

ADAPTACION DEL ALGORITMO DE OR AL VRPTW CON CARGA Y DESCARGA SIMULTANEA

Joaquín Antonio Pacheco Bonrostro (+) y Cristina Rocio Delgado Sema(++)

(+)E.U.E.WRESARLES BURGOS

(++)S.A.A.T. Proyecto SIILT

1.- INTRODUCCION

El Problema de Rutas de Vehículos (VRP) puede ser descrito de la forma siguiente: un conjunto de individuos $\{2, 3, n_l\}$ requieren les sea transportada una cantidad de mercancía $q(i)$, $i=2, \dots, n_l$, desde un origen 1. Para ello se dispone de m vehículos con capacidad Q ; cada cliente debe ser servido por un vehículo. Se trata de diseñar m rutas para que los vehículos lleven la mercancía requerida a cada cliente de forma que se respeten las restricciones de capacidad y con distancia total recorrida mínima. Cuando cada punto del problema debe ser visitado dentro de un intervalo de tiempo determinado $[e_i, l_i]$, $i = 1, \dots, n_l$, tenemos el VRP con Ventanas de Tiempo (VRPTW). En el problema que se analiza en este trabajo, además de los puntos de descarga, existen un conjunto de puntos $\{n_l+1, n_l+2, \dots, n_l+n_2\}$ donde hay que recoger una cierta cantidad de mercancía $q(i)$, $i = n_l+1, \dots, n_l+n_2$. Así definido tenemos el VRPTW con Carga y Descarga, o sencillamente, VRPTW Mixto. (En adelante denotaremos $n = n_l+n_2$, es decir el número de puntos que intervienen en el problema).

En este trabajo se propone un algoritmo para el VRPTW Mixto consistente en una extensión y adaptación a este problema del algoritmo de Or para el TSP, (1.976). El algoritmo de Or es una variante de los conocidos algoritmos r -óptimos desarrollados por Lin, (1.965), para el Problema del Viajante (TSP) simétrico. En el caso del algoritmo de Or se toma $r = 3$ y se reduce la búsqueda de los 3-intercambios a aquellos que supongan una recolocación de cadenas de 1, 2 o 3 elementos entre otros dos consecutivos en la ruta actual. La ventaja de este método es que el número de operaciones que se realiza es de $O(n')$, frente al algoritmo 3-óptimo original que utiliza $O(n^3)$. Además, como se verá mas adelante, el algoritmo de Or se puede utilizar en problemas asimétricos. Adaptaciones de este algoritmo al TSP con ventanas de tiempo (TSPTW) se pueden encontrar en los trabajo de Savelsbergh, (1.985), y Salomon y otros, (1.988).

Esta adaptación consiste en incorporar un método que determine o *filtre* aquellos intercambios que vayan a respetar la factibilidad de la nueva solución, sin tener que chequear cada intercambio uno por uno. El resultado es un algoritmo de mejora que reduce mucho el tiempo de computación y que puede ser utilizado en problemas de gran tamaño en ordenadores personales. En este trabajo se ha utilizado este método de filtrado en el algoritmo de Or, pero la misma idea puede ser utilizada para otras adaptaciones de algoritmos de mejora.

2.- IDEA BASICA

Or, (1.976), propone restringir la búsqueda de intercambios a los 3-intercambios en los que cadenas de uno, dos o tres clientes consecutivos son recolocadas entre otros dos clientes (ver figura 1). Esto hace reducir el número de 3-intercambios a considerar a una cantidad de orden $\theta(n^2)$. Nótese que con estos intercambios no se cambia el sentido de los diferentes tramos. En nuestro caso seguiremos la misma idea, pero no pondremos límite al tamaño de la cadena a recolocar salvo el determinado por el tamaño del problema, además debemos chequear la factibilidad de cada posible recolocación respecto a las restricciones del problema. En este caso el número de operaciones es de orden $\theta(n^3)$.

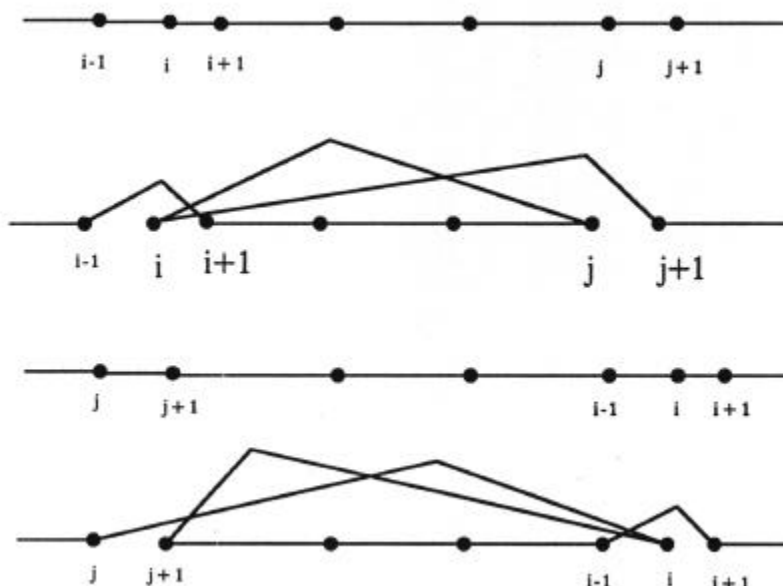


Figura 1.- Posible recolocación del elemento j hacia adelante y hacia atrás entre j y $j+1$.

En el caso general de recolocación de cadenas de k elementos consecutivos, comenzando en la posición i , entre dos puntos que ocupan las posiciones j y $j+1$, supone suprimir los arcos $(i-1,i)$, $(i+k-1,i+k)$, $(j,j+1)$ e incorporar los arcos $(i-1,i+k)$, (j,i) y $(i+k-1,j+1)$.

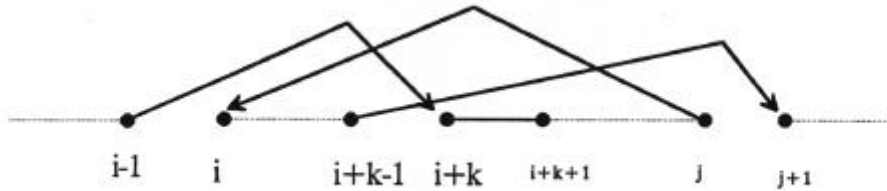


Figura 2. - Recolocación de una cadena de k elementos.

Se define i como el elemento inicial de la cadena; k como el número de elementos que la componen y j el punto tras el cual se recoloca. Los valores que puede tomar i varían desde 2, hasta $n-1$. Para cada valor de i los valores que puede tomar k varían desde 1 a $n-i$. Para cada valor de i y cada valor de k , los posibles valores que puede tomar j varían desde $i+k$ hasta n , (en nuestra adaptación solo se considera la recolocación hacia adelante). Por comodidad, se añade a las rutas el punto $n+1$ que se interpreta como la vuelta al origen 1.

El algoritmo básicamente actúa de la forma siguiente: a partir de una solución inicial examina todos los intercambios o recolocaciones posibles que sean factibles. Para cada uno de estos intercambios factibles calcula la variable *ganancia* definida como la suma de las distancias de los arcos que se suprimen menos la suma de las distancias de los arcos que se añaden. Se realiza el intercambio correspondiente al mayor valor de *ganancia*, y se vuelve a repetir el proceso con la nueva ruta. El algoritmo termina cuando el mayor valor de *ganancia* no sea positivo..

El chequeo de la factibilidad de cada intercambio, es decir de la nueva solución a la que este da lugar, puede suponer un tiempo de computación excesivo. En los apartados siguientes se va a proponer un método que de forma rápida examine las recolocaciones factibles para cada cadena, es decir, para cada valor de i y de k determinar los valores de j factibles.

3.- CASO DE UN SOLO VEHICULO

El chequeo de la factibilidad de los intercambio en el problema que se analiza supone examinar si estos intercambios pueden hacer que las rutas resultantes violen o no las restricciones de capacidad del vehículo en cada momento y los horarios de visita de cada punto.

A partir de ahora se define la variable $r(s)$, $s = 1, \dots, n+1$, como el punto que ocupa la posición i -ésima en la ruta actual en cada iteración; así n -ésimo, para cada posible valor de i , k , j se definen las siguientes cadenas:

cadena_0: $1-r(2)-\dots-r(i-1)$

cadena_1: $r(i)-r(i+1)-\dots-r(i+k-1)$

cadena_2: $r(i+k)-r(i+k+1)-\dots-r(j)$

cadena_3: $r(j)-r(j+1)-\dots-r(n)-1$;

obsérvese que la recolocación en realidad supone un intercambio de posición entre *cadena_1* y *cadena_2*.

3.1- Estudio de la factibilidad respecto al espacio disponible en el vehículo

Para determinar que intercambios pueden *violar* la capacidad del vehículo definimos las siguientes variables:

esp_libre(s): Espacio libre disponible en el vehículo después de visitar el punto $r(s)$;

esp_ocupado_1: Espacio ocupado (si es positivo) o liberado (si es negativo) en el vehículo tras visitar los puntos de *cadena_1*;

esp_ocupado_2: Espacio ocupado o liberado en el vehículo tras visitar los puntos de *cadena_2*;

minimo-cadena_1: Mínimo espacio libre disponible en el vehículo durante la visita a los puntos de *cadena_1*;

minimo-cadena_2: Mínimo espacio libre disponible en el vehículo durante la visita a los puntos de *cadena_2*;

los valores de la variable *esp_libre* se pueden calcular fácilmente en cada iteración. De igual forma se tiene que:

$$esp_ocupado_1 = esp_libre(i-1) - esp_libre(i+k-1);$$

$$esp_ocupado_2 = esp_libre(i+k-1) - esp_libre(j);$$

$$y \quad minimo-cadena-1 = \min_{s=i, \dots, i+k-1} esp_libre(s);$$

$$\text{minimo-cadena-2} = \min_{s=i+k, \dots, j} \text{esp_libre}(s).$$

La recolocación definida por los valores de i , k y j no afecta a *cadena_0* y *cadena-3* en cuanto a espacio disponible en los vehículos, solamente afecta a *cadena-1* y *cadena-2*. Se tienen las siguientes proposiciones:

Proposición 1

Si $\text{esp_ocupado}_1 + \text{minimo_cadena}_2 < 0$ entonces la recolocación definida por i , j y k es infactible.

Demostración.- Trivial: vale darse cuenta que en este caso se violarían las restricciones de capacidad del vehículo en al menos un punto de *cadena_2*.

Corolario 1

Sea la cadena a recolocar definida por los valores de i y k ; se define

$$s = \min \{ r / r > i+k-1 \text{ y } \text{esp_libre}(r) + \text{esp_ocupado}_2 < 0 \}$$

entonces para los valores de j comprendidos entre s y n la recolocación definida por i , j , k es infactible

Proposición 2

Si $\text{esp_ocupado}_2 > \text{minimo_cadena}_1$ entonces la recolocación definida por i , j y k es infactible.

Demostración.- Trivial: vale darse cuenta que se violarían las restricciones de capacidad del vehículo en al menos un punto de *cadena_1*.

3.2- Estudio de la factibilidad respecto las ventanas de tiempo

En este caso hay que tener presente que la recolocación definida por i , j y k además de afectar a los tiempos llegada y salida de los puntos de *cadena_1* y *cadena_2*, también podría afectar a los tiempos de los puntos de *cadena_3*. Se van a definir las siguientes variables:

$A(s)$: Tiempo de llegada a $r(s)$,

$D(s)$: Tiempo de salida de $r(s)$,

$M(s)$: Margen de tiempo en la llegada a $r(s)$,

$E(s)$: Tiempo de espera en $r(s)$,

son inmediatas de calcular en cada iteración. Así mismo para una determinada cadena $r(p)r(p+1)\dots-r(p+h)$ se define

$maxretraso =$ Máximo retraso en la llegada a $r(p)$ que no produce *violaciones* de las ventanas de tiempo en los puntos de dicha cadena.

Proposición 3

Se tiene que los valores de $maxretraso$ se pueden obtener de la forma siguiente

$$maxretraso = \min\left\{M(p), \min_{j=1\dots h}\left[M(p+j) + \sum_{l=p}^{p+j-1} E(l)\right]\right\}$$

Demostración.- Vale ver que en cada punto de la cadena $r(p)\dots-r(p+h)$ el máximo retraso permitido que no viola la ventana de tiempo correspondiente se obtiene sumando los tiempos de espera acumulados en los puntos anteriores de la cadena más el margen en dicho punto.-

Corolario 2

Un método recursivo para obtener el valor de $maxretraso$ viene dado por la siguiente fórmula:

$$maxretraso(r(p)\dots-r(p+h)) = \min\left\{maxretraso(r(p)\dots-r(p+h-1)), M(p+h) + \sum_{j=p}^{p+h-1} E(j)\right\}$$

con $maxretraso(r(p)) = M(p)$ como valor inicial.

Análogamente se define para la cadena $r(p)r(p+1)\dots-r(p+h)$

$maxadelanto =$ Máximo adelanto en la llegada a $r(p)$ que no produce tiempo de espera en dicha cadena.-

Proposición 4

Los valores de máximo adelanto puede obtenerse de la siguiente forma:

$$maxadelanto = \min_{j=0\dots h}\left\{\max[0, A(p+j) - e_{r(p+j)}]\right\}$$

Demostración.- Trivial.-

Corolario 3

A partir de aquí se puede obtener un método recursivo para su cálculo:

$$maxadelanto(r(p)\dots-r(p+h)) = \min\{maxadelanto(r(p)\dots-r(p+h-1)), \max[0, A(p+h) - e_{r(p+h)}]\}.-$$

Estas variables son importantes a la hora de determinar como afecta la alteración en el tiempo de llegada al primer punto de una cadena en el tiempo de salida del último punto de esa cadena y a la posible *violación* o no de las ventanas de tiempo en los puntos de la cadena. Concretamente sea para una determinada cadena $r(p)-r(p+1)-\dots-r(p+h)$, *adelanto_inicial* el tiempo en que se adelanta la llegada a $r(p)$, y *adelanto_final* el tiempo en que se adelanta la salida de $r(p+h)$, se tiene que

$$adelanto_final = \min \{adelanto_inicial, maxadelanto\};$$

análogamente sean *retraso_inicial* y *retraso_final* respectivamente los atrasos en los tiempos de llegada a $r(p)$ y salida de $r(p+h)$, se tiene que

$$retraso_final = \max \{retraso_inicial \sum_{j=1}^h E(p+j), 0\}.$$

De esta forma para cualquier recolocación se puede determinar secuencialmente:

- La alteración en el tiempo de llegada a *cadena_2*, la violación o no de las ventanas de tiempo en los puntos de esa cadena, y la alteración en el tiempo de salida;
- Para *cadena_1* se realiza el n-ésimo proceso;
- Para *cadena-3* finalmente se determina el adelanto o atraso de llegada, y si esta alteración respeta o no las ventanas de tiempo.

4.- CASO DE VARIOS VEHICULOS

4.1.- Planteamiento General

En este caso se va a utilizar un método análogo al utilizado en el apartado anterior. La solución en cada momento va a estar representada también por vector que representa una secuencia ordenada de puntos que empiezan y finalizan el origen 1. Pero en este caso, además, se incorpora el origen 1 un cierto número de veces para indicar la finalización de una ruta y el comienzo de la siguiente. Concretamente en una solución con m rutas habrá $m+1$ "1" en total, (el origen, el final y $m-1$ en puntos intermedios). Por consiguiente, el número total de elementos que componen la solución es de $n+m$.

Se ha de tener en cuenta que cada recolocación puede suponer un cambio solo en la posición de los elementos de cada ruta, sino también un cambio en los elementos que componen cada ruta e incluso, en el número de rutas.

El efecto de las recolocaciones en los tiempos de llegada y salida, y en los espacios disponibles en los vehículos en cada punto dependerán de la presencia o no de 1 en las cadenas que intervienen en la recolocación, concretamente de *cadena_1* y *cadena_2*. Por ello vamos a considerar, además de las ya deferidas, las nuevas cadenas siguientes:

$$\begin{array}{ll}
 \text{cadena}_a: & 1^* \dots -i-1; \\
 \text{cadena}_b: & i \dots - 1^{**}; \\
 \text{cadena}_c: & i\$ \dots - i+k-1; \\
 \text{cadena}_d: & i+k \dots - 1\$\$; \\
 \text{cadena}_e: & 1\& \dots -j; \\
 \text{cadena}_f: & j+1 \dots - i\&\&
 \end{array}$$

donde se han definido: 1^* como siguiente posición a la del último 1 de *cadena_0*; 1^{**} como anterior posición a la del primer 1 de *cadena_1*; $1\$$ como siguiente posición a la del último 1 de *cadena_1*; $1\$\$$ como anterior posición a la del primer 1 de *cadena_2*, y $1\&$ como siguiente posición a la del último 1 de *cadena_2*; y finalmente $1\&\&$ como anterior posición a la del primer 1 de *cadena_3*.

Si no existe ningún 1 en *cadena_1* se toma $1^{**} = i+k-1$ y $1\$ = i$; en este caso *cadena_1* = *cadena_b* = *cadena_c*. De igual forma, si no existe ningún 1 en *cadena_2* se toma $1\$\$ = j$ y $1\& = j+k$; en este caso *cadena_2* = *cadena_d* = *cadena_e*.

De esta forma, solamente habrá que considerar los efectos de las recolocaciones en los puntos de estas seis cadenas, no en el resto.

A continuación vamos a determinar la composición de las nuevas rutas que se forman en cada recolocación. (En adelante a la cadena resultante de la concatenación de otras dos cadenas A y B la denotaremos por A + B). En principio hay que considerar cuatro casos dependiendo de la existencia o no de 1 en *cadena_1* y *cadena_2*:

Si existe 1 en ambas cadenas tras la recolocación definida por i, k y j se tiene las siguientes nuevas rutas:

$1 - \text{cadena}_a + \text{cadena}_d - 1$, $1 - \text{cadena}_e + \text{cadena}_b - 1$ y $1 - \text{cadena}_c + \text{cadena}_f - 1$;
si no existe 1 en *cadena_1* y si existe en *cadena_2* se tiene

$1 - \text{cadena}_a + \text{cadena}_d - 1$, y $1 - \text{cadena}_e + \text{cadenas} + \text{cadena}_f - 1$;
si existe 1 en *cadena_1* y no existe en *cadena_2* se tiene

$1 - \text{cadena}_a + \text{cadena}_d + \text{cadena}_b - 1$ y $1 - \text{cadena}_c + \text{cadena}_f - 1$;
y finalmente, si no existe 1 en ambas cadenas se tiene

$$1 - \text{cadena}_a + \text{cadena}_d + \text{cadena}_b + \text{cadena}_f - 1.$$

4.2- Estudio de la factibilidad respecto al espacio disponible en cada vehículo

El posible cambio de la composición de las rutas hace que haya que estudiar la factibilidad respecto a la capacidad del vehículo en cada una de las seis cadenas *implicadas* en cada recolocación.

Para este estudio vamos a hacer uso del siguiente

Lema

El espacio disponible en un vehículo en cada momento en una determinada ruta va a ser igual a Q menos la suma de las $q(r)$ correspondientes a los puntos de carga r que se hayan visitado previamente y las $q(s)$ correspondientes a los puntos de descarga s que restan por visitar en esa ruta.

Demostración. Obvio: En cada momento en un vehículo se encuentra la mercancía recogida previamente y la que hay que descargar posteriormente.-

Sea una determinada cadena cualquiera *cadena* que no contiene ningún 1, vamos a definir la siguientes variables

minimoespacio(cadena): Mínimo espacio libre disponible en un vehículo que partiendo desde el origen visitara sólo los puntos de *cadena* en ese orden;

qtotalcarga(cadena): Suma total de las cantidades correspondientes a los puntos de carga de *cadena*;

qtotaldescarga(cadena): Suma total de las cantidades correspondientes a los puntos de descarga de *cadena*; con estas definiciones se tiene la siguiente

Proposición 5

Considérese una ruta formada secuencialmente por las cadenas siguientes 1, *cadenaanterior*, *cadena*, *cadenaposterior* y 1, es decir 1- *cadenaanterior* + *cadena* + *cadenaposterior* -1; se tiene que dicha ruta es factible respecto la capacidad del vehículo en los puntos de *cadena* si y sólo si

$$qtotalcarga(cadenaanterior) + qtotaldescarga(cadenaposterior) \leq \text{minimoespacio}(cadena).$$

Demostración. Trivial a partir de los resultados del Lema anterior.-

De esta forma la estrategia que se sigue para comprobar la factibilidad respecto la capacidad del vehículo de cada recolocación es la siguiente:

- Determinar los valores de *minimoespacio*, *qtotalcarga* y *qtotaldescarga* para *cadena_a*, *cadena_b*, *cadena_c*, *cadena_d*, *cadena_e* y *cadena_f*,
- Establecer, según la existencia o no de 1 en *cadena_1* y *cadena_2*, que nuevas rutas se forman;
- Para cada una de las cadenas *cadena_a*,..., *cadena_f*, determinar si se viola o no la factibilidad de la ruta respecto al espacio del vehículo utilizando la proposición 5.

4.3- Estudio de la factibilidad respecto las ventanas de tiempo

En el caso de varias rutas no complica demasiado el estudio de la factibilidad respecto las ventanas de tiempo. Se va a seguir la misma estrategia que en el caso de un solo vehículo, pero teniendo en cuenta que si en una determinada cadena existe un 1, cualquier posible alteración en el tiempo de llegada al comienzo de dicha cadena solamente afectará a los puntos que sean anteriores a dicho 1; por consiguiente:

- Solamente hay que chequear la factibilidad respecto a las ventanas de tiempo en dichos puntos;
- No hay alteraciones en el tiempo de salida de esa cadena.

Por tanto el proceso recursivo para calcular el valor de *maxretraso1*, tal como se ha definido en el apartado 3.3, se ha de parar en el momento en que en *cadena_1* aparezca un 1. De igual forma, el cálculo del valor de *maxretraso2* se ha de parar cuando en *cadena-2* aparezca un 1, y el cálculo de *maxretraso3* se ha de parar cuando aparezca un 1 en *cadena-3*.

Análogamente, los tiempos de *máximoadelanto* y el cálculo de los nuevos tiempos de salida en una cadena se pueden obviar si en dicha cadena se ha detectado un 1.

5.- RESULTADOS COMUTACIONALES

El objeto de los apartados 3 y 4 es desarrollar un método para determinar de forma más *rápida* que intercambios son factibles. Este método se basa, como se ha descrito anteriormente, en definir un conjunto de variables globales cuyo carácter recursivo permite ahorrar cálculos y que determinan la factibilidad de los intercambios fácilmente.

Para establecer la eficacia del método propuesto, se han hecho pruebas para problemas con un solo vehículos y con varios: para ello se han simulado problemas de diferente tamaño, concretamente 20 para cada tamaño de problema. Los problemas generados se han resuelto utilizando el algoritmo con el método de chequeo propuesto y chequeando los intercambios uno a uno. La forma de generar distancias es asignar a cada punto del problema

dos coordenadas x e y, cuyos valores son generados aleatoriamente con distribución uniforme entre 0 y 100. La distancia entre cada par de puntos se define como la distancia euclídea correspondiente. La cantidad de puntos de carga y descarga para cada problema, las cantidades $q(i)$ de cada punto y los horarios de entrega, se han generado también de forma aleatoria.

En todos los problemas se ha utilizado como solución inicial la dada por una adaptación para este tipo de problemas del algoritmo de inserción más cercana. Los algoritmos se han programado en PASCAL, utilizando el compilador BORLAND PASCAL 7.0; en un ordenador tipo PC PENTIUM-100 Mhz.

5.1.- Caso de un sólo vehículo

En este caso se han simulado problemas de los siguientes tamaños: 10, 20, 30, 40, 50, 60 y 70 puntos: A continuación se muestra un cuadro con los siguientes resultados medios

Tamaño	Chequeo uno a uno	Método propuesto	Reducción Porcentual
10	0'022	0'006	72'72
20	0'445	0'077	82'69
30	4'731	0'5 92	87'48
40	14'276	1'464	89'74
50	45'024	4'016	91,08
60	109'297	8'375	92'3 3
70	305'940	20'3 5 9	93'34

5.2.- Caso de varios vehículos

En este caso los tamaños de los problemas son los siguientes: 10, 15, 20, 25, 30, 35 y 40. Los resultados (tiempo medio en segundos) son los siguientes:

Tamaño	Chequeo uno a uno	Método Propuesto	Reducción Porcentual
10	0'402	0'1 04	74'12
15	3'565	0'589	83'47
20	15'862	2'044	87'1 1
25	49'972	5'1 96	89'60
30	129'805	1 0'627	9 1'81

35	262'753	19,015	92'76
40	596'540	40'419	9322

Para el caso de varios vehículos se han realizado además el mismo tipo de pruebas limitando el tamaño de las cadenas a recolocar a $k = 3$, siguiendo la idea original propuesta por Or; los resultados fueron los

Tamaño	Chequeo uno a uno	Método Propuesto	Reducción Porcentual
10	0'143	0'031	78'32
15	0'960	0'1 71	82'1 8
20	3'024	0'401	86'73
25	8'957	0'922	89'70
30	16'604	1'346	91,89
35	3 1'479	2'609	9 1'71
40	6 1'978	4'131	93'33

6.- RESUMEN Y CONCLUSIONES

A la vista de los resultados anteriores se pueden extraer las siguientes conclusiones:

-Los métodos de intercambio r-óptimos tienen una ainplía aceptación dentro de las técnicas heurísticas para problemas de rutas. Sin embargo a medida que aumenta la complejidad de los problemas y se debe chequear la factibilidad de cada intercambio, el tiempo de computación aumenta de forma importante. Se hace pues, necesario incorporar métodos alternativos al del chequeo intercambio a intercambio.

-Este trabajo propone una técnica para determinar los intercambios factibles, en el caso de la adaptación al modelo tratado del algoritmo de Or, basado en definiciones de variables globales que sean calculadas de forma recursiva, con el consiguiente ahorro de tiempo computacional. Una idea semejante a esta fue propuesta para el TSPTW por Savelsbergh, (1.985); en el modelo que se trataba en dicho trabajo se minimizaba el tiempo total empleado en la ruta, por lo que se simplificaban los cálculos al tratar simultáneamente el problema de la factibilidad y *deseabilidad*, en el sentido de ahorro de tiempo, de los intercambios. También para el TSPTW, Salomon y otros, (1.988), proponen un método para filtrar intercambios factibles basado en la definición de una matriz de precedencia que prevé en muchos casos la infactibilidad de los intercambios; sin embargo este método no siempre detecta todos los

intercambios infactibles y por consiguiente es necesario un chequeo posterior. En cualquier caso, según los autores, se conseguía un ahorro medio del 69% para un conjunto de 20 problemas, en la adaptación de Or, y se llegaba a un ahorro del 85% en los algoritmos 3-óptimos.

- En el método de filtrado que se ha propuesto en este trabajo se consiguen ahorros medios superiores a los conseguidos en estos trabajos anteriores. Además se han tratado un modelo más complejo que el TSPTW, al incorporarse varios vehículos y carga y descarga simultánea. Estos ahorros permiten tratar problemas en ordenadores personales de tamaño muy superior a los que se podrían resolver sin este método de filtrado; al menos esto permite concluir la evolución de los tiempos de computación en los diferentes casos. Concretamente, en nuestro caso, se han conseguido resolver problemas con 200 puntos, en menos de 115 segundos de tiempo medio; para ello se ha partido de soluciones iniciales mejores que las obtenidas por los algoritmos de inserción, concretamente con las obtenidas por una adaptación a este modelo del conocido algoritmo de Fisher y Jaikumar para el VRP, (1. 98 l).

- Así mismo el método propuesto, a diferencia de otros métodos anteriores como los mencionados de Solomon y otros, y de Savelsbergh, permite chequear intercambios *interrutas*, y no solamente intercambios *intrarutas*. Esto hace que se puedan llegar a ahorros mayores en las soluciones obtenidas en las sucesivas iteraciones. Muy importante es este punto sobretodo en problemas en los que hay que minimizar el número de vehículos o, en general, en los que hay un coste fijo por vehículo usado, independientemente de la distancia recorrida por estos. Para estos problemas, y en general cuando se utilizan algoritmos de mejora que permiten intercambios entre rutas diferentes, se sugiere redefinir la matriz de instancias de la siguiente manera:

$d(1,i) := d(1,i) + \text{coste por vehículo}$, para $i = 2, \dots, nl+n2$; de esta forma se ven "favorecidos" los intercambios que den lugar a soluciones que contengan al arco (1,1) deshaciendo arcos del tipo (1,i), para $i > 1$, ya que $d(1, 1) = 0$, lo que hace que se vacíen rutas, es decir que se reduzca su número. (El valor de coste por vehículo, obviamente, vendría dado por la relación entre el valor del coste fijo por uso de cada camión y el coste que suponen los kilómetros recorridos). Aunque no es objetivo de este trabajo analizar la *bondad* de este tipo de algoritmos, - sino más bien estudiar la reducción del tiempo de computación que se pueden conseguir en ellos -, según experiencias realizadas, el algoritmo que se consigue incorporando este método de filtrado *compite* en la relación *calidad de la solución obtenida* - *tiempo de computación empleado* con otros heurísticos. En cualquier caso, al ser algoritmos

de mejora, siempre pueden combinarse con cualquiera de estos otros heurísticos. En trabajos posteriores se analizará y comparará con detalle la calidad de las soluciones que se puede obtener con dichas combinaciones.

7.- REFERENCIAS Y BIBLIOGRAFIA

-BODIN,L.D. and GOLDEN,B.L. (1.981)

"Classification in Vehicle Routing and Scheduling".

Networks, vol. 11, no 2, 97-108.

-FISHER,M.L. and JAIKUMARR. (1.981)

"A Generalized Assignment Heuristic for Vehicle Routing".

Networks, vol. 11, no 2, 109-124.

-LIN,S. (1.965)

"Computer Solutions to the Traveling Salesman Problem".

Bell Syst. Tech.Jou., vol 44, 2245-2269.

-OR,I. (1.976)

"Traveling Salesman Type Combinatorial Problems and their Relations to the Logistics of Blood Banking".

Ph. Thesis, Dept. of Industrial Engineering and Management Sciences, Northwestern Univ.

-SAVELSBERGH,M.W.P. (1.985)

"Local Search for Routing Problems with Time Windows".

Ann.Oper.Res., 4, 285-305.

-SOLOMON,M., BAKERE. and SCHAFFERJ. (1.988)

"Vehicle Routing and Scheduling Problems with Time Window Constraints: Efficient Implementations of Solution Improvement Procedures".

In Vehicle Routing: Methods and Studies, (Studies in Management Sciences and Systems, vol. 16), eds: GOLDEN,B.L. and ASSAD,A.A., North-Holland, 85-106.