matematica: Una publicación del ICM – ESPOL
2003, Vol. 2, No. 1 Publicado: 2003-04-30

UNA HEURÍSTICA DE TIPO TABU PARA RESOLVER EL PROBLEMA DE RUTEO DE VEHÍCULOS CON VENTANAS DE TIEMPO SUAVES

Sandoya Fernando[1]

Resumen. *En este trabajo se describe una variante del Problema de Ruteo de Vehículos (VRP) denominada el Problema de Ruteo de Vehículos con Ventanas de Tiempo Suaves (VRPSTW). Para resolver este problema se desarrolla una heurística de Búsqueda Tabú y una heurística de intercambio de aristas para la etapa de post optimización. El procedimiento es implementado en Mathematica 4 y fue probado en el conjunto de problemas de prueba de Salomón para el caso de Ventanas Duras (VRPHTW), que es un caso particular del VRPSTW. Se reportan resultados computacionales y se realizan comparaciones con los mejores resultados conocidos en la literatura.*

Palabras Claves: Optimización Combinatoria, ruteo de vehículos, metaheurísticas, búsqueda tabú, logística.

1. INTRODUCCIÓN

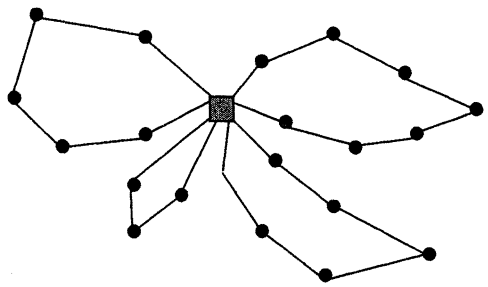
Cuando usamos un teléfono, compramos en la tienda de víveres del vecindario o en un centro comercial, leemos nuestro correo o viajamos por negocios o placer, estamos siendo beneficiarios de algún sistema que ha "ruteado" mensajes, mercancías, personas o información desde un lugar a otro. Muchos de los grandes avances de la tecnología, como el teléfono, los trenes, los aviones, la Internet representan saltos gigantescos en nuestra habilidad para transportar información, bienes o servicios. A la par, ha surgido una creciente necesidad de optimizar cada uno de los eslabones de la cadena logística, especialmente el área del transporte, sobre todo en cumplir con los compromisos de entrega a los clientes. Así, la incorporación de herramientas orientadas a automatizar y optimizar estos recursos adquiere gran importancia en el mundo actual, pues los ahorros y beneficios que se pueden lograr permiten obtener ventajas comparativas con la competencia y garantizar un alto nivel de servicio a los clientes.

Así, en general, todas las empresas deben hacer frente a estos problemas relacionados con el transporte de personas, de mercancías o de información, denominados comúnmente problemas de ruteo. Estos problemas no se restringen únicamente al sector del transporte en sí mismo, sino también se aplican a otras empresas, como por ejemplo las fábricas pueden tener que transportar partes y piezas hacia y desde los diferentes sitios de la fábrica. Estas empresas tienen que optimizar sus sistemas de transportación, pues gran cantidad de sus recursos económicos podrían estar involucrados en esta actividad; y, ya que la economía mundial se vuelve cada vez más y más globalizada, se puede asegurar que el transporte llegará a ser aún más importante en el futuro.

Por lo tanto resolver las diferentes clases de problemas de ruteo es una parte muy importante de la investigación de operaciones. Incluso la reducción de una fracción pequeña de los costos puede dar lugar a grandes ahorros y a reducir el impacto en el medio ambiente causado por la contaminación y el ruido.

El Problema de Ruteo de Vehículos (VRP: Vehicle Routing Problem) es el problema en el cual se tiene que servir a cierto número de clientes, a través de m vehículos que salen de un depósito, cada cliente tiene una demanda a ser satisfecha, y cada vehículo tiene una cierta capacidad, m generalmente es desconocido y se determina como una solución del problema. A veces también se refiere en la literatura a este problema como el Problema de Ruteo de Vehículos Capacitado (CVRP: Capacitated Vehicle Routing Problem). La suma de las demandas en una ruta no puede exceder la capacidad del vehículo asignado a esta ruta. El objetivo es reducir al mínimo la suma de las distancias de las rutas. Hay que tener presente, que el VRP no es puramente geográfico ya que aparecen restricciones debido a la demanda. El VRP es el modelo básico para una gran cantidad de problemas de ruteo de vehículos. En la Figura 1 se muestra una solución típica para el VRP.

Figura 1
Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves
Una solución para el VRP (4 rutas). El cuadrado denota el depósito



[1] *Fernando Sandoya Sánchez,* Matemático *Profesor de la Escuela Superior Politécnica del Litoral (ESPOL), (e-mail: fsandoya@espol.edu.ec).*

2. EL PROBLEMA DE RUTEO CON VENTANAS DE TIEMPO SUAVES Y DURAS

Muchos de los modelos y algoritmos diseñados para resolver problemas de ruteo de vehículos solo consideran la dimensión espacial del problema y no son capaces de satisfacer todo el conjunto de restricciones del problema práctico, por ejemplo las restricciones ligadas a la dimensión temporal del problema. Una de estas restricciones es que la llegada del vehículo a la posición de cada cliente para el inicio del servicio deberá producirse dentro de una ventana de tiempo asociada y el vehículo deberá permanecer en la ubicación del cliente durante el servicio.

Así obtenemos el Problema de Ruteo de Vehículos con Ventanas de Tiempo (VRPTW: Vehicle Routing Problem with Time Windows). Además de las restricciones de capacidad, ahora un vehículo tiene que visitar a un cliente dentro de cierto intervalo de tiempo. El vehículo puede llegar antes del inicio de la ventana de tiempo del cliente, y en ese caso deberá esperar para realizar su servicio, pero ningún cliente puede ser servido luego del fin de su ventana de tiempo.

En este trabajo se presenta una extensión del VRPTW, el Problema de Ruteo de Vehículos con Ventanas de Tiempo Suaves (VRPSTW: Vehicle Routing Problem with Soft Time Windows), en el cual se permiten llegadas con retraso; es decir, se permite dar servicio a un cliente luego del fin de su ventana de tiempo, pero bajo un cierto costo o penalidad por el retraso.

La mayoría de las investigaciones se han realizado sobre ventanas de tiempo duras, pero hay varias razones para desarrollar el VRPSTW siendo las más importantes las siguientes:

1. El modelo VRPSTW es más general y abarca al VRPTW. Además, un algoritmo para el programa con ventana de tiempo suave sería más flexible (debido a menos restricciones) y por lo tanto sería menos probable conseguir alcanzar un óptimo local. Si se requieren ventanas de tiempo duras, aumentando las penalidades apropiadamente se alienta soluciones con menos violaciones a las ventanas de tiempo.
2. En el mundo real las características de las ventanas de tiempo son normalmente suaves por naturaleza. Pues podría ser bueno establecer prioridades, atendiendo a tiempo a ciertos clientes pero retrasando un poco el servicio a otros. Los coeficientes de penalidad de tiempo pueden ajustarse para reflejar esta situación. Se ponen penalidades altas a los clientes con requerimientos de tiempo de entrega estrictos, mientras que los clientes que pueden aceptar violaciones de sus ventanas de tiempo reciben coeficientes de penalidad pequeños. Esto podría redundar en que se necesiten menos vehículos para servir a todos los clientes, y por tanto disminuiría el costo total del ruteo.
3. El modelo para el VRPSTW puede ser muy útil en un nivel de planificación estratégico en el cual se consideran las relaciones entre el tamaño, los costos de operación de la flota, y el nivel de servicio.
4. Finalmente, el modelo VRPSTW es capaz de encontrar soluciones en casos en los cuales falla una formulación de ventana de tiempo dura. Muchos problemas con ventanas de tiempo duras y flota pequeña de vehículos podrían no tener una solución factible que satisfaga a todos los clientes. En este caso, el VRPSTW produciría una solución en la cual algunos de los clientes no se servirían a tiempo. Naturalmente, la solución sería no factible para el modelo de ventana de tiempo dura pero el usuario habría, por lo menos, encontrado una solución alternativa. Esta solución proporcionaría amplia información sobre los clientes que causan la no-factibilidad en el horario de planificación; los "causantes de la dificultad" se identificarían fácilmente en las rutas resultantes.

El VRPSTW ha sido tradicionalmente muy poco estudiado en la literatura de la programación matemática; entre los pocos trabajos que se han desarrollado en esta área destacan sobre todo los trabajos de Koskosidis, Powel y Solomon [2], y el de Taillard et al. [4].

3. FORMULACION DEL VRPTW

El VRPTW es definido en la red G = (V, A), donde se tienen las siguientes notaciones:

1 $V = \{0, 1, 2, ..., n, n+1\}$ es el conjunto de nodos, que representan a cada cliente, excepto los nodos 0 y n+1 que representan al depósito.

2 $A = \{(i,j) \ / \ i, j \in V\}$ es el conjunto de aristas que conectan nodos en V.

3 Con cada cliente i está asociado un intervalo de tiempo $[a_i, b_i]$, que constituye su ventana de tiempo.

4 t_{ij} : el tiempo de viaje, para cada arco $(i, j) \in A$.

5 s_i : el tiempo de servicio para cada cliente i.
d_i : la demanda del cliente i que debe ser cubierta por algún vehículo.
C : la capacidad de los vehículos.

Normalmente, se asume que las matrices de costo y de tiempo de viaje coinciden, y que todos los vehículos dejan el depósito en el momento 0.

Es más, es fácil observar que las restricciones de ventana de tiempo inducen una orientación implícita de cada ruta aún cuando las matrices originales sean simétricas. Por consiguiente, el VRPTW debe modelarse como un problema asimétrico. Todas las rutas factibles de vehículos corresponden a caminos en G que parten del nodo 0 y terminan en el nodo n+1.

El problema consiste en buscar un conjunto de circuitos simples, o tours, que minimicen el costo total, o la longitud total de las rutas, y tal que se satisfaga lo siguiente:

1. Cada circuito parte y termina en el depósito;
2. Cada nodo es visitado por exactamente un circuito
3. La suma de las demandas de los nodos visitados en un circuito no excede la capacidad del vehículo, C;
4. Para cada cliente i, el servicio empieza dentro de la ventana de tiempo, $[a_i, b_i]$, y el vehículo da servicio durante s_i unidades de tiempo.

Se asocia una ventana de tiempo al depósito, así: $[a_0, b_0] = [a_{n+1}, b_{n+1}] = [E, L]$, donde E y L representan la primera partida posible desde el depósito y la última llegada posible al depósito, respectivamente. Se define la demanda y el tiempo de servicio como cero para estos dos nodos, es decir, $d_0 = d_{n+1} = s_0 = s_{n+1} = 0$.

Dadas las variables de decisión y temporales:

1.- Variables de flujo:

$$x_{i,j,k} = \begin{cases} 1 & \text{si el arco } (i, j) \text{ es usado por el veh. } k \\ 0 & \text{si no} \end{cases}$$

2.- Variables de tiempo: w_{ik} = inicio de servicio en i cuando está siendo servido por el vehículo k, $i \in V$, $k \in K$.

El VRPTW puede entonces ser descrito formalmente como el siguiente modelo de flujo en una red multi-producto con restricciones de ventanas de tiempo y capacidad, donde, la función objetivo de esta formulación expresa el costo total:

$$\min \quad \sum_{k \in K} \sum_{(i,j) \in A} c_{ij} x_{ijk}$$

sujeto a:

$$\sum_{k \in K} \sum_{j \in \Delta^+(i)} x_{ijk} = 1 \quad \forall i \in N \quad (3.1)$$

$$\sum_{j \in \Delta^+(0)} x_{0jk} = 1 \quad \forall k \in K, \quad (3.2)$$

$$\sum_{i \in \Delta^-(j)} x_{ijk} - \sum_{i \in \Delta^+(j)} x_{jik} = 0 \quad \forall k \in K, j \in N, \quad (3.3)$$

$$\sum_{i \in \Delta^-(n+1)} x_{i,n+1,k} = 1 \quad \forall k \in K, \quad (3.4)$$

$$x_{ijk}\left(w_{ik} + s_i + t_{ij} - w_{jk}\right) \le 0 \quad \forall k \in K, (i, j) \in A, \quad (3.5)$$

$$a_i \sum_{j \in \Delta^+(i)} x_{ijk} \le w_{ik} \le b_i \sum_{j \in \Delta^+(i)} x_{ijk} \quad \forall k \in K, i \in N, \quad (3.6)$$

$$E \le w_{ik} \le L \quad \forall k \in K, i \in \{0, n+1\}, \quad (3.7)$$

$$\sum_{i \in N} d_i \sum_{j \in \Delta^+(i)} x_{ijk} \le C \quad \forall k \in K, \quad (3.8)$$

$$x_{ijk} \ge 0 \quad \forall k \in K, (i, j) \in A, \quad (3.9)$$

$$x_{ijk} \in \{0, 1\} \quad \forall k \in K, (i, j) \in A, \quad (3.10)$$

Como $N = V \setminus \{0, n+1\}$ representa el conjunto de clientes, la restricción (3.1) restringe la asignación de cada cliente a una ruta de vehículo exactamente.

Las restricciones (3.2), (3.3), y (3.4) caracterizan el flujo en el camino que sigue el vehículo k. Para un k dado, las restricciones (3.6) obligan a que $w_{ik} = 0$ cuando el cliente i no es visitado por el vehículo k. Adicionalmente, las restricciones (3.5), (3.7) y (3.8) garantizan la factibilidad de la programación con respecto a consideraciones de tiempo y aspectos de capacidad, respectivamente. Finalmente, las restricciones (3.10) imponen el carácter binario de las variables de flujo.

4. LOS METODOS EXACTOS Y LAS METAHEURISTICAS EN EL TRATAMIENTO DEL VRPTW

Como el VRP es NP-duro también su extensión VRPTW es NP-duro y fundamentalmente más difícil que el VRP simple. Un problema de talla mediana podría involucrar fácilmente millones de variables y restricciones. Como tal, no se debe esperar obtener soluciones óptimas dentro de un tiempo polinomial; y hasta ahora las heurísticas han ofrecido los resultados más prometedores resolviendo los problemas de tamaño real. Los algoritmos heurísticos han tenido bastante éxito tratando problemas de gran escala con requerimientos de cálculo mínimos, pero estos pueden padecer limitaciones serias concernientes a la calidad de las soluciones. Lo que siempre se busca es un equilibrio entre la calidad de la solución y los requerimientos de tiempo computacional.

Las meta-heurísticas son la fuente de la mayoría de los trabajos recientes en métodos de aproximación para el VRPTW, e incluyen los métodos: recocido simulado, búsqueda tabú, y algoritmos evolutivos tales como la búsqueda genética. A diferencia de las heurísticas de búsqueda local que terminan una vez que es alcanzado un óptimo local, estos métodos exploran un gran subconjunto del espacio solución en la espera de lograr una solución cercana a la óptima. Mientras que recocido simulado depende en su mayor parte de pasos aleatorios para escapar de un óptimo local, la búsqueda tabú usa la memoria de corto y largo plazo para evitar entrar en un ciclo y para orientar la búsqueda hacia regiones inexploradas del espacio solución. Los algoritmos de evolución se desarrollan desde una analogía con el proceso de evolución natural y consisten de seleccionar, re-combinar y mutar iterativamente soluciones codificadas para obtener individuos superiores.

5. BÚSQUEDA TABU

La Búsqueda Tabú (TS: Tabu Search) es un procedimiento meta-heurístico utilizado para guiar un algoritmo heurístico que explora el espacio de soluciones sin entramparse en óptimos locales, Los orígenes de la búsqueda tabú se extienden hasta la década de 1970. En 1986, *Fred Glover* lo desarrolló en su forma actual, aunque últimamente también han aparecido híbridos de búsqueda tabú con otras heurísticas o procedimientos algorítmicos. Los éxitos que ha tenido la búsqueda tabú en las aplicaciones prácticas han promovido la investigación hacia formas de explotar con mayor intensidad sus ideas subyacentes. Al mismo tiempo, aún falta por explorar muchas facetas de estas ideas. Temas como la identificación de las mejores combinaciones de memoria de corto y largo plazo, y los mejores equilibrios de estrategias de intensificación y diversificación, sin duda alguna abrigan descubrimientos importantes para desarrollar métodos de solución más poderosos en el futuro.

La búsqueda tabú se distingue de otros métodos en que incorpora una *memoria adaptativa* y una *exploración sensible*. En el lenguaje común y corriente lo tabú se entiende como un tipo de prohibición con connotaciones culturales, es decir, algo que está sujeto a la influencia de la historia y el contexto, y que sin embargo puede superarse bajo condiciones apropiadas. En el método de búsqueda tabú, durante la exploración del espacio de soluciones factibles, se consideran prohibidas ciertas soluciones (temporalmente), como veremos más adelante.

El uso de la memoria adaptativa hace a este procedimiento superior a otras metaheurísticas como son los métodos recocido simulado y genéticos, que podemos considerarlos métodos "sin memoria", y con diseños de "memoria rígida", como aquellos que utilizan la técnica de ramificación y acotamiento.

La exploración sensible aparece como un concepto importante en búsqueda tabú, gracias a la suposición, bastante lógica, de que una mala elección estratégica puede producir más información que una buena elección al azar. En un sistema que emplea memoria, una mala elección basada en una estrategia puede dar claves útiles acerca de cómo podrían hacerse modificaciones provechosas a la estrategia.

En términos generales la búsqueda tabú funciona de la siguiente manera. Dado el problema de programación matemática general:

$$\max/\min\ g(x)$$
s.a.r.:
$$x \in X$$

Donde $g(x)$ es la función objetivo a ser optimizada (maximizada o minimizada) y X es el conjunto de soluciones factibles.

A cada solución factible $x \in X$ se le asocia una vecindad $N(x) \subset X$, tal que cada solución $x' \in N(x)$ se puede alcanzar desde x mediante una operación llamada movimiento. La búsqueda tabú empieza de la misma manera que cualquier procedimiento de búsqueda local, yendo iterativamente de un punto a otro hasta satisfacer un criterio de parada, pero supera a los métodos de búsqueda local empleando una estrategia de modificación de $N(x)$ a medida que la búsqueda progresa, reemplazándola por otra vecindad $N^*(x)$. Una clave de la búsqueda tabú es cómo se determina el conjunto $N^*(x)$. La forma en que las soluciones son admitidas en $N^*(x)$ por las estructuras de memoria se determinan de varias maneras, una de ellas es identificando soluciones encontradas sobre un horizonte especificado, y determinando de acuerdo a cierta regla a cuales se les prohíbe pertenecer a $N^*(x)$ clasificándolas como soluciones tabú.

En resumen, la idea básica consiste en permitir el paso de una solución a otra, a través de "movimiento", aún cuando temporalmente ésta última empeore el resultado. Para evitar el posible ciclado se introduce la lista tabú. En ella se guarda durante cierto tiempo, un atributo que permita identificar la solución o el movimiento realizado. Así, todo movimiento que tenga un atributo en la lista tabú se considera prohibido, y no se permite su realización. El proceso mediante el cual las soluciones adquieren la calidad de tabú tiene varias facetas, diseñadas para promover un examen agresivo y guiado de nuevos puntos. Una manera útil de visualizar e implementar este proceso es el reemplazo de la evaluación original de soluciones mediante evaluaciones tabú, las cuales introducen penalizaciones para desalentar en forma significativa la elección de soluciones tabú. Además, las evaluaciones tabú incluyen también periódicamente incentivos para estimular la elección de otro tipo de soluciones, como resultado de niveles de aspiración e influencias a largo plazo.

6. PROCEDIMIENTO EMPLEADO PARA RESOLVER EL VRPSTW

La metodología propuesta se incluye algunas componentes diferentes, tales como un procedimiento de inicialización, una etapa de optimización intra-ruta, una memoria adaptativa, la búsqueda Tabú como tal, y luego un proceso de post optimización,

En la etapa de ***inicialización*** se construye una solución inicial que contiene ***p*** rutas diferentes usando una heurística de inserción del vecino mas cercano. Luego se aplica una heurística de

intercambio de aristas a cada solución y se guardan las rutas resultantes en la memoria adaptativa. Luego actúa el procedimiento de búsqueda tabú de la siguiente manera:

Mientras no se verifique el criterio de parada:

1. Se construye una solución inicial a partir de las rutas halladas en la memoria adaptativa, y se define esta solución como la solución actual
2. Aplicar una heurística de λ-intercambio para generar la vecindad de la solución actual, donde actúa el procedimiento de búsqueda Tabú, la exploración de la vecindad se realiza por el criterio Primero-mejor.
3. El conjunto de las nuevas rutas halladas en la búsqueda Tabú se define como la nueva solución actual.
4. Se guardan estas rutas de la solución actual en la memoria adaptativa

Por último se aplica un procedimiento de post-optimización a cada ruta individual de la mejor solución.

A continuación se describe con mas detalle las componentes del procedimiento.

INICIALIZACIÓN: Para llenar la memoria adaptativa con una solución inicial se usa una heurística de inserción del vecino mas cercano para construir ***p*** rutas iniciales. Para esto, se inicializa las rutas seleccionando aleatoriamente ***p*** clientes semilla. Por lo tanto, cada ruta inicial solo sirve a un cliente. Se inicia una nueva ruta seleccionando un cliente semilla y luego insertando secuencialmente los clientes dentro de la ruta que está siendo construida, hasta que o bien la capacidad del vehículo es excedida o no es factible insertar otro cliente dentro de esta ruta en términos de tiempo ya que no se respeta la restricción de la ventana de tiempo del depósito (recuérdese que la ventana de tiempo del depósito no puede ser violada). El cliente seleccionado para la inserción en cada paso es el más cercano al cliente que se insertó en el paso inmediatamente anterior. El proceso de selección es repetido hasta que no pueda ser insertado ningún cliente adicional; así, ha sido creada una nueva ruta. Se repite todo el procedimiento para todos los clientes semilla hasta que todos los clientes del problema hayan sido ruteados. El número actual de vehículos empleados, ***p***, provee una cota superior para el número de rutas de la solución optimal.

OPTIMIZACION INTRA-RUTA: La heurística de construcción simple planteada en la etapa de inicialización no produce soluciones de alta calidad, como se puede ver en los ejemplos numéricos. Por consiguiente, una heurística de optimización intra-ruta es aplicada a cada ruta antes de que las rutas resultantes sean guardadas en la memoria adaptativa. Esta fase reduce significativamente el costo total del ruteo en la función objetivo, lo cual es explotado después por la búsqueda Tabú. Utilizar un procedimiento de optimización intra-ruta significa, en definitiva, resolver ***m*** problemas del tipo TSP (Travel Salesman Problem), problema para el cual se han descrito en la literatura numerosos procedimientos heurísticos efectivos para su tratamiento.

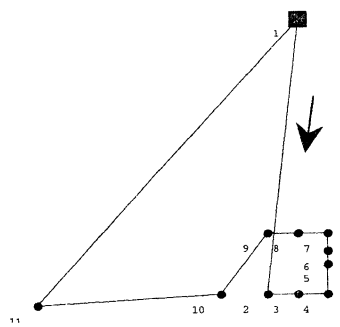
La heurística empleada en este trabajo para la optimización intra-ruta es la heurística 2-opt, la cual se aplica a cada ruta individual de la solución inicial. Según Osman[1] esta es una de las heurísticas de mejor desempeño para el tratamiento del TSP, además en el peor de los casos su tiempo de corrida es $O\left((n/p)^2\right)$. En la heurística 2-opt una ruta es mejorada con el siguiente procedimiento: En cada paso se borran dos arcos cualquiera no consecutivos, se da la "vuelta" a uno de los caminos resultantes y luego se los reconecta, evaluándose la función objetivo para esta nueva ruta, si la función objetivo es mejor que la de la ruta actual, esta es ahora la nueva ruta actual. Se repite este procedimiento para la ruta actual hasta que no pueda obtenerse ninguna mejora adicional. En el siguiente ejemplo se observa el resultado de la acción de esta heurística en nuestra implementación en una ruta individual para el problema de prueba RC204 con 100 clientes, C=100 y α_i=1:

Figura 2

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Ruta inicial (1 2 3 4 5 6 7 8 9 10 11 1)

Función objetivo: 135.63



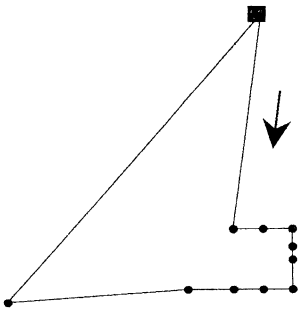
[1] Metastrategy Simulated Annealing and Tabu search algorithms for the vehicle routing problem. Osman H., Annals of Operations Research, 1993

Figura 3

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Ruta 2-Opt (1 9 8 7 6 5 4 3 2 10 11 1)

Función objetivo: 118.203



BÚSQUEDA TABÚ: Luego de realizado el procedimiento de optimización Intra-ruta, se aplica la heurística de búsqueda Tabú mientras no se verifique el criterio de parada. La efectividad de cualquier método iterativo de búsqueda local está determinada por la eficiencia del mecanismo de generación de la vecindad y la manera en que es realizada la búsqueda en la vecindad. En nuestra implementación utilizamos el mecanismo de generación 1-intercambio y que se basa en el intercambio de clientes entre los conjuntos de rutas de vehículos. Básicamente puede describirse como: dada una solución factible representada por $S = \{R_1, ..., R_p, ..., R_q, ..., R_k\}$, donde R_p es el conjunto de clientes servidos por el vehículo P. Un 1-intercambio entre un par de rutas R_p y R_q es el reemplazo de un subconjunto $S_1 \subseteq R_p$ de tamaño $|S_1| \leq 1$ por otro subconjunto $S_2 \subseteq R_q$ de tamaño $|S_2| \leq 1$, para conseguir los dos nuevos conjuntos de rutas $R'_p = (R_p - S_1) \cup S_2$ y $R'_q = (R_q - S_2) \cup S_1$ y una nueva solución vecina $S' = \{R_1, ..., R'_p, ..., R'_q, ..., R_k\}$. La vecindad $N_2(S)$ de una solución dada S es el conjunto de todos los vecinos $\{S'\}$ generado por el método de 1-intercambio. Luego de que es generada una solución, se requiere un criterio para aceptar o rechazar un movimiento. Nosotros utilizamos la estrategia Primer-Mejor, la cual seleccionará la primera solución en S' en $N_2(S)$ en la vecindad de S que resulte en un decrecimiento en el costo. El tiempo de permanencia de los movimientos Tabú es administrado por el Tamaño de la Lista Tabú (TLS). Nosotros fijamos TLS en 10 en base a nuestra experiencia computacional. La Lista Tabú también emplea una medida de frecuencia que cuenta el número de instantes en los cuales las soluciones y movimientos tabú han sido determinados por la búsqueda. Las frecuencias altas indican que la búsqueda ha caído en un óptimo local y por lo tanto debe ser tomada cierta acción.

POST-PROCESAMIENTO DE CADA RUTA INDIVIDUAL: Por último se aplica un procedimiento de post-optimización a cada ruta individual de la mejor solución, el procedimiento elegido es nuevamente el 2-Opt.

IMPLEMENTACIÓN DE LA LISTA TABÚ:

La identificación de las soluciones tabú permite que soluciones que han sido recientemente visitadas no puedan ser visitadas nuevamente, evitando así el ciclaje del procedimiento, pero guardar las soluciones tabú en una lista requiere de un gran esfuerzo computacional y capacidad de memoria de la máquina, pues recordemos que cada solución está constituida por un conjunto de rutas y su valor objetivo, cada una de las cuales está determinada a su vez por un conjunto de nodos y una matriz de adyacencia que define el conjunto de aristas. Esto puede ser evitado en gran medida con el uso de alguna estructura de datos especial, como la planteada por Osman para el tratamiento del VRP general[2] que fue usada en nuestra implementación y que reduce de gran manera el esfuerzo computacional requerido, esta consiste en lo siguiente:

Se construye una tabla, que denominaremos TTABU, como un arreglo de $n \times p$, donde n es el número de clientes y p es el número de rutas. TTABU(i,j) registra el número de iteración en la cual el cliente i fue removido de la ruta j para insertarlo en la otra ruta de acuerdo al criterio indicado antes. De esta manera los movimientos de clientes que se hacen durante el procedimiento son rápidamente representados y pueden ser fácilmente chequeados. Inicialmente, la matriz TTABU se inicializa con valores negativos grandes para impedir que se identifique falsamente a los movimientos como tabú durante las iteraciones iniciales. Con esta estructura de datos para el registro de las soluciones visitadas recientemente también se vuelve fácil el chequeo de sí una solución debe abandonar o no la lista tabú en una iteración dada. En efecto, consideremos que Ψ es el número de iteraciones durante las cuales una solución que ingresa a la lista debe permanecer en ella, a menos que se cumpla el criterio de aspiración, este número Ψ se suele denominar tamaño de la lista tabú, y depende de las características del problema y de la estrategia de selección usada, su valor en general debe ser determinado empíricamente a través de la experimentación sobre los problemas test. Una forma simple de chequear si la solución actual es tabú es la siguiente:

Si k-TTABU(i,p) $< \Psi$, entonces tal movimiento sigue siendo tabú y permanece en la lista, a menos que con esa solución se satisface el criterio de

[2] Metastrategy Simulated Annealing and Tabu search algorithms for the vehicle routing problem. Osman H., Annals of Operations Research, 1993

aspiración. Así el status tabú puede ser fácilmente chequeado con el uso de una operación y una comparación.

RESULTADOS COMPUTACIONALES:

El método fue probado en el conjunto de Solomon de problemas de prueba VRPHTWs. En lo que sigue, primeramente son introducidos estos problemas de prueba. Luego, se reportan las soluciones producidas.

APLICACION

Para probar la eficiencia de los algoritmos heurísticos desarrollados para resolver problemas de ruteo de vehículos con ventanas de tiempo algunos investigadores expertos en el tema han desarrollado numerosos problemas de prueba, los mismos que tratan de representar múltiples situaciones con distintos grados de complejidad.

En particular, este algoritmo fue probado en problemas test estándar proporcionados por Solomon. En estos problemas euclideanos, los tiempos de viaje son equivalentes a las distancias euclideanas correspondientes. Las ubicaciones de los clientes están distribuidas dentro de un cuadrado $[0, 100]^2$. Se definen seis conjuntos diferentes de problemas, a saber C1, C2, R1, R2, RC1, y RC2. En los problemas de tipo R los clientes están uniformemente distribuidos, y en los problemas de tipo C están amontonados en grupos o *clusters*. Estas características son mezcladas en los problemas de tipo RC. Además, la ventana de tiempo es angosta en el depósito central de los problemas de tipo 1, de modo que solamente unos pocos clientes pueden ser servidos en cada ruta. Inversamente, esta ventana de tiempo es amplia para los problemas del tipo 2, de modo que muchos clientes pueden ser servidos en la misma ruta. Finalmente, se tiene un tiempo de servicio fijo en cada ubicación de cliente (es decir, el tiempo para descargar las mercancías).

Este tiempo de servicio es de 10 unidades de tiempo para clientes de los problemas de tipo R y RC, y 90 unidades de tiempo por cliente para los problemas de tipo C.

RESULTADOS NUMÉRICOS:

En general, es muy difícil comparar diferentes metodologías para resolver problemas con respecto a sus requerimientos computacionales, a causa del hardware específico, las estructuras de datos y el lenguaje de computación usado en la implementación. Además, hay que recalcar que el problema que se quiere resolver lleva una gran carga computacional adicional asociada con las ventanas de tiempo suaves, lo cual hace que no sea muy efectiva la comparación con los resultados reportados en la literatura, los mismos que se limitan al caso de ventanas de tiempo duras, y que han sido encontrados con algoritmos desarrollados exclusivamente para ese propósito.

En esta parte se incluyen algunos resultados numéricos de la implementación para comprobar su operatividad, rendimiento y eficacia así como sus desventajas y limitaciones. La implementación del algoritmo se ha realizado en Mathematica versión 4.0.2, que resultó la plataforma ideal para implementar este tipo de aplicaciones informáticas, concretamente con Mathematica se pueden procesar grandes volúmenes de datos numéricos, incluye cajas de herramientas para manejar cálculos simbólicos, y sobretodo contiene el paquete Combinatorica que incorpora funciones para construir grafos y otros objetos de la teoría de grafos y matemáticas discretas; finalmente, permite interactuar con otros lenguajes de programación, especialmente con el lenguaje de programación "C.

Para la implementación realizada en este trabajo muchas de las pruebas se desarrollaron en una computadora portátil Pentium IV de 2.40 GHz. Con 512 MB de RAM. Por la naturaleza heurística de los algoritmos implementados, la duración de las pruebas es también variable. El número de iteraciones que se requieren realizar en la parte TABÚ puede ser muy grande para obtener buenas soluciones, y por tanto podrían necesitar de un gran tiempo de ejecución, algunos autores que han desarrollado algoritmos basados en búsqueda tabú parecidos al nuestro señalan que se tiene que experimentar por lo menos 500 iteraciones antes de terminar[3].

EJEMPLO DE UNA CORRIDA:

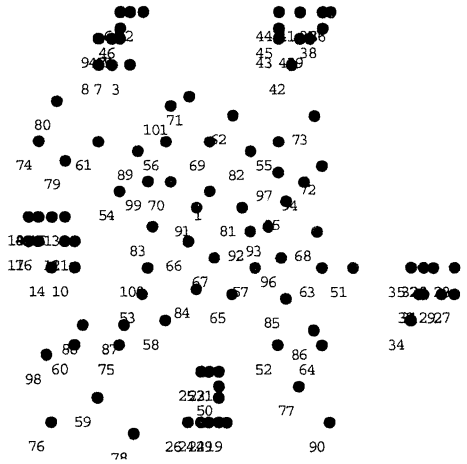
A continuación se presenta el resultado de una corrida para el problema de prueba RC204 con 100 clientes y vehículos de capacidad 1000:

PARÁMETROS:

Número de clientes semilla: 4
Coeficiente de penalidad: 1
Longitud de la lista tabú: 10
Número de iteraciones: 40

[3] " Heuristics for Vehicle Routing Problem with Time Windows", Zhu K., Tan C., National University of Singapore, 1999

Figura 4
Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves
Localización de los clientes en el problema de prueba RC204



En lo que sigue se presentan los resultados de cada etapa del procedimiento planteado; esto es, la solución inicial hallada por el método de inserción del vecino mas cercano, luego lo obtenido con el procedimiento de optimización intra-ruta, seguido de la solución obtenida por el método tabú, y por último la solución final reportada por el procedimiento de post-optimización.

Figura 5ª
Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves
Ubicación de los clientes semilla

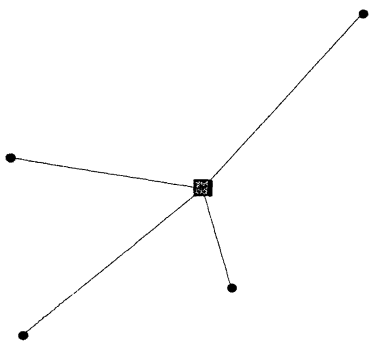


Figura 5b
Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves
Los clientes ruteados con el método de inserción del vecino mas cercano

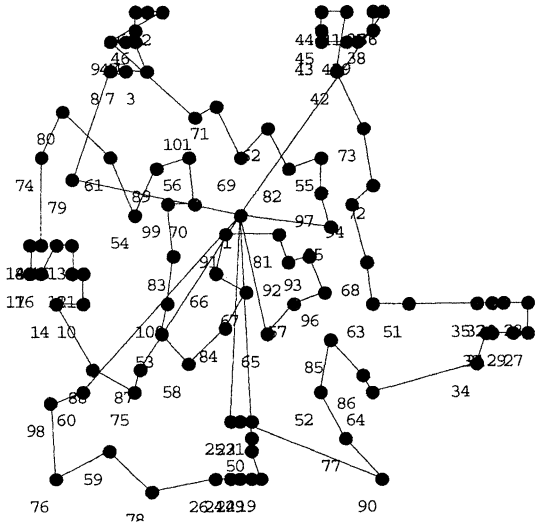


Figura 6

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Las rutas individuales creadas con el método de inserción del vecino mas cercano

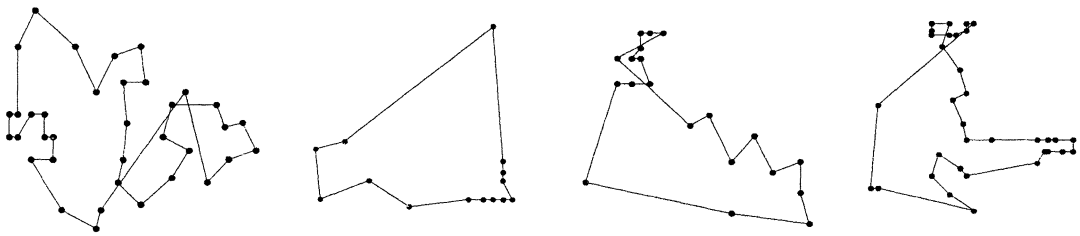


Tabla 1

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Resultados en la ruta inicial

RUTA	Recorrido	Longitud	Penalidad	Costo tot
1	0,64,56,95,92,91,80,90,65,66,83,57,52, 99,82,98,69,55,88,53,60,79,73,47,17,16, 15,14,12,11,10,9,13,87,74,86,0	288.571	2300.66	2589.23
2	0,59,97,75,58,77,25,23,21,48,18,19,49, 20,0	150.208	506.977	657.184
3	0,78,7,6,2,4,46,45,5,3,1,8,100,70,68,61, 81,54,96,94,0	175.483	231.009	406.492
4	0,35,36,37,38,39,42,44,43,40,41,72,71, 93,67,62,50,34,31,29,27,26,28,30,32,33, 63,85,84,51,76,89,22,24,0	264.334	15877.5	16141.8
TOTAL		**878.596**	**18916.146**	**19794.74**

Tabla II

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Resultados luego de la etapa de optimización intra-ruta

RUTA	Recorrido	Longitud	Penalidad	Costo tot
1	0,64,56,95,92,91,80,90,65,66,83,57,52, 99,82,98,69,55,88,53,60,79,73,47,17,16, 15,14,12,11,10,9,13,87,74,86,0	288.571	2300.66	2589.23
2	0,59,97,75,58,77,25,23,21,48,18,19,49, 20,0	150.208	506.977	657.184
3	0,78,7,6,2,4,46,45,5,3,1,8,100,70,68,61, 81,54,96,94,0	175.483	231.009	406.492
4	0,35,36,37,38,39,42,44,43,40,41,72,71, 93,67,62,50,34,31,29,27,26,28,30,32,33, 63,85,84,51,76,89,22,24,0	264.334	15877.5	16141.8
TOTAL		**878.596**	**18916.146**	**19794.74**

Tabla III

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Resultados luego de las iteraciones tabú

RUTA	Recorrido	Longitud	Penalidad	Costo total
1	0,64,56,95,92,91,80,90,65,66,83,57,52, 99,82,98,69,9,10,11,12,14,15,16,17,47, 73,79,60,53,88,55,13,87,74,86,0	337.882	763.061	1100.94
2	0,20,49,19,18,48,21,23,25,77,58,75,97, 59,0	150.208	0	150.208
3	0,78,7,6,8,1,3,5,45,46,4,2,100,70,68,61, 81,54,96,94,0	168.74	0	168.74
4	0,85,63,33,32,30,28,26,27,29,31,34,50, 62,67,93,71,72,41,40,43,44,42,39,38,37, 36,35,84,51,76,89,22,24,0	339.056	300.703	639.759
TOTAL		**995.886**	**1063.764**	**2059.65**

Tabla IV

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

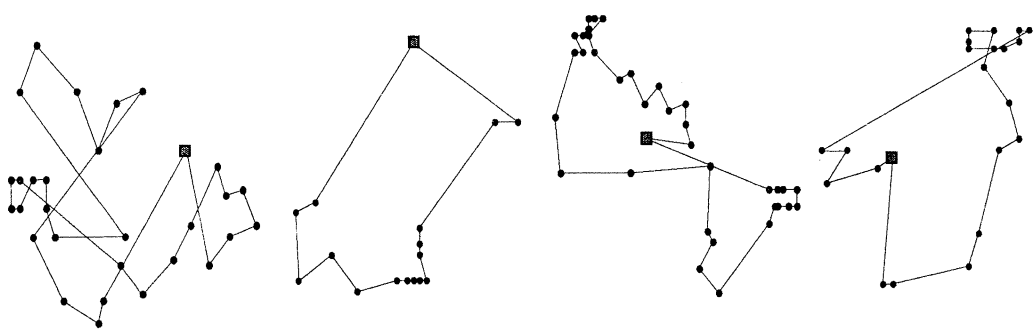
Resultados finales luego de las etapa de post-optimización

RUTA	Recorrido	Longitud	Penalidad	Costo totl
1	0,64,56,95,92,91,80,66,83,57,52,47,17, 16,15,14,12,11,9,99,73,79,60,53,88,55, 13,87,74,86,0	332.821	205.86	538.681
2	0,50,62,20,49,19,18,48,21,23,25,77,58, 75,97,59,0	187.971	0	187.971
3	0,34,31,29,27,26,28,30,32,33,89,76,63, 85,67,65,10,78,7,6,8,46,45,5,3,1,4,2,100, 70,68,61,81,54,96,94,0	347.958	2.34	350.297
4	0,90,82,69,98,35,36,37,38,39,42,44,43, 40,41,72,71,93,84,51,22,24,0	230.108	0	230.108
TOTAL		**1098.858**	**208.2**	**1307.06**

Figura 7

Una heurística de tipo tabú para resolver el problema de ruteo de vehículos con ventanas de tiempo suaves

Las rutas individuales en la solución final



7. CONCLUSIONES

1. La solución inicial obtenida por el método de inserción es lo más parecido a lo que obtendría el administrador de la flota usando solo su sentido común; es decir, localizaría grupos más o menos cercanos de clientes y luego enviaría un vehículo a servir a cada uno de esos grupos yendo cada vez al siguiente cliente mas próximo. Sin embargo podemos ver que esta puede ser una pésima práctica, pues se obtiene una solución muy alejada del óptimo, en el ejemplo esta solución tiene un costo de 19795, mientras que la solución reportada en nuestro método tiene un costo total de 1307, es decir el 6.6% del costo total de la solución inicial.

2. El uso y la implementación de las técnicas de intercambio de aristas denominadas: 1-intercambio y 2-Opt utilizadas en nuestro procedimiento resultó muy efectivo en el algoritmo desarrollado, el método así es simple, directo y el ahorro de esfuerzo computacional es enorme.

3. La búsqueda tabú es una plataforma eficiente para tratar el VRPSTW, pues el algoritmo desarrollado permitió resolver muchos de los conjuntos de problemas de Salomón con soluciones cercanas a las mejor conocidas.

4. Este tipo de métodos y su implementación en la computadora pueden servir de núcleo para la generación de software de interés comercial

5. Se puede enriquecer el procedimiento planteado haciendo modificaciones para bajar el tiempo de computación, específicamente en la forma de generación de la vecindad, lo cual permitiría efectuar más iteraciones del método tabú.

REFERENCIAS BIBLIOGRAFICAS

1. **BARD, JONATHAN F. y KONTORAVDIS, GEORGE; YU, GANG.** (2002). *"A Branch-and-Cut Procedure for the Vehicle Routing Problem with Time Windows"* Transportation Science, Vol. 36, No. 2, pp. 250-269.

2. **CORDEAU, J. GENDREAU M., LAPORTE G., POTVIN J., SEMET F.** (2002). *"A guide to vehicle routing heuristics"*. Journal of the Operational Research Society, Vol 53, pp 512-522.

3. **TOTH, P., VIGO, D.** (2002). *"The Vehicle Routing Problem"*. SIAM Monographs on Discrete Mathematics and Applications. Philadelphia, PA.

4. **TAILLARD E., BADEAU P., GENDREAU M., GUERTIN F., POTVIN J.** (1997). *"A Tabu Search Heuristic for the Vehicle Routing Problem with Soft Time Windows"*. Transportation Science, Vol.31, No.2, pp 170-186

5. **NEMHAUSER, G. L. y RINNOOY KAM, A. H. G.** (1995). *"Handbooks in Operations Research and Management Science. Network Routing"*. Volume 8. Elsevier Science B. V. The Netherlands

6. **KOSKOSIDIS, YIANNIS A. POWELL, WARREN B.; SOLOMON, MARIUS M.** (1992). *"An Optimization-Based Heuristic for Vehicle Routing and Scheduling with Soft Time Window Constraints"* Transportation Science, Vol. 26, No. 2, pp. 69-85.