaciones de la solución de búsqueda que no sean de mejora mediante criterios de aceptación estocásticos o utilizando la memoria del proceso de búsqueda [6].

Las metaheurísticas evolutivas establecen estrategias para conducir la evolución en el espacio de búsqueda de conjuntos de soluciones (Población) con el objetivo de acercarse a la solución óptima. La metaheurística más conocida perteneciente a este género son los Algoritmos Genéticos [9]. Otras metaheurísticas se basan en mecanismos de aprendizaje, que ayudan a construir mejores soluciones basándose en el conocimiento de soluciones anteriores. Ejemplo de ello son las metaheurísticas basadas en Redes Neuronales y el Sistema de Colonia de Hormigas.

2.7 Búsqueda Tabú (Tabu Search)

La búsqueda tabú es una metaheurística que usa un procedimiento de tipo iterativo para resolver problemas de optimización combinatoria discretos. Fue propuesta inicialmente por Fred Glover[11].

El principal objetivo de la búsqueda tabú es escapar de óptimos locales, para ello emplea algunas metodologías como el uso de memorias flexibles o cambio de la estructura del problema creando o relajando las restricciones con la finalidad de realizar la búsqueda en áreas no factibles pero que de alguna manera nos puede llevar a una solución de buena calidad.

“La búsqueda tabú se basa en tres principios:

11 “Tabu Search Methods in Artificial Intelligence and Operations Research, " ORSA Artificial Intelligence, Vol. 1, No. 2, 6, 1987

1. *Uso de estructuras de memoria basadas en atributos diseñados para permitir criterios de evaluación e información de búsqueda histórica, la cual se explota más a fondo que las estructuras de memoria rígida (como en ramificación y acotamiento) o por sistemas de pérdida de memoria (como recocido simulado y otros métodos aleatorizados)*
2. *Un mecanismo asociado de control, mediante el empleo de estructuras de memoria, basado en el inter- juego entre las condiciones que restringen y liberan al proceso de búsqueda (envuelto en las restricciones tabú y el criterio de aspiración).*
3. *La incorporación de funciones de memoria de diferentes lapsos de tiempo, desde término corto hasta de término largo.*

El tipo de memorias que se usan en TS se clasifican en:

a) Memoria a corto plazo: También llamada lista tabú; en esta memoria se almacena el historial de los últimos movimientos realizados (movimientos tabú) a través de una lista FIFO (First Int First Out), prohibiendo que los movimientos se repitan en las próximas iteraciones, así evita ciclos y escapa de mínimos locales.
b) Memoria a medio plazo: en ella se registran los atributos más comunes de un conjunto de soluciones.
c) Memoria de largo plazo: La memoria guarda un registro de las zonas que no han sido exploradas aún, diversificando de esta manera la búsqueda.

Las memorias a mediano plazo y largo plazo usan como base las estrategias de intensificación y diversificación. La estrategia de intensificación modifica los parámetros usados en la búsqueda para mejorar la elección de combinaciones de movimientos y características de las soluciones encontradas. Es necesario encontrar e identificar un conjunto de soluciones con características especiales y cuyas características puedan ser incorporadas a las nuevas soluciones.

La estrategia de diversificación trata de llevar la búsqueda a zonas del espacio de soluciones no visitadas antes, generando de esta manera soluciones que difieren significativamente de las ya encontradas. Además de emplear estas estrategias para la búsqueda del óptimo global emplea un método llamado nivel de aspiración. El objetivo de aplicar este criterio es determinar en que condiciones se puede admitir un movimiento clasificado como tabú, cambiando su clasificación de tabú cuando su memoria a corto plazo expire.

El criterio de aspiración más sencillo de aplicar, es considerar una solución $\boldsymbol{S}$ cuando $\boldsymbol{f(s_0) < f(s^*)}$, este movimiento facilita una nueva dirección en la búsqueda y garantiza que no se creen ciclos.

A continuación se presenta un algoritmo muy común de la búsqueda tabú:

$Seleccionar\ una\ solución\ inicial\ s_0$

$Hacer\ s^* = s_0; T = \emptyset, niter = 0, kiter = 0$

$Repetir$

$niter = niter + 1;$

$Seleccionar\ s\ \in N(s_0)\ /\ s \notin T\ o\ s\ Verifica\ criterio\ de\ aspiración\ con\ f(s) mínimo$

$Hacer\ \ s_0 = s, si\ f(s_0) < f(s^*) entonces:\ hacer\ s^* = s\ y\ kiter = niter$

$Actualizar\ T; hasta\ niter - kiter \geq maxiter$

Fuente: "**Concentración Heurística: descripción y comparación con otros Metaheurísticos"** Pacheco Bonrostro, Joaquín A. Facultad de C. Económicas y Empresariales, Departamento de Economía (matemáticas). Universidad de Burgos.

2.7.1 Definiciones y parámetros usados en el algoritmo

A continuación se describen algunos términos usados en el algoritmo desarrollado.

2.7.1.1 Secuencias o tours

La representación de las soluciones se da a través una secuencia creada por el vector solución. Este incluye el orden en que los clientes son visitados por cada vehículo. En el tour se pueden identificar las rutas, por ejemplo, para una instancia con 5 clientes un tour se representa mediante la secuencia:

1	2	3	1	4	5	1	6	1

Aquí el número 1 representa el depósito y el número de vehículos es igual a 3.

Al dividir el tour en rutas tenemos:

Ruta 1

1	2	3	1

Ruta 2

1	4	5	1

Ruta 3

1	6	1

2.7.1.2 Matriz de Costos o distancias.

La matriz asociada representa los costos o distancias de ir de un elemento $\boldsymbol{i}$ a un elemento $\boldsymbol{j}$. Es importante aclarar que los problemas que se tratan en el algoritmo son simétricos, esto es $\boldsymbol{C_{i,j} = C_{j,i}}$ para todo $\boldsymbol{i, j}$. La matriz asociada es la siguiente:

$$C = \begin{pmatrix} c_{11} & c_{12} & c_{13} \dots c_{1n} \\ c_{21} & c_{12} & c_{23} \dots c_{2n} \\ c_{31} & c_{32} & c_{23} \dots c_{3n} \\ \vdots & \vdots & \vdots & \vdots \\ c_{n1} & c_{n2} & c_{n3} & c_{nn} \end{pmatrix}$$

2.7.1.3 Listado de Demandas

En esta lista se muestran las demandas asociadas a cada uno de los clientes. Este listado nos sirve para determinar la cantidad de vehículos que se van a utilizar en total. Cabe recordar que cuando se asignen las rutas, los vehículos no pueden sobrepasar su capacidad.

Tabla 2 Listado de demandas.

Cliente	Demanda
1	D1
2	D2
3	D3
…	…
n	Dn

Fuente Autores

2.7.1.4 Solución Factible

Una solución factible, es aquella solución que cumple a cabalidad con las restricciones del problema (capacidad de los vehículos, número de vehículos, etc…)

2.7.1.5 Solución Inicial

Para el cálculo de la solución de partida o solución inicial, se usaron dos técnicas muy conocidas en la literatura que son: “el vecino más cercano” y “Clarke and Wright”. A continuación se presenta la forma como operan estas técnicas:

- **Heurística del Vecino más Cercano (Nearest Neighbor)**

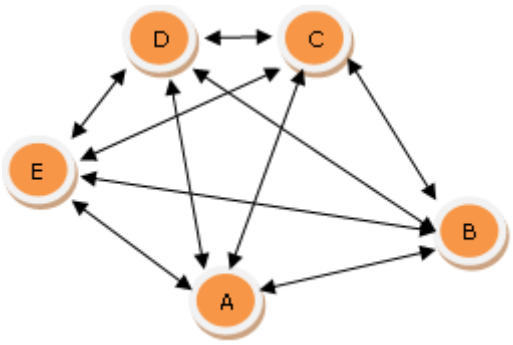
El algoritmo es clasificado como goloso o greedy, ya que en cada paso que se efectúa hace lo que se supone es mejor para ese estado.

En el caso del CVRP se parte del depósito (cliente con demanda=0) y se busca entre todos los clientes restantes el más cercano (menor distancia) a este depósito, luego busca el más cercano al último cliente encontrado y se va armando el tour de manera sucesiva hasta que se visiten todos los clientes. Además de tener en cuenta que se deben visitar todos los clientes, no se debe exceder la capacidad de los vehículos, a medida que se va alcanzando la capacidad de un vehículo se va empleando otro para la visita de los clientes faltantes. Cuando se visita el último cliente se debe retornar al depósito donde se inició el tour

El algoritmo se ejecuta en un tiempo proporcional a n^2, lo que representa una mejora significativa frente a $n!$. La solución que genera el algoritmo depende de que tan uniforme es la distribución de los clientes en el plano, la solución puede estar tan cerca a la óptima que no habría duda en aceptarla o tan lejos que no es mucho lo que nos puede aportar. Esto se debe a que no se tienen en cuenta estados posteriores ni anteriores al estado actual, sin embargo, las soluciones encontradas usando este algoritmo pueden ser un buen punto de partida a la hora de aplicar el algoritmo de búsqueda tabú o recocido simulado.

Ejemplo: en la ilustración 4 se muestra la red de transporte conformada por los clientes (nodos) B, C, D y el nodo depósito A. La tabla 3 muestra las respectivas demandas de los clientes. Por último la ilustración 5 muestra la matriz de costos, la capacidad y el número de vehículos.

Ilustración 4. Red de transporte.



Fuente autores

Tabla 3. Tabla de Demandas.

Cliente	Demanda
A	0 (Depósito)
B	10
C	10
D	10
E	10

Fuente autores

Ilustración 5. Matriz de Costo.

	A	B	C	D	E
A	0	8	2	16	4
B	8	0	4	12	2
C	2	4	0	16	6
D	16	12	16	0	8
E	4	2	6	8	0

Fuente autores

Capacidad de los Vehículos

$$\boldsymbol{Q = 20}$$

Número de Vehículos

$$\boldsymbol{k = 2}$$

Para la construcción de un tour se llevan a cabo los pasos descritos a continuación:

1. Se empieza a construir el tour usando como punto de partida el cliente A (depósito).
2. El próximo cliente elegido es el cliente más cercano (menor distancia), siempre que no esté incluido en el tour.
3. Repetir el paso 2 hasta que todos los clientes estén en el tour. En cada momento hay que verificar que no se viole la restricción de capacidad de los vehículos.

En el ejemplo anterior se parte desde el cliente A (depósito) y se elige el arco A-C pues C tiene el menor costo, desde C, el arco de menor costo es el que va hacia B. En este momento el primer vehículo alcanzo su máximo de capacidad por lo que debe regresar al depósito. El tour hasta el momento queda de la siguiente manera A-C-B-A. Quedan por visitar los clientes D y E, por tanto se visita el cliente más cercano que es E. Desde el punto E solo nos falta visitar el cliente D y el tour quedaría de la siguiente manera A-C-B-A-E-D-A, con un costo total de 42.

- **Heurística de "Clarke and Wright Savings"**

La heurística de Clarke and Wright se clasifica dentro de las heurísticas de tipo constructivo, es muy utilizada para la solución del VRP, pero haciéndole unas pequeñas modificaciones se puede usar para hallar soluciones del CVRP

obteniendo buenos resultados. Para describir cómo funciona el algoritmo hay que definir las siguientes variables:

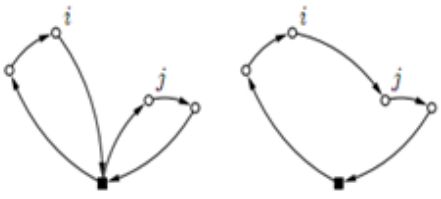
- $(\mathbf{0},\ldots,\boldsymbol{i},\mathbf{0})$ **=** Orden de la ruta generada tomando cero como el depósito hasta el cliente $\boldsymbol{i}$ y retornado al depósito.
- $(\mathbf{0},\boldsymbol{j},..,\mathbf{0})$ **=** Orden de la ruta generada tomando cero como el depósito al cliente $\boldsymbol{j}$ y retornando al depósito.
- $(\mathbf{0},..,\boldsymbol{i},\boldsymbol{j},..,\mathbf{0})$ **=** Orden de la ruta generada tomando cero como el depósito al cliente $\boldsymbol{i}$, al cliente $\boldsymbol{j}$ y retornando al depósito.
- $\boldsymbol{S_{i,j}}$ **=** es el ahorro que se obtiene al unir los clientes $(\boldsymbol{i},\boldsymbol{j})$ en una sola ruta.
- $\boldsymbol{d_{i,0}}$**=** distancia recorrida del cliente $\boldsymbol{i}$ al depósito.
- $\boldsymbol{d_{0,j}}$**=** distancia recorrida del cliente $\boldsymbol{j}$ al depósito.
- $\boldsymbol{d_{i,j}}$ **=**distancia recorrida del cliente $\boldsymbol{i}$ al cliente $\boldsymbol{j}$.

El algoritmo básicamente opera de la siguiente manera:

Se parte de una solución inicial donde se asume que cada cliente es visitado por un solo vehículo y se combinan las rutas que den los ahorros más significativos, de manera tal que no se violen las restricciones del problema.

Se combinan las rutas $(\mathbf{0},\ldots,\boldsymbol{i},\mathbf{0})$ y $(\mathbf{0},\boldsymbol{j},..,\mathbf{0})$, para formar una nueva ruta $(\mathbf{0},..,\boldsymbol{i},\boldsymbol{j},..,\mathbf{0})$, obteniéndose un ahorro en distancia descrito por $\boldsymbol{S_{i,j}} = \boldsymbol{d_{i,0}} + \boldsymbol{d_{0,j}} - \boldsymbol{d_{i,j}}$ puesto que en la nueva ruta los arcos $(\boldsymbol{i},\mathbf{0})$ y $(\mathbf{0},\boldsymbol{j})$ desaparecen al agregar el arco $(\boldsymbol{i},\boldsymbol{j})$.

Ilustración 6. Dos rutas antes y después de Unirlas.



Fuente: Alfredo Olivera, “Heurísticas para el problema de ruteo de vehículos”, Universidad de la República, Montevideo Uruguay 2004.

Existe una versión paralela en la que se trabaja sobre todas las rutas simultáneamente y otra secuencial que construye las rutas de a una[12].

Tabla 4. Pasos para elaborar un algoritmo de ahorros versión paralela.

Paso 1 (Inicialización)	Para cada cliente $\boldsymbol{i}$ construir la ruta $(\mathbf{0},\ldots,\boldsymbol{i},\mathbf{0})$.
Paso 2 (Cálculo de ahorros)	Calcular $\boldsymbol{S_{i,j}}$ para cada par de clientes $\boldsymbol{i}$ y $\boldsymbol{j}$.
Paso 3 (Mejor Unión)	Sea $\boldsymbol{S_{i*,j*}}$= max $\boldsymbol{S_{i,j}}$, donde el máximo se toma entre los ahorros que no han sido considerados aún. Sean $\boldsymbol{r_{i*}}$ y $\boldsymbol{r_{j*}}$ las rutas que contienen a los clientes $\boldsymbol{i}*$ y $\boldsymbol{j}*$ respectivamente. Si $\boldsymbol{i}*$ es el último cliente de $\boldsymbol{r_{i*}}$ y $\boldsymbol{j}*$es el primer cliente de $\boldsymbol{r_{j*}}$ y la combinación de $\boldsymbol{r_{i*}}$ y $\boldsymbol{j}*$ es factible, combinarlas. Eliminar$\boldsymbol{S_{i*,j*}}$ de futuras consideraciones. Si quedan ahorros por examinar ir a 3, si no terminar.

Fuente: Alfredo Olivera, "Heurísticas para el problema de ruteo de vehículos", Universidad de la República, Montevideo Uruguay 2004

Tabla 5. Pasos para elaborar un algoritmo de ahorros versión secuencial.

Paso 1 (Inicialización)	Para cada cliente $\boldsymbol{i}$ construir la ruta $(\mathbf{0},\ldots,\boldsymbol{i},\mathbf{0})$.
Paso 2 (Cálculo de ahorros)	Calcular $\boldsymbol{S_{i,j}}$ para cada par de clientes $\boldsymbol{i}$ y $\boldsymbol{j}$.
Paso 3 (Selección)	Si todas las rutas fueron consideradas, terminar. Si no, seleccionar una ruta que aún

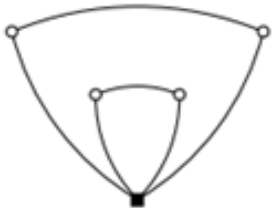
12 Alfredo Olivera, "Heurísticas para el problema de ruteo de vehículos", Universidad de la República, Montevideo Uruguay 2004.

	no haya sido considerada.
Paso 4 (Extensión)	Sea $(\mathbf{0},..,\boldsymbol{i},\boldsymbol{j},..,\mathbf{0})$ la ruta actual. Si no existe ningún ahorro conteniendo a $\boldsymbol{i}$ o a $\boldsymbol{j}$, ir a 3. Sea $\boldsymbol{S_{k*i}}$ (o $\boldsymbol{S_{jl*}}$), el máximo ahorro conteniendo a $\boldsymbol{i}$ o a $\boldsymbol{j}$. Si $\boldsymbol{k*}$ (o $\boldsymbol{l*}$) es el último (o primer) cliente de su ruta y la combinación de dicha ruta con la actual es factible, realizar dicha combinación. Eliminar $\boldsymbol{S_{k*i}}$ (o $\boldsymbol{S_{jl*}}$) de futuras consideraciones. Ir a 4.

Fuente: Alfredo Olivera, "Heurísticas para el problema de ruteo de vehículos", Universidad de la República, Montevideo Uruguay 2004.

Algunos autores han propuesto una mejora al algoritmo de ahorros debido a que alguna veces este genera rutas circulares lo cual no es deseable.

Ilustración 7. Rutas circulares.



Fuente: Alfredo Olivera, "Heurísticas para el problema de ruteo de vehículos", Universidad de la República, Montevideo Uruguay 2004.

Partiendo del concepto que a menor distancia entre los clientes mayores son los ahorros obtenidos, se propone añadir un parámetro λ (parámetro de forma o *shape parameter*) cuyo objetivo es penalizar la unión entre clientes lejanos. El parámetro puede usarse también para crear un nuevo conjunto de soluciones

cuando se ejecuta varias veces el algoritmo y se le dan valores diferentes a λ en cada iteración[13].

El nuevo ahorro queda de la siguiente manera:

$$S_{i,j} = d_{i,0} + d_{0,j} - \lambda d_{i,j}$$

Las soluciones encontradas con este algoritmo, pueden ser mejoradas usando operadores de búsqueda local, tal como inserción intercambio.

2.7.1.6 Definición de Vecindad

La vecindad de una solución se puede definir como el conjunto de todas las soluciones que se pueden encontrar a partir de una solución s' por medio de un movimiento σ; un movimiento puede ser una inserción, un intercambio o una eliminación de componentes en una solución s.[14]

$$N(s) = \{s' \in S : s \xrightarrow{\sigma} s'\}$$

Donde,

$N(s)$ Representa la vecindad con respecto a s.

s Representa una solución tomada del espacio de soluciones S.

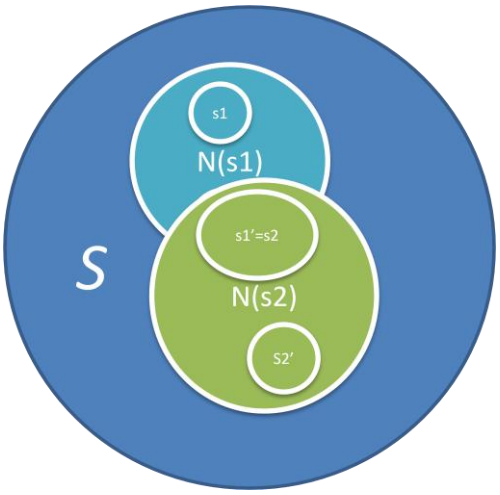
s' Es el vecino de s generado a partir del movimiento σ.

En la siguiente ilustración muestra la representación gráfica de vecindad:

13 Alfredo Olivera, “Heurísticas para el problema de ruteo de vehículos”, Universidad de la República, Montevideo Uruguay 2004

14 A. A. MARTÍNEZ MORALES.: "Algoritmo Basado en Tabu Search para el Cálculo del Índice de Transmisión de un Grafo". Departamento de computación Facultad de ciencias y tecnología. FARAUTE de Ciencias y Tecnología, Vol. 1, No. 1, páginas 31-39. Universidad de Carabobo, Valencia, Estado Carabobo, Venezuela (2006).

Ilustración 8. Vecindad generada a partir de una solución s.



Fuente autores.

2.7.1.7 Estrategia de Vecindad

- **Operador λ-intercambio**

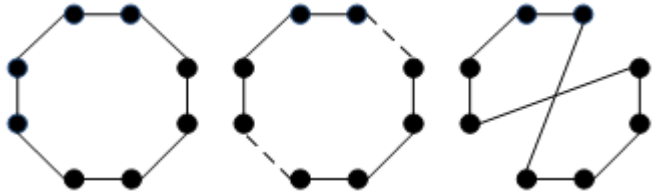
Uno de los operadores de búsqueda local para una ruta más conocido es el λ-intercambio definido por Lin[15]. Un λ-intercambio, consiste en eliminar λ-arcos de la solución (λ>1) y reconectar los λ elementos restantes. Una solución se llama λ-óptima si puede ser mejorada utilizando λ-intercambios. Se llama λ-opt a un algoritmo de búsqueda local que utiliza λ-intercambios hasta alcanzar una solución λ-óptima.

Usualmente se implementan 2-intercambios y 3-intercambios. En general, estas movidas invierten el orden de algunas visitas. Asumiendo que las movidas no afectan la factibilidad y que los costos son simétricos, puede buscarse movimientos que mejoren el costo de una ruta sin necesidad de explorar todas las

15 Lin, S.: Computer solutions of the traveling salesman problem. Bell SystemTechnical Journal 44 (1965) 2245–2269

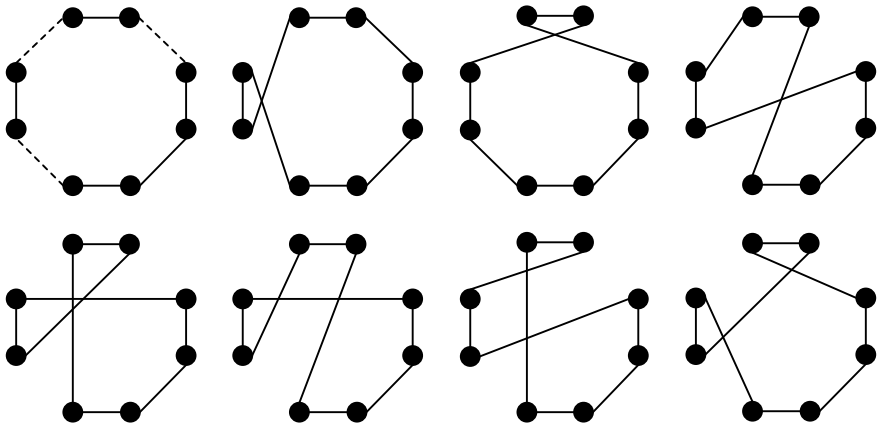
posibilidades[16]. Si no se cumplan dichos supuestos, buscar λ-intercambios tendría un costo computacional más elevado debido a la necesidad de incorporar chequeos de factibilidad y re-calcular algunos costos.

Ilustración 9. El único 2-intercambio posible para los arcos marcados..



Fuente: Alfredo Olivera, "Heurísticas para el problema de ruteo de vehículos", Universidad de la República, Montevideo Uruguay 2004.

Ilustración 10. Todos los 3-intercambios posibles para los arcos marcados.



Fuente: Alfredo Olivera, "Heurísticas para el problema de ruteo de vehículos", Universidad de la República, Montevideo Uruguay 2004.

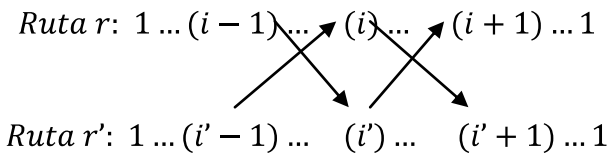
Vecindarios tipo Gendreu-Clarke

Se pueden considerar tres tipos de intercambios entre dos rutas diferentes de la solución actual:

16 Johnson, D., McGeoch., L.: The Traveling Salesman Problem: a case studyin local optimization. In: Local Search in Combinatorial Optimization.John Wiley and Sons (1997) 215–310.

I. Mutación (4-intercambio)

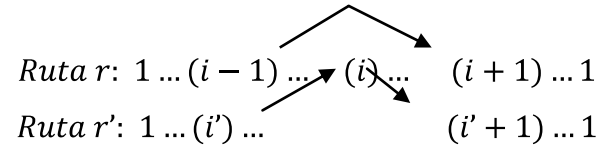
Es el intercambio del elemento i de la ruta r con el elemento i' de la ruta r' [26].



Eliminación de los arcos $(i-1,i),(i,i+1),(i'-1,i'),(i'.i+1')$ e incorporación de los arcos $(i-1,i'),(i',i+1),(i'-1,i),(i,i'+1)$.

II. Inserción (3-intercambio)

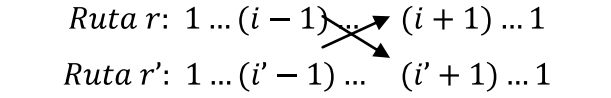
Es la inserción del elemento i de la ruta r con el elemento i' e $i'+1$ de la ruta r' [26].



Eliminación de los arcos $(i-1,i),(i,i+1),(i',i'+1)$ e incorporación de los arcos $(i',i),(i,i'+1),(i'-1,i+1)$.

III. 2-opt(2-intercambio)

Con esta estrategia lo que se hace es invertir completamente una parte de la secuencia de visitas. [26].



Eliminación de los arcos $(i,i+1),(i'.i'+1)$ e incorporación de los arcos $(i',i+1),(i,i'+1)$.

2.7.1.8 Lista Tabú

Es una lista creada para guardar una memoria de los movimientos que no se permiten (movimientos tabú) en la actual iteración. Esto con el fin de evitar movimientos que regresen a un punto de alguna iteración anterior y de esta manera producir ciclos en la búsqueda. El buen manejo de la lista tabú nos lleva a explorar nuevas regiones.

La lista tabú puede contener:

- Soluciones visitadas recientemente
- Movimientos realizados recientemente o
- Atributos o características que tenían las soluciones visitadas.

Un movimiento permanece como tabú solo durante un cierto número de iteraciones. Pasadas estas iteraciones, la búsqueda puede estar en una región diferente y los movimientos guardados en la lista se pueden liberar del status tabú.

2.7.1.9 Criterio de Aspiración

El criterio de aspiración es un elemento indispensable al aplicar el algoritmo de búsqueda tabú ya que permite omitir las restricciones tabú, con el fin de eliminar la clasificación tabú a un movimiento, cuando la solución de éste es mejor que la solución obtenida hasta el momento.[17]

- Aspiración por Default: Si todos los movimientos posibles son clasificados como tabú, entonces se selecciona el movimiento “menos tabú”.
- Aspiración por Objetivo: Una aspiración de movimiento se satisface, permitiendo que un movimiento x sea un candidato para seleccionarse si, por ejemplo.

[17] Melián, B., Glover, F. Introducción a la Búsqueda Tabú.

$F(x)$ < mejor costo. (En un problema de minimización)

- Aspiración por Dirección de Búsqueda: Un atributo de aspiración para la solución “s” se satisface si la dirección en “s” proporciona un mejoramiento y el actual movimiento es un movimiento de mejora. Entonces “s” se considera un candidato.

2.7.1.10 Estrategia de Intensificación

La estrategia de intensificación tiene como objetivo explorar más detalladamente una zona determinada del espacio de soluciones con la posibilidad de encontrar en este conjunto una solución o un grupo de soluciones mejores que las conocidas.

Para implementar esta técnica existen varios procedimientos. Una manera sencilla de hacerlo es disminuir el tamaño de la lista tabú, de manera que se permitan más movimientos de retroceso con el fin de explorar más detalladamente una determinada zona. Llevar a cabo este procedimiento puede generar ciclos por lo que hay que tener cierto cuidado al hacerlo. Otra estrategia planteada puede ser la de tomar un conjunto de buenas soluciones encontradas usando el algoritmo básico y repetir el algoritmo de búsqueda, tomando con solución inicial cada una de estas buenas soluciones.

2.7.1.11 Estrategia de Diversificación

Cuando una solución no se puede mejorar después de un determinado número de iteraciones es conveniente emplear el criterio de diversificación con el fin de generar soluciones que agreguen atributos o características significativamente diferentes a los encontrados en soluciones anteriores.

2.7.1.12 Criterio de Parada

El criterio de parada se establece al inicio del problema y se pueden usar los siguientes supuestos:

- Establecer un número fijo de iteraciones sin que se mejore la última solución encontrada.
- Estipular un tiempo límite de procesamiento del algoritmo.
- Fijar un número determinado de iteraciones.

2.8 Pasos para la implementación de TS (Forma General)

A continuación se presentan los pasos más relevantes en la implementación del algoritmo de TS[18]:

- **Paso 1**

  Generar una solución inicial, puede ser de manera aleatoria o a través de algoritmos heurísticos. Generar una solución de manera aleatoria puede traer grandes desventajas debido a que la solución encontrada puede ser de mala calidad; en el presente algoritmo las soluciones iniciales se generaron con las heurísticas "Vecino más Cercano" y "Clarke and Wright".
- **Paso 2**

  Elegir el entorno o vecindario de la solución inicial, con el fin de generar nuevas soluciones a partir de esta. Para generar el vecindario se aplican técnicas como: Inserción, Intercambio, etc…

- **Paso 3**

  Evaluar la función objetivo. Si la solución es factible y mejor que la anterior (criterio de aspiración) se toma como nueva solución, si no se pasa al paso 4.
- **Paso 4**

  Se revisa que la nueva solución no esté restringida por un movimiento tabú, y se toma como nueva solución, si está restringida se toma la siguiente mejor solución factible y no restringida por la lista tabú.

[18] Pertuz Alfredo José y Rojas Kimberly, 2007. "Formulación y evaluación de un algoritmo, basado en la meta-heurística búsqueda tabú para la optimización del ruteo de vehículos con capacidad 2007". tesis de pregrado.

- **Paso 5**

  Se actualiza la lista tabú reemplazando el valor más antiguo de la lista por el nuevo movimiento tabú.

- **Paso 6**

  Se ejecuta el algoritmo hasta que se cumpla el criterio de parada. Este criterio puede ser definido por la cantidad de iteraciones, la cantidad de iteraciones sin mejora, el tiempo de ejecución entre otros.

3 EJEMPLO DE SOLUCIÓN USANDO LA METAHEURÍSTICA BÚSQUEDA TABÚ

3.1 Soluciones Iníciales

A continuación se desarrolla el problema E016_3m como ejemplo ilustrativo de la creación de las soluciones iníciales, Vecino más Cercano" y "Clarke and Wright" utilizadas para los algoritmos de búsqueda tabú y recocido simulado.

Datos del Problema E016_3m

Coordenadas de los nodos

Ilustración 11. Coordenadas nodos del problema E016_3m.

NODO	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
X	37	49	52	20	40	21	17	31	52	51	42	31	5	12	36	30
Y	52	49	64	26	30	47	63	62	33	21	41	32	25	42	16	40

Fuente Librerías de Christophides and Eilon[19].

Vector Demanda

Ilustración 12. Vector demanda.

NODO	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
DEMANDA	7	30	16	9	21	15	19	23	11	5	19	29	23	21	10	0

Fuente Librerías de Christophides and Eilon.

Cantidad de vehículos

$$k = 3$$

Capacidad de los vehículos

$$Q = 90$$

[19] Librerías de Christophides and Eilon, disponible en http://www.or.deis.unibo.it/research_pages/ORinstances/VRPLIB/VRPLIB.html

Matriz de Costos

El problema E016_3m es un problema simétrico para lo cual se usó el método euclidiano para calcular la matriz de costos.

Ilustración 13. Matriz de costos ejemplo..

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	M	12,37	19,21	31,06	22,20	16,76	22,83	11,66	24,21	34,01	12,08	20,88	41,87	26,93	36,01	13,89
2	12,37	M	15,30	37,01	21,02	28,07	34,93	22,20	16,28	28,07	10,63	24,76	50,12	37,66	35,47	21,02
3	19,21	15,30	M	49,68	36,06	35,36	35,01	21,10	31,00	43,01	25,08	38,28	61,07	45,65	50,60	32,56
4	31,06	37,01	49,68	M	20,40	21,02	37,12	37,64	32,76	31,40	26,63	12,53	15,03	17,89	18,87	17,20
5	22,20	21,02	36,06	20,40	M	25,50	40,22	33,24	12,37	14,21	11,18	9,22	35,36	30,46	14,56	14,14
6	16,76	28,07	35,36	21,02	25,50	M	16,49	18,03	34,01	39,70	21,84	18,03	27,20	10,30	34,44	11,40
7	22,83	34,93	35,01	37,12	40,22	16,49	M	14,04	46,10	54,04	33,30	34,01	39,85	21,59	50,70	26,42
8	11,66	22,20	21,10	37,64	33,24	18,03	14,04	M	35,81	45,62	23,71	30,00	45,22	27,59	46,27	22,02
9	24,21	16,28	31,00	32,76	12,37	34,01	46,10	35,81	M	12,04	12,81	21,02	47,68	41,00	23,35	23,09
10	34,01	28,07	43,01	31,40	14,21	39,70	54,04	45,62	12,04	M	21,93	22,83	46,17	44,29	15,81	28,32
11	12,08	10,63	25,08	26,63	11,18	21,84	33,30	23,71	12,81	21,93	M	14,21	40,31	30,02	25,71	12,04
12	20,88	24,76	38,28	12,53	9,22	18,03	34,01	30,00	21,02	22,83	14,21	M	26,93	21,47	16,76	8,06
13	41,87	50,12	61,07	15,03	35,36	27,20	39,85	45,22	47,68	46,17	40,31	26,93	M	18,38	32,28	29,15
14	26,93	37,66	45,65	17,89	30,46	10,30	21,59	27,59	41,00	44,29	30,02	21,47	18,38	M	35,38	18,11
15	36,01	35,47	50,60	18,87	14,56	34,44	50,70	46,27	23,35	15,81	25,71	16,76	32,28	35,38	M	24,74
16	13,89	21,02	32,56	17,20	14,14	11,40	26,42	22,02	23,09	28,32	12,04	8,06	29,15	18,11	24,74	M

Fuente Autores

3.1.1 Método vecino más cercano

El depósito se iguala en la posición número 16, con demanda 0. El primer paso para empezar a resolver el problema, es generar una solución inicial a través de las heurísticas como "Vecino más Cercano" o "Clarke and Wright". A continuación se muestra el procedimiento usado por la heurística del vecino más cercano.

Para hallar la primera arista que se incluye en el tour se toma el vector de costos asociado al depósito como se muestra a continuación:

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
16	13,89	21,02	32,56	17,20	14,14	11,40	26,42	22,02	23,09	28,32	12,04	**8,06**	29,15	18,11	24,74	M

De la tabla se observa que el vecino más cercano al depósito es el nodo número 12, con un costo asociado de 8,06. Se debe verificar en el vector de demandas los nodos que se pueden atender sin sobrepasar la capacidad disponible del vehículo y de ellos escoger el nodo de menor costo. Si no hay ningún nodo que cumpla con el criterio, se debe retornar al depósito y asignar la carga del nodo de menor costo a un nuevo vehículo. Repetir el procedimiento hasta que se asignen todos los nodos a algún vehículo. A medida que se asignen los nodos se va creando la ruta y asociando su costo total.

DEMANDA															
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
7	30	16	9	21	15	19	23	11	5	19	**29**	23	21	10	0

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	29
CAPACIDAD DISPONIBLE	61
ASIGNACIÓN ACTUAL	29

RUTA	16	12
COSTO	0	8,06
COSTO TOTAL	8,06	

Asignadas las demandas al primer vehículo y sus costos procedemos a continuar buscando el vecino más cercano al nodo en el cual nos encontramos (nodo 12).

COSTO															
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
12	20,88	24,76	38,28	12,53	**9,22**	18,03	34,01	30,00	21,02	22,83	14,21	**M**	26,93	21,47	16,76	M

El nodo 5 es el más cercano al 12, por lo que se vuelven a realizar los pasos anteriores.

DEMANDA															
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
7	30	16	9	**21**	15	19	23	11	5	19	**M**	23	21	10	0

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	50
CAPACIDAD DISPONIBLE	40
ASIGNACIÓN ACTUAL	21

RUTA	16	12	5
COSTO	0	8,06	9,22
COSTO TOTAL	**17,28**		

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
5	22,2	21,02	36,06	20,4	M	25,5	40,22	33,24	12,37	14,21	**11,18**	M	35,36	30,46	14,56	14,14

DEMANDA															
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
7	30	16	9	M	15	19	23	11	5	**19**	M	23	21	10	0

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	69
CAPACIDAD DISPONIBLE	21
ASIGNACIÓN ACTUAL	19

RUTA	16	12	5	11
COSTO	0	8,06	9,22	11,18
COSTO TOTAL	**28,46**			

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
11	**12,08**	M	25,08	26,63	M	21,84	33,3	M	12,81	21,93	M	M	M	30,02	25,71	12,04

DEMANDA															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
7	30	16	9	M	15	19	23	11	5	M	M	23	21	10	0

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	76
CAPACIDAD DISPONIBLE	14
ASIGNACIÓN ACTUAL	7

RUTA	16	12	5	11	1
COSTO	0	8,06	9,22	11,18	12,08
COSTO TOTAL	**40,54**				

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	M	M	M	31,06	M	M	M	M	**24,21**	34,01	M	M	M	M	36,01	13,89

DEMANDA															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
M	30	16	9	M	15	19	23	**11**	5	M	M	23	21	10	0

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	87
CAPACIDAD DISPONIBLE	3
ASIGNACIÓN ACTUAL	11

RUTA	16	12	5	11	1	9
COSTO	0	8,06	9,22	11,18	12,08	24,21
COSTO TOTAL	**64,75**					

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
9	M	M	M	M	M	M	M	M	M	M	M	M	M	M	M	23,09

DEMANDA																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	30	16	9	M	15	19	23	M	5	M	M	23	21	10	0	

RUTA	16	12	5	11	1	9	16
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09
COSTO TOTAL	87,84						

Como se puede observar el vehículo número 1 se encuentra con una capacidad disponible de 3, por lo tanto no se puede atender a ningún cliente; La demanda mínima de los clientes que faltan por atender es de 9. A continuación se devuelve al vehículo número 1 al depósito y se procede a iniciar una nueva ruta usando el vehículo número 2.

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
16	M	21,02	32,56	17,20	M	**11,40**	26,42	22,02	M	28,32	M	M	29,15	18,11	24,74	M

DEMANDA																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	30	16	9	M	**15**	19	23	M	5	M	M	23	21	10	0	

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	15
CAPACIDAD DISPONIBLE	75
ASIGNACIÓN ACTUAL	15

RUTA	16	12	5	11	1	9	16	6
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40
COSTO TOTAL	99,24							

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
6	M	28,07	35,36	21,02	M	M	16,49	18,03	M	39,70	M	M	27,20	**10,30**	34,44	11,40

DEMANDA															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
M	30	16	9	M	M	19	23	M	5	M	M	23	**21**	10	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	36
CAPACIDAD DISPONIBLE	54
ASIGNACIÓN ACTUAL	21

RUTA	16	12	5	11	1	9	16	6	14
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30
COSTO TOTAL	109,54								

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
14	M	37,66	45,65	**17,89**	M	M	21,59	27,59	M	44,29	M	M	18,38	M	35,38	18,11

DEMANDA															
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
M	30	16	**9**	M	M	19	23	M	5	M	M	23	M	10	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	45
CAPACIDAD DISPONIBLE	45
ASIGNACIÓN ACTUAL	9

RUTA	16	12	5	11	1	9	16	6	14	4
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89
COSTO TOTAL	**127,43**									

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
4	M	37,01	49,68	M	M	M	37,12	37,64	M	31,40	M	M	**15,03**	M	18,87	17,20

DEMANDA																
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
M	30	16	M	M	M	19	23	M	5	M	M	**23**	M	10	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	68
CAPACIDAD DISPONIBLE	22
ASIGNACIÓN ACTUAL	23

RUTA	16	12	5	11	1	9	16	6	14	4	13
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03
COSTO TOTAL	**142,46**										

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
13	M	M	61,07	M	M	M	39,85	M	M	46,17	M	M	M	M	**32,28**	29,15

DEMANDA																
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
M	30	16	M	M	M	19	23	M	5	M	M	M	M	**10**	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	78
CAPACIDAD DISPONIBLE	12
ASIGNACIÓN ACTUAL	10

RUTA	16	12	5	11	1	9	16	6	14	4	13	15
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03	32,28
COSTO TOTAL	**174,74**											

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
15	M	M	M	M	M	M	M	M	M	**15,81**	M	M	M	M	M	24,74

DEMANDA																
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
M	30	16	M	M	M	19	23	M	**5**	M	M	M	M	M	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	83
CAPACIDAD DISPONIBLE	7
ASIGNACIÓN ACTUAL	5

RUTA	16	12	5	11	1	9	16	6	14	4	13	15	10	16
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03	32,28	15,81	28,32
COSTO TOTAL	**218,87**													

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
16	M	**21,02**	32,56	M	M	M	26,42	22,02	M	M	M	M	M	M	M	M

DEMANDA																
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
M	30	16	M	M	M	19	23	M	M	M	M	M	M	M	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	83
CAPACIDAD DISPONIBLE	7
ASIGNACIÓN ACTUAL	5

RUTA	16	12	5	11	1	9	16	6	14	4	13	15	10	16
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03	32,28	15,81	28,32
RUTA	2													
COSTO	21,02													
COSTO TOTAL	**239,89**													

COSTO																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
2	M	M	**15,30**	M	M	M	34,93	22,20	M	M	M	M	M	M	M	21,02

DEMANDA																
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
M	M	16	M	M	M	19	23	M	M	M	M	M	M	M	0

CAPACIDAD DEL VEHÍCULO 3	90
CARGA ASIGNADA TOTAL	46
CAPACIDAD DISPONIBLE	44
ASIGNACIÓN ACTUAL	16

RUTA	16	12	5	11	1	9	16	6	14	4	13	15	10	16
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03	32,28	15,81	28,32
RUTA	2	3												
COSTO	21,02	15,30												
COSTO TOTAL	**255,19**													

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	M	M	M	M	M	M	35,01	**21,10**	M	M	M	M	M	M	M	32,56

DEMANDA																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	

CAPACIDAD DEL VEHÍCULO 3	90
CARGA ASIGNADA TOTAL	46
CAPACIDAD DISPONIBLE	44
ASIGNACIÓN ACTUAL	16

RUTA	16	12	5	11	1	9	16	6	14	4	13	15	10	16
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03	32,28	15,81	28,32
RUTA	2	3	8											
COSTO	21,02	15,30	21,10											
COSTO TOTAL	**276,29**													

COSTO																
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
8	M	M	M	M	M	M	**14,04**	M	M	M	M	M	M	M	M	22,02

DEMANDA																
1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	M	M	M	M	M	**19**	M	M	M	M	M	M	M	M	0	

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	87
CAPACIDAD DISPONIBLE	3
ASIGNACIÓN ACTUAL	11

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	83
CAPACIDAD DISPONIBLE	7
ASIGNACIÓN ACTUAL	5

CAPACIDAD DEL VEHÍCULO 3	90
CARGA ASIGNADA TOTAL	46
CAPACIDAD DISPONIBLE	44
ASIGNACIÓN ACTUAL	16

RUTA	16	12	5	11	1	9	16	6	14	4	13	15	10	16
COSTO	0	8,06	9,22	11,18	12,08	24,21	23,09	11,40	10,30	17,89	15,03	32,28	15,81	28,32
RUTA	2	3	8	7	16									
COSTO	21,02	15,30	21,10	14,04	26,42									
COSTO TOTAL	**316,75**													

3.1.2 Solución Inicial Usando la Heurística "Clarke and Wrigth" (Algoritmo de Ahorros)

A continuación se presenta la solución del problema anterior (E016_3m) usando el algoritmo de ahorros.

1. El primer paso es calcular la matriz de ahorros usando la siguiente fórmula:

$$S_{i,j} = c_{i,0} + c_{0,j} - c_{i,j}$$

Donde

- $S_{i,j}$ **=** es el ahorro que se obtiene al unir los clientes (i, j) en una sola ruta.
- $c_{i,0}$ **=** Costo de ir del cliente i al depósito.
- $c_{0,j}$ **=** Costo de ir del cliente j al depósito.

- $c_{i,j}$ = Costo de ir del cliente i al cliente j.

2. El siguiente paso que realiza el algoritmo es establecer como punto inicial el depósito.
3. Usando la matriz de ahorros se elige cual es la combinación de nodos que representa el mayor ahorro y se toma esta combinación agregándola a la ruta. En caso de que el vehículo no pueda atender a un cliente porque se viola la restricción de capacidad se escoge la combinación subsiguiente. Si no hay ningún nodo que cumpla con la restricción de capacidad se termina la ruta y se inicia con un nuevo vehículo.

A continuación se muestra la matriz de ahorros:

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0,00	22,55	27,24	0,03	5,83	8,53	17,49	24,25	12,77	8,20	13,85	1,07	1,18	5,08	2,62	0
2	22,55	0,00	38,28	1,21	14,14	4,35	12,51	20,84	27,83	21,27	22,44	4,33	0,06	1,48	10,29	0
3	27,24	38,28	0,00	0,08	10,64	8,60	23,96	33,49	24,64	17,87	19,52	2,34	0,64	5,02	6,70	0
4	0,03	1,21	0,08	0,00	10,95	7,58	6,50	1,58	7,53	14,12	2,62	12,74	31,33	17,43	23,08	0
5	5,83	14,14	10,64	10,95	0,00	0,05	0,34	2,92	24,86	28,25	15,00	12,98	7,94	1,79	24,32	0
6	8,53	4,35	8,60	7,58	0,05	0,00	21,33	15,40	0,47	0,02	1,60	1,44	13,35	19,22	1,70	0
7	17,49	12,51	23,96	6,50	0,34	21,33	0,00	34,41	3,41	0,70	5,16	0,47	15,72	22,94	0,46	0
8	24,25	20,84	33,49	1,58	2,92	15,40	34,41	0,00	9,30	4,72	10,36	0,08	5,96	12,55	0,49	0
9	12,77	27,83	24,64	7,53	24,86	0,47	3,41	9,30	0,00	**39,36**	22,32	10,13	4,57	0,20	24,48	0
10	8,20	21,27	17,87	14,12	28,25	0,02	0,70	4,72	**39,36**	0,00	18,43	13,56	11,30	2,14	37,25	0
11	13,85	22,44	19,52	2,62	15,00	1,60	5,16	10,36	22,32	18,43	0,00	5,89	0,89	0,14	11,07	0
12	1,07	4,33	2,34	12,74	12,98	1,44	0,47	0,08	10,13	13,56	5,89	0,00	10,29	4,70	16,04	0
13	1,18	0,06	0,64	31,33	7,94	13,35	15,72	5,96	4,57	11,30	0,89	10,29	0,00	28,88	21,61	0
14	5,08	1,48	5,02	17,43	1,79	19,22	22,94	12,55	0,20	2,14	0,14	4,70	28,88	0,00	7,47	0
15	2,62	10,29	6,70	23,08	24,32	1,70	0,46	0,49	24,48	37,25	11,07	16,04	21,61	7,47	0,00	0
16	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0

Como se puede observar el mayor ahorro se puede encontrar con la combinación de 9-10 ó 10-9, por lo que seleccionamos para crear la ruta la combinación 10-9.

RUTA	16	10	9
COSTO	0	28,32	12,04
COSTO TOTAL	**40,36**		

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	16
CAPACIDAD DISPONIBLE	74
ASIGNACIÓN ACTUAL	16

4. Con el fin de no volver a escoger un nodo ya asignado en una ruta, se deben penalizar los valores correspondientes a ese nodo en la matriz de ahorros asignándole un valor mínimo (0), como se muestra en la tabla No. 8.

Se deben repetir los pasos 3 y 4 hasta asignar todos los nodos.

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
1	0,00	22,55	27,24	0,03	5,83	8,53	17,49	24,25	12,77	0	13,85	1,07	1,18	5,08	2,62	0
2	22,55	0,00	38,28	1,21	14,14	4,35	12,51	20,84	27,83	0	22,44	4,33	0,06	1,48	10,29	0
3	27,24	38,28	0,00	0,08	10,64	8,60	23,96	33,49	24,64	0	19,52	2,34	0,64	5,02	6,70	0
4	0,03	1,21	0,08	0,00	10,95	7,58	6,50	1,58	7,53	0	2,62	12,74	31,33	17,43	23,08	0
5	5,83	14,14	10,64	10,95	0,00	0,05	0,34	2,92	24,86	0	15,00	12,98	7,94	1,79	24,32	0
6	8,53	4,35	8,60	7,58	0,05	0,00	21,33	15,40	0,47	0	1,60	1,44	13,35	19,22	1,70	0
7	17,49	12,51	23,96	6,50	0,34	21,33	0,00	34,41	3,41	0	5,16	0,47	15,72	22,94	0,46	0
8	24,25	20,84	33,49	1,58	2,92	15,40	34,41	0,00	9,30	0	10,36	0,08	5,96	12,55	0,49	0
9	12,77	27,83	24,64	7,53	24,86	0,47	3,41	9,30	0,00	0	22,32	10,13	4,57	0,20	24,48	0
10	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
11	13,85	22,44	19,52	2,62	15,00	1,60	5,16	10,36	22,32	0	0,00	5,89	0,89	0,14	11,07	0
12	1,07	4,33	2,34	12,74	12,98	1,44	0,47	0,08	10,13	0	5,89	0,00	10,29	4,70	16,04	0
13	1,18	0,06	0,64	31,33	7,94	13,35	15,72	5,96	4,57	0	0,89	10,29	0,00	28,88	21,61	0
14	5,08	1,48	5,02	17,43	1,79	19,22	22,94	12,55	0,20	0	0,14	4,70	28,88	0,00	7,47	0
15	2,62	10,29	6,70	23,08	24,32	1,70	0,46	0,49	24,48	0	11,07	16,04	21,61	7,47	0,00	0
16	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
9	12,77	27,83	24,64	7,53	24,86	0,47	3,41	9,30	0,00	0,00	22,32	10,13	4,57	0,20	24,48	0

RUTA	16	10	9	2
COSTO	0	28,32	12,04	16,28
COSTO TOTAL	50,64			

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	46
CAPACIDAD DISPONIBLE	44
ASIGNACIÓN ACTUAL	30

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
2	22,55	0,00	38,28	1,21	14,14	4,35	12,51	20,84	0,00	0,00	22,44	4,33	0,06	1,48	10,29	0

RUTA	16	10	9	2	3
COSTO	0	28,32	12,04	16,28	15,30
COSTO TOTAL	71,94				

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	62
CAPACIDAD DISPONIBLE	28
ASIGNACIÓN ACTUAL	16

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
3	27,24	0,00	0,00	0,08	10,64	8,60	23,96	33,49	0,00	0,00	19,52	2,34	0,64	5,02	6,70	0

RUTA	16	10	9	2	3	8	16
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02
COSTO TOTAL	115,05						

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	85
CAPACIDAD DISPONIBLE	5
ASIGNACIÓN ACTUAL	23

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
8	24,25	0,00	0,00	1,58	2,92	15,40	34,41	0,00	0,00	0,00	10,36	0,08	5,96	12,55	0,49	0

RUTA	16	10	9	2	3	8	16
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02
COSTO TOTAL	**115,05**						

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	85
CAPACIDAD DISPONIBLE	5
ASIGNACIÓN ACTUAL	23

En este paso no hay ningún nodo que cumpla con la restricción de capacidad por tanto se finaliza la primera ruta y se escoge otro vehículo para continuar creando una nueva ruta. A continuación se muestra la nueva matriz de ahorros en la que se debe escoger la pareja que genere el mejor ahorro.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0,00	0,00	0,00	0,03	5,83	8,53	17,49	0,00	12,77	0,00	13,85	1,07	1,18	5,08	2,62	0
2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
3	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
4	0,03	0,00	0,00	0,00	10,95	7,58	6,50	0,00	7,53	0,00	2,62	12,74	31,33	17,43	23,08	0
5	5,83	0,00	0,00	10,95	0,00	0,05	0,34	0,00	24,86	0,00	15,00	12,98	7,94	1,79	24,32	0
6	8,53	0,00	0,00	7,58	0,05	0,00	21,33	0,00	0,47	0,00	1,60	1,44	13,35	19,22	1,70	0
7	17,49	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
8	24,25	0,00	0,00	1,58	2,92	15,40	34,41	0,00	9,30	0,00	10,36	0,08	5,96	12,55	0,49	0
9	12,77	0,00	0,00	7,53	24,86	0,47	3,41	0,00	0,00	0,00	22,32	10,13	4,57	0,20	24,48	0
10	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
11	13,85	0,00	0,00	2,62	15,00	1,60	5,16	0,00	22,32	0,00	0,00	5,89	0,89	0,14	11,07	0
12	1,07	0,00	0,00	12,74	12,98	1,44	0,47	0,00	10,13	0,00	5,89	0,00	10,29	4,70	16,04	0
13	1,18	0,00	0,00	31,33	7,94	13,35	15,72	0,00	4,57	0,00	0,89	10,29	0,00	28,88	21,61	0
14	5,08	0,00	0,00	17,43	1,79	19,22	22,94	0,00	0,20	0,00	0,14	4,70	28,88	0,00	7,47	0
15	2,62	0,00	0,00	23,08	24,32	1,70	0,46	0,00	24,48	0,00	11,07	16,04	21,61	7,47	0,00	0
16	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0

Como se puede observar el mayor ahorro se puede encontrar con la combinación de 4-13 ó 13-4, por lo que seleccionamos para crear la ruta la combinación 13-4.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
13	1,18	0,00	0,00	31,33	7,94	13,35	15,72	0,00	4,57	0,00	0,89	10,29	0,00	28,88	21,61	0

RUTA	16	10	9	2	3	8	16	13	4
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03
COSTO TOTAL	159,24								

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	32
CAPACIDAD DISPONIBLE	58
ASIGNACIÓN ACTUAL	32

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
4	0,03	0,00	0,00	0,00	10,95	7,58	6,50	0,00	7,53	0,00	2,62	12,74	0,00	17,43	23,08	0

RUTA	16	10	9	2	3	8	16	13	4	15
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87
COSTO TOTAL	**178,11**									

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	42
CAPACIDAD DISPONIBLE	48
ASIGNACIÓN ACTUAL	10

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
15	2,62	0,00	0,00	0,00	24,32	1,70	0,46	0,00	24,48	0,00	11,07	16,04	0,00	7,47	0,00	0

RUTA	16	10	9	2	3	8	16	13	4	15	5
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87	14,56
COSTO TOTAL	**192,67**										

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	63
CAPACIDAD DISPONIBLE	27
ASIGNACIÓN ACTUAL	21

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
5	5,83	0,00	0,00	0,00	0,00	0,05	0,34	0,00	24,86	0,00	15,00	12,98	0,00	1,79	0,00	0

RUTA	16	10	9	2	3	8	16	13	4	15	5
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87	14,56
RUTA	11										
COSTO	11,18										
COSTO TOTAL	**203,85**										

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	82
CAPACIDAD DISPONIBLE	8
ASIGNACIÓN ACTUAL	19

	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**
11	13,85	0,00	0,00	0,00	0,00	1,60	5,16	0,00	22,32	0,00	0,00	5,89	0,00	0,14	0,00	0

RUTA	16	10	9	2	3	8	16	13	4	15	5
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87	14,56
RUTA	11	1	16								
COSTO	11,18	12,08	13,89								
COSTO TOTAL	**229,83**										

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	89
CAPACIDAD DISPONIBLE	1
ASIGNACIÓN ACTUAL	7

En este paso no hay ningún nodo que cumpla con la restricción de capacidad, se finaliza la segunda ruta y se escoge otro vehículo para continuar creando una nueva ruta. A continuación se muestra la nueva matriz de ahorros en la que se debe escoger la pareja que genere el mejor ahorro.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
1	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
2	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
3	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
4	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
5	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
6	0,00	0,00	0,00	0,00	0,00	0,00	21,33	0,00	0,00	0,00	0,00	1,44	0,00	19,22	0,00	0,00
7	0,00	0,00	0,00	0,00	0,00	21,33	0,00	0,00	0,00	0,00	0,00	0,00	0,00	22,94	0,00	0,00
8	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
9	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
10	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
11	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
12	0,00	0,00	0,00	0,00	0,00	1,44	0,47	0,00	0,00	0,00	0,00	0,00	0,00	4,70	0,00	0,00
13	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
14	0,00	0,00	0,00	0,00	0,00	19,22	22,94	0,00	0,00	0,00	0,00	4,70	0,00	0,00	0,00	0,00
15	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00
16	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00

Como se puede observar el mayor ahorro se puede encontrar con la combinación de 7-14 ó 14-7, por lo que seleccionamos para crear la ruta la combinación 14-7.

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
14	0,00	0,00	0,00	0,00	0,00	19,22	22,94	0,00	0,00	0,00	0,00	4,70	0,00	0,00	0,00	0,00

RUTA	16	10	9	2	3	8	16	13	4	15	5
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87	14,56
RUTA	11	1	16	14	7						
COSTO	11,18	12,08	13,89	18,11	21,59						
COSTO TOTAL	**269,53**										

CAPACIDAD DEL VEHÍCULO 3	90
CARGA ASIGNADA TOTAL	40
CAPACIDAD DISPONIBLE	50
ASIGNACIÓN ACTUAL	40

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
7	0,00	0,00	0,00	0,00	0,00	21,33	0,00	0,00	0,00	0,00	0,00	0,47	0,00	0,00	0,00	0,00

RUTA	16	10	9	2	3	8	16	13	4	15	5
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87	14,56
RUTA	11	1	16	14	7	6					
COSTO	11,18	12,08	13,89	18,11	21,59	16,49					
COSTO TOTAL	**286,02**										

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	85
CAPACIDAD DISPONIBLE	5
ASIGNACIÓN ACTUAL	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	89
CAPACIDAD DISPONIBLE	1
ASIGNACIÓN ACTUAL	0

CAPACIDAD DEL VEHÍCULO 3	90
CARGA ASIGNADA TOTAL	55
CAPACIDAD DISPONIBLE	35
ASIGNACIÓN ACTUAL	15

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
6	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	0,00	1,44	0,00	0,00	0,00	0,00

RUTA	16	10	9	2	3	8	16	13	4	15	5
COSTO	0	28,32	12,04	16,28	15,30	21,09	22,02	29,15	15,03	18,87	14,56
RUTA	11	1	16	14	7	6	12	16			
COSTO	11,18	12,08	13,89	18,11	21,59	16,49	18,03	8,06			
COSTO TOTAL	**312,11**										

CAPACIDAD DEL VEHÍCULO 1	90
CARGA ASIGNADA TOTAL	85
CAPACIDAD DISPONIBLE	5
ASIGNACIÓN ACTUAL	0

CAPACIDAD DEL VEHÍCULO 2	90
CARGA ASIGNADA TOTAL	89
CAPACIDAD DISPONIBLE	1
ASIGNACIÓN ACTUAL	0

CAPACIDAD DEL VEHÍCULO 3	90
CARGA ASIGNADA TOTAL	84
CAPACIDAD DISPONIBLE	6
ASIGNACIÓN ACTUAL	29

Anteriormente, usamos dos algoritmos para encontrar una solución inicial al problema E016_3m, estos algoritmos arrojaron los siguientes resultados:

ALGORITMO	RUTA	COSTOS
VECINO MÁS CERCANO	16-12-5-11-1-9-16-6-14-4-13-15-10-16-2-3-8-7-16	316,75
CLARKE AND WRIGHT (AHORROS)	16-10-9-2-3-8-16-13-4-15-5-11-1-16-14-7-6-12-16	312,11

Como podemos observar para el problema E016_3m, el algoritmo de ahorros arrojó el mejor resultado, por lo cual, usaremos está solución como solución inicial para continuar realizando los pasos del algoritmo de Búsqueda Tabú.

3.2 Estrategias de Vecindad

A continuación se presentan dos estrategias de vecindad con el fin de generar nuevas soluciones para la búsqueda tabú. Las estrategias usadas son inserción y Intercambio.

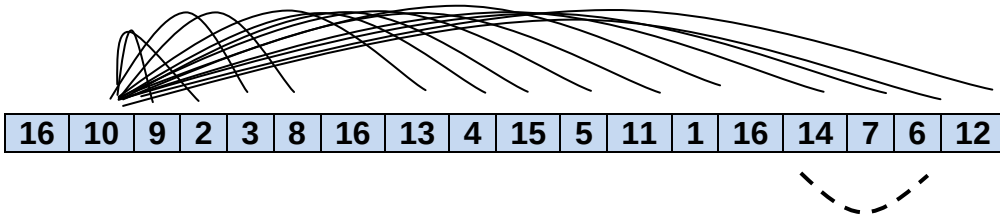
3.2.1 Inserción

Para aplicar la estrategia de inserción se hace lo siguiente:

1. Tomar la ruta inicial (Mejor solución encontrada después de aplicar el algoritmo de ahorros y del vecino más cercano) y se prueba insertando el primer cliente en cada una de las posiciones del vector excepto los depósitos, lo que nos da, una población de vecinos de n*(n-1).

16	10	9	2	3	8	16	13	4	15	5	11	1	16	14	7	6	12	16

Costo asociado: 312,11



2. Se realiza este procedimiento con cada uno de los clientes de la ruta hasta agotar todas las posibilidades. Luego de encontrar las nuevas rutas, se evalúa cuál ruta factible es la más económica.

En consecuencia para la ruta inicial, el mejor vecino está dado por la inserción del cliente 14 en la posición del cliente 6.

16	10	9	2	3	8	16	13	4	15	5	11	1	16	14	7	6	12	16

Por lo que la nueva ruta es:

B	10	9	2	3	8	16	13	4	15	5	11	1	16	7	6	14	12	16

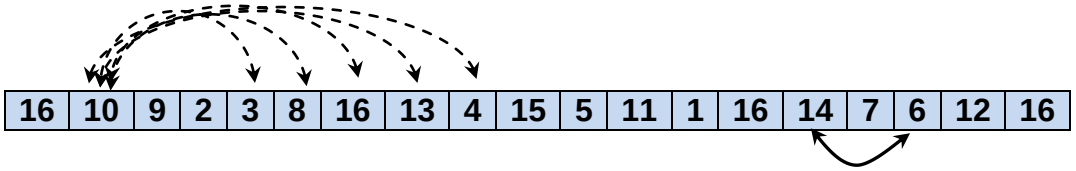
Costo Asociado: 312,567

3. La nueva ruta encontrada se toma como solución inicial y se realiza un número x de iteraciones hasta que se cumpla el criterio de parada.

3.2.2 Intercambio

Para aplicar la estrategia de Intercambio, se realizan los siguientes pasos:

1. Tomar la ruta inicial (Mejor solución encontrada después de aplicar el algoritmo de ahorros y del vecino más cercano) y se prueba intercambiando el primer cliente con cada una de la posiciones del vector excepto los depósitos.
2. Se realiza este procedimiento con cada uno de los clientes de la ruta hasta agotar todas las posibilidades. Luego de encontrar las nuevas rutas, se evalúa cual ruta factible es la más económica.



En consecuencia para la ruta inicial, el mejor vecino es el dado por el intercambio del cliente 14 y el cliente 6.

Por lo que la nueva ruta es:

16	10	9	2	3	8	16	13	4	15	5	11	1	16	6	7	14	12	16

Costo asociado: 308,84

3. La nueva ruta encontrada se toma como solución inicial y se realiza un número x de iteraciones hasta que se cumpla el criterio de parada.

3.3 Determinación de la Lista Tabú.

Para continuar con la solución del algoritmo es necesario definir el tamaño de la lista tabú y el tipo de datos que se van a guardar en ella. Algunos autores afirman

que la lista tabú no debería restringir más de la mitad de los movimientos posibles y mínimo 1/10 de dichos movimientos. Para el caso del problema E016_3m se definió un tamaño de lista tabú igual a 4. Los datos que se van a guardar en esta lista son los dos clientes que generaron el movimiento de vecindario.

Para el caso en el que se usa como estrategia de vecindario el método de inserción, la lista tabú guarda los clientes 14 y 6 durante mínimo 4 iteraciones.

Para el caso en el que se usa como estrategia de vecindario el método de intercambio, la lista tabú guarda los clientes 14 y 6 durante mínimo 4 iteraciones.

3.4 Criterio de Aspiración

Si un vecino evaluado presenta una solución factible cuyo costo sea menor que la mejor encontrada, se actualiza automáticamente la solución anterior reemplazándola por la nueva, sin necesidad de verificar la lista tabú.

Tomando la última solución encontrada con un costo asociado de 308,84 tenemos:

16	**10**	**9**	**2**	**3**	**8**	**16**	**13**	**4**	**15**	**5**	**11**	**1**	**16**	**6**	**7**	**14**	**12**	**16**

La lista tabú de tamaño cuatro está representada por:

Lista	Tabú
3	**11**
14	**6**
3	**4**
2	**6**

En la siguiente iteración se obtiene que el mejor vecino se encuentra al mover los clientes 3 y 11, que están restringidos por la lista y cumplen con la restricción de capacidad; el costo asociado de esta nueva solución es de 306.53 y es menor que la mejor solución encontrada hasta el momento, por lo cual, se aplica el criterio de aspiración y se toma la nueva solución.

16	10	9	2	11	8	16	13	4	15	5	3	1	16	6	7	14	12	16

3.5 Criterio de Parada

Para el presente algoritmo se estableció que el criterio de parada sería un número de iteraciones entre 20 y 400 iteraciones según elija el usuario de la herramienta.

3.6 Criterio de Intensificación

Se considera que esta última solución es una solución de buena calidad, por lo cual, se toma como solución inicial para aplicar el criterio de intensificación y explorar más detalladamente esta zona del espacio de soluciones. Para la aplicación de este criterio se deben llevar a cabo los siguientes pasos:

1. Se toma el vector solución y se divide en sub-vectores que representan la ruta de cada vehículo.

16	10	9	2	11	8	16	13	4	15	5	3	1	16	6	7	14	12	16

*Vector original, costo asociado de 306,53.

Vehículo 1:

16	10	9	2	11	8	16

Costo Asociado de 90,98

Vehículo 2:

16	13	4	15	5	3	1	16

Costo Asociado de 136,54

Vehículo 3:

16	6	7	14	12	16

Costo Asociado de 79,01

2. Se toma cada uno de los vectores y se aplica la estrategia de búsqueda exhaustiva, hasta encontrar la combinación de clientes que de la solución óptima para cada vector. Después de un número x de iteraciones se obtiene:

Vehículo 1:

16	10	9	2	8	11	16

Costo Asociado de 87,61

Vehículo 2:

16	4	15	13	5	3	1	16

Costo Asociado de 117,79

Vehículo 3:

16	6	7	14	12	16

Costo Asociado de 79,01

3. Se reagrupan nuevamente los vectores y se calcula el costo total de la ruta.

16	10	9	2	8	11	16	4	15	13	5	3	1	16	6	7	14	12	16

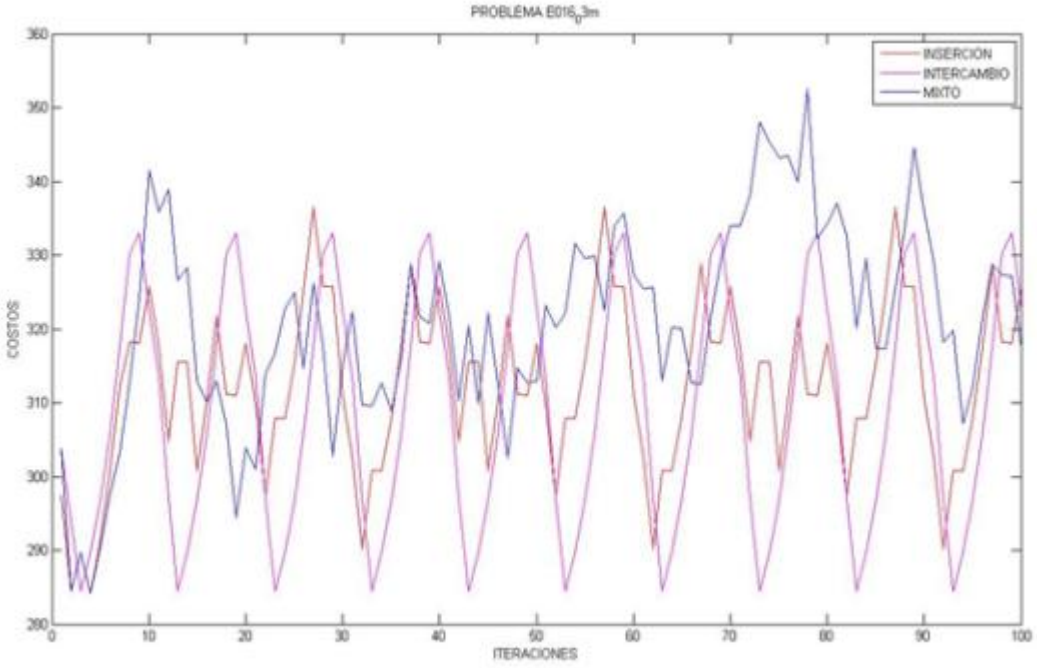
Costo Asociado de 284,41

Este último vector, es la solución encontrada después de aplicar el algoritmo de búsqueda tabú.

A continuación se muestra la mejora realizada por el algoritmo y su comparación con la solución encontrada en la literatura[20].

Problema	Heurística del Vecino más Cercano	Heurística de Clarke and Wright	Búsqueda Tabú antes de la intensificación	Búsqueda Tabú con intensificación	Literatura	Diferencia
E016_03m	316,74	331,31	306,53	284,41	262	8,55%

Ilustración 14. Comportamiento de los costos vs iteraciones..



Fuente autores

La ilustración anterior muestra la evolución de los costos vs la cantidad de iteraciones realizadas (100 iteraciones para este caso) usando las metodologías tabú inserción, tabú intercambio y tabú mixto.

20 Librerías de Christophides and Eilon, disponible en http://www.or.deis.unibo.it/research_pages/ORinstances/VRPLIB/VRPLIB.html

4 RECOCIDO SIMULADO (SIMULATED ANNEALING)

El método de recocido se utiliza muy comúnmente para mejorar las cualidades físicas de un material. El proceso básicamente consiste en calentar un material a una temperatura muy alta y de manera paulatina se hace descender la temperatura dejando que los átomos queden en equilibrio. Al finalizar el proceso, los átomos crean una estructura cristalina con características muy regulares alcanzando el material la máxima resistencia. De forma experimental se puede comprobar que si se hace descender la temperatura del material violentamente, al finalizar el proceso esta estructura no va a ser la más adecuada. Mediante una serie de experimentos se demostró que los átomos de un sistema en un proceso de recocido se comportan de acuerdo al factor de probabilidad de Boltzmann.[21]

En 1953 Metropolis[22] modeló el proceso de recocido mediante un algoritmo, el cual da a los átomos un desplazamiento al azar y mide el cambio de ΔE . Si $\Delta E \leq 0$ se acepta el desplazamiento. Si $\Delta E \geq 0$, se acepta el desplazamiento con probabilidad $e^{\left(\frac{-\Delta E}{K_B T}\right)}$, donde *T* es la temperatura y K_B es la constante de Boltzmann.

A principios de los ochenta, Kirkpatrick[23] y V. Cerny[24] dentro de sus investigaciones establecieron una analogía entre los parámetros que intervienen en la simulación termodinámica y los que aparecen en los métodos de optimización local. A continuación se muestra el resultado de esta analogía.

---

21 En física, el factor de Boltzmann es un factor de ponderación que determina la probabilidad relativa de un estado i en un sistema con múltiples estados en equilibrio termodinámico a temperatura T.

22 N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, Y. Teller, Equation of State Calculations by Fast Computing Machines, Journal of Chemical Physics 21, 1087 (1953).

23 DÍAZ, Adenso, GLOVER, Fred, GHARZIRI, Hassan, GONZALEZ, J, LAGUNA, Manuel, MOSCATO, Pablo. Optimización Heurística y Redes Neuronales. Editorial Parafino. 1996. P.41

24 Idem pag 41

Tabla 6. Analogía entre la termodinámica y la optimización

TERMODINÁMICA	OPTIMIZACIÓN
Configuración	Solución Factible
Configuración fundamental	Solución óptima
Energía de la configuración	Costo de la solución
Temperatura	Parámetro de control

Fuente: Autores

Tomando como base la anterior analogía los autores plantearon lo siguiente; Sea (S, c) una instancia de un problema de optimización combinatoria donde:

- S es el conjunto de soluciones factibles
- c es la función de costo (a valores reales positivos)
- T Parámetro de control (Temperatura)

El problema se reduce a hallar un estado i en S que minimice c . Para alcanzar este estado, el algoritmo de SA plantea los siguientes pasos:

1. Generar una solución inicial. Se puede usar una solución factible tomada de manera aleatoria o usando un algoritmo de tipo *Greedy.*
2. Hacer uso del parámetro Temperatura, pues este parámetro determinará la aceptación de soluciones vecinas peores que la actual.
3. Generar en cada iteración un número de vecinos, $V(T)$.
4. Aplicar el criterio de aceptación a estos vecinos, para ver si se puede sustituir la solución actual. Si c(j)<c(i), acepto automáticamente la nueva solución, de lo contrario paso 5
5. Aceptar la nueva solución aplicando la siguiente probabilidad:

$$PA = e^{(x)} donde\ x = -\frac{c(j) - c(i)}{T}$$

Esto significa que:

- A mayor temperatura, mayor es la probabilidad de aceptar soluciones peores.
- A menor diferencia entre los costes aumenta la probabilidad de aceptar soluciones peores.

6. Tras haber generado $V(T)$ soluciones vecinas, se debe enfriar la temperatura mediante una función o un criterio de reducción hasta que se alcanza una T_f.

Existen varios mecanismos de enfriamiento[25]:

- Enfriamiento basado en sucesivas temperaturas descendentes fijadas por el usuario.
- Enfriamiento con descenso constante de temperatura.
- Descenso exponencial: $T_{k+1} = \alpha * T_k$;, α
- Criterio de Boltzmann: $T_k = T_0/(1 + \log(k))$
- Esquema de Cauchy: $T_k = T_0/(1 + \mathrm{k})$
- Si se necesita controlar el número de iteraciones (Cauchy Modificado): $T_{k+1} = \frac{T_k}{1+\beta * T_k}; \beta = (T_0 - T_f)/(M * T_0 * T_f)$

Donde,

T_0, Es la temperatura inicial.

T_f, Es la temperatura final.

T_k , Es la temperatura al cabo de *k* iteraciones.

α , Es una constante cercana a 1 (usualmente $\alpha \epsilon [0.8; 0.99]$)

M , Número de iteraciones a las que se desea terminar.

[25] K.A. Dowsland, A. Díaz. Diseño de heurísticas y fundamentos del recocido simulado. Inteligencia Artificial VII:2 (2003) 93-101. http://sci2s.ugr.es/docencia/algoritmica/Tema03-EnfriamientoSimulado-09-10.pdf

7. Se lleva a cabo el criterio de parada que puede ser cuando:
    - $T = 0$
    - $T \leq T_f$
    - Después de un número fijo de iteraciones

5 EJEMPLO DE SOLUCIÓN USANDO LA METAHEURÍSTICA RECOCIDO SIMULADO

Para mostrar un ejemplo de aplicación del algoritmo de SA usamos el mismo problema usado en el algoritmo de TS (E016_3m). A continuación se presentan los pasos desarrollados en la aplicación del algoritmo.

5.1 Generar una Solución Inicial

Las heurísticas usadas para generar una solución inicial en el ejemplo de aplicación del algoritmo de TS, generan resultados relativamente eficientes, es por ello, que para el desarrollo del presente algoritmo usaremos las mismas heurísticas.

ALGORITMO	RUTA	COSTOS
VECINO MÁS CERCANO	16-12-5-11-1-9-16-6-14-4-13-15-10-16-2-3-8-7-16	316,75
CLARK AND WRIGHT (AHORROS)	16-10-9-2-3-8-16-13-4-15-5-11-1-16-14-7-6-12-16	312,11

5.2 Definición del Parámetro de Temperatura

Se definió el parámetro de temperatura usando el siguiente criterio:

$$T_0 = 10\mu$$

Donde,

μ Es el 2% del Costo de la Solución Inicial (error esperado).

Como solución inicial tomamos la mejor encontrada después de aplicar las heurísticas correspondientes, para este caso la heurística que arrojó la mejor solución fue la de Clarke and Wright con los resultados presentados a continuación:

ALGORITMO	RUTA	COSTOS
CLARKE AND WRIGHT (AHORROS)	16-10-9-2-3-8-16-13-4-15-5-11-1-16-14-7-6-12-16	312,11

Continuamos aplicando la fórmula para hallar la temperatura inicial.

$$\mu = 0{,}02 * 312{,}11 = 6{,}2422$$

$$T_0 = 10\mu = 10 * 6{,}2422 = 62{,}422$$

5.3 Generar en cada Iteración un Número de Vecinos

Para la generación de los vecinos de la nueva solución, se utilizan las mismas estrategias utilizadas en el algoritmo de búsqueda tabú (Inserción e Intercambio).

Al aplicar la heurística de inserción en la primera iteración nos da el siguiente resultado:

RUTA:

16	10	9	2	3	8	16	13	4	15	5	11	1	16	6	7	14	12	16

COSTO:

308,84

5.4 Aplicar el Criterio de Aceptación

Después de generar las soluciones vecinas, se evalúan estas soluciones y si el costo de la nueva solución es menor que el costo de la mejor solución encontrada acepto la nueva solución de lo contrario continuamos al siguiente paso.

Como la solución encontrada en el paso anterior es mejor que la mejor encontrada hasta el momento entonces se acepta la nueva solución y se buscan los nuevos

vecinos para continuar con el algoritmo. Después de una siguiente iteración se obtiene los siguientes resultados:

RUTA:

16	10	9	2	11	8	16	13	4	15	5	3	1	16	6	7	12	14	16

COSTO:

338,78

Como esta nueva solución es de menor calidad que la mejor solución actual, se continúa con el siguiente paso.

5.5 Aceptación Bajo una Función de Probabilidad

Si la solución encontrada es más costosa que la mejor solución inicial, podemos aceptar esta nueva solución aplicando el siguiente criterio:

$$PA = e^{(x)}\ donde\ x = -\frac{c(j) - c(i)}{T} = \frac{Error}{T}$$

Donde,

$c(j)$ Es el costo de la última solución.

$c(i)$ Es el costo de la mejor solución encontrada.

T Es la temperatura.

Después de realizar este cálculo se debe generar un número aleatorio entre 0 y 1, este número se compara con el dato obtenido de la probabilidad y si el número aleatorio es menor que PA se acepta la solución correspondiente a esta iteración.

A continuación se calcula la probabilidad de aceptación basándonos en los datos obtenidos en el último paso.

$$x = -\frac{303{,}82 - 297{,}33}{64{,}22} = -\frac{6{,}49}{64{,}22} = -0{,}101$$

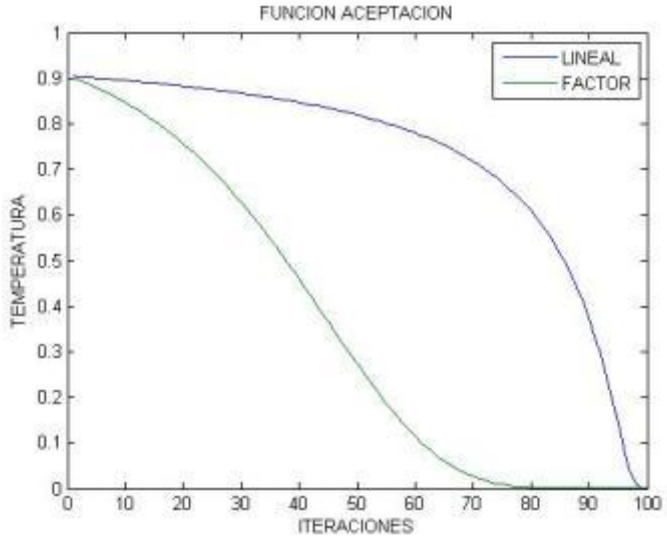
$$PA = e^{(x)} = e^{(-0{,}1)} = 0.9048$$

$$Número\ Aleatorio = 0{,}82$$

La solución es aceptada y se continúa con el siguiente paso. Si la solución no es aceptada se debe regresar al paso 5.3.

En la siguiente figura se muestra las funciones de probabilidad de aceptación que dependen de la estrategia de la razón de enfriamiento usada (Lineal o Factor).

Ilustración 15. Función de probabilidad de aceptación.



Fuente: autores

5.6 Secuencia de Enfriamiento

Una vez aceptada una solución del vecindario, se debe actualizar la temperatura. La actualización se realiza usando dos estrategias de enfriamiento. La primera consiste en hacer un decrecimiento de la temperatura de forma lineal, empezando en T_0 y disminuyendo la temperatura en cada iteración hasta llegar a un valor mínimo después de x iteraciones. El valor se calcula usando las siguientes formulas:

$$\Delta T = \frac{T_0}{102}$$

$$Ti = T_{i-1} - \Delta T$$

Donde,

T_i Es la Temperatura en la iteración i

La segunda estrategia consiste en disminuir T_i de manera exponencial usando la siguiente ecuación:

$$T_i = \propto T_{i-1}$$

Donde $\propto$ es una constante (Factor de Enfriamiento), usualmente $\propto \epsilon [0{,}8; 0{,}99]$. Para nuestro caso la constante $\propto$ fue calculada de la siguiente manera:

$$\propto^{\#iteraciones} = 3\%$$

$$\#de\ iteraciones * \ln(\propto) = \ln(0{,}03)$$

$$\ln(\propto) = -3{,}506/\#de\ iteraciones$$

$$\propto = e^{-3{,}506/\#de\ iteraciones}$$

Para 100 iteraciones $\propto = e^{-3{,}506/100} = 0{,}965$

A continuación se presenta el cálculo para el decrecimiento de la temperatura de forma lineal:

$$\Delta T = \frac{64{,}22}{102} = 0{,}629$$

$$T_1 = 64{,}22 - 0{,}629 = 63{,}59$$

Después de 100 iteraciones la temperatura es igual a:

$$T_{100} = 1{,}888 - 0{,}629 = 1{,}2592$$

Estos cálculos se deben realizar para 100 iteraciones.

Otra forma de llevar a cabo el enfriamiento de la temperatura es la siguiente:

$$T_i = \propto T_{i-1}$$

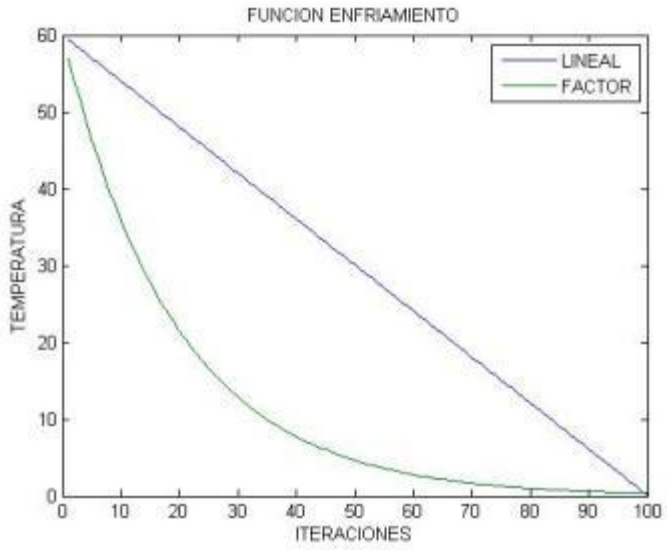
$$T_1 = 0{,}965 * T_0 = 0{,}965 * 64{,}22 = 61{,}972$$

Después de 100 iteraciones la temperatura es igual a:

$$T_{100} = 0{,}965 * T_{99} = 0{,}965 * 1{,}888 = 1{,}822$$

En la siguiente figura se muestran las funciones de decrecimientos para 100 iteraciones.

Ilustración 16. Función de Enfriamiento para el algoritmo de recocido simulado.



Fuente autores.

Por último se presenta la mejor solución encontrada por el método de recocido simulado.

RUTA:

16	**12**	**5**	**10**	**9**	**11**	**16**	**1**	**6**	**14**	**13**	**4**	**15**	**16**	**2**	**3**	**8**	**7**	**16**

COSTO:

284.2309

6 RESULTADOS OBTENIDOS POR LOS ALGORITMOS

A continuación se presentan las tablas que contiene de manera general, los resultados obtenidos después de ejecutar los algoritmos desarrollados para la aplicación de las metaheurísticas de Búsqueda Tabú y Recocido Simulado. Estos algoritmos fueron probados con 42 problemas encontrados en la bibliografía y que pertenecen a diferentes autores, de los cuales, se escogieron 10 problemas pues se consideran representativos para el diseño de experimentos. Los problemas se caracterizan por ser de tipo simétrico (coordenadas euclidianas).

La herramienta fue desarrollada en el software MATLAB®. versión R2009a y ejecutada en un computador HACER ASPIRE 4736Z, con un procesador Intel® Pentium® T4500 (2,3 GHz, 800 MHz FSB) y 4 GB de memoria RAM.

Para el algoritmo de búsqueda tabú se programaron dos estrategias de vecindad (Inserción e intercambio) con el fin de encontrar diversas soluciones a los problemas. Además se realizó una combinación de estas estrategias que se aplica de la siguiente manera:

- Si el número de la iteración en la que se encuentra el algoritmo es par, se aplica la estrategia tabú inserción para encontrar la siguiente solución.

- Si el número de la iteración en la que se encuentra el algoritmo es impar se aplica la estrategia tabú intercambio para encontrar la siguiente solución.

Esta forma de combinar las estrategias arroja mejores resultados que si se aplicaran las estrategias de forma independiente. Los problemas fueron evaluados con (50,100, 200 y 400 iteraciones) y con tamaños de lista tabú (4, 5, 6, 7, 8, 9,10)

A continuación se presentan los resultados arrojados por el algoritmo:

Tabla 7. Resultados obtenidos al aplicar la estrategia tabú inserción..

PROBLEMA	TABÚ INSERCIÓN						
	50 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.018,32	1.018,32	1.038,30	1.018,32	1.048,57	1.018,32	1.018,32
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	687,43	687,43	634,40	627,92	656,40	651,95	712,44
E_n33_k4	911,71	911,71	907,87	906,62	907,87	907,87	907,87
P_n50_k10	787,65	787,65	787,65	787,65	787,65	787,65	787,65
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	749,82	749,82	749,82	749,82	749,82	746,02	746,02
P_n60_k15	1.075,36	1.075,36	1.045,96	1.045,96	1.047,08	1.047,08	1.047,08
P_n101_k4	778,44	778,44	778,44	778,44	778,44	778,44	778,44
M_n200_k17	1.595,14	1.595,14	1.595,14	1.595,14	1.595,14	1.593,61	1.593,76
PROBLEMA	100 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.018,32	1.038,30	1.038,30	1.038,64	1.038,64	1.038,30	1.038,30
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	656,54	687,43	627,92	568,56	656,40	651,95	712,44
E_n33_k4	911,71	911,71	907,64	906,62	907,87	907,87	907,87
P_n50_k10	787,65	787,65	787,65	787,65	787,65	787,65	787,65
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	749,82	749,82	749,82	749,82	749,82	746,02	746,02
P_n60_k15	1.075,36	1.075,36	1.045,96	1.045,96	1.047,08	1.047,08	1.047,08
P_n101_k4	772,77	772,08	772,08	771,99	771,99	770,34	771,76
M_n200_k17	1.595,14	1.595,14	1.595,14	1.593,48	1.593,48	1.593,48	1.584,99
PROBLEMA	200 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.018,32	1.038,30	1.038,30	1.038,30	1.038,64	1.038,30	1.038,30
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	656,54	687,43	627,92	568,56	656,40	651,95	712,44
E_n33_k4	910,51	902,80	899,48	906,62	903,87	900,59	907,87
P_n50_k10	787,65	787,65	787,65	787,65	787,65	787,65	787,65
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	749,82	749,82	749,82	749,82	749,82	746,02	746,02
P_n60_k15	1.075,36	1.075,36	1.045,96	1.045,96	1.047,08	1.047,08	1.047,08
P_n101_k4	772,77	771,65	771,65	770,56	770,56	770,34	768,96
M_n200_k17	1.595,14	1.595,14	1.595,14	1.593,48	1.593,48	1.593,52	1.584,99
PROBLEMA	400 ITERACIONES						

	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.018,32	1.038,30	1.038,30	1.038,30	1.038,64	1.038,30	1.038,30
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	656,54	687,43	627,92	568,56	656,40	651,95	712,44
E_n33_k4	910,51	902,80	899,48	896,46	896,65	900,59	907,87
P_n50_k10	787,65	787,65	787,65	787,65	787,65	787,65	787,65
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	749,82	749,82	749,82	749,82	749,82	746,02	746,02
P_n60_k15	1.075,36	1.075,36	1.045,96	1.045,96	1.047,08	1.047,08	1.047,08
P_n101_k4	772,77	771,65	771,65	770,56	770,56	770,34	768,96
M_n200_k17	1.595,14	1.595,14	1.595,14	1.593,48	1.593,48	1.593,52	1.584,99

Fuente autores

La siguiente tabla muestra los problemas con los resultados encontrados en las librerías:

Tabla 8. Mejores resultados encontrados en librerías.

PROBLEMA	RESPUESTA LIBRERÍAS
A_n45_k6	**944**
E016_03m	**262**
E_n23_k3	**569**
E_n33_k4	**845**
P_n50_k10	**696**
P_n22_k2	**216**
P_n55_k10	**694**
P_n60_k15	**968**
P_n101_k4	**681**
M_n200_k17	**1373**

Fuente autores

Con los datos mostrados en la tabla anterior y los resultados obtenidos en cada uno de los algoritmos, se calculó el porcentaje de diferencia entre las mejores soluciones encontradas por los algoritmos y la respuesta encontrada en las librerías. Los resultados para la metodología tabú inserción se muestran en la siguiente tabla:

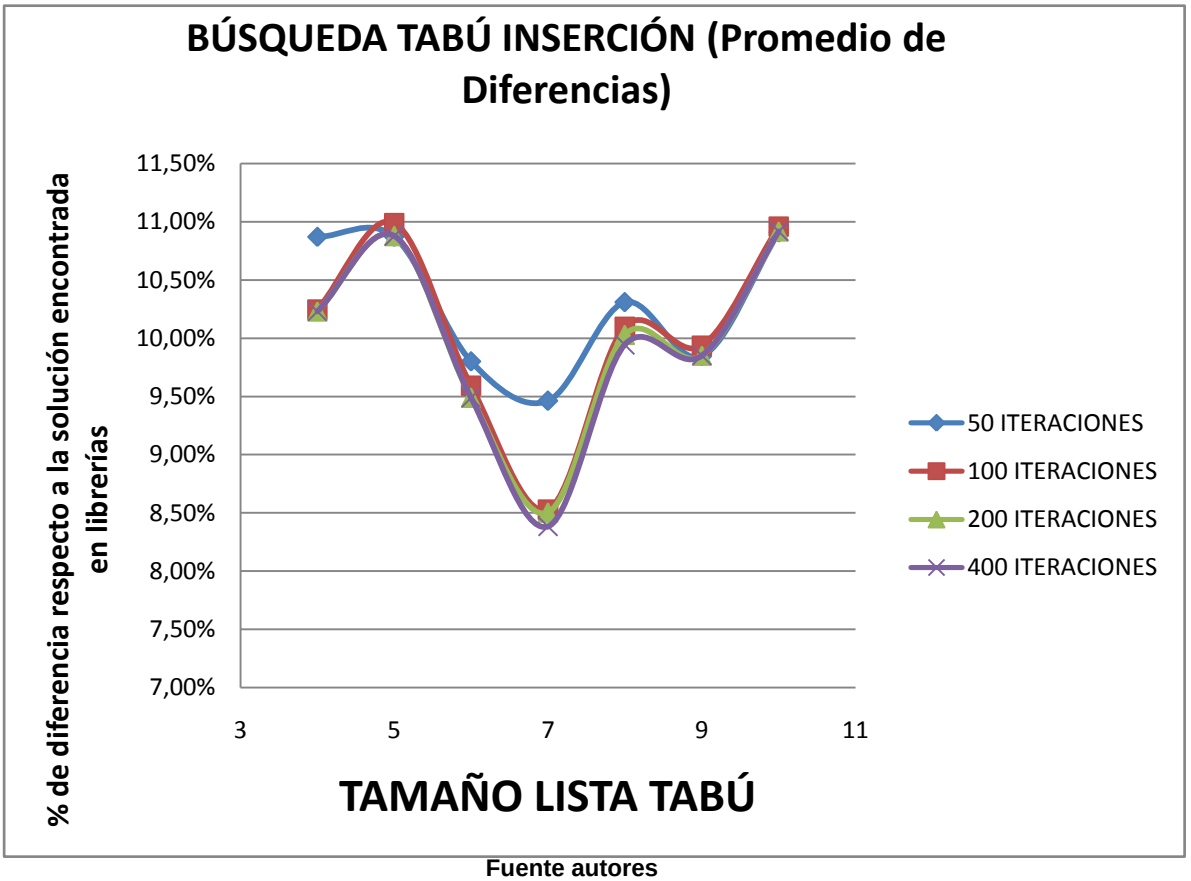
Tabla 9. Porcentajes de diferencia respecto a la solución encontrada en librerías (inserción).

PROBLEMA	TABÚ INSERCIÓN						
	50 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	7,87%	7,87%	9,99%	7,87%	11,08%	7,87%	7,87%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,48%
E_n23_k3	20,81%	20,81%	11,49%	10,35%	15,36%	14,58%	25,21%
E_n33_k4	7,89%	7,89%	7,44%	7,29%	7,44%	7,44%	7,44%
P_n50_k10	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	8,04%	8,04%	8,04%	8,04%	8,04%	7,50%	7,50%
P_n60_k15	11,09%	11,09%	8,05%	8,05%	8,17%	8,17%	8,17%
P_n101_k4	14,31%	14,31%	14,31%	14,31%	14,31%	14,31%	14,31%
M_n200_k17	16,18%	16,18%	16,18%	16,18%	16,18%	16,07%	16,08%
PROMEDIO	10,87%	10,87%	9,80%	9,46%	10,31%	9,84%	10,91%
PROBLEMA	100 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	7,87%	9,99%	9,99%	10,03%	10,03%	9,99%	9,99%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,48%
E_n23_k3	15,39%	20,81%	10,35%	-0,08%	15,36%	14,58%	25,21%
E_n33_k4	7,89%	7,89%	7,41%	7,29%	7,44%	7,44%	7,44%
P_n50_k10	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	8,04%	8,04%	8,04%	8,04%	8,04%	7,50%	7,50%
P_n60_k15	11,09%	11,09%	8,05%	8,05%	8,17%	8,17%	8,17%
P_n101_k4	13,48%	13,37%	13,37%	13,36%	13,36%	13,12%	13,33%
M_n200_k17	16,18%	16,18%	16,18%	16,06%	16,06%	16,06%	15,44%
PROMEDIO	10,25%	10,99%	9,59%	8,53%	10,10%	9,94%	10,96%
PROBLEMA	200 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	7,87%	9,99%	9,99%	9,99%	10,03%	9,99%	9,99%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,48%
E_n23_k3	15,39%	20,81%	10,35%	-0,08%	15,36%	14,58%	25,21%
E_n33_k4	7,75%	6,84%	6,45%	7,29%	6,97%	6,58%	7,44%
P_n50_k10	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	8,04%	8,04%	8,04%	8,04%	8,04%	7,50%	7,50%
P_n60_k15	11,09%	11,09%	8,05%	8,05%	8,17%	8,17%	8,17%
P_n101_k4	13,48%	13,31%	13,31%	13,15%	13,15%	13,12%	12,92%

M_n200_k17	16,18%	16,18%	16,18%	16,06%	16,06%	16,06%	15,44%
PROMEDIO	10,23%	10,88%	9,49%	8,50%	10,03%	9,85%	10,92%
PROBLEMA	400 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	7,87%	9,99%	9,99%	9,99%	10,03%	9,99%	9,99%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,48%
E_n23_k3	15,39%	20,81%	10,35%	-0,08%	15,36%	14,58%	25,21%
E_n33_k4	7,75%	6,84%	6,45%	6,09%	6,11%	6,58%	7,44%
P_n50_k10	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%	13,17%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	8,04%	8,04%	8,04%	8,04%	8,04%	7,50%	7,50%
P_n60_k15	11,09%	11,09%	8,05%	8,05%	8,17%	8,17%	8,17%
P_n101_k4	13,48%	13,31%	13,31%	13,15%	13,15%	13,12%	12,92%
M_n200_k17	16,18%	16,18%	16,18%	16,06%	16,06%	16,06%	15,44%
PROMEDIO	10,23%	10,88%	9,49%	8,38%	9,94%	9,85%	10,92%

Fuente autores.

Ilustración 17. Resultados obtenidos al aplicar la estrategia tabú inserción.



Fuente autores

De la ilustración anterior se puede decir que el promedio de las mejores soluciones encontradas usando la estrategia tabú inserción es cercano al 8,5% y se alcanza con una combinación de tamaño de la lista tabú 7 y 400 iteraciones.

Tabla 10. Resultados obtenidos al aplicar la estrategia tabú intercambio.

PROBLEMA	TABÚ INTERCAMBIO						
	50 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.104,77	1.104,77	1.104,77	1.104,77	1.104,77	1.076,64	1.104,77
E016_03m	284,42	284,42	284,42	284,42	284,42	284,42	284,42
E_n23_k3	676,80	676,80	676,80	650,20	676,80	668,10	676,80
E_n33_k4	981,19	981,19	959,27	958,36	958,36	958,36	958,36
P_n50_k10	750,54	738,58	738,58	739,11	739,11	739,11	746,06
P_n22_k2	249,58	245,28	249,58	249,58	249,58	243,46	249,58
P_n55_k10	750,16	748,68	748,68	750,16	750,16	750,16	750,16
P_n60_k15	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83
P_n101_k4	989,96	989,96	989,96	989,96	989,96	989,96	989,96
M_n200_k17	1.713,03	1.709,19	1.709,19	1.709,19	1.709,19	1.709,19	1.708,91
PROBLEMA	100 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.104,77	1.104,77	1.104,77	1.104,77	1.104,77	1.076,64	1.104,77
E016_03m	284,42	284,42	284,42	284,42	284,42	284,42	284,42
E_n23_k3	676,80	676,80	676,80	650,20	676,80	655,59	676,80
E_n33_k4	981,19	981,19	959,27	958,36	958,36	958,36	958,36
P_n50_k10	750,54	738,58	738,58	739,11	739,11	739,11	746,06
P_n22_k2	249,58	245,28	249,58	237,82	249,58	243,46	249,58
P_n55_k10	750,16	748,68	748,68	750,16	750,16	750,16	750,16
P_n60_k15	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83
P_n101_k4	989,96	989,96	989,96	989,96	989,96	989,96	989,96
M_n200_k17	1.713,03	1.709,19	1.709,19	1.709,19	1.709,19	1.709,19	1.708,91
PROBLEMA	200 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1.104,77	1.104,77	1.104,77	1.104,77	1.104,77	1.076,64	1.100,94
E016_03m	284,42	284,42	284,42	284,42	284,42	284,42	284,42
E_n23_k3	676,80	676,80	676,80	650,20	676,80	655,59	676,80
E_n33_k4	981,19	981,19	959,27	958,36	958,36	958,36	958,36
P_n50_k10	750,54	738,58	738,58	739,11	737,56	737,56	738,58
P_n22_k2	249,58	245,28	249,58	231,65	249,58	243,46	249,58
P_n55_k10	750,16	748,68	748,68	750,16	750,16	750,16	750,16

P_n60_k15	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83
P_n101_k4	989,96	989,96	989,96	989,96	989,96	989,96	989,96
M_n200_k17	1.713,03	1.709,19	1.709,19	1.709,19	1.709,19	1.709,19	1.708,91
PROBLEMA	**400 ITERACIONES**						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	1.104,77	1.104,77	1.104,77	1.104,77	1.104,77	1.076,64	1.096,48
E016_03m	284,42	284,42	284,42	284,42	284,42	284,42	284,42
E_n23_k3	676,80	676,80	676,80	650,20	676,80	655,59	676,80
E_n33_k4	981,19	981,19	959,27	958,36	958,36	958,36	958,36
P_n50_k10	750,54	738,58	738,58	739,11	737,56	737,56	738,58
P_n22_k2	249,58	245,28	249,58	231,65	249,58	243,46	249,58
P_n55_k10	750,16	748,68	748,68	750,16	750,16	750,16	750,16
P_n60_k15	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83	1.059,83
P_n101_k4	989,96	989,96	989,96	989,96	989,96	989,96	989,96
M_n200_k17	1.713,03	1.709,19	1.709,19	1.709,19	1.709,19	1.709,19	1.708,91

Fuente autores

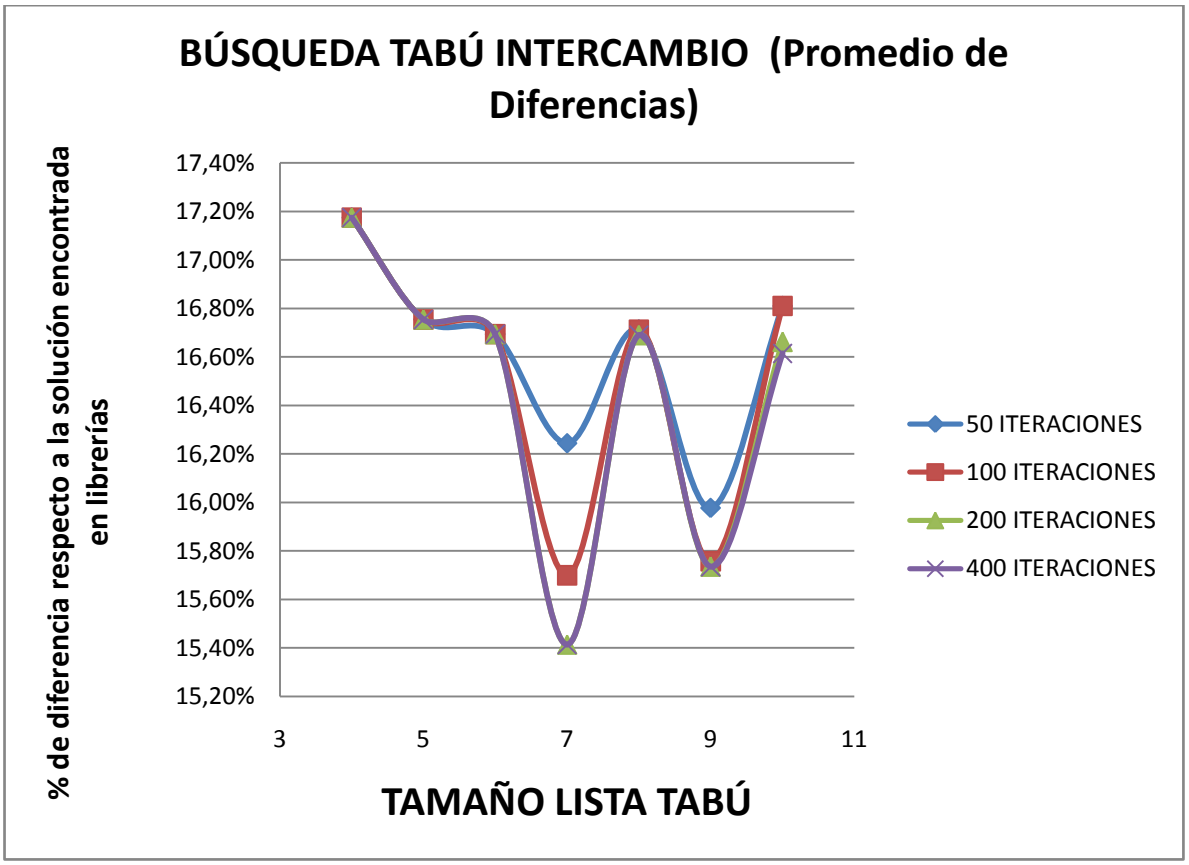
Tabla 11. Porcentajes de diferencia respecto a la solución encontrada en librerías (intercambio). Fuente autores.

PROBLEMA	**TABÚ INTERCAMBIO**						
	50 ITERACIONES						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	17,03%	17,03%	17,03%	17,03%	17,03%	14,05%	17,03%
E016_03m	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%
E_n23_k3	18,95%	18,95%	18,95%	14,27%	18,95%	17,42%	18,95%
E_n33_k4	16,12%	16,12%	13,52%	13,42%	13,42%	13,42%	13,42%
P_n50_k10	7,84%	6,12%	6,12%	6,19%	6,19%	6,19%	7,19%
P_n22_k2	15,55%	13,56%	15,55%	15,55%	15,55%	12,71%	15,55%
P_n55_k10	8,09%	7,88%	7,88%	8,09%	8,09%	8,09%	8,09%
P_n60_k15	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%
P_n101_k4	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%
M_n200_k17	24,77%	24,49%	24,49%	24,49%	24,49%	24,49%	24,47%
PROMEDIO	17,17%	16,75%	16,69%	16,24%	16,71%	15,98%	16,81%
PROBLEMA	**100 ITERACIONES**						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	17,03%	17,03%	17,03%	17,03%	17,03%	14,05%	17,03%
E016_03m	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%
E_n23_k3	18,95%	18,95%	18,95%	14,27%	18,95%	15,22%	18,95%
E_n33_k4	16,12%	16,12%	13,52%	13,42%	13,42%	13,42%	13,42%
P_n50_k10	7,84%	6,12%	6,12%	6,19%	6,19%	6,19%	7,19%

P_n22_k2	15,55%	13,56%	15,55%	10,10%	15,55%	12,71%	15,55%
P_n55_k10	8,09%	7,88%	7,88%	8,09%	8,09%	8,09%	8,09%
P_n60_k15	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%
P_n101_k4	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%
M_n200_k17	24,77%	24,49%	24,49%	24,49%	24,49%	24,49%	24,47%
PROMEDIO	17,17%	16,75%	16,69%	15,70%	16,71%	15,76%	16,81%
PROBLEMA	**200 ITERACIONES**						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	17,03%	17,03%	17,03%	17,03%	17,03%	14,05%	16,63%
E016_03m	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%
E_n23_k3	18,95%	18,95%	18,95%	14,27%	18,95%	15,22%	18,95%
E_n33_k4	16,12%	16,12%	13,52%	13,42%	13,42%	13,42%	13,42%
P_n50_k10	7,84%	6,12%	6,12%	6,19%	5,97%	5,97%	6,12%
P_n22_k2	15,55%	13,56%	15,55%	7,25%	15,55%	12,71%	15,55%
P_n55_k10	8,09%	7,88%	7,88%	8,09%	8,09%	8,09%	8,09%
P_n60_k15	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%
P_n101_k4	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%
M_n200_k17	24,77%	24,49%	24,49%	24,49%	24,49%	24,49%	24,47%
PROMEDIO	17,17%	16,75%	16,69%	15,41%	16,69%	15,74%	16,66%
PROBLEMA	**400 ITERACIONES**						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	17,03%	17,03%	17,03%	17,03%	17,03%	14,05%	16,15%
E016_03m	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%	8,56%
E_n23_k3	18,95%	18,95%	18,95%	14,27%	18,95%	15,22%	18,95%
E_n33_k4	16,12%	16,12%	13,52%	13,42%	13,42%	13,42%	13,42%
P_n50_k10	7,84%	6,12%	6,12%	6,19%	5,97%	5,97%	6,12%
P_n22_k2	15,55%	13,56%	15,55%	7,25%	15,55%	12,71%	15,55%
P_n55_k10	8,09%	7,88%	7,88%	8,09%	8,09%	8,09%	8,09%
P_n60_k15	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%	9,49%
P_n101_k4	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%	45,37%
M_n200_k17	24,77%	24,49%	24,49%	24,49%	24,49%	24,49%	24,47%
PROMEDIO	17,17%	16,75%	16,69%	15,41%	16,69%	15,74%	16,61%

Fuente autores

Ilustración 18. Resultados obtenidos al aplicar la estrategia tabú intercambio.



Fuente autores

De la ilustración anterior se puede decir que el promedio de las mejores soluciones encontradas usando la estrategia tabú intercambio es cercano al 15,5% y se alcanza con una combinación de tamaño de la lista tabú 7 para 200 y 400 iteraciones.

Tabla 12. Resultados obtenidos al aplicar la estrategia tabú mixto.

PROBLEMA	TABÚ MIXTO						
	50 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	960,42	960,42	960,42	960,42	960,42	960,42	960,42
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	634,60	634,60	639,24	639,24	639,24	639,24	639,24
E_n33_k4	865,72	895,53	860,10	895,53	895,53	895,53	855,14
P_n50_k10	756,33	754,57	754,57	756,33	754,57	756,33	754,57
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	744,26	750,15	744,26	744,04	744,26	742,78	730,22
P_n60_k15	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88
P_n101_k4	811,32	811,43	811,76	805,55	810,33	811,76	811,43
M_n200_k17	1.596,06	1.596,06	1.727,82	1.596,06	1.591,98	1.587,53	1.591,41
PROBLEMA	**100 ITERACIONES**						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	960,42	960,42	960,42	960,42	956,88	960,42	960,42
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	634,60	634,60	639,24	639,24	639,24	570,01	639,24
E_n33_k4	860,34	895,53	860,10	889,77	892,87	895,53	844,29
P_n50_k10	756,33	754,57	754,57	756,33	738,02	756,33	739,76
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	744,26	750,15	724,66	742,78	744,26	742,78	730,22
P_n60_k15	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88
P_n101_k4	789,14	787,76	787,76	791,44	791,55	796,07	783,91
M_n200_k17	1.596,06	1.596,06	1.727,82	1.596,06	1.587,70	1.587,44	1.579,08
PROBLEMA	**200 ITERACIONES**						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	960,42	960,42	960,42	960,42	953,25	960,42	960,42
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	634,60	634,60	639,24	634,60	639,24	568,56	639,24
E_n33_k4	860,34	895,53	860,10	869,11	892,87	875,50	844,29
P_n50_k10	756,33	754,57	754,57	756,33	737,33	754,57	739,76
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	744,26	750,15	722,38	742,78	743,74	742,78	730,22
P_n60_k15	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88
P_n101_k4	789,14	781,96	776,96	791,44	781,86	789,20	783,91
M_n200_k17	1.596,06	1.596,06	1.727,82	1.596,06	1.587,70	1.587,44	1.579,08
PROBLEMA	**400 ITERACIONES**						

	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	960,42	960,42	960,42	953,25	953,25	960,42	960,42
E016_03m	284,23	284,23	284,23	284,23	284,23	284,23	284,23
E_n23_k3	634,60	634,60	639,24	634,60	596,26	568,56	639,24
E_n33_k4	860,34	895,53	860,10	869,11	891,90	870,92	844,29
P_n50_k10	756,33	754,57	754,57	756,33	732,50	738,02	738,70
P_n22_k2	217,85	217,85	217,85	217,85	217,85	217,85	217,85
P_n55_k10	744,26	750,15	722,38	742,78	720,76	742,78	730,22
P_n60_k15	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88	1.036,88
P_n101_k4	789,14	781,96	776,96	791,44	781,86	789,20	776,30
M_n200_k17	1.596,06	1.596,06	1.727,82	1.596,06	1.587,70	1.587,44	1.579,08

Fuente autores

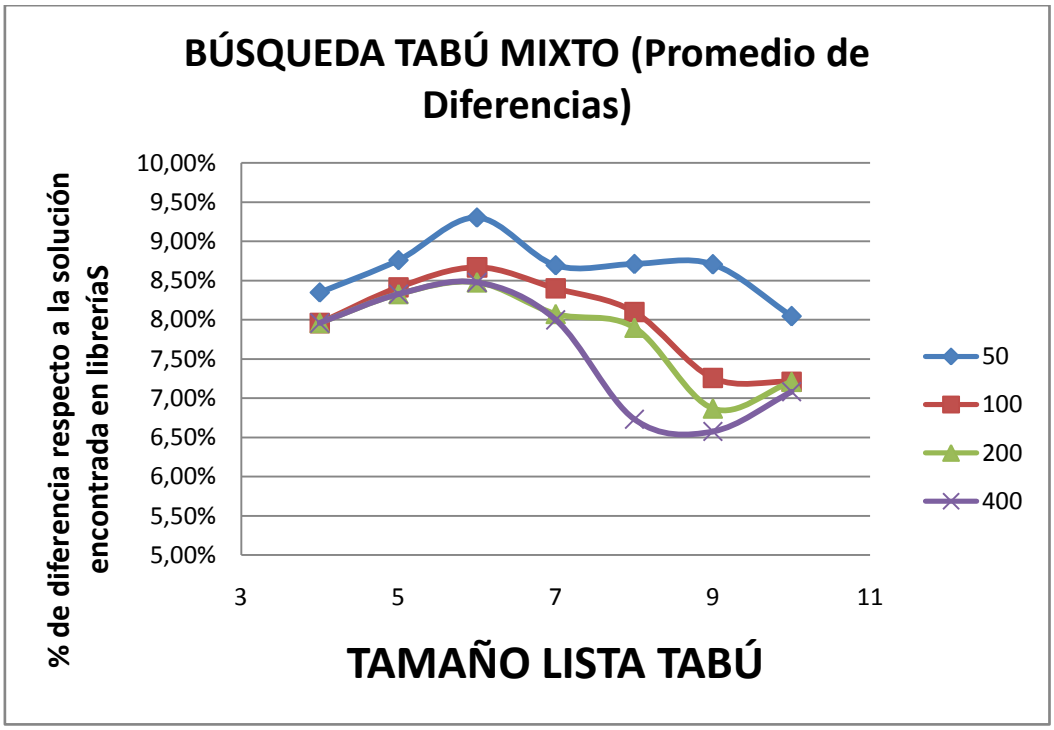
Tabla 13. Porcentajes de diferencia respecto a la solución encontrada en librerías (mixto).

PROBLEMA	TABÚ MIXTO						
	50 ITERACIONES						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1,74%	1,74%	1,74%	1,74%	1,74%	1,74%	1,74%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%
E_n23_k3	11,53%	11,53%	12,34%	12,34%	12,34%	12,34%	12,34%
E_n33_k4	2,45%	5,98%	1,79%	5,98%	5,98%	5,98%	1,20%
P_n50_k10	8,67%	8,41%	8,41%	8,67%	8,41%	8,67%	8,41%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	7,24%	8,09%	7,24%	7,21%	7,24%	7,03%	5,22%
P_n60_k15	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%
P_n101_k4	19,14%	19,15%	19,20%	18,29%	18,99%	19,20%	19,15%
M_n200_k17	16,25%	16,25%	25,84%	16,25%	15,95%	15,62%	15,91%
PROMEDIO	8,35%	8,76%	9,30%	8,69%	8,71%	8,70%	8,04%
PROBLEMA	**100 ITERACIONES**						
	LISTA TABÚ 4	LISTA TABÚ 5	LISTA TABÚ 6	LISTA TABÚ 7	LISTA TABÚ 8	LISTA TABÚ 9	LISTA TABÚ 10
A_n45_k6	1,74%	1,74%	1,74%	1,74%	1,36%	1,74%	1,74%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%
E_n23_k3	11,53%	11,53%	12,34%	12,34%	12,34%	0,18%	12,34%
E_n33_k4	1,82%	5,98%	1,79%	5,30%	5,67%	5,98%	-0,08%
P_n50_k10	8,67%	8,41%	8,41%	8,67%	6,04%	8,67%	6,29%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	7,24%	8,09%	4,42%	7,03%	7,24%	7,03%	5,22%
P_n60_k15	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%
P_n101_k4	15,88%	15,68%	15,68%	16,22%	16,23%	16,90%	15,11%

M_n200_k17	16,25%	16,25%	25,84%	16,25%	15,64%	15,62%	15,01%
PROMEDIO	7,96%	8,41%	8,67%	8,40%	8,10%	7,26%	7,21%
PROBLEMA	**200 ITERACIONES**						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	1,74%	1,74%	1,74%	1,74%	0,98%	1,74%	1,74%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%
E_n23_k3	11,53%	11,53%	12,34%	11,53%	12,34%	-0,08%	12,34%
E_n33_k4	1,82%	5,98%	1,79%	2,85%	5,67%	3,61%	-0,08%
P_n50_k10	8,67%	8,41%	8,41%	8,67%	5,94%	8,41%	6,29%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	7,24%	8,09%	4,09%	7,03%	7,17%	7,03%	5,22%
P_n60_k15	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%
P_n101_k4	15,88%	14,83%	14,09%	16,22%	14,81%	15,89%	15,11%
M_n200_k17	16,25%	16,25%	25,84%	16,25%	15,64%	15,62%	15,01%
PROMEDIO	7,96%	8,33%	8,48%	8,07%	7,90%	6,87%	7,21%
PROBLEMA	**400 ITERACIONES**						
	LISTA TABÚ 4	**LISTA TABÚ 5**	**LISTA TABÚ 6**	**LISTA TABÚ 7**	**LISTA TABÚ 8**	**LISTA TABÚ 9**	**LISTA TABÚ 10**
A_n45_k6	1,74%	1,74%	1,74%	0,98%	0,98%	1,74%	1,74%
E016_03m	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%	8,49%
E_n23_k3	11,53%	11,53%	12,34%	11,53%	4,79%	-0,08%	12,34%
E_n33_k4	1,82%	5,98%	1,79%	2,85%	5,55%	3,07%	-0,08%
P_n50_k10	8,67%	8,41%	8,41%	8,67%	5,24%	6,04%	6,13%
P_n22_k2	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%	0,86%
P_n55_k10	7,24%	8,09%	4,09%	7,03%	3,86%	7,03%	5,22%
P_n60_k15	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%	7,12%
P_n101_k4	15,88%	14,83%	14,09%	16,22%	14,81%	15,89%	13,99%
M_n200_k17	16,25%	16,25%	25,84%	16,25%	15,64%	15,62%	15,01%
PROMEDIO	7,96%	8,33%	8,48%	8,00%	6,73%	6,58%	7,08%

Fuente autores

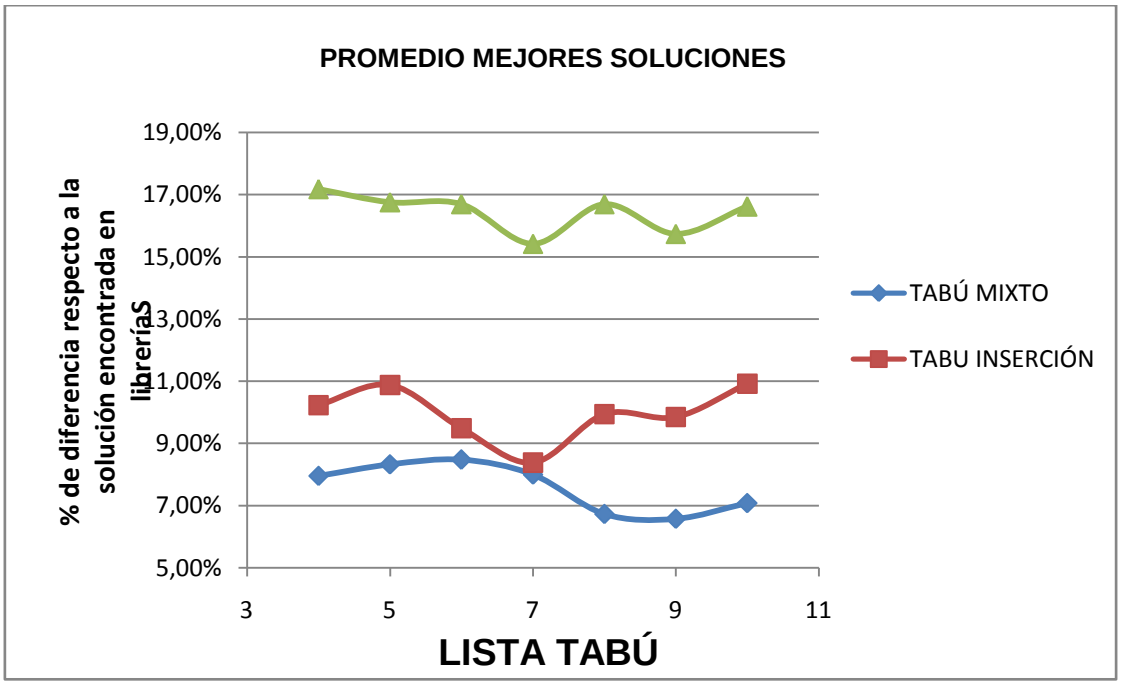
Ilustración 19. Resultados obtenidos al aplicar la estrategia tabú mixto.



Fuente autores

De la ilustración anterior se puede decir que el promedio de las mejores soluciones encontradas usando la estrategia tabú mixto es cercano al 6,5% y se alcanza con una combinación de tamaño de la lista tabú 9 y 400 iteraciones.

Tabla 14. Resumen, mejores soluciones encontradas por los algoritmos.



Fuente autores

Las mejores soluciones encontradas usando el algoritmo de búsqueda tabú se obtienen haciendo correr el algoritmo con cada una de sus metodologías 400 iteraciones. La ilustración anterior muestra que la metodología tabú mixto con 400 iteraciones y lista tabú de tamaño 9 arroja mejores resultados que las otras dos metodologías. Cabe aclarar que para corroborar o rechazar estas afirmaciones es necesario hacer un análisis estadístico el cual se encuentra en el siguiente capítulo.

Para el algoritmo de recocido simulado se programaron dos estrategias cada una de ellas contiene una función de enfriamiento diferente. La primera contiene una función de enfriamiento que decrece de manera lineal y la segunda contiene una función de enfriamiento que decrece de acuerdo a un factor definido. Tanto el algoritmo de búsqueda tabú como el algoritmo de recocido simulado, usan como solución inicial la mejor solución encontrada por las heurísticas de Clarke and Wrigth y Vecino más cercano.

A continuación se presentan los resultados obtenidos por el algoritmo de recocido simulado:

Tabla 15. Respuestas encontradas al aplicar el algoritmo de recocido simulado con una función de enfriamiento que decrece de acuerdo a un factor.

PROBLEMA	RECOCIDO FACTOR				RESPUESTA LIBRERÍAS
	50 ITERACIONES	100 ITERACIONES	200 ITERACIONES	400 ITERACIONES	
A_n45_k6	956,88	928,97	953,25	953,25	**944**
E016_03m	284,23	284,23	284,23	284,42	**262**
E_n23_k3	634,60	634,60	634,60	634,60	**569**
E_n33_k4	886,65	886,65	859,54	859,54	**845**
P_n50_k10	756,33	756,33	750,16	737,00	**696**
P_n101_k4	792,35	816,42	795,47	757,42	**681**
P_n22_k2	217,85	217,85	217,85	217,85	**216**
P_n55_k10	749,82	743,74	724,51	723,56	**694**
P_n60_k15	1.012,28	1.014,98	1.012,51	1.004,78	**968**
M_n200_k17	1.574,35	1.586,70	1.569,63	1.532,40	**1373**

Fuente autores

Tabla 16. Porcentajes de diferencia respecto a la solución encontrada en librerías (recocido factor).

PROBLEMA	DIFERENCIA RECOCIDO FACTOR			
	50	100	200	400
A_n45_k6	1,36%	-1,59%	0,98%	0,98%
E016_03m	8,49%	8,49%	8,49%	8,56%
E_n23_k3	11,53%	11,53%	11,53%	11,53%
E_n33_k4	4,93%	4,93%	1,72%	1,72%
P_n50_k10	8,67%	8,67%	7,78%	5,89%
P_n101_k4	16,35%	19,89%	16,81%	11,22%
P_n22_k2	0,86%	0,86%	0,86%	0,86%
P_n55_k10	8,04%	7,17%	4,40%	4,26%
P_n60_k15	4,57%	4,85%	4,60%	3,80%
M_n200_k17	14,66%	15,56%	14,32%	11,61%
PROMEDIO	7,95%	8,03%	7,15%	6,04%

Fuente autores

Tabla 17. Respuestas encontradas al aplicar el algoritmo de recocido simulado con una función de enfriamiento que decrece de manera lineal. Fuente autores.

PROBLEMA	RECOCIDO LINEAL				RESPUESTA LIBRERÍAS
	50 ITERACIONES	100 ITERACIONES	200 ITERACIONES	400 ITERACIONES	
A_n45_k6	960,42	971,91	928,97	928,97	**944**
E016_03m	284,23	284,23	284,23	284,23	**262**
E_n23_k3	630,09	634,60	627,92	627,92	**569**
E_n33_k4	886,42	887,91	888,47	885,93	**845**
P_n50_k10	752,02	754,57	738,02	729,71	**696**
P_n22_k2	217,85	217,85	219,14	217,85	**216**
P_n55_k10	744,10	724,07	720,60	724,26	**694**
P_n60_k15	1.014,98	1.014,98	1.014,98	998,99	**968**
P_n101_k4	803,10	784,75	772,34	770,88	**681**
M_n200_k17	1.596,50	1.583,70	1.555,77	1.555,77	**1373**

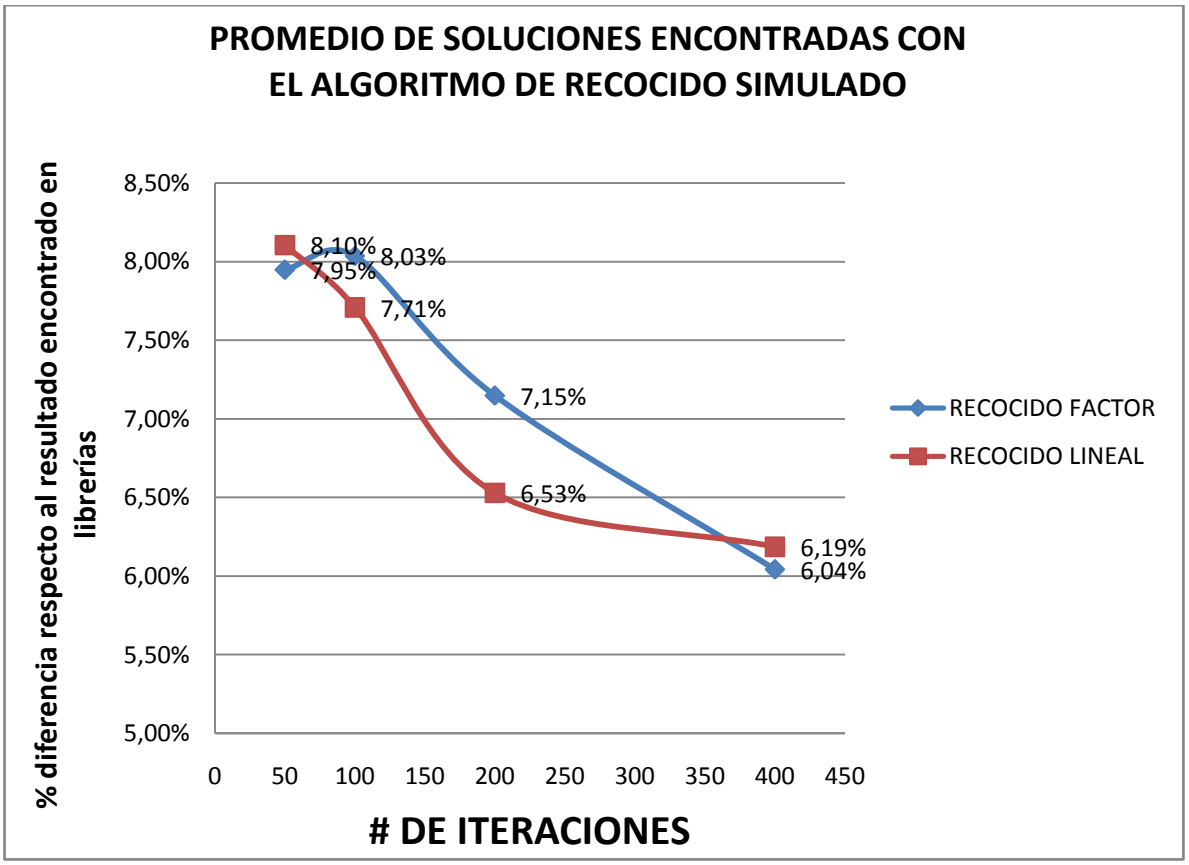
Fuente autores.

Tabla 18. Porcentajes de diferencia respecto a la solución encontrada en librerías (recocido lineal).

PROBLEMA	DIFERENCIA RECOCIDO LINEAL			
	50 ITERACIONES	100 ITERACIONES	200 ITERACIONES	400 ITERACIONES
A_n45_k6	1,74%	2,96%	-1,59%	-1,59%
E016_03m	8,49%	8,49%	8,49%	8,49%
E_n23_k3	10,74%	11,53%	10,35%	10,35%
E_n33_k4	4,90%	5,08%	5,14%	4,84%
P_n50_k10	8,05%	8,41%	6,04%	4,84%
P_n22_k2	0,86%	0,86%	1,45%	0,86%
P_n55_k10	7,22%	4,33%	3,83%	4,36%
P_n60_k15	4,85%	4,85%	4,85%	3,20%
P_n101_k4	17,93%	15,24%	13,41%	13,20%
M_n200_k17	16,28%	15,35%	13,31%	13,31%
PROMEDIO	8,10%	7,71%	6,53%	6,19%

Fuente autores

Ilustración 20. Promedio de las soluciones encontradas aplicando el algoritmo de recocido simulado.



Fuente autores

De la ilustración anterior se puede decir que no existe gran diferencia al aplicar las estrategias de recocido factor y recocido lineal. Al parecer el número de iteraciones que se use no es relevante ya que usando 50 iteraciones se obtiene un valor promedio cercano al 8%, mientras que si se usan 400 iteraciones el valor promedio se reduce a aproximadamente 6%. Cabe aclarar que para corroborar o rechazar estas afirmaciones es necesario hacer un análisis estadístico el cual se encuentra en el siguiente capítulo.

A continuación se muestra una tabla comparativa que contiene los mejores resultados obtenidos por cada una de los algoritmos:

Tabla 19. Comparativo de las mejores soluciones encontradas por los algoritmos TS y SA.

PROBLEMA	MEJOR SOLUCIÓN TABÚ	MEJOR SOLUCIÓN RECOCIDO	MEJOR ENCONTRADA	MEJOR SOLUCIÓN LIBRERÍAS	PORCENTAJE DE DIFERENCIA
A_n45_k6	953,25	928,97	928,97	**944,00**	-1,59%
E016_03m	284,23	284,23	284,23	**262,00**	8,48%
E_n23_k3	568,56	627,92	568,56	**569,00**	-0,08%
E_n33_k4	844,29	859,54	844,29	**845,00**	-0,08%
P_n50_k10	732,50	729,71	729,71	**696,00**	4,84%
P_n22_k2	217,85	217,85	217,85	**216,00**	0,86%
P_n55_k10	720,76	720,60	720,60	**694,00**	3,83%
P_n60_k15	1.036,88	998,99	998,99	**968,00**	3,20%
P_n101_k4	768,96	757,42	757,42	**681,00**	11,22%
M_n200_k17	1.579,08	1.532,40	1.532,40	**1.373,00**	11,61%
				Promedio	**4,23%**

Fuente Autores.

De la tabla No 12 se puede afirmar que los algoritmos de búsqueda tabú y recocido simulado arrojan buenas soluciones, ya que contrastando los resultados con los de las librerías la mayor diferencia es de 11,61%. En promedio, los resultados obtenidos por los algoritmos están a un 4,23 % de la solución encontrada en librerías. Hay casos donde la solución encontrada por los algoritmos es menor que la solución de las librerías.

7 DISEÑO DE EXPERIMENTOS PARA DETERMINAR LA INFLUENCIA DE LOS PARÁMETROS UTILIZADOS EN LOS ALGORITMOS DE BÚSQUEDA TABÚ Y RECOCIDO SIMULADO

Este capítulo está dedicado a realizar el análisis del comportamiento de los algoritmos desarrollados, bajo diferentes parámetros que se consideran pueden causar algún efecto sobre la respuesta de cada una de las que meta-heurísticas. Luego de identificar dichos parámetros, se hizo el diseño del experimento usando 10 problemas encontrados en las librerías y considerados por nosotros representativos. Estos problemas tienen (16, 22, 23, 45, 50, 55, 60,101 y 200) nodos. El experimento se llevó a cabo usando los resultados obtenidos en la aplicación desarrollada por los autores “CVRP”.

La variable de salida es el porcentaje de diferencia que presenta los algoritmos propuestos respecto a la solución dada en las librerías.

$$\%\boldsymbol{\Delta} = (\boldsymbol{y_{meta\ heuristica}} - \boldsymbol{y_{librerías}})/\boldsymbol{y_{librerías}}$$

El objetivo es determinar cuáles factores minimizan el $\%\Delta$.

7.1 Algoritmo de Búsqueda Tabú

En el algoritmo de BT se usaron los parámetros nombrados a continuación:

- Tamaños de la lista tabú
- Número de Iteraciones
- Función de vecindad (intercambio, inserción y mixto)
- La solución inicial
- El tamaño del problema

A continuación se muestra el análisis estadístico realizado para encontrar cuáles fueron los parámetros incidentes en la variable respuesta. El modelo estadístico que se utiliza para construir la ANOVA del diseño factorial general es el siguiente.

$$\boldsymbol{y_{ijk}} = \boldsymbol{\mu} + \boldsymbol{\tau_i} + \boldsymbol{\beta_j} + (\boldsymbol{\tau\beta})_{ij} + \boldsymbol{\epsilon_{ijk}} = \boldsymbol{y_{meta\ heuristica}}$$

Donde,

$\boldsymbol{y_{ijk}}$: Respuesta observada cuando el factor tamaño de la lista tabú está en el $\boldsymbol{i-}$ é$\boldsymbol{simo}$ nivel $(\boldsymbol{i=4,5,6,7,8,9,10})$ y el factor función de vecindad está en el $\boldsymbol{j-}$ é$\boldsymbol{simo}$ nivel
$(\boldsymbol{j=tab}$ú $\boldsymbol{inserci}$ó$\boldsymbol{n(1)}, \boldsymbol{tab}$ú $\boldsymbol{intercambio}\ \boldsymbol{(2)}, \boldsymbol{tab}$ú $\boldsymbol{mixto}\ \boldsymbol{(3)})$ para el $\boldsymbol{k-}$ é$\boldsymbol{simo}$ problema $(\boldsymbol{k=1,2,3,\ldots,10})$.

$\boldsymbol{\mu}$: Es el efecto promedio global

$\boldsymbol{\tau_i}$: Es el efecto del *i*-ésimo nivel del factor tamaño de la lista tabú.

$\boldsymbol{\beta_j}$: Es el efecto del *j*-ésimo nivel del factor función de vecindad.

$(\boldsymbol{\tau\beta})_{ij}$: Es el efecto de la interacción entre $\boldsymbol{\tau_i}$ y $\boldsymbol{\beta_j}$.

$\boldsymbol{\epsilon_{ijk}}$: Es el componente de error aleatorio

Con el experimento se desean probar las siguientes hipótesis:

$\boldsymbol{H_0: \tau_4 = \tau_5 = \tau_6 = \tau_7 = \tau_8 = \tau_9 = \tau_{10} = 0}$

$\boldsymbol{H_1: al\ menos\ una\ \tau_i \neq 0}$

$\boldsymbol{H_0: \beta_1 = \beta_2 = \beta_3 = 0}$

$\boldsymbol{H_1: al\ menos\ una\ \beta_j \neq 0}$

$\boldsymbol{H_0: (\tau\beta)_{ij} = 0\ \forall\ i,j.}$

$\boldsymbol{H_1: al\ menos\ un\ (\tau\beta)_{ij} \neq 0}$

A continuación se presentan los resultados del análisis estadístico realizado con la ayuda de la herramienta SPSS v18.

PRUEBA DE EFECTOS ENTRE FACTORES (ANOVA)

Fuente	Suma de Cuadrados Tipo III	Grados de Libertad	Media Cuadrática	F	Significancia
Modelo Corregido	,290[a]	20	,015	2,254	,003
Intersección	2,739	1	2,739	425,737	,000
Lista	,004	6	,001	,113	,995
Vecindad	,281	2	,141	21,873	,000
Lista * Vecindad	,004	12	,000	,055	1,000
Error	1,216	189	,006		
Total	4,245	210			
Total Corregida	1,506	209			

*R Cuadrado = 0,193 (R cuadrado ajustado = 107)

De la tabla anterior se puede concluir que no existe una interacción significativa entre los dos factores principales (Tamaño de la lista tabú y Función de vecindad) dado que la significancia (1,00) es mayor que el **α** (0,05) usado en el experimento, por lo tanto, no existe evidencia suficiente para rechazar $\boldsymbol{H_0}$**.** De igual manera se concluye que el efecto producido por el factor tamaño de la lista no es significativo y no hay evidencia suficiente para rechazar la $\boldsymbol{H_0}$**.** Para el caso del factor función de vecindad, la significancia (0,00) es menor que el **α** (0,05) usado en el experimento, por lo tanto, se puede afirmar que existe un efecto significativo sobre la variable respuesta y se rechaza la $\boldsymbol{H_0}$**.**

Puesto que uno de los factores principales analizados tiene un efecto significativo sobre la variable respuesta, se decidió realizar la prueba de Tukey con el fin de evaluar la significancia de todas las diferencias entre los tratamientos. Para ello se plantean las siguientes hipótesis:

$\boldsymbol{H_0}: \boldsymbol{\mu_a = \mu_b}$ $\quad$ $\boldsymbol{H_0}: \boldsymbol{\mu_a = \mu_c}$ $\quad$ $\boldsymbol{H_0}: \boldsymbol{\mu_b = \mu_c}$

$\boldsymbol{H_1}: \boldsymbol{\mu_a \neq \mu_b}$ $\quad$ $\boldsymbol{H_1}: \boldsymbol{\mu_a \neq \mu_c}$ $\quad$ $\boldsymbol{H_1}: \boldsymbol{\mu_b \neq \mu_c}$

A continuación se muestran los resultados obtenidos por la prueba.

Comparaciones Múltiples

Variable Dependiente: Diferencia

	(I) Vecindad	(J) Vecindad	Diferencia de Medias (I-J)	Std. Error	Significancia.
Tukey HSD	Tabú Inserción	Tabú intercambio	-,064613*	,0135581	,000
		Tabú Mixto	,021544	,0135581	,253
	Tabú intercambio	Tabú Inserción	,064613*	,0135581	,000
		Tabú Mixto	,086158*	,0135581	,000
	Tabú Mixto	Tabú Inserción	-,021544	,0135581	,253
		Tabú intercambio	-,086158*	,0135581	,000

Subconjuntos Homogéneos

Diferencia

	Vecindad	N	Subconjunto 1	Subconjunto 2
Tukey HSD[a,b]	Tabú Mixto	70	,078307	
	Tabú Inserción	70	,099851	
	Tabú intercambio	70		,164465
	Sig.		,253	1,000

De lo anterior se puede concluir que el tratamiento Tabú Intercambio, presenta diferencias significativas con el resto de los tratamientos. Si observamos los resultados arrojados por los algoritmos es evidente que el tratamiento tabú intercambio arroja resultados de menor calidad por lo que se recomienda usar las estrategias tabú inserción y tabú mixto para la solución de los problemas de CVRP.

Los demás parámetros no se consideraron de gran incidencia sobre la respuesta debido a:

- Número de iteraciones: para el parámetro se realizó un análisis previo el cual tuvo como resultado que el número de iteraciones que se debe usar para la aplicación del algoritmo es de 200, dado que se obtuvieron buenas soluciones en un tiempo de ejecución del algoritmo aceptable.
- Solución inicial: no se tuvo en cuenta debido a que el algoritmo toma la mejor solución encontrada por los algoritmos de Vecino más Cercano y Clarke and Wrigth, como solución inicial.
- Tamaño del problema: es un parámetro que incide sobre el tiempo de ejecución del algoritmo y no se puede determinar su incidencia sobre la variable respuesta puesto que esto depende de las características propias del problema.

7.2 Recocido Simulado

Se considera que los factores más incidentes en la variable de salida son los parámetros enunciados a continuación:

- Función de enfriamiento
- Número de iteraciones

El modelo estadístico que se utiliza para construir la ANOVA del diseño factorial general es el siguiente:

$$\boldsymbol{y_{ijk} = \mu + \tau_i + \beta_j + (\tau\beta)_{ij} + \epsilon_{ijk} = y_{meta\ heuristica}}$$

Donde,

$\boldsymbol{y_{ijk}}$: Respuesta observada cuando el factor función de enfriamiento está en el $\boldsymbol{i-}$ é$\boldsymbol{simo}$ nivel $(\boldsymbol{i = factor(1), lineal(2)})$ y el factor número de iteraciones está

en el $\boldsymbol{j-}$ é$\boldsymbol{simo}$ nivel $(\boldsymbol{j=50,100,200,400})$ para el $\boldsymbol{k-}$ é$\boldsymbol{simo}$ problema $(\boldsymbol{k=1,2,3,\dots,10})$.

$\boldsymbol{\mu}$: Es el efecto promedio global

$\boldsymbol{\tau_i}$: Es el efecto del $\boldsymbol{i-}$ é$\boldsymbol{simo}$ nivel del factor función de enfriamiento.

$\boldsymbol{\beta_j}$: Es el efecto del $\boldsymbol{j-}$ é$\boldsymbol{simo}$ nivel del factor número de iteraciones.

$(\boldsymbol{\tau\beta})_{\boldsymbol{ij}}$: Es el efecto de la interacción entre $\boldsymbol{\tau_i}$ y $\boldsymbol{\beta_j}$.

$\boldsymbol{\epsilon_{ijk}}$: Es el componente de error aleatorio

Con el experimento se desean probar las siguientes hipótesis:

$\boldsymbol{H_0: \tau_1 = \tau_2 = 0}$

$\boldsymbol{H_1: \tau_1 \neq \tau_2}$

$\boldsymbol{H_0: \beta_{50} = \beta_{100} = \beta_{200} = \beta_{400} = 0}$

$\boldsymbol{H_1: al\ menos\ una\ \beta_j \neq 0}$

$\boldsymbol{H_0: (\tau\beta)_{ij} = 0\ \forall\ i,j.}$

$\boldsymbol{H_1: al\ menos\ un\ (\tau\beta)_{ij} \neq 0}$

A continuación se presentan los resultados del análisis estadístico realizado con la ayuda de la herramienta SPSS v18.

PRUEBA DE EFECTOS ENTRE FACTORES (ANOVA)

Fuente	Suma de Cuadrados Tipo III	Grados de Libertad	Media Cuadrática	F	Significación
Modelo Corregido	,005a	7	,001	,260	,967
Intersección	,416	1	,416	147,323	,000
FE	5,153E-5	1	5,153E-5	,018	,893
Iteraciones	,005	3	,002	,576	,632
FE * Iteraciones	,000	3	7,184E-5	,025	,994
Error	,203	72	,003		
Total	,625	80			
Total Corregida	,209	79			

*R cuadrado = 0,025 (R cuadrado ajustado = -0,070)

De la tabla anterior se puede concluir que no existe una interacción significativa entre los dos factores principales (Número de iteraciones y Función de enfriamiento) dado que la significancia (0,994) es mayor que el **α** (0,05) usado en el experimento, por lo tanto, se puede afirmar que no hay evidencia suficiente para rechazar $\boldsymbol{H_0}$**.** De igual manera se concluye que los efectos producidos por los factores principales tampoco son significativos.

8 CONCLUSIONES

- Se alcanzaron todos los objetivos propuestos en la investigación; los algoritmos desarrollados arrojan resultados bastante buenos para los problemas propuestos del CVRP, en tiempos computacionales aceptables.

- Dado que los problemas de tipo combinatorio como el VRP y sus variaciones tienen gran complejidad matemática es recomendable el uso de heurísticas y metaheurísticas para obtener soluciones eficientes.

- Los algoritmos de Búsqueda tabú y recocido simulado son flexibles y se pueden aplicar a diversos problemas de tipo combinatorio.

- En la programación realizada se creyó conveniente dejar abiertos los códigos de la aplicación de tal manera que el usuario pueda realizar modificaciones que le sean útiles en la solución de otros tipos de problemas o para el avance y mejora de esta aplicación.

- Los algoritmos metaheurísticos producen mejores soluciones que los algoritmos heurísticos ya que no se encierran en óptimos locales y se permite la exploración de más regiones del problema.

- Teniendo en cuenta los resultados arrojados por el diseño de experimentos se puede afirmar que:

    - En el algoritmo de búsqueda tabú el tratamiento Tabú Intercambio, presenta diferencias significativas con el resto de los tratamientos. Si observamos los resultados arrojados por los algoritmos, es evidente que el tratamiento tabú intercambio arroja resultados de menor calidad por lo que se recomienda usar las estrategias tabú inserción y tabú mixto para la solución de los problemas de CVRP.

- El segundo factor evaluado (tamaño de la lista tabú) al igual que la interacción entre los factores principales no tienen un efecto significativo sobre la variable respuesta.

- Aunque el diseño de experimentos muestra que no es incidente el factor tamaño de la lista tabú sobre la variable respuesta, nos atrevemos a recomendar que se use un tamaño de lista tabú igual a 7 cuando la cantidad de clientes sea menor a 100, dado que, en la gran mayoría de los problemas que se ejecutaron se obtuvieron buenos resultados usando este tamaño de lista.

- En el algoritmo de recocido simulado no existe una interacción significativa entre los dos factores principales (Número de iteraciones y Función de enfriamiento) dado que la significancia (0,994) es mayor que el **α** (0,05) usado en el experimento, por lo tanto, se puede que no hay evidencia suficiente para rechazar $\boldsymbol{H_0}$**.** De igual manera se concluye que los efectos producidos por los factores principales tampoco son significativos.

- Todos los problemas evaluados presentan características diferentes, por lo cual no se puede afirmar a ciencia cierta cuál de las dos metaheurísticas arroja los mejores resultados. Teniendo en cuenta que el tiempo computacional de los algoritmos no es muy grande, se recomienda que se desarrollen los problemas usando las diversas metodologías incluidas en la aplicación, de esta manera se puede escoger entre las soluciones encontradas cual es la que mejor resultado presenta.

9 RECOMENDACIONES

- Mostrar a los estudiantes de pregrado y de postgrado la herramienta desarrollada para usarla como una aplicación en las clases de logística e investigación de operaciones ya que consideramos puede ser un buen apoyo en el proceso de aprendizaje.

- Después de investigar el problema de ruteo de vehículos, se recomienda introducir a la aplicación matrices con costos reales de desplazamiento o datos de georeferenciación con el fin de obtener resultados aplicables a la realidad.

- Motivar a los estudiantes para que investiguen en áreas como la logística, investigación de operaciones, control de producción para que desarrollen herramientas útiles para solucionar los problemas que día a día se presentan en las organizaciones respecto a estas temáticas.

- Se sugiere que los resultados obtenidos por la aplicación sean contrastados con los resultados encontrados en otro tipo de aplicación, con el fin de comparar la eficiencia de las mismas.

- Se espera que la aplicación sea utilizada como referencia para futuros estudios tanto teóricos como aplicativos en el grupo de investigación OPALO.

10 TRABAJOS FUTUROS

Para futuras investigaciones se recomienda:

- Modificar el software en los módulos de solución inicial y tipo de vecindario, agregando otros tipos de metodologías que mejoren el desempeño de la herramienta tanto en tiempo computacional como calidad de los resultados.

- Incorporar al software metaheurísticas como los algoritmos genéticos, colonia de hormigas etc. para solucionar los mismos problemas y poder hacer una comparación de cual algoritmo es más eficiente en resultados y tiempo computacional.

- Ampliar la investigación a otros problemas del VRP tales como CVRPTW (Capacitated Vehicle Routing Problem With Time Windows), PVRP (Periodic Vehicle Routing Problem) entre otros.

- Desarrollar software en otros lenguajes de programación que permitan ser aplicados no solo a la parte académica sino también a la parte empresarial sin tener que asumir altos costos por la compra de licencias.

REFERENCIAS BIBLIOGRÁFICAS

1. H. Papadimitriou, Kenneth Steiglitz “Combinatorial Optimization: Algorithms And Complexity” Englewood Cliff N.J 2 Editions (1982) pag. 3.
2. Dantzig, G y Ramser, J. (1959)], “The Truck Dispatching Problem”, Management Science 6. Pag. 80-91.
3. Clarke, G. y Wright, W. (1964), “Scheduling of vehicles from a central depot to a number of delivery points”, Operations Research 12. Pag. 568-581.
4. Toth, P., y Vigo, D. (2000), “An Overview of Vehicle Routing Problems, Monographs on Discrete Mathematics and Applications. In the Vehicle Routing Problem”, SIAM, 1–26.
5. B. Nag, BL Golden, and A. Assad, “Vehicle Routing with Site Dependencies,in: Vehicle Routing; Methods and Studies“ edited by BL Golden N.J 2 Editions (1988) pag. 35.
6. Melián, B., Moreno, J. y Moreno, M. (2003), “Metaheuristics: A global view”, Inteligencia Artificial, Revista Iberoamericana de Inteligencia Artificial 19, pag. 7-28.
7. Guignard, M. (2002), Lagrangian Relaxation, Handbook of Applied Optimization, Oxford University Press, pag. 465-474.
8. Resende, M. y Ribeiro, C. (2003), Greedy randomized adaptive search procedures, Handbook of Metaheuristics, Kluwer Academia Publishers.
9. Goldberg, D. (1989), Genetic Algorithms in Search, Optimization and Machine Learning, Addison Wesley.
10. Glover, F. (1986) “Future Paths for Integer Programming and Links to Artificial Intelligence”,Computers and Operations Research. 5. Pag, 533-549.
11. Gutiérrez de Los Cobos Pérez. Búsqueda Tabú: Un procedimientos heurístico para solucionar problemas de optimización combinatoria. Disponible en http://www.azc.uam.mx/publicaciones/enlinea2/1-3.htm

12. Alfredo Olivera, “Heurísticas para el problema de ruteo de vehículos”, Universidad de la República, Montevideo Uruguay 2004.
13. Yellow, P.: A computational modification to the savings method of vehicle scheduling. Operational Research Quarterly 21 (1970). Pag. 281–283.
14. Golden, B., Magnanti, T., Nguyen, H.: Implementing vehicle routing algorithms. Networks 7 (1977). Pag. 113–148.
15. Gaskell, T.: Bases for vehicle fleet scheduling. Operational Research Quarterly 18 (1967). Pag. 281–295.
16. Lin, S.: Computer solutions of the traveling salesman problem. Bell System Technical Journal 44 (1965). Pag. 2245–2269.
17. Johnson, D., McGeoch., L.: The Traveling Salesman Problem: a case studyin local optimization. In: Local Search in Combinatorial Optimization.John Wiley and Sons (1997). Pag. 215–310.
18. Lin, S., Kernighan, B.: An effective heuristic algorithm for the traveling salesman problem. Operations Research (1973). Pag. 498–516.
19. Melián, B., Glover, F. Introducción a la Búsqueda Tabú.
20. H. Li and A. Lim, “Local search with annealing-like restarts to solve the vehicle routing problem with time windows” Proceedings of the 2002 ACM symposium on Applied computing, ACM, (2002), Pag. 560-565.
21. S. Lin et. al, “Vehicle Routing Problems with Time Windows Using Simulated Annealing”, IEEE International Conference on Systems, Man and Cybernetics, 2006. SMC '06. Vol. 1, (2006), Pag. 645-650.
22. H. C. B. de Oliveira et. al, “A Multi-Start Simulated Annealing Algorithm for the Vehicle Routing Problem with Time Windows”, Ninth Brazilian Symposium on Neural Networks, 2006. SBRN '06., (2006), Pag. 137-142.
23. R. Tavakkoli-Moghaddam et. al, “A New Capacitated Vehicle Routing Problem with Split Service for Minimizing Fleet Cost by Simulated Annealing” Journal of the Franklin Institute Vol. 344, (2007). Pag, 406-425.
24. K.A. Dowsland, A. Díaz. Diseño de heurísticas y fundamentos del recocido simulado. Inteligencia Artificial VII:2 (2003) 93-101.

Disponible en http://sci2s.ugr.es/docencia/algoritmica/Tema03-EnfriamientoSimulado-09-10.pdf

25. Or. Traveling Salesman-type Combinatorial Problems and Their Relation to the Logistics of Regional Blood Banking. PhD thesis, Department of Industrial Engineering and Management Sciences, Northwestern University, Evanston, IL.
26. Delgado, c. Uso de conjunto de concentración en busqueda tabú. Nuevas soluciones para TSPLIB. Actas de la VII Jornadas de ASEPMA, Valencia, 24-25 de Septiembre de 1999. Pag: 210-220.
27. A. A. Martínez morales.: "Algoritmo Basado en Tabu Search para el Cálculo del Índice de Transmisión de un Grafo". Departamento de computación Facultad de ciencias y tecnología. FARAUTE de Ciencias y Tecnología, Vol. 1, No. 1, páginas 31-39. Universidad de Carabobo, Valencia, Estado Carabobo, Venezuela (2006). Disponible en http://campusv.uaem.mx/revista_os/articulos/vol1no1articulo6.pdf
28. Augerat, et al. Librería CVRP, disponible en, http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/
29. Christofides and Eilon. Librería CVRP, disponible en, http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/
30. Fisher. Librería CVRP, disponible en, http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/
31. Gillet and johnson. Librería CVRP, disponible en, http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/
32. Christofides, Mingozzi and Toth. Librería CVRP, disponible en, http://www.coin-or.org/SYMPHONY/branchandcut/VRP/data/

ANEXO A

PROGRAMACIÓN DE LOS ALGORITMOS BÚSQUEDA TABÚ Y RECOCIDO SIMULADO EN MATLAB

Este capítulo muestra una descripción general las funciones creadas especialmente para la aplicación de los algoritmos usados en el proyecto.

Función Precio Ruta precioruta(ruta,c)

Objetivo: Calcular el precio de un vector

Variables de Entrada:

Ruta= Es el vector ruta al que se desea calcular el costo.

c= la matriz de costo con la que se calcula el costo de la ruta.

Variables de Salida:

Costo=costo calculado de la ruta.

```
function costo= precioruta(ruta,c)%COSTO= la respuesta; ruta=vector; c= matriz costo
for i=2:length(ruta)% for del tamaño del vector  menos 1
precio(i-1)=c(ruta(i-1),ruta(i));%costo de ir de un cliente al siguiente
end
costo=sum(precio);%costo de ir a todos los clientes del vector
```

Función Demanda Asociada demanda_asociada(ruta,d):

Objetivo: Calcular la carga asociada a cada vehículo.

Variables de Entrada:

ruta= Es el vector al que se le va a calcular la demanda.

d=Es el vector demanda asociado a la ruta.

Variables de Salida:

demanda= demanda asociada a cada vehículo

```
function demanda= demanda_asociada(ruta,d)%demanda= la respuesta;
ruta=vector; d= vector demanda del problema
y=d(ruta);%y= vector demanda de la ruta
ceros=find(y==0);%ceros = halla la posición de los depósitos en el vector ruta
for i=1:length(ceros)-1;% halla el numero de depósitos que hay en el vector
   u(i)=sum(y(ceros(i):ceros(i+1)));%u= halla la demanda o caga que hay entre dos
depósitos consecutivos
end
demanda =u;% halla la demanda asociada a cada vehículo
end
```

Función Restricción Tabú restabu(listabu,movtabu):

Objetivo: Calcular cuantas veces está un movimiento en la lista tabú.

Variables de Entrada:

listabu=matriz de tamaño 2*Tamaño lista tabú donde se almacenaron los últimos movimientos.

movtabu=es un vector de 2 celdas, donde se almacenan los movimientos que se pretenden hacer.

Variables de Salida:

Debe arrojar un valor entre [0, 2].

```
function restabu=restabu(listabu,movtabu)%restabu= la respuesta; listatabu= la
lista tabu actual; movtabu= el movimiento  que se va analizar
for i=1:length(listabu(:,1));%for del tamaño de la lista tabu
```

```
    s(i)=sum(listabu(i,:)==movtabu(1))+sum(listabu(i,:)==movtabu(2));% halla las
veces que se encuentra el movimiento en la lista
    restabu=sum(s);% debe arrojar un valor < 2
end
```

Función Factibilidad facti(ruta,d,q):

Esta función utilizada para rutas donde hay sobrecupo en un vehículo.

Objetivo: Esta función intenta equilibrar las cargas en los vehículos y se usa generalmente en las soluciones iniciales.

Variables de Entrada:
ruta= Es el vector al que se le va a calcular la factibilidad.
d= Es el vector demanda asociado a la ruta.
q=Capacidad de un vehículo.

Variables de Salida:
Factib=Es un vector que contiene la ruta y en la mayoría de los casos equilibra la carga de cada vehículo.

```
function      factib =facti(ruta,d,q)%factib= la respuesta; ruta=vector; d= vector
demanda, q=capacidad de los vehículos
dep=max(ruta);%hallo cual es el deposito
rsob=find(demanda_asociada(ruta,d)==max(demanda_asociada(ruta,d)));%halla
cual es el vehículo  con sobrecupo
rdis=find(demanda_asociada(ruta,d)==min(demanda_asociada(ruta,d)));%halla
cual es el vehículo  con disponible
deps=find(ruta==dep);% halla la posición de los depósitos
rso=ruta(deps(rsob(1))+1:deps(rsob(1)+1)-1);%vector del vehículo con sobrecupo
rdi=ruta(deps(rdis(1))+1:deps(rdis(1)+1)-1);%vector del vehículo con disponible
xa=find(d(rso)==min(d(rso)))+deps(rsob(1));%posición a trasportar
```

```
xb=deps(rdis(1));%posición a insertar
if(xa>xb)% si la posición del cliente a transportar es menor que la del vehículo a insertar
 factib=[ruta(1:xb(1)),ruta(xa(1)),ruta(xb(1)+1:xa(1)- 1),ruta (xa (1) +1 :end)]; % modifica el vector ruta
else
 factib=[ruta(1:xa(1)-1),ruta(xa(1)+1:xb(1)), ruta(xa(1)), ruta(xb(1) +1:end)];% modifica el vector ruta
end
end
```

Función Mejorar **mejorar(ruta,c,pos):**

Esta función subdivide el vector ruta en sub-vectores donde cada sub-vector es la asignación de clientes de cada vehículo, luego se hace un barrido de todas las posibles combinaciones de las posiciones de cada cliente en los sub-vectores y se calculan los costos de estas rutas con el fin de elegir la ruta óptima del sub-vector.

Objetivo: Optimizar la ruta asignada a un vehículo que tenga menos de 9 clientes.

Variables de Entrada:
ruta= ruta que se desea optimizar.
c=costo de la ruta a optimizar.
pos=posición del vehículo en el vector.

Variables de Salida:
Una nueva ruta para la cual se espera que sus costos sean más bajos que la ruta inicial.

```
%funion para optimizar la ruta de un vehículo con clientes entre 1 & 9
function tsp =mejorar(ruta,c,pos)%ruta a optimizar, costos, posición del vehículo
en el vector
dep=max(ruta);% hallo cual es el deposito
ins=find(ruta==dep);%hallo posición de los depósitos
sub=ruta(ins(pos)+1:ins(pos+1)-1);%sub ruta a mejorar
   if(length(sub)>2 && length(sub)<9)%máximo tamaño de la permutación
w=perms(sub);%perm posibles
r=99999;%costo inicial a mejorar
for i=1:length(w(:,1))/2%barrido de todas las permutaciones
   s=[dep,w(i,:),dep];%sub vector de clientes
if (r>precioruta(s,c))% si hay una mejor
   tsp=s;%actualice respuesta
   r=precioruta(s,c);% actualice precio
end %fin mejora
end %fin barrido
   else
      tsp=[dep,sub,dep];
   end
   end
```

Función Unir unir(ruta,c):

Esta función une cada una de las rutas obtenidas luego de aplicar la función mejorar.

Objetivo: Unir las rutas que de manera individualmente se han optimizado.

Variables de Entrada:

ruta= es la ruta de un vehículo.

c= es el costo de la ruta de un vehículo.

Variables de Salida:

Un vector que contiene todos los sub-vectores unidos y el costo de esta nueva ruta.

```
function rutaopt =unir(ruta,c)%ruta completa a mejorar, c= matriz de costo
g=mejorar(ruta,c,1);% primer vehículo a optimizar
h=length(find(ruta==max(ruta)));% halla el número de vehículos en el vector
for jj=2:h-1%
   g2=mejorar(ruta,c,jj);% optimización del enrutamiento del vehículo jj
   g=[g(1:end-1),g2];% reconstrucción del vector total
end
rutaopt=g;% vector respuesta
end
```

Función De Vecindario Inserción inser(ruta,demanda,costo,q):

Objetivo: Generar el vecindario tipo inserción de la última ruta obtenida.

Variables de Entrada:

ruta=ruta a la que se le va a aplicar la estrategia de vecindad.

demanda=es la demanda asociada a la ruta que se le va aplicar la estrategia de vecindad.

costo=es el costo de la ruta a la que se le va aplicar la estrategia de vecindad.

q=capacidad de los vehículos.

Variables de Salida:

Una matriz organizada de manera descendente de acuerdo a los costos que contiene todas las rutas vecinas factibles además de los costos de cada ruta y los

movimientos realizados desde la ruta original hasta cada una de las vecinas encontradas.

```
function ruta1=inser(ruta,demanda,costo,q)% ruta a trabajar, demanda, matriz de costo, capacidad de los vehículos
cont=0;
      if length(ruta)>72% criterio de barrido según tamaño
  for i=2:2:length(ruta)-1% celda a tomar
     for j=3:2:length(ruta)-1;% posición de a insertar
        if(i~=j)%no repetición
           vc=[ruta(1:i-1),ruta(i+1:end)];% vector corto sin celda tomada
           r=[vc(1:j-1),ruta(i),vc(j:end)];%vector nuevo con celda insertada
    if (max(demanda_asociada(r,demanda))<=q) %restricción de capacidad
       cont=cont+1;
        rts(cont,:)=r;%almaceno rutas
        cts(cont)=precioruta(r,costo);%almaceno el valor del costo
        mov(cont,:)=[r(i),r(j)];;%almaceno el movimiento tabú
    else,end %fin restricción capacidad
     else,end %fin restricción  desigualdad
  end;end; %fin i j
     else % si la ruta es menor de 80
          for i=2:length(ruta)-2% celda inferior
    for j=3:length(ruta)-1;% celda superior
        if(i~=j)%no repetición
          vc=[ruta(1:i-1),ruta(i+1:end)];% vector corto sin celda tomada
           r=[vc(1:j-1),ruta(i),vc(j:end)];%vector nuevo con celda insertada
    if (max(demanda_asociada(r,demanda))<=q) %restricción de capacidad
       cont=cont+1;
        rts(cont,:)=r;% almaceno rutas
```

```
        cts(cont)=precioruta(r,costo);%almaceno el valor del costo
        mov(cont,:)=[r(i),r(j)];;%almaceno el movimiento tabú
    else,end %fin restricción capacidad
     else,end %fin restricción  desigualdad
  end;end; %fin i j
     end% fin restricción tamaño de la ruta
ruta1=double(sortrows([cts',rts,mov]));% ordeno las respuesta en una matriz en
orden descendente del costo
end
```

<u>Función De Vecindario Tipo Intercambio</u> inter(ruta,demanda,costo,q):

Objetivo: Generar un vecindario tipo intercambio de la ultima ruta obtenida.

Variables de Entrada:

ruta=ruta a la que se le va a aplicar la estrategia de vecindad.

demanda=es la demanda asociada a la ruta que se le va aplicar la estrategia de vecindad.

costo=es el costo de la ruta a la que se le va aplicar la estrategia de vecindad.

q=capacidad de los vehículos.

Variables de Salida:

Una matriz organizada de manera descendente de acuerdo a los costos que contiene todas las rutas vecinas factibles además de los costos de cada ruta y los movimientos realizados desde la ruta original hasta cada una de las vecinas encontradas.

```
function ruta1=inter(ruta,demanda,costo,q)% ruta a trabajar, demanda, matriz de
costo, capacidad de los vehículos
cont=0;
if (length(ruta)>80)
```

```
  for i=2:2:length(ruta)-3;% celda inferior
     for j=i+1:2:length(ruta)-1;% celda superior
        if(ruta(i)~=ruta(j))%no intercambio de depósitos
        r=ruta;r(i)=ruta(j);r(j)=ruta(i);%intercambio de 2 celdas
     if (max(demanda_asociada(r,demanda))<=q) %restricción de capacidad
        cont=cont+1;%contador
        rts(cont,:)=r;%almaceno rutas
        cts(cont)=precioruta(r,costo);%almaceno el valor del costo
        mov(cont,:)=[ruta(i),ruta(j)];%almaceno el movimiento tabú
     else;end
      else,end %fin restriction
  end;end; %fin i j
else
  for i=2:length(ruta)-3;% celda inferior
     for j=i+1:length(ruta)-1;% celda superior
        if(ruta(i)~=ruta(j))%no intercambio de depósitos
        r=ruta;r(i)=ruta(j);r(j)=ruta(i);%intercambio de 2 celdas
     if (max(demanda_asociada(r,demanda))<=q) %restricción de capacidad
        cont=cont+1;%contador
        rts(cont,:)=r;%almaceno rutas
        cts(cont)=precioruta(r,costo);%almaceno el valor del costo
        mov(cont,:)=[ruta(i),ruta(j)];%almaceno el movimiento tabú
     else;end
      else,end %fin restriction
  end;end; %fin i j
end
ruta1=double(sortrows([cts',rts,mov]));% ordeno las respuesta en una matriz en
orden descendente del costo
end
```

Código Para la Heurística del Vecino más Cercano: Código utilizado para generar la solución inicial utilizando la heurística del mejor vecino más cercano.

```
load (strcat(archivo,'.mat'));% se carga los parámetros del archivo
r=dep;%primer deposito
ct=c;% se crea una matriz de costos temporal
dt=d(1:n-1);%demanda temporal, la ultima demanda es la del deposito= no se
toma
for i=1:n+k-1
   ins2=max(find(r==dep));%hallo ultimo deposito de la ruta
   carga=sum(d(r(ins2:end)));%hallo ocupación del ultimo vehículo asignado
   if((q-carga)>=min(dt))%si hay demanda por satisfacer
      x=find(dt<=(q-carga));%hallo  vector de cliente factibles
      cf=ct(r(i),:);%vector de costo de ir de i a cualquier j
      cf(r(i))=9999;%penalizo el costo de ir a si mismo
      cff=cf(x);%vector de costo de ir de i a j factibles
      y=find(cf==min(cff));%hallo cliente posible de menor costo
      r=[r,y(1)];%asigne el cliente al final de la ruta
      dt(y(1))=10*q;% penalice la demanda del cliente
      ct(:,y(1))=9999;
   else%si el margen es menor que la demanda posible mínima
      r=[r,dep];%asigne un nuevo vehículo
   end
end
format short g
ruta=unir(r,c);% mejora la solución inicial
costo=precioruta(ruta,c);%calcula el costo de la ruta
clear carga ccf cf  dt final h i j ins2 r ult w x z y name  matriz_ahorros
```

Código Clarke And White (algoritmo de ahorros): Código utilizado para generar la solución inicial utilizando la heurística Clarke And White.

```
load (strcat(archivo,'.mat'))% se carga los parámetros del archivo
    for j=1:n
        h(i,j)=c(dep,i)+c(dep,j)-c(i,j);% genero matriz de ahorros
h(i,i)=0;
    end
end
r=dep;%primer depósito
dt=d(1:end-1);%vector demanda temporal
matriz_ahorros=h;deps=find(r==dep);
while (length(deps)<k)%mientras los depósitos
     ult=r(end);%ultima  asignación
          %%%%%%%% deposito
    if (ult==dep);%si la ultima asignación fue un deposito
    [x,y]=find(h==max(max(h)));%halle la pareja de mayor ahorro
        if(d(x(1))+d(y(1))<q)%si la si la demanda de la pareja es menor que q
        r=[r,x(1),y(1)];%asigne pareja a la  ruta
        dt(x(1))=10*q;%demanda  del primero penalizada
        h(:,x(1))=0;h(x(1),:)=0;%<ahorros del primero penalizada la fila y la columna
        else%si la si la demanda de la pareja es mayor que q
        r=[r,x(1)];
        end%fin restricción de la demanda de la pareja

    else%si la última asignación no fue un depósito
   dt(ult)=10*q;%penalice demanda de la ultima asignación
   ins2=max(find(r==dep));%hallo ultimo deposito de la ruta
   carga=sum(d(r(ins2:end)));%hallo ocupación del ultimo vehículo asignado
```

```
  if((q-carga)>=min(dt))%si hay demanda por satisfacer
      cf=find(dt<=(q-carga));%hallo  vector de cliente factibles
      w=h(ult,:);%ahorros de los clientes
      ccf=w(cf);% ahorros de los clientes factibles
      z=find(w==max(ccf));%hallar el mayor ahorro de los clientes factibles
      r=[r,z(1)];%asigne el de mayor ahorro
      h(:,ult)=0;h(ult,:)=0;%ahorros del anterior penalizada
  else% si no hay demanda por satisfacer
     h(:,ult)=0;h(ult,:)=0;dt(ult)=11*q;%ahorros del anterior penalizada
     r=[r,dep];%asigno deposito
    deps=find(r==dep);
  end%fin restricción demanda por satisfacer
   end%fin asignaciones
end%fin iteraciones
   dt(r(end))=10*q;%penalizo la última asignación
  final=find(dt<q);%hallo los nodos no asignados
ruta=[r,final',dep];% finalizo ruta
if(max(demanda_asociada(ruta,d))>q)%restricción de cumplimiento de demanda
vs q
for i=1:10
  ruta=facti(ruta,d);%vuelvo factible vector
end
r=inter(ruta,d,c,q);
ruta=r(1,2:end-2);
end
ruta=double(unir(ruta,c));
costo=precioruta(ruta,c);
clear carga ccf cf deps dt final  i j  ins2 r ult w x z y name matriz_ahorros
```

Código Inicial: Código utilizado para hallar la mejor solución inicial entre las heurísticas de Clarke and Wrigth y la de Vecino más Cercano.

```
load (strcat(archivo,'.mat'))% se carga los parámetros del archivo
clark;rutaclark=ruta;costoclark=costo;% ejecuto el código de clarke and wright y guardo sus valores
mejor_vecino;rutavec=ruta;costovec=costo;mejor vecino% ejecuto el código de clark and white y guardo sus valores
if(costovec<costoclark)% si costo de vecino menor que costo clarke
    costo_ini=costovec;% actualice costo_inicial
    ruta_ini=rutavec;% actualice ruta inicial
else
    costo_ini=costoclark;
    ruta_ini=rutaclark;% actualice ruta inicial
end
```

Código Búsqueda Tabú Inserción: Código utilizado para hacer una búsqueda tabú utilizando el vecindario tipo inserción.

```
rt=ruta_ini;%valor inicial de la ruta
opti=costo_ini*1.10;listabu=[0,0];mj=0;% mejor costo , lista tabú en ceros, mejoras en ceros
itera=0;% numero de vecindarios a recorrer
for itera=1:100% haga 100 iteraciones
    b=inser(rt,d,c,q);% matriz respuesta de la función inser
t=1;% contador temporal
if(opti>b(1,1)+1)%criterio de aspiración
    opti=b(1,1);%actualización mejor global
    rt=b(1,2:end-2);%nuevo vecindario
```

```
   movtabu=b(1,end-1:end);
else
while (t<length(b(:,1)))
   movtabu=b(t,end-1:end);
      if(tabu(listabu,movtabu)<1); %restricción tabú
      listabu(mod(itera,tam_lista)+1,:)=movtabu;%actualiza la lista del movimiento
tabú
      rt=b(t,2:end-2);%nuevo vecindario
      t=length(b(:,1))+5;%salir del vecindario
      else
      t=t+1;%analizar siguiente vecino
      end% fin restriction tabú
end%fin while
end% fin criterio de aspiración
rutas(itera,:)=rt;% almacene la mejor ruta de este vecindario en una matriz
costos(itera)=precioruta(rt,c); % almacene el costo de la solución en una matriz
end;
elg=find(costos==min(costos));% ubique el mejor costo
ruta_ins=double(unir(rutas(elg(1),:),c));% halle la ruta con el mejor costo
costo_ins=precioruta(ruta_ins,c)% costo de la ruta solución
```

Código Búsqueda Tabú Intercambio: Código utilizado para hacer una búsqueda tabú utilizando el vecindario tipo intercambio.

```
rt=ruta_ini;%valor inicial de la ruta
tam_lista=4
opti=costo_ini*1.10;listabu=[0,0];
for itera= 1:100
b=inter(rt,d,c,q);% vecindario tipo intercambio
t=1;
```

```
if(opti>b(1,1))%criterio de aspiración
    opti=b(1,1);%actualización mejor global
    rt=b(1,2:end-2);%nuevo vecindario
    movtabu=b(1,end-1:end)
else
while (t<length(b(:,1)))
    movtabu=b(t,end-1:end);
        if(tabu(listabu,movtabu)<1); %restricción tabú
        listabu(mod(itera,tam_lista)+1,:)=movtabu;%actualiza la lista del movimiento
tabu
        rt=b(t,2:end-2);%nuevo vecindario
        t=length(b(:,1))+5;%salir del vecindario
        else
            t=t+1;%analizar siguiente vecino
        end% fin restricción tabú
end%fin while
end% fin criterio de aspiración
rutas(itera,:)=rt;
costostabu(itera)=precioruta(rt,c);
end
elg=find(costostabu==min(costostabu));
ruta_mix=unir(rutas(elg(1),:),c);
costo_mix=precioruta(ruta_mix,c)
```

Código Búsqueda Tabú Intercambio-Inserción: Código utilizado para hacer una búsqueda tabú utilizando los vecindarios tipo intercambio e inserción.

```
load (strcat('inicial_',archivo,'.mat'))
rt=ruta_ini;%valor inicial de la ruta
tam_lista=4
```

```
opti=costo_ini*1.10;listabu=[0,0];
for itera= 1:100
if (mod(itera,2)==0)% intercala los tipos de vecindario
b=inser(rt,d,c,q);% vecindario tipo intercambio
else
b=inser(rt,d,c,q);% vecindario tipo inserción
end
t=1;
if(opti>b(1,1))%criterio de aspiración
   opti=b(1,1);%actualización mejor global
   rt=b(1,2:end-2);%nuevo vecindario
   movtabu=b(1,end-1:end)
else
while (t<length(b(:,1)))
   movtabu=b(t,end-1:end);
      if(tabu(listabu,movtabu)<1); %restricción tabú
      listabu(mod(itera,tam_lista)+1,:)=movtabu;%actualiza la lista del movimiento
tabu
      rt=b(t,2:end-2);%nuevo vecindario
      t=length(b(:,1))+5;%salir del vecindario
      else
         t=t+1;%analizar siguiente vecino
      end% fin restricción tabú
end %fin while
end % fin criterio de aspiración
rutas(itera,:)=rt;
costostabu(itera)=precioruta(rt,c);
end
elg=find(costostabu==min(costostabu));
ruta_mix=unir(rutas(elg(1),:),c);
```

```
costo_mix=precioruta(ruta_mix,c)
```

ANEXO 2

PASOS PARA EL USO DE LA HERRAMIENTA “CVRP”

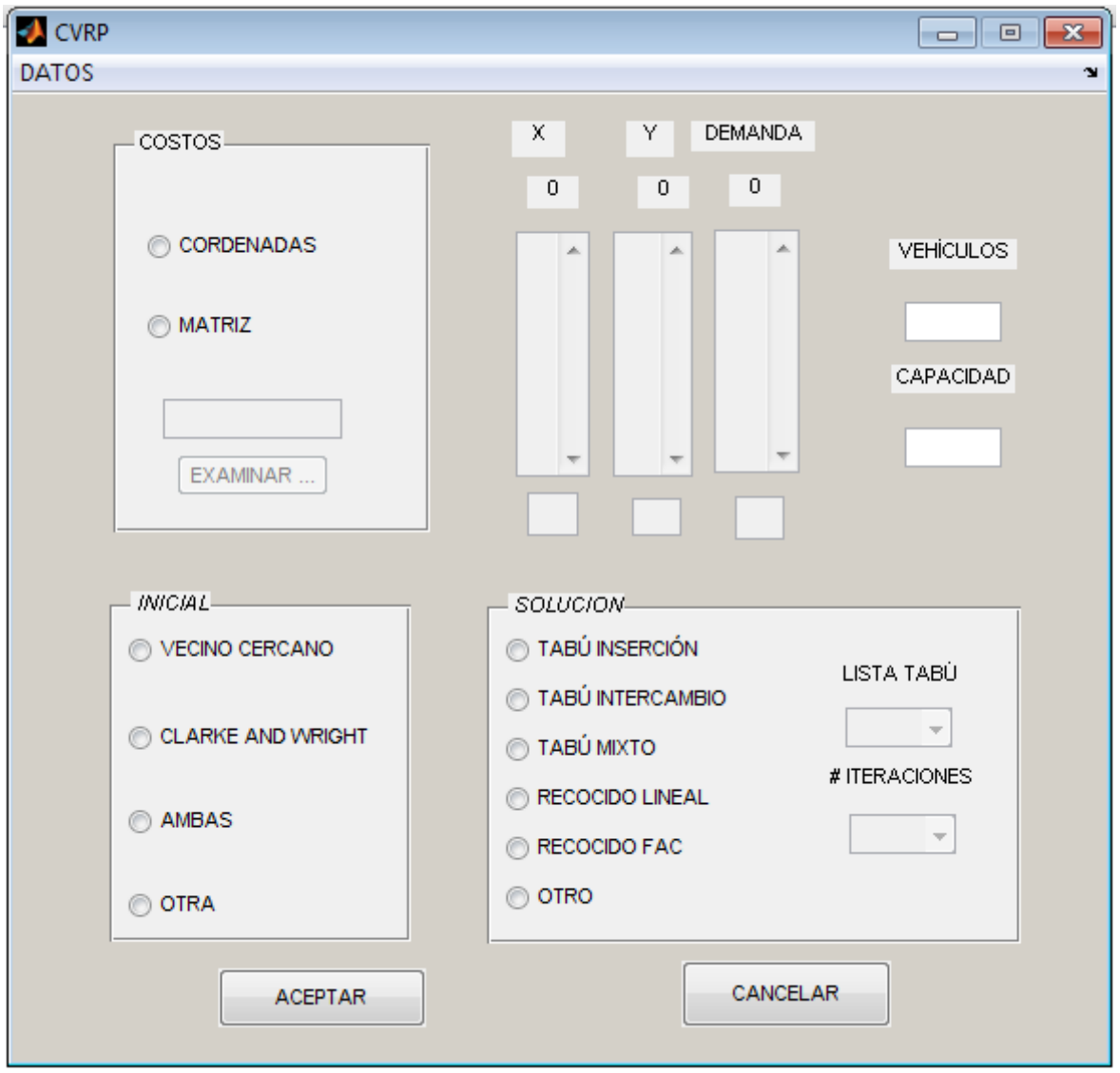
Este capítulo tiene como finalidad dar a conocer al usuario de la herramienta “CVRP”, los elementos y las condiciones que se deben tener en cuenta para el buen manejo de la aplicación y el posterior desarrollo de la misma.

1. Instalación

Para empezar a trabajar con la herramienta es necesario realizar la instalación del software MATLAB®.versión 2007 o una versión posterior. Después de realizada la instalación de MATLAB®.se debe copiar la carpeta CVRP y pegarla en la carpeta de destino que desee el usuario, la carpeta contiene las funciones creadas para poder ejecutar bien la aplicación, además del archivo con la herramienta.

2. Abrir aplicación

El programa se puede ejecutar dando doble clic sobre el archivo de la aplicación. El archivo desplegará el siguiente entorno gráfico en el cual se cargaran los datos del problema que se desea analizar.



3. Ingreso de datos

Para ingresar los datos del problema la herramienta presenta dos maneras para capturar estos datos:

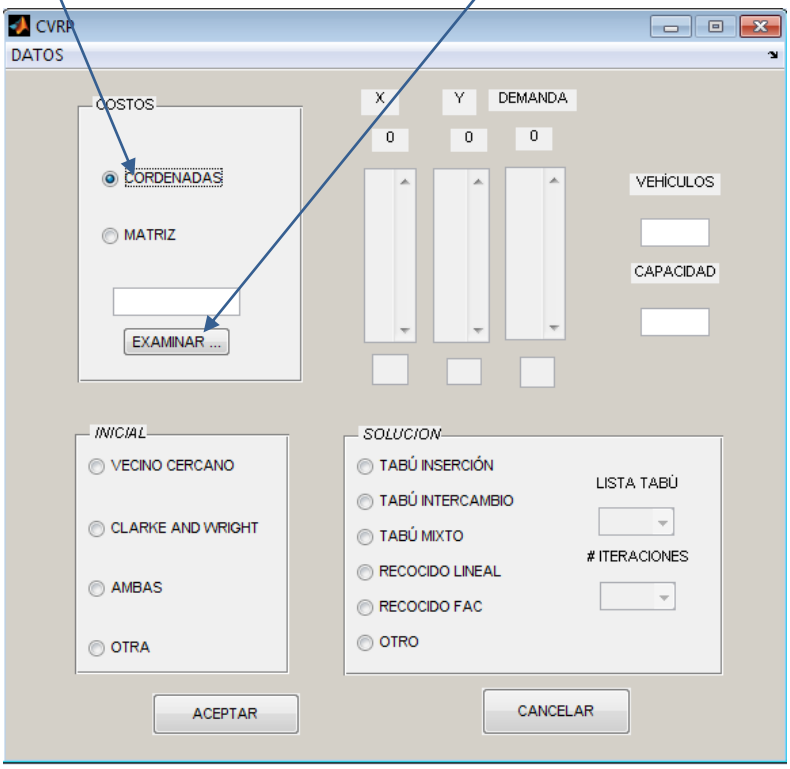
a. Cargando los vectores con coordenadas en X y en Y de los clientes, el vector de demanda asociada a cada uno de los clientes, la cantidad de vehículos y la capacidad de los mismos desde un archivo de excel. A

continuación se presenta la manera como se hace la captura de datos desde la aplicación.

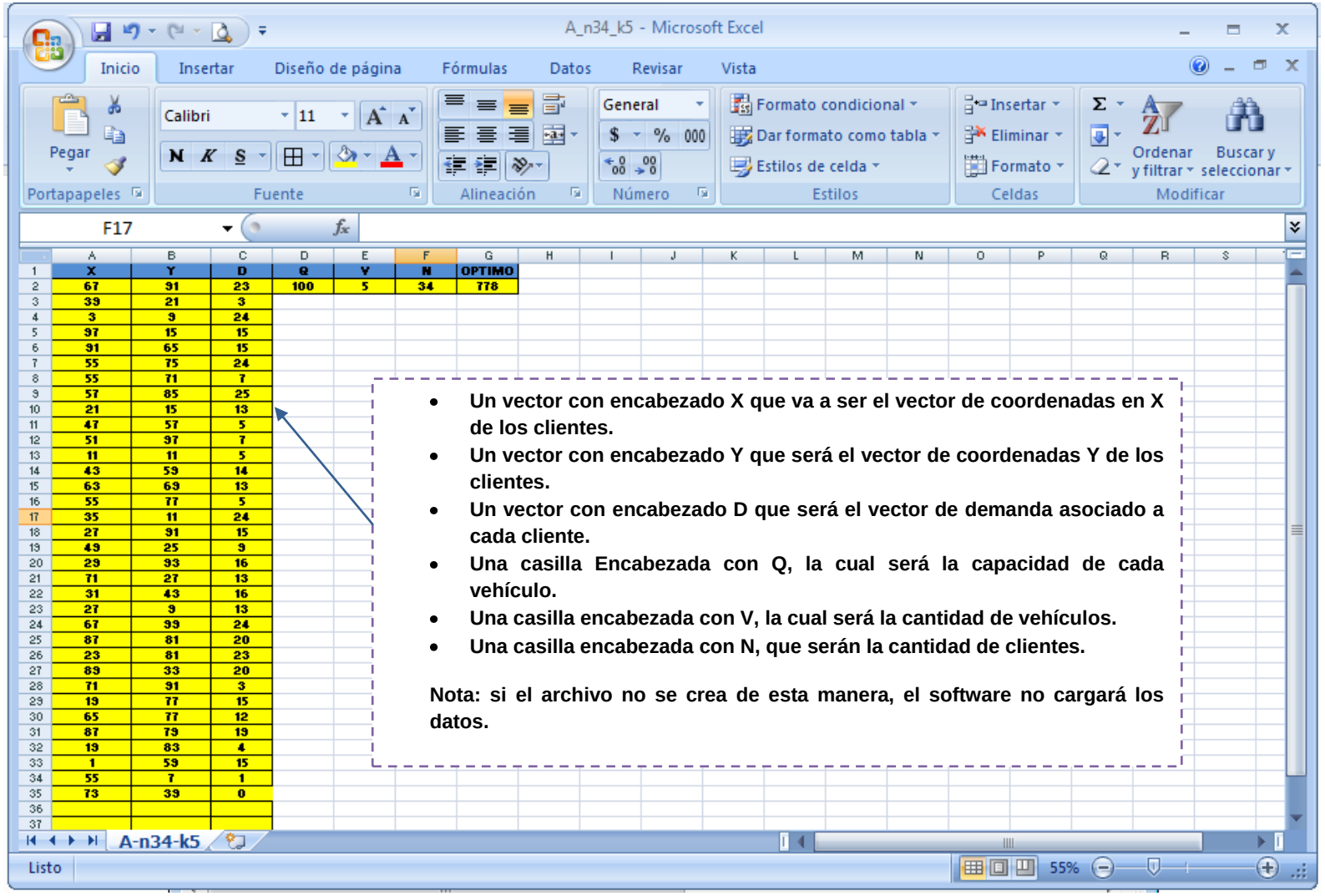
b. Cargando la matriz de costos o distancias desde un archivo de Excel junto con la demanda asociada a cada cliente, la cantidad de vehículos y la capacidad de los mismos.

1. Se debe marcar la opción coordenadas

2. Una vez elegida la forma como se capturan los datos, se activa la casilla examinar. En la casilla aparecerá una ventana de Windows donde se elegirá el archivo de Excel (.xlsx) que contiene los datos del problema.



Si se selecciona la opción COORDENADAS el archivo de Excel debe tener las siguientes características

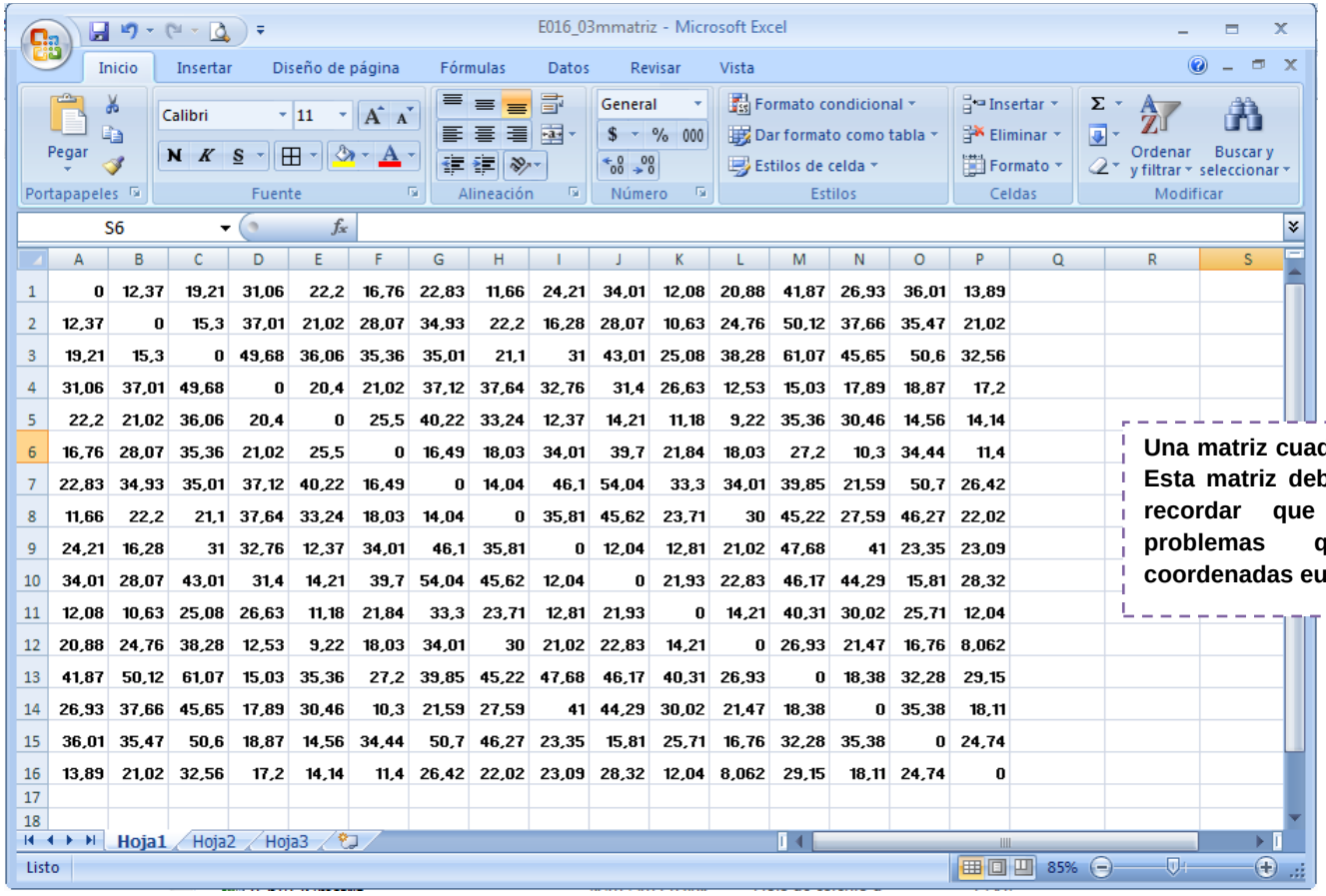


X	Y	D	Q	V	N	OPTIMO
67	91	23	100	5	34	778
39	21	3				
3	9	24				
37	15	15				
31	65	15				
55	75	24				
55	71	7				
57	85	25				
21	15	13				
47	57	5				
51	97	7				
11	11	5				
43	59	14				
63	69	13				
55	77	5				
35	11	24				
27	91	15				
49	25	9				
29	93	16				
71	27	13				
31	43	16				
27	9	13				
67	99	24				
87	81	20				
23	81	23				
89	33	20				
71	91	3				
19	77	15				
65	77	12				
87	79	19				
19	83	4				
1	59	15				
55	7	1				
73	39	0				

- **Un vector con encabezado X que va a ser el vector de coordenadas en X de los clientes.**
- **Un vector con encabezado Y que será el vector de coordenadas Y de los clientes.**
- **Un vector con encabezado D que será el vector de demanda asociado a cada cliente.**
- **Una casilla Encabezada con Q, la cual será la capacidad de cada vehículo.**
- **Una casilla encabezada con V, la cual será la cantidad de vehículos.**
- **Una casilla encabezada con N, que serán la cantidad de clientes.**

Nota: si el archivo no se crea de esta manera, el software no cargará los datos.

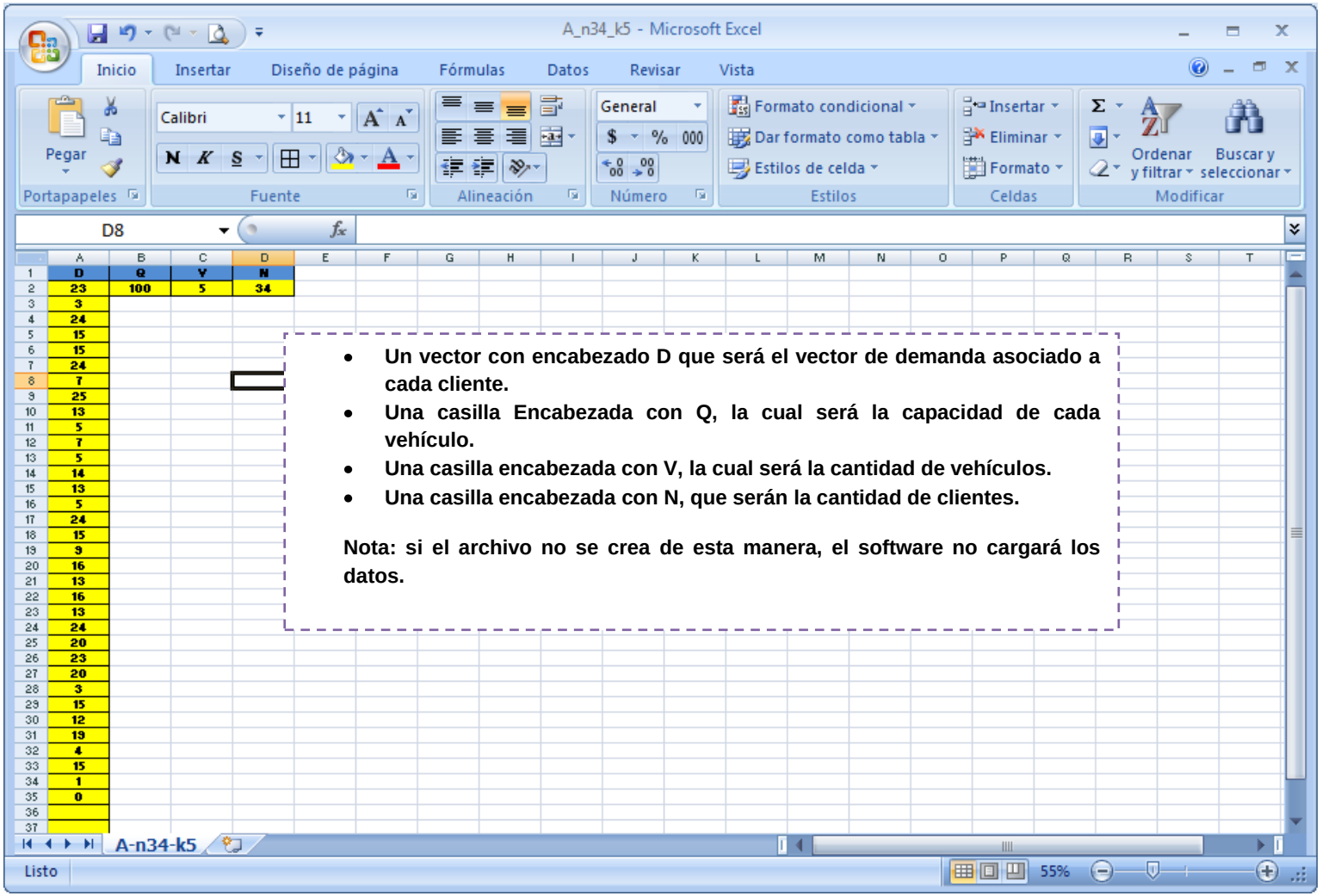
Si se selecciona la opción MATRIZ se deben crear dos archivos; uno para la matriz de coordenadas o costos y el otro para las instancias del problema (Demanda, cantidad de vehículos, capacidad de los vehículos). El archivo de Excel que contiene la matriz de costos o distancias debe tener las siguientes características.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	0	12,37	19,21	31,06	22,2	16,76	22,83	11,66	24,21	34,01	12,08	20,88	41,87	26,93	36,01	13,89
2	12,37	0	15,3	37,01	21,02	28,07	34,93	22,2	16,28	28,07	10,63	24,76	50,12	37,66	35,47	21,02
3	19,21	15,3	0	49,68	36,06	35,36	35,01	21,1	31	43,01	25,08	38,28	61,07	45,65	50,6	32,56
4	31,06	37,01	49,68	0	20,4	21,02	37,12	37,64	32,76	31,4	26,63	12,53	15,03	17,89	18,87	17,2
5	22,2	21,02	36,06	20,4	0	25,5	40,22	33,24	12,37	14,21	11,18	9,22	35,36	30,46	14,56	14,14
6	16,76	28,07	35,36	21,02	25,5	0	16,49	18,03	34,01	39,7	21,84	18,03	27,2	10,3	34,44	11,4
7	22,83	34,93	35,01	37,12	40,22	16,49	0	14,04	46,1	54,04	33,3	34,01	39,85	21,59	50,7	26,42
8	11,66	22,2	21,1	37,64	33,24	18,03	14,04	0	35,81	45,62	23,71	30	45,22	27,59	46,27	22,02
9	24,21	16,28	31	32,76	12,37	34,01	46,1	35,81	0	12,04	12,81	21,02	47,68	41	23,35	23,09
10	34,01	28,07	43,01	31,4	14,21	39,7	54,04	45,62	12,04	0	21,93	22,83	46,17	44,29	15,81	28,32
11	12,08	10,63	25,08	26,63	11,18	21,84	33,3	23,71	12,81	21,93	0	14,21	40,31	30,02	25,71	12,04
12	20,88	24,76	38,28	12,53	9,22	18,03	34,01	30	21,02	22,83	14,21	0	26,93	21,47	16,76	8,062
13	41,87	50,12	61,07	15,03	35,36	27,2	39,85	45,22	47,68	46,17	40,31	26,93	0	18,38	32,28	29,15
14	26,93	37,66	45,65	17,89	30,46	10,3	21,59	27,59	41	44,29	30,02	21,47	18,38	0	35,38	18,11
15	36,01	35,47	50,6	18,87	14,56	34,44	50,7	46,27	23,35	15,81	25,71	16,76	32,28	35,38	0	24,74
16	13,89	21,02	32,56	17,2	14,14	11,4	26,42	22,02	23,09	28,32	12,04	8,062	29,15	18,11	24,74	0

Una matriz cuadrada N*N, con diagonal 0. Esta matriz debe ser simétrica. Hay que recordar que el software resuelve problemas que contengan solo coordenadas euclidianas.

El que contiene las instancias debe tener las siguientes características.

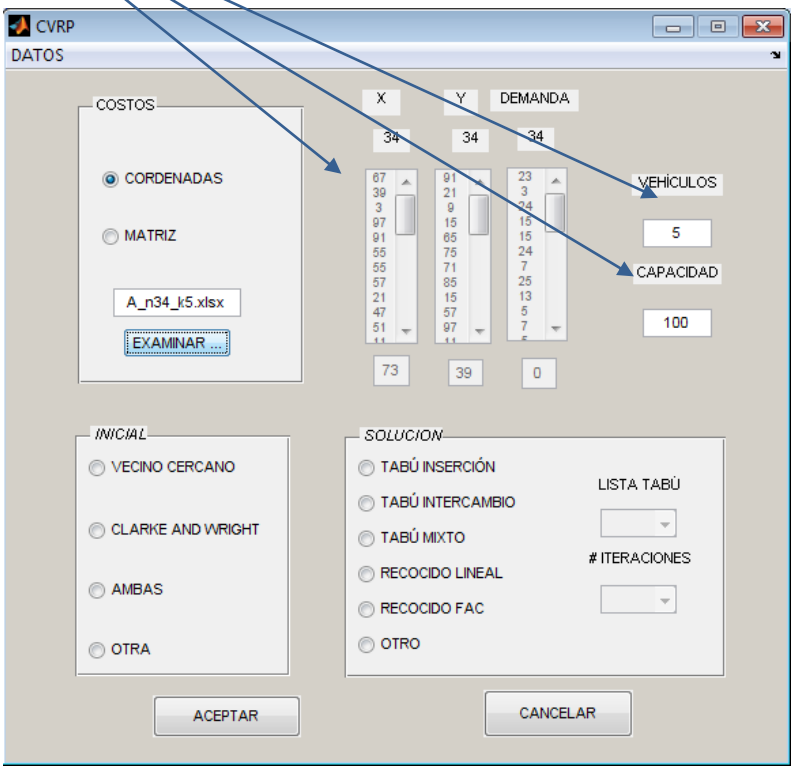


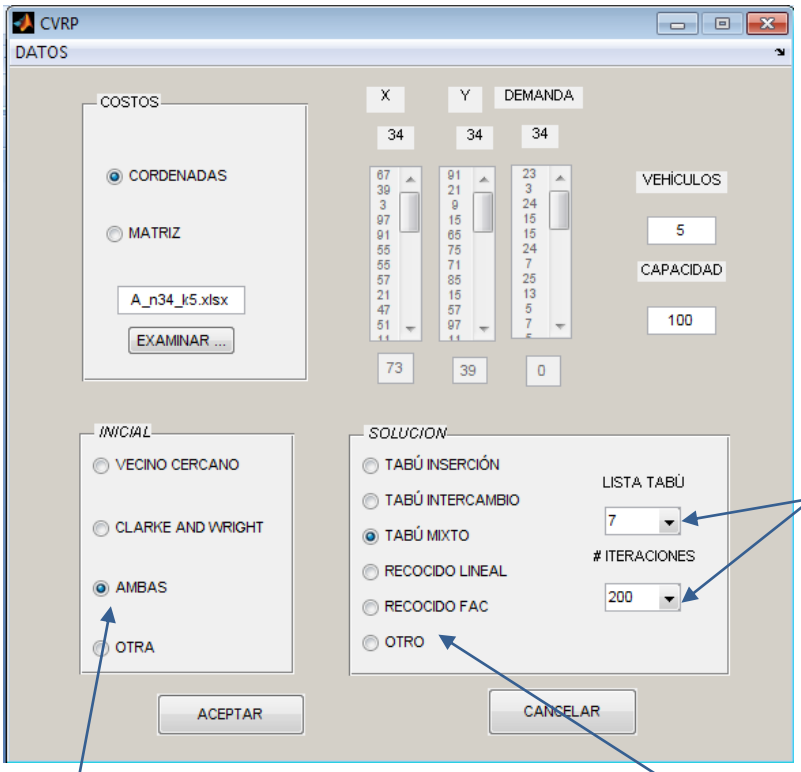
D	Q	V	N
23	100	5	34
3			
24			
15			
15			
24			
7			
25			
13			
5			
7			
5			
14			
13			
5			
24			
15			
9			
16			
13			
16			
13			
24			
20			
23			
20			
3			
15			
12			
19			
4			
15			
1			
0			

- **Un vector con encabezado D que será el vector de demanda asociado a cada cliente.**
- **Una casilla Encabezada con Q, la cual será la capacidad de cada vehículo.**
- **Una casilla encabezada con V, la cual será la cantidad de vehículos.**
- **Una casilla encabezada con N, que serán la cantidad de clientes.**

Nota: si el archivo no se crea de esta manera, el software no cargará los datos.

3. Después de elegir el archivo que contiene los datos, el software cargará las coordenadas X y Y así como el vector demanda, la cantidad de vehículos y su respectiva capacidad.





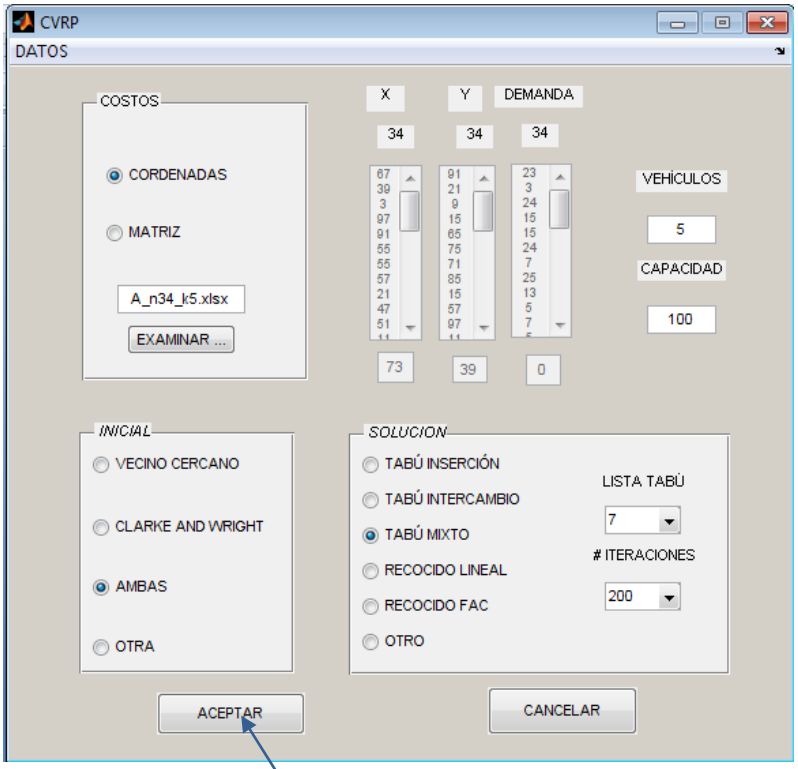
6. Si se elige alguna de la opciones que aplica la metaheuristica BT, se activan automáticamente las casillas lista tabú y # de iteraciones. Lista tabú tiene opciones para un tamaño un tamaño de lista desde 4 hasta 10. # de iteraciones tiene opciones desde 50 hasta 400 iteraciones. Si se elige algunas de las opciones de la metaheurística RS solo se activa la casilla correspondiente al número de iteraciones.

4. Hay que elegir el método heurístico para encontrar la solución inicial de la que partirán las metaheurísticas. Para ello se presentan las opciones Vecino más Cercano, Clarke and Wright, Ambas y otra. La opción ambas, calcula la solución inicial por ambas heurísticas y escoge la mejor respuesta para usarla como solución inicial. La opción otra se creó con el fin de que el usuario pueda modificar la herramienta y le pueda programar otra heurística para construir la solución inicial.

5. Después de elegir el método para encontrar la solución inicial, se debe elegir el método con el cual se dará solución al problema planteado. La aplicación da como opciones:

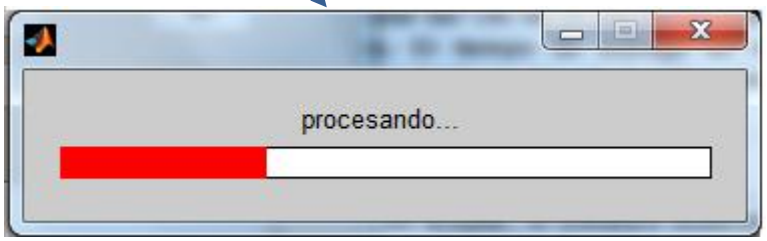
- **Tabú inserción: Aplica la metaheurística búsqueda tabú usando la metodología inserción para crear el vecindario.**
- **Tabú intercambio: Aplica la metaheurística búsqueda tabú usando la metodología intercambio para crear el vecindario.**
- **Tabú mixto: Aplica la metaheurística búsqueda tabú usando una combinación de las metodologías inserción e intercambio para crear el vecindario.**
- **Recocido Lineal: Aplica la metaheurística recocido simulado con una función de enfriamiento que decrece de forma lineal.**
- **Recocido Fac: Aplica la metaheurística recocido simulado con una función de enfriamiento que decrece de acuerdo a un factor establecido.**

4. Ejecutar



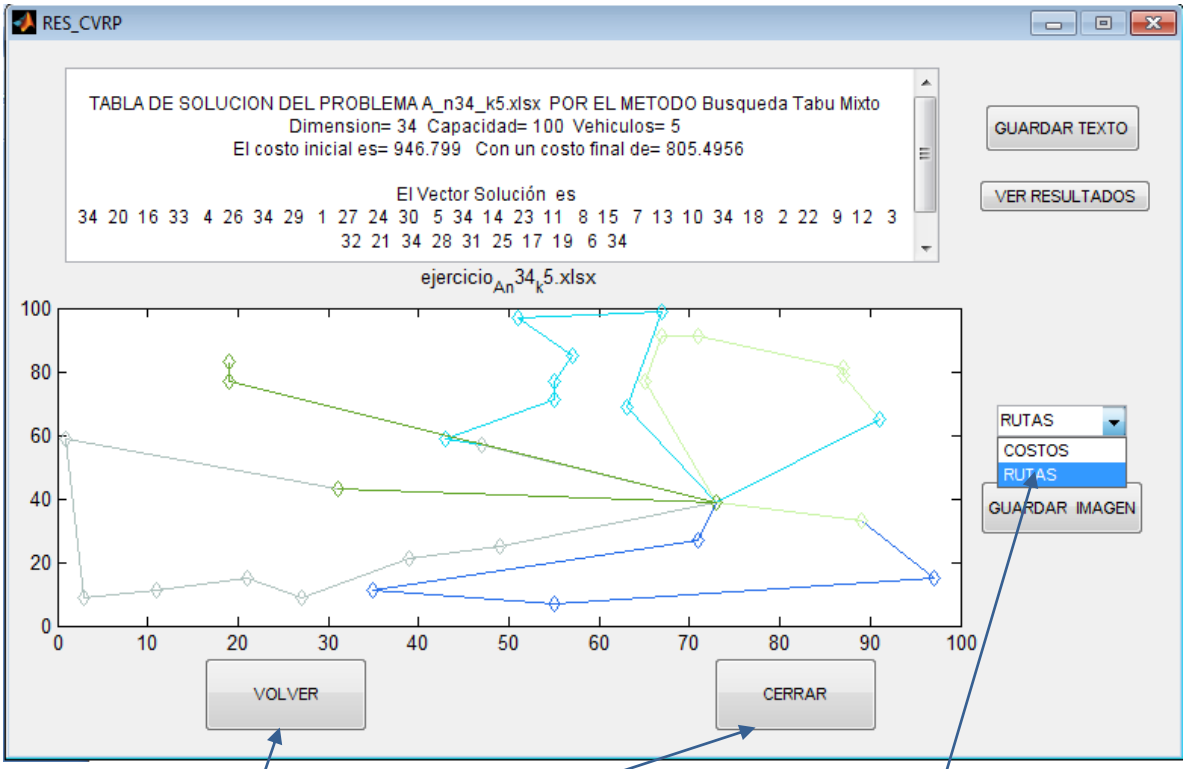
7. A continuación se debe dar clic en el vinculo aceptar para correr el algoritmo y obtener los resultados. El tiempo de entrega de resultados depende de las características del equipo donde se ejecute la aplicación, el tamaño del problema y el número de iteraciones escogido.

8. después de dar clic en aceptar, el software mostrará el siguiente recuadro que indica que el algoritmo está en proceso.



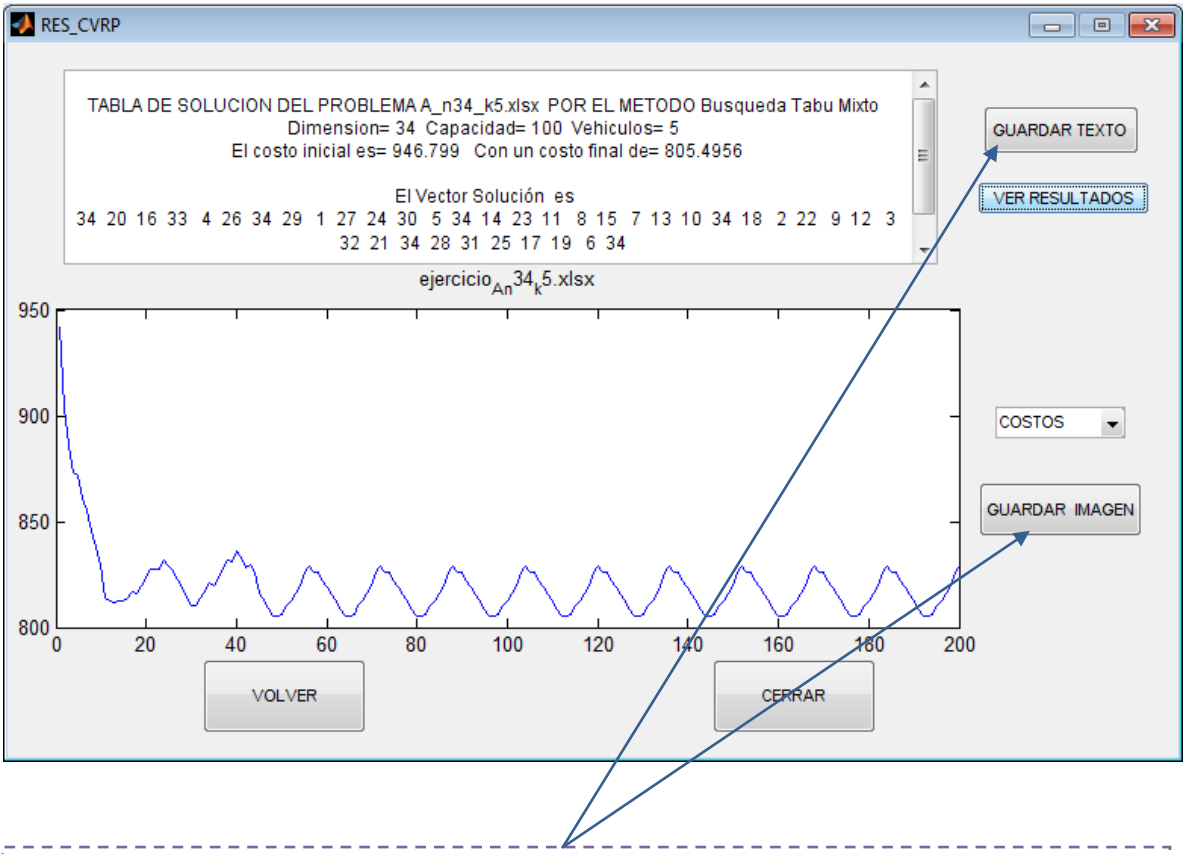
5. Guardar resultados

En cuanto el algoritmo termine de realizar las iteraciones, aparecerá el siguiente entorno gráfico que muestra de manera parcial los resultados obtenidos.



El software presenta dos recuadros (VOLVER y CERRAR). El recuadro VOLVER sirve para eliminar los datos cargados en el software y empezar a cargar nuevos datos. El recuadro CERRAR sirve para cerrar la aplicación.

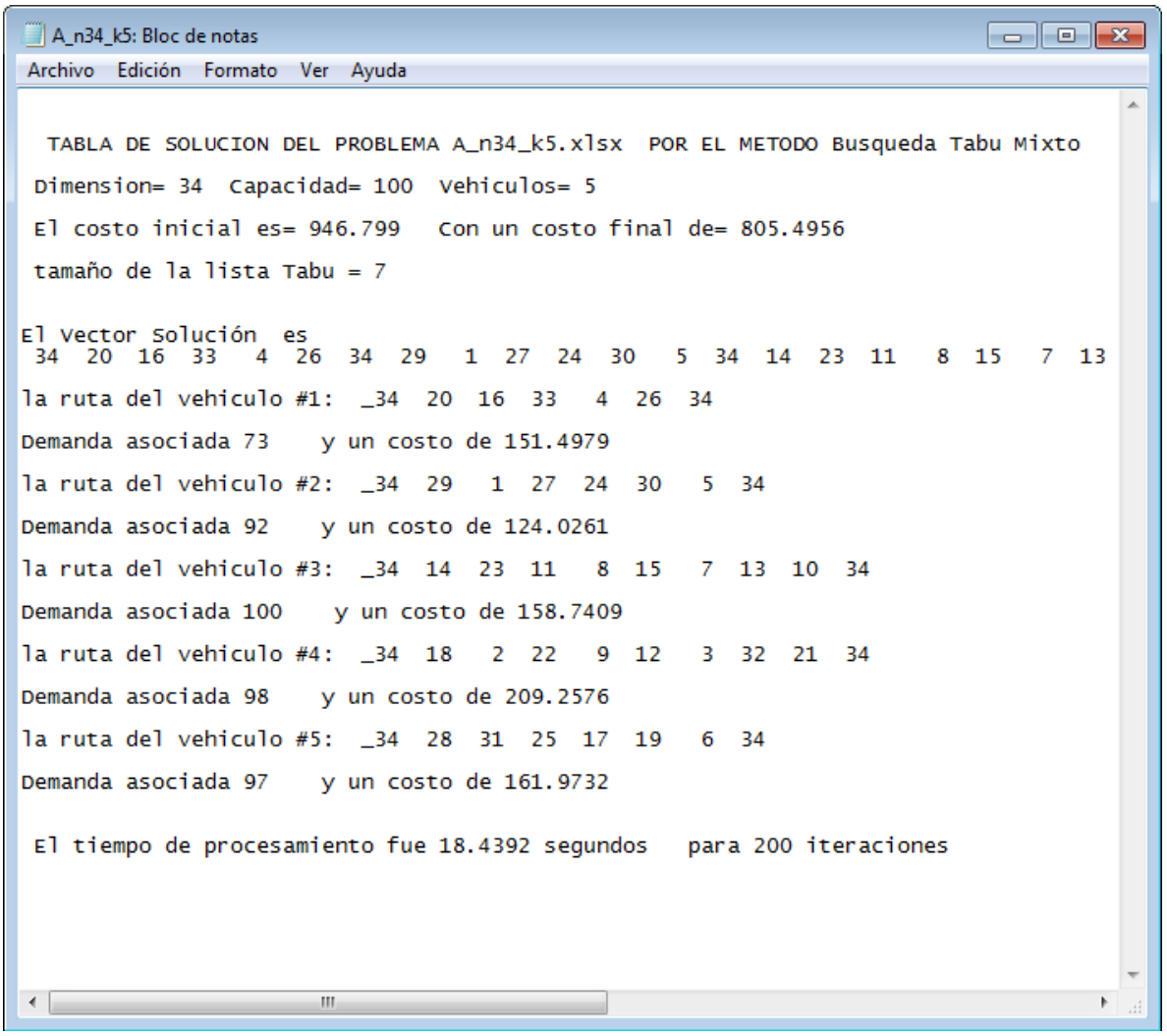
El software permite ver la gráfica de las rutas obtenidas. La opción rutas se grafica con las coordenadas reales de cada uno de los clientes solo en el caso en que los datos son ingresados como coordenadas; cuando los datos son ingresados a como matriz, el software gráfica coordenadas al azar.



9. Para guardar los resultados se muestran dos recuadros, uno sirve para guardar los resultados obtenidos en un archivo de texto y el otro sirve para guardar las gráficas. Es necesario dar clic en cada uno de estos recuadros para seleccionar la ubicación de la carpeta donde se desean guardar los resultados.

NOTA: los datos que no sean guardados se perderán y será necesario volver a correr el algoritmo para obtenerlos.

Una vez guardados los resultados obtenidos, se puede ir a la carpeta y abrir el archivo de texto que contiene la siguiente información:



Se recomienda que los resultados sean guardados en sola carpeta y con nombres diferentes con el fin de no reemplazar archivos creados anteriormente y que se facilite el acceso a los mismos.