ÍNDICE

Estudios de Economía Aplicada
Nº 14, 2000. Págs. 137-151

Diseño de Metaheurísticos para Problemas de Rutas con Flota Heterogénea: Concentración Heurística

PACHECO, J.A.
DELGADO SERNA, C.R.
Universidad de Burgos

Agradecen al evaluador sus comentarios y sugerencias, las cuales han sido tomados en cuenta y han sido fundamentales para mejora del trabajo.

RESUMEN

En este trabajo se propone un algoritmo Metaheurístico para el problema de rutas con ventanas de tiempo, carga y descarga simultánea y flota heterogénea, basado en un proceso de tipo *Concentración Heurística*, un tipo de Metaheurístico dado a conocer muy recientemente por Rosing (1997) y Rosing y ReVelle (1997), compuesto en dos fases: en la primera se obtienen un conjunto de optimos locales y se forma un conjunto con los elementos de los mejores optimos locales; en la segunda se utiliza un algoritmo que concentre la búsqueda en las soluciones que contengan elementos de dicho conjunto. En este trabajo se comprueba como de esta forma, y para este modelo concreto, se mejoran los resultados obtenidos en la primera fase. Mejorando en muchos casos las soluciones obtenidas por otros metaheurísticos.

Palabras clave: Metaheurísticos, GRASP, Concentración Heurística, Problemas de Ruta con Flotas Heterogéneas

ABSTRACT

In this study, a metaheuristic algorithm is proposed for the study of routes with time windows, simultaneous pick-up and delivery, and a heterogeneous fleet based on a process of the Heuristic Concentration type, a Metaheuristic type recently introduced by Rosing (1997) and Rosing and ReVelle (1997)-. It is a two stages process: In stage one a set of locals optimums is generated and a set with the

ÍNDICE

elements of the best optimums locals is building; in stage two an algorithm is executed to concentrate the search in the solutions holding elements of this set. For this model, this work show that the solution obtained in stage two improve the best solution obtained in stage one. Almost, the solutions are better that solutions obtained by others metaheuristics.

Artículo recibido el 7 de junio de 1999. Aceptado en octubre de 1999.

ÍNDICE

1. Introducción

El Problema de Rutas de Vehículos con Ventanas de tiempo y con Carga y Descarga Simultánea o sencillamente VRPTW Mixto (Mixed VRPTW) con flota heterogénea puede ser descrito de la forma siguiente: considérese un conjunto de puntos $\{2, 3, ..., n1\}$ donde hay que entregar unas determinadas cantidades de mercancía *q(i)*, $i = 2,...,n1$, en forma de *palés*, desde un origen 1; además considérese otro conjunto de puntos $\{n1+1, n1+2,..., n1+n2\}$ donde hay que recoger otras cantidades de *palés* *q(i)*, $i = n1+1,..., n1+n2$, y llevarlas al origen 1. Para cumplir estos requerimientos se dispone de una flota heterogénea con diferentes tipos de vehículos; cada tipo de vehículo tiene una capacidades de carga diferente; cada punto del problema lleva asociado un intervalo de tiempo de visita $[e_i, l_i]$, $i = 2,...,n1+n2$, (si se llega a i antes del instante e_i hay espera, y no puede visitar más tarde del instante l_i). Las distancias d_{ij} y tiempos t_{ij} entre cada par de puntos $i, j \in \{1,2,...,n1+n2\}$ son conocidas. (A partir de ahora $n = n1+n2$). El número de vehículos disponibles se denota por *ntipos* y las capacidades y costes de cada tipo por *capactipo(i)* y *costetipo(i)*, $i = 1,...,$ *ntipos*; obviamente a más capacidad mas coste.

En este problema se ha de diseñar un conjunto de rutas de coste mínimo verificando las siguientes restricciones:

- Cada ruta comience y finalice en el punto 1;
- se lleve la mercancía correspondiente a cada uno de los puntos del conjunto $\{2,..,n1\}$, y se recoja de cada uno de los puntos del conjunto $\{n1+1, n1+2,..., n1+n2\}$;
- el número de *palés* que en cada momento deba llevar cada vehículo no supere su capacidad;
- cada punto i, $i = 2,...,n1+n2$, sea visitado exactamente una vez;
- se respeten los intervalos de tiempo de visita.

El coste total a minimizar f se descompone en dos partes:

- una parte proporcional a la distancia total recorrida;
- el coste fijo por cada vehículo (según su tipo).

Este problema es una generalización del conocido *Problema de Rutas de Vehículos* o VRP y del *Problema de Rutas de Vehículos con Ventanas de Tiempo* o VRPTW. Existen muchos algoritmos de solución para el VRP y el VRPTW en la literatura. Se pueden encontrar recopilaciones de los principales en trabajos como los de Bodin y Golden (1981), Desrochers y otros (1988), Haouri y otros (1990), y Laporte (1992).

En los últimos años han tomado importancia el desarrollo de algoritmos basados en procesos denominados Metaheurísticos como *Algoritmos Genéticos, Temple Simulado, Búsqueda Tabú, GRASP, Búsqueda Local Guiada (GLS), Colonias de Hórmigas*... especialmente a partir de los trabajo de Gendreu y otros, (1991), ((1994), en versión posterior) y de Osman, (1993); y más recientemente en los de Potvin y otros, (1993), (1994), Thangiah y otros, (1993) y (1994), Campos y Mota (1995), Kantoravdis, (1995), Rochat et al, (1995), Kilby y otros (1997), Backer y otros, (1997), Bullnheimer y otros, (1997), o Rego (1998).

En este trabajo se adapta para este modelo una estrategia Metaheurística dada a conocer por Rosing (1997), Rosing y ReVelle (1997), y Rosing y otros (1998), y que en estos trabajos se aplicaba al problema de las p-medianas. La idea básica es sencilla: obtener un conjunto de buenas soluciones (por ejemplo óptimos locales, aunque no se descartan otros métodos de obtención); formar un conjunto, denominado *Conjunto de Concentración*, con los elementos que componen dichas soluciones; y finalmente ejecutar un algoritmo que busque solamente en las soluciones que contengan (en su totalidad o en la mayor parte, según el diseño que se haga) elementos de este Conjunto de Concentración.

En la sección siguiente se describe de forma general el algoritmo de *Concentración Heurística*; en la sección tres se explica el diseño de la Fase I de esta técnica; en la cuatro se describe el diseño de la Fase II; en la sección cinco se muestran los resultados computacionales y finalmente se comentan las conclusiones.

2. Concentración Heurística: Descripción del Algoritmo

Esta es una de la más reciente de las estrategias metaheurísticas y sus ideas básicas son usadas en este trabajo. Ha sido propuesta en los trabajos de Rosing (1997), Rosing y ReVelle (1997), y Rosing y otros (1998), aplicándola al problema de las p-medianas. Consta de las dos fases siguiente:

ALGORITMO CONCENTRACIÓN HEURÍSTICA

Fase I: *Repetir*
Generar una solución aleatoria y aplicar Búsqueda local durante varias iteraciones
Registrar las m mejores diferentes soluciones obtenidas (donde m es un parámetro preestablecido)

Fase II: *Definir CS (Conjunto de Concentración) como el conjunto de elementos que aparecen en alguna de dichas soluciones.*
Aplicar un método (exacto o heurístico) al problema original pero restringiendo (o concentrando) la selección de elementos a CS.

Según el propio Rossing *"cada óptimo local puede ser considerado como una fuente de información acerca de la estructura de una parte de la solución óptima... ...se espera que un conjunto de aquellos den información sobre todas las partes de esta"* (Rosing y ReVelle, (1997) pg.78).

Por otra parte *CS* se espera que sea bastante más reducido que el conjunto de elementos original, con lo cual la aplicación de un algoritmo exacto en el último paso puede no requerir un tiempo de computación excesivo. En este sentido en Rosing y ReVelle (1997), y Rosing y otros (1998), para el problema de las p-medianas, se plantea una variante consistente en descomponer *CS* en dos subconjuntos CS_0 (abierto) y CS_f (libre); CS_0 es el conjunto de elementos que aparecen en todos los *m* óptimos locales seleccionados y CS_f el conjunto de elementos que aparecen en alguno de ellos pero no en todos; posteriormente se resuelve el problema dejando fijos los elementos de CS_0 en la solución y seleccionando los elementos que quedan en CS_f. De esta forma el tamaño del problema se reduce aún más.

En este trabajo se opta por diseñar un algoritmo heurístico para la segunda fase que intensifique la búsqueda en soluciones que contengan elementos de *CS*, en vez de usar un exacto.

3. Diseño de la Fase I

Se va a seguir la misma idea básica del algoritmo descrito en la sección anterior pero con algunas modificaciones que se describen a continuación. En primer lugar hay que definir que son los elementos que componen una solución; obsérvese que cada solución viene representada por una secuencia de puntos, por ejemplo:

$$1 - 3 - 5 - 1 - 4 - 2 - 1$$

representa a una solución con dos rutas: 1 - 3 - 5 - 1 y 1 - 4 - 2 - 1. En realidad, tales secuencias de puntos pueden considerarse cadenas de arcos de la forma $(1,i_1)$, (i_1,i_2), (i_2,i_3), ..., $(i_s,1)$. Así, en este trabajo consideramos que los elementos que componen una solución son los arcos que aparecen en la cadena correspondiente; en el ejemplo anterior (1,3), (3,5), (5,1), (1,4), (4,2) y (2,1). (Esto no excluye otras opciones para trabajos futuros; por ejemplo si se considera el problema de asignación subyacente en estos modelos, los elementos de una solución pueden considerarse como las asignaciones de cada uno de los puntos de visita a la ruta correspondiente).

Obsérvese que en este caso, las dos estrategias propuestas por Rosing, -considerar un único conjunto de elementos *CS* o dividirlo en dos subconjuntos CS_0 (abierto) y CS_f (libre)-, coinciden, ya que en este modelo de todos los arcos que pueden aparecer ($n\cdot(n$-1) en total) en cada solución debe aparecer uno y solamente uno que salga de cada punto de visita $\{2,3,...,n\}$ y uno y solamente uno que llegue a él.

Además el número de óptimos locales, *m*, seleccionados para la segunda fase no se va a fijar previamente, sino que va a venir dado por un número de arcos diferentes prefijado, (que en adelante denotaremos *num_arcos*). Más concretamente, se ordenan los óptimos locales obtenidos en la primera fase según el valor de la función obtenidos y, comenzando por el mejor, se van añadiendo los elementos de cada uno de ellos a *CS* hasta que el cardinal de *CS* sea mayor o igual que *num_arcos*. (Téngase en cuenta que *m* soluciones son muy parecidas producen CS pequeño; por el contrario si las *m* soluciones son muy diferentes dan lugar a CS muy grande. Por esta razón para evitar excesos en ambos sentidos se opta por fijar el número de elementos diferentes *num_arcos*).

La soluciones iniciales de cata iteración no se generan de forma aleatoria sino que son de tipo ávido-aleatorias, ya que se producen por lo general mejores soluciones. De esta forma, en realidad, esta primera fase es un GRASP. La forma de generar estas soluciones ávido-aleatorias consisten básicamente en: considerar cada problema del VRPTW Mixto como un Problema de Asignación Lineal Generalizada como se hace Fisher y Jaikumar (1.981), y resolver cada problema de Asignación Lineal Generalizada usando el método de aproximación MTHG propuesto en el texto de Martello y Toth (1990). (Para más detalles ver Pacheco y Delgado, (1999)).

El procedimiento de Búsqueda Local usado es el mismo que el propuesto en el trabajo anterior, Pacheco y Delgado (1999): sea *s* una determinada solución se define $N_k(s)$ como el conjunto de soluciones factibles obtenidas por intercambios de Or (1976), de cadenas de a lo sumo *k* elementos en posiciones posteriores realizadas en dicha solución (considerando cada solución como una unica secuencia de puntos, como se ha comentado antes). En la siguiente figura se muestra un ejemplo de intercambio de Or.

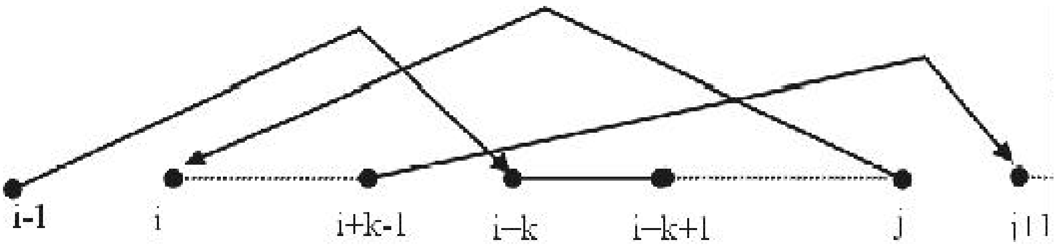

Figura 1. Recolocación de una cadena de k elementos

El procedimiento de Búsqueda Local queda de la forma siguiente:

PROCEDIMIENTO BÚSQUEDA LOCAL

Leer solución inicial s_0
Repetir
Determinar s1 / f(s1) = min {f(s) / s N(s_0)}
Si f(s1) < f(s_0) hacer s_0 = s1
hasta f(s1) f(s_0)

donde *N*(*s*) es el conjunto de soluciones vecinas de cada solución s. Concretamente se toma *N*(*s*) = N_3(*s*). De esta forma la primera fase queda de la siguiente forma:

CONCENTRACIÓN HEURÍSTICA: FASE I

Repetir
Generar una solución Avida Aleatoria;
Aplicar Búsqueda local
durante un determinado número de iteraciones;
Registrar las soluciones obtenidas en una lista ordenada según el valor de la función objetivo f comenzando con la mejor
Hacer i = 0 y CS = ∅
Repetir
hacer i = i+1
Añadir los elementos de la i-ésima solución de la lista a CS
hasta Cardinal(CS) ³ num_arcos

4. Diseño de la Fase II

Según se ha comentado anteriormente se va a diseñar un algoritmo heurístico que *concentre* la búsqueda de elementos en *CS*. Por otra parte para contrastar la eficacia de esta búsqueda dirigida a *CS* el algoritmo diseñado va a ser similar al de la primera fase. Inicialmente se define la matriz de distancias auxiliar *d*1 de la siguiente forma:

*d*1(*i*,*j*) = *d*(*i*,*j*) si (*i*,*j*) 0 *CS*
*d*1(*i*,*j*) = *d*(*i*,*j*) + 10**max_d*, si (*i*,*j*) ó *CS*

donde *max_d* = max{*d*(*i*,*j*) / *i*,*j* = 1,...,*n*}; además se define *f*1(*s*) como el valor de la función objetivo considerando el coste de los arcos dado por *d*1. Se propone el siguiente procedimiento de búsqueda que tiene en cuenta ambas matrices *d* y *d*1:

ÍNDICE

PROCEDIMIENTO BÚSQUEDA LOCAL GUIADA

Leer solución inicial s_0
Hacer $s^* = s_0$
Repetir
  Hacer $coste\_anterior = f(s^*)$
  Repetir
    Buscar $s' \ 0 \ N_3(s_0) \ / \ f1(s') = min\{f1(s) \ / \ s \ 0 \ N_3(s_0)\}$
    Si $f1(s') < f1(s_0)$ *entonces* $s_0 = s'$
    Buscar $s'' \ 0 \ N_3(s_0) \ / \ f(s'') = min\{f(s) \ / \ s \ 0 \ N_3(s_0)\}$
    Si $f(s'') < f(s^*)$ *entonces* $s^* = s''$
  hasta $f1(s') \geq f1(s_0)$
  Hacer $s_0 = s^*$
hasta $f(s^*) = coste\_anterior$

Se trata de un procedimiento de *búsqueda local anidado*: en cada paso la solución actual s_0 se sustituye por otra mejor según *f*1, es decir según *d*1 y buscando por tanto soluciones que contengan elementos de *CS*; cuando no hay mejora en *f*1, se sustituye s_0 por s^*, la mejor solución según *f* observada en los vecindarios explorados y se reinicia la búsqueda local. El proceso acaba cuando no hay mejora en $f(s^*)$.

Es un procedimiento de búsqueda "guiado" por *f*1, es decir, por *d*1, hacia soluciones que contengan el mayor número de elementos de *CS* posibles. Obsérvese que *d*1 ?penaliza? a los arcos no pertenecientes a *CS*, pero no ?impide? su elección en cada paso, ya que esto podría hacer excesivamente reducido el número de soluciones *s* a considerar y ?encajonar? el proceso. Obviamente, si se usara un algoritmo exacto la estrategia debería ser impedir en vez de penalizar. Este procedimiento se inserta en la segunda fase que queda de la siguiente forma:

CONCENTRACIÓN HEURÍSTICA: FASE II

Hacer $iter\_mejor = 0$, $niter = 0$, $coste\_min = inf.$;
Repetir
  $niter = niter + 1$;
  Generar una solución Avida-Aleatoria (considerando función objetivo f1);
  Ejecutar Búsqueda_Local_Guiada;
  Si $f(s_0) < coste\_min$ *hacer:* $coste\_min = f(s_0)$, $s\_mejor = s_0$, $iter\_mejor = niter$
hasta $niter - itermejor > 100$

Esta fase es similar a la primera salvo las siguientes consideraciones: se toma *f*1 en vez de *f* al generar las soluciones iniciales y se ejecuta el procedimiento de *Búsqueda_Local_Guiada* dirigido por *f*1, en vez de la búsqueda local habitual. En definitiva, se "concentra" la búsqueda de soluciones en los elementos de *CS*. Veamos

ÍNDICE

a continuación los resultados obtenidos.

5. Resultados Computacionales

Según se ha comentado se quiere comprobar si la segunda fase, dirigiendo la elección de elementos a *CS*, puede mejorar o no los resultados de la primera. También se quiere determinar con que "níveles de concentración", es decir, con que valores del parámetro *num_arcos* se consiguen mejores resultados. De igual forma se quiere comprobar si las listas más grandes en la primera fase llevan a mejores soluciones en la segunda.

Para ello se han generado 50 instancias: 25 con 39 puntos de descarga y 25 con 19 puntos de carga y 20 de descarga (más el origen, obviamente). Los datos de cada problema se definen de la forma siguiente:

- Se asigna 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. Los tiempos (en minutos) de trayecto se toman igual a la distancia (en kilómetros.). (Es decir consideramos una velocidad de 60 kilómetros/hora). El coste por kilómetro es de 1 u.m.
- A e_i y l_i, para $i = 2, ..., n1+n2$, se les asigna respectivamente dos valores enteros aleatorios generados uniformemente: entre 600 y 720 (minutos) en el primer caso y entre 960 y 1.080 en el segundo. Se hace e_1 igual a 480 y l_1 igual a 1.200.
- A cada $q(i)$, $i = 2, ..., n1+n2$, se le asigna un valor entero generado uniformemente de forma aleatoria entre 1 y 12.
- En todos los casos se supone dos tipos de vehículos, con capacidades 10 y 20, y con costes 1.000 y 1.200 u.m. respectivamente.
- De igual forma, en la fase I se han realizado 10.000 iteraciones obteniéndose 4 listas ordenadas de óptimos locales según el valor de f correspondientes a las 1.000 primeras, 2.000 primeras, 5.000 primeras y a las 10.000 iteraciones respectivamente. Además se ha generado una quinta lista ordenada de la siguiente forma: a cada una de las 500 mejores soluciones de las generadas anteriormente se les vuelve a aplicar el procedimiento de Búsqueda Local usado en la primera fase pero tomando los vecindarios $N(s) = N_{nt-3}(s)$ en vez de $N(s) = N_3(s)$, (donde nt es el número de puntos que componen una solución s), es decir, se consideran todo el conjunto de recolocaciones.

PROCEDIMIENTO OBTENCIÓN_LISTA_5

Desde i = 1 hasta 500 hacer:
Tomar la i-ésima mejor solución de las 10.000 de la Fase I
Ejecutar Búsqueda_Local considerando $N = N_{nt-3}$
Guardar estas nuevas soluciones en una lista ordenada (Lista _5)

Para la segunda fase se han establecido 4 niveles de concentración correspondientes a fijar *num_arcos* igual a 260 (16'6% de los arcos), 195 (12'5%), 130 (8'3 %) y 78 (5%), formándose para cada nivel y para cada lista el conjunto *CS* correspondiente, (5 listas x 4 niveles = 20 conjuntos en cada instancia).

A continuación se muestran tablas con el resumen de los resultados obtenidos para cada tipo de instancias de las mejores soluciones.

Tipo de Instancias I: Origen + 39 puntos de descarga

	Lista 1	*Lista 2*	*Lista 3*	*Lista 4*	*Lista 5*
Fase I	14971.00	14956.44	14946.00	14935.04	14916.32
Nivel 1 (16'66%)	14940.52	14929.32	14932.40	14925.32	14923.40
Nivel 2 (12'5%)	14940.48	14931.08	14926.32	14923.32	14914.68
Nivel 3 (8'33%)	14937.40	14926.24	14921.20	14915.20	14900.48
Nivel 4 (5%)	14945.24	14926.64	14925.28	14914.48	14903.84

Resultados medios de las mejores soluciones obtenidos en cada fase para cada lista

Tipo de Instancias II:
Origen + 19 puntos de descarga + 20 puntos de carga

	Lista 1	*Lista 2*	*Lista 3*	*Lista 4*	*Lista 5*
Fase I	8730.68	8722.36	8717.56	8710.88	8680.68
Nivel 1 (16'66%)	8737.36	8730.68	8716.92	8725.92	8711.88
Nivel 2 (12'5%)	8731.52	8729.48	8717.40	8718.96	8712.84
Nivel 3 (8'33%)	8724.80	8718.88	8713.56	8720.16	8698.16
Nivel 4 (5%)	8719.80	8721.08	8705.88	8717.12	8696.16

Resultados medios de las mejores soluciones obtenidos en cada fase y en cada lista

Los resultados observados son interesantes, especialmente para el primer tipo de instancias: usando una estrategia análoga las soluciones encontradas en la segunda fase superan al mejor óptimo local de la primera fase de la correspondiente lista, (en valores medios y en la mayoría de las instancias analizadas). Para el segundo tipo de instancias también se observa este hecho para las primeras listas, pero de forma menos clara, e incluso para las últimas listas (4 y 5) no mejora los resultados de la primera. Entendemos sin embargo a que es debido al tipo de vecindario usado en la segunda fase (no se contemplan recolocaciones hacia detrás). En cualquier caso parece claro, aunque de forma general, la capacidad de mejorar las soluciones obtenidas que tiene el hecho "concentrar" la búsqueda en un conjunto de arcos pertenecientes a esas mismas soluciones.

Se confirma una idea expresada anteriormente: de una lista más grande se obtiene mejores soluciones en la primera fase y también en la segunda; o mas concretamente: a mejores soluciones de la primera fase mejores en la segunda. En cuanto a los niveles de concentración parece que a menor tamaño de *CS* mejores soluciones, siendo los niveles 3 y 4 los mejores.

Finalmente se han generado otras 10 instancias de cada tipo para comparar los 4 metaheurísticos diseñados en este trabajo y en dos anteriores, Pacheco y Delgado, (1998) y (1999): *Temple Simulado*, *Búsqueda Tabú*, *GRASP* y *Concentración Heurística*. Indicar que, para evitar redundancias, en el algoritmo GRASP se hace una pequeña modificación de forma que realice 1.000 iteraciones justas en vez de usar el criterio de parada original; de esta forma sirve simultáneamente como Fase I en el algoritmo Concentración Heurística. Para la Fase II se establece un nivel de concentración del 10%. (A pesar de la aleatoriedad con que dichas instancias han sido generadas el autor pone a disposición de los lectores interesados los datos de los problemas así como los algoritmos usados programados en Pascal para contrastar los resultados). Los resultados se muestran a continuación:

Tipo de Instancias I: Origen + 39 puntos de descarga

	Temple Simulado	*Búsqueda Tabú*	*GRASP- 1.000 it (CH-Fase I)*	*Concentración Heurística Fase II*
N. 1	14670	14666	14530	14514*
N. 2	14742	14585	14501	14471*
N. 3	15739	15628	15586	15570*
N. 4	15241	15114*	15174	15123
N. 5	15726	14806*	14864	14836
N. 6	14582	14534	14524	14468*
N. 7	15708	15741	15662	15649*
N. 8	14466	14466	14490	14450*
N. 9	15890	15883	15878	15795*
N. 10	13649	13428	13384*	13394
C. Medio	**15041.30**	**14885.10**	**14859.30**	**14827.00***

Tipo de Instancias II:
Origen + 19 puntos de descarga + 20 puntos de carga

	Temple Simulado	*Búsqueda Tabú*	*GRASP- 1.000 it (CH-Fase I)*	*Concentración Heurística Fase II*
N. 1	6774	6921	6622	6581*
N. 2	8004	7885	7921	7883*
N. 3	9297	9319	9242	9227*
N. 4	8313*	8355	8331	8314
N. 5	9334*	9473	9436	9365
N. 6	9267	9289	8333*	8343
N. 7	8291	8352	8131*	8133
N. 8	9305	9339	9316	9301*
N. 9	8284	8306	8242	8215*
N. 10	9540	8738	8533	8514*
C. Medio	8640.90	8597.70	8410.70	8387.60*

*(Con * se indica el mejor resultado de cada instancia y del conjunto)*

Obsérvese como la Fase II del algoritmo *Concentración Heurística* mejora en la mayoría de los casos a los de la Fase I; además es el mejor de los 4 metaheurísticos propuestos para la mayoría de las instancias y en los resultados medios.

6. Resumen y Conclusiones

En este trabajo se expone un tipo de metaheurístico denominado Concentración Heurística, propuesto recientemente por Rosing (1997) y Rosing y ReVelle (1997), que se aplica al Problema de Rutas de Vehículos con flota heterogénea y Carga y Descarga Simultánea.

La técnica se desarrolla en dos fases básicas:

1) Generar un conjunto de óptimos locales a partir de soluciones aleatorias y registrar las mejores.
2) Formar el conjunto *CS* (conjunto de concentración) de elementos que aparecen en esas soluciones, y ejecutar un algoritmo exacto o heurístico del problema original pero restringiendo o ?concentrando? la búsqueda de elementos a *CS*.

En este trabajo se propone una variante para la Fase I: generar óptimos locales pero a partir de soluciones Avido-Aleatorias, como en el GRASP propuesto por Pacheco y Delgado (1999) y no prefijar el número de óptimos locales a seleccionar sino el tamaño del conjunto *CS*.

Para la segunda fase se usa una estrategia análoga a la Fase I: realizar una serie de iteraciones en cada una de las cuales se genera una solución Ávida-Aleatoria y se aplica el procedimiento de *Búsqueda Local* de la Fase I modificado (*Busqueda_local_Guiada)* que dirige la búsqueda a los elementos de *CS*.

Realizando una serie de pruebas se comprueba como para diferentes tamaños de *CS* y diferentes número de iteraciones de la Fase I, los resultados de la Fase II mejoran los de la Fase I. Esto pone de manifiesto la eficacia de concentrar la búsqueda a los elementos de *CS*, sobretodo teniendo en cuenta que el número de iteraciones realizadas en esta segunda fase es menor.

Además el algoritmo resultante ofrece mejores soluciones que otros Metaheurísticos conocidos al menos en las instancias analizadas.

A partir de aquí se plantean las siguientes cuestiones y sugerencias:

- ¿Podría seguir siendo eficaz esta estrategia considerando los elementos de cada solución como las asignaciones de puntos a rutas (en vez de arcos), y los vecindarios propuestos en trabajos como los de Gendreu y otros (1991), Osman (1993) o Campos y Mota (1995)?
- ¿Se podrían mejorar los resultados de la Fase II eligiendo un algoritmo más adecuado, aumentando las iteraciones, afinando el parámetro a que determina el grado de Aleatoriedad o 'Avídez' en las soluciones iniciales de cada iteración, utilizando un vecindario mayor, etc...?.
- ¿Podría utilizarse otro método para generar buenas soluciones, es decir, otra forma de diseñar la Fase I?.
- Relacionado con esto último ¿Qué resultado daría el uso de un Conjunto de Concentración, como se hace en este trabajo, en la fase de Intensificación de la Búsqueda Tabú?

En cualquier caso entendemos que es interesante tomar en consideración este Metaheurístico para la resolución de diferentes modelos en trabajos futuros tanto en problemas de rutas como en otros.

7. Bibliografía

BACKER (DE), B. FURNON,V. KILBY,P., PROSSER,P. and SHAW,P. (1997): «Solving Vehicle Routing Problems using Constraint Programming and Metaheuristics». *Journal of Heuristics*, vol. nº 1 - 160.

BODIN L.D. and GOLDEN B.L. (1981): Classification in Vehicle Routing and Scheduling. *Networks*, vol.11, nº 2.

ÍNDICE

BULLHEIMER,B., HARTI,R.F. and STRAUSS,C. (1997): Applying the Ant System for the Vehicle Routing Problem. *2nd Metaheuristics International Conference* (MIC-97), Sophie-Antipolis, France, July 1997.

CAMPOS V. y MOTA E. (1995): Metaheurísticos para el CVRP. *XXII Congreso Nacional de Estadística e Investigación Operativa*. Sevilla, Noviembre.

DESROCHERS M., LENSTRA J.K. SAVELSBERGH M.W.P. and SOUMIS F. (1988): Vehicle Routing with Time Windows: Optimization and Approximation. In *Vehicle Routing: Methods and Studies*, (Studies in Management Sciences and Systems, vol.16), eds: GOLDEN,B.L. and ASSAD,A.A., Nort-Holland.

FISHER,M.L. y JAIKUMAR,R. (1.981): «A Generalized Assignment Heuristic for Vehicle Routing». *Networks*, vol.11, nº 2, 109-124.

GENDREU M., HERTZ A. and LAPORTE G. (1991): A Tabu Search Heuristic for Vehicle Routing Problem. Report CRT-777. *Centre de Recherche sur les Transports*. Univ. Montréal.

HAOURI M., DEJAX P. et DESROCHERS M. (1990): Les Problems de Tournées avec Contraintes des Fenênetres de Temps: L'Etat de l'Art. *Recherche Operationnelle/Operations Research*, vol. 24, n? 3.

KILBY,P., PROSSER,P. and SHAW,P. (1997): Guided Local Search for the Vehicle Routing Problem. *2nd Metaheuristics International Conference* (MIC-97), Sophie-Antipolis, France, July 1997.

KONTORAVDIS G. and BARD J.F. (1995): A GRASP for the Vehicle Routimg Problem with Time Windows. *ORSA Journal on Computing*, 7: 10-23.

LAPORTE G. (1992): The Vehicle Routing Problem: An overview of exact and approximate algorithms. *European Journal of Operations Research*, 59.

MARTELLO,S y TOTH,P. (1990): Knapsack Problems. *John Wiley & Sons*. Chichester.

OR,I. (1976): Traveling Salesman Type Combinatorial Problems y their Relations to the Logistics of Blood Banking. Ph.Thesis, Dpt. of *Industrial Enginnering y Management Sciences*, Northwestern Univ.

OSMAN I.H. (1993): Metastrategy Simulated Annealing and Tabu Search Algorithms for the Vehicle Routing Problem. *Annals of Operations Research*, 41.

PACHECO J. y DELGADO C. (1998): Diseño de Metaheurísticos híbridos para Problemas de Rutas con Flota Heterogénea. *XII Reunión ASEPELT-ESPAÑA*. Córdoba.

PACHECO J. y DELGADO C. (1999): Diseño de Metaheurísticos para Problemas de Rutas con Flota Heterogénea: GRASP. Aceptado para su publicación en el nº 9 de *Cuadernos de Ciencias Económicas y Empresariales*. Universidad Complutense de Madrid.

POTVIN,J.Y. and BENGIO,S. (1994): «A Genetic Approach to the Vehicle Routing Problem with Time Windows». Technical Report CRT-953, *Centre de Recherche sur les Transports*. Univ. Montréal.

ÍNDICE

POTVIN,J.Y., KERVAHUT,T., GARCIA,B.L. and ROUSSEAU,J.M. (1993).«A Tabu Search Heuristic for Vehicle Routing Problem with Time Windows». Report CRT-777. *Management Sci*. 40 (10), pg.1276-1290.

REGO, C. (1998): «A Subpath Ejection Method for the Vehicle Routing Problem». *Management Science*, vol. 44, nº 10, pg. 1447-1459.

ROCHAT,Y. and TAILLARD,E.D. (1995): «Probabilistic Diversification and Intensification in Local Search for Vehicle Routing». *Journal of Heuristics*, 1 (1), 147-167.

ROSING K.E. (1997): Heuristic Concentration: An Introduction with Examples. *The Tenth Meeting of the European Chapter on Combinatorial Optimization*. Tenerife. Spain. May.

ROSING K.E. and REVELLE C.S. (1997): Heuristic Concentration: Two Stage solution Construction. *European Journal of Operational Research* 97.

ROSING K.E., REVELLE C.S., ROLLAND E., SCHILLING D.A. and CURRENT J.R. (1998): Heuristic Concentration and Tabu Search: A head to head comparison. *European Journal of Operational Research* 104, 93-99.

TAILLARD, E., BADEAU,P., GENDREU,M., GUERTAIN,F. and POTVIN,J.Y. (1995): «A new Neighbourhood structure for the Vehicle Routing Problem with Time Windows». Technical Repport CRT-95-66, *Centre de Recherche sur les Transports*. Univ. Montréal.

TAILLARD, E., BADEAU,P., GENDREU,M., GUERTAIN,F. and POTVIN,J.Y. (1997): «A Tabu Search heuristic for the Vehicle Routing Problem with Time Windows». *Transportation Science* 31, 170-186.

THANGIAH,S.R., OSMAN,I.H., and SUN,T. (1994): «Hybrid Genetic Algorithm, Simulated Annealing, and Tabu Search methods for the Vehicle Routing Problem with Time Windows». Working paper UKC/OR94/4, *Institute of Mathematics and Statistics*, University of Kent, Canterbury.

THANGIAH,S.R., VINAYAGAMOORTY,R., and SUN,T. (1993): «Vehicle Routing Problem with Time Deadlines using Genetic and Local Algorithms». In *5th International Conference on Genetic Algorithms*, 1993.