MAEB'09

VI Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados

Certificado de Participación

El comité organizador del VI Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados (MAEB'09), celebrado los días 11 al 13 de febrero de 2009 en Málaga,

certifica que

Irma Delia García Calvillo

ha participado en el congreso.

Málaga, a 11 de febrero de 2009

MAEB 2009 UNIVERSIDAD DE MALAGA

Fdo.: Antonio J. Nebro
Secretario de MAEB'09

MAEB'09

VI Congreso Español sobre Metaheurísticas, Algoritmos Evolutivos y Bioinspirados

11, 12 y 13 de Febrero de 2009

Málaga, España

Editores

Enrique Alba, Francisco Chicano, Francisco Luna, Gabriel Luque

Un problema de distribución de productos alimenticios con flexibilidad en la fecha de entrega

Ada Álvarez[1], Joaquín Pacheco[2], Francisco Angel–Bello[3], Irma García[4],

Resumen— **Se presenta una aplicación de un problema de ruteo de vehículos, donde se permite cierta flexibilidad en la fecha de entrega de productos. Se propone una modelación del problema y un método GRASP para resolverlo. Se presentan algunos resultados computacionales.**

Palabras clave— **ruteo de vehículos, GRASP, heurística GENI.**

I. Introducción

Se presenta una problemática real planteada por una compañía de productos de repostería que tiene su depósito central o fábrica en Briviesca, Burgos. La empresa tiene que satisfacer los pedidos de sus centros de distribución, ubicados en el norte de España, en el período de una semana utilizando una flota de vehículos homogénea. La ruta de cada vehículo finaliza el mismo día que comienza y deben regresar al depósito al finalizar el recorrido. Cada orden o pedido consiste en un número predeterminado de pallets solicitados y un día de la semana en que es requerido.

Actualmente, para el diseño de las rutas de los vehículos, la compañía resuelve un problema de ruteo de vehículos (VRP por sus siglas en inglés) clásico cada día de la semana, de acuerdo a los centros que requieren los productos cada día. Sin embargo, la compañía considera que los costos pueden mejorarse.

Para reducir los costos de transporte, y teniendo en cuenta que los productos se envían a distribuidores y no a usuarios finales, es posible considerar una cierta flexibilidad en la fecha de entrega de productos. Sin embargo, como se trata de productos alimenticios, la propuesta es entregar los productos el día originalmente solicitado o unos pocos días antes (uno o dos). Obviamente, esto generará un costo de almacenamiento, pero será controlado ya que se almacenan justamente las cantidades que han sido solicitadas y además es un costo aceptable para la compañía.

El modelo propuesto es una variante de la formulación de dos índices para el problema de ruteo de vehículos capacitado (CVRP), descrita por Toth y Vigo [9], considerando todos los pedidos de la semana y agregando las restricciones correspondientes a la flexibilidad en la fecha de entrega. Para resolverlo se propone un método basado en la metaheurística GRASP.

[1]Universidad Autónoma de Nuevo León, México. E-mail: adalvarez@mail.uanl.mx
[2]Universidad de Burgos, España. E-mail: jpacheco@ubu.es
[3]Instituto Tecnológico y de Estudios Superiores de Monterrey, Campus Monterrey, México. E-mail: fangel@itesm.mx
[4]Universidad Autónoma de Nuevo León, Universidad Autónoma de Coahuila, México. E-mail: irma.igarcia@gmail.com

II. El modelo propuesto

El problema se modela utilizando un grafo completo cuyos nodos son el depósito central y los centros de distribución de la compañía. Introducimos la siguiente notación.

n	=	número total de órdenes en la semana.
G	=	(V, A) un grafo completo.
V	=	$\{v_0, v_1, v_2, \ldots, v_n\}$ nodos del grafo, donde v_0 es el depósito y v_i corresponde a la localización del pedido i, $i = 1, 2, \ldots, n$.
A	=	$V \times V$ arcos del grafo,
k	=	número de vehículos.
Q	=	capacidad de los vehículos.
c_{ij}	=	costo (distancia) del nodo i al nodo j.
d_i	=	demanda del nodo i.
e_j	=	día en que el nodo j requiere los productos, $e_j \in \{1, 2, \ldots, 5\}$.
F	=	máximo de días permitidos para adelantar pedidos. En la aplicación real $F = 1$.

Las variables de decisión

$$x_{ij} = \begin{cases} 1, & \text{si el nodo } j \text{ se visita justo después del nodo } i \\ 0, & \text{en otro caso} \end{cases}$$

Para garantizar que las fechas de requerimiento de productos de todos los nodos que pertenecen a la misma ruta no difieran en más de F días, se agregan las siguientes restricciones: Si r y s son nodos en la misma ruta, entonces

$$|e_r - e_s| \leq F$$

Para que el modelo permanezca lineal, introducimos una variable z_j para cada nodo j, que representa una cota inferior de las fechas de requerimiento de productos de los nodos de la ruta. De esta forma, si un vehículo viaja del nodo i al nodo j, entonces

$$z_j \leq z_i$$

Esto puede modelarse como

$$z_j - z_i \leq M(1 - x_{ij})$$

para cada $i, j \in V\backslash\{0\}$, con $M > 0$ una constante.

Además, sea w_j una cota superior de las fechas de requerimientos de productos de los nodos de la ruta. Si $x_{ij} = 1$ entonces

$$w_j \geq w_i$$

y agregamos las restricciones

$$w_i - w_j \leq M(1 - x_{ij})$$

para cada $i, j \in V\backslash\{0\}$. Para que las fechas de requerimientos de productos de los nodos que pertenezcan a una ruta no difieran en más de F, agregamos las restricciones

$$w_j - z_j \leq F$$

para cada nodo j.

El modelo entero–mixto es el siguiente.

$$\text{mín} \sum_{i \in V} \sum_{j \in V} c_{ij} x_{ij}$$

sujeto a

$$\sum_{i \in V} x_{ij} = 1, \qquad \forall j \in V\backslash\{0\} \qquad (1)$$

$$\sum_{j \in V} x_{ij} = 1, \qquad \forall i \in V\backslash\{0\} \qquad (2)$$

$$\sum_{i \in V} x_{i0} \leq k \qquad (3)$$

$$\sum_{i \in V} x_{i0} = \sum_{j \in V} x_{0j} \qquad (4)$$

$$u_i - u_j + Q x_{ij} \leq Q - d_j, \quad \forall i, j \in V\backslash\{0\}, i \neq j, \text{ tal que } d_i + d_j \leq Q \qquad (5)$$

$$d_i \leq u_i \leq Q, \qquad \forall i \in V\backslash\{0\} \qquad (6)$$

$$z_j - z_i \leq M(1 - x_{ij}), \qquad \forall i, j \in V\backslash\{0\} \qquad (7)$$

$$w_i - w_j \leq M(1 - x_{ij}), \qquad \forall i, j \in V\backslash\{0\} \qquad (8)$$

$$w_j - z_j \leq F, \qquad \forall j \in V\backslash\{0\} \qquad (9)$$

$$1 \leq z_j \leq e_j, \qquad \forall j \in V\backslash\{0\} \qquad (10)$$

$$e_j \leq w_j \leq 5, \qquad \forall j \in V\backslash\{0\} \qquad (11)$$

$$u_j, z_j, w_j \geq 0, \qquad \forall j \in V\backslash\{0\} \qquad (12)$$

$$x_{ij} \in \{0, 1\}, \qquad \forall i, j \in V \qquad (13)$$

Las restricciones (1) y (2) son las restricciones de grado, exactamente un arco llega y sale de cada nodo. Análogamente, (3) y (4) imponen las restricciones de grado para el depósito; (5) y (6) son la familia de restricciones alternativas equivalentes a las restricciones de capacidad y corte y (7) a (9) son las restricciones adicionales que se han introducido para modelar la flexibilidad en la fecha de entrega. (10) y (11) son las restricciones impuestas para las cotas inferior y superior.

Note que se han agregado $2n$ variables continuas y $2n^2 + n$ restricciones al modelo VRP clásico.

El modelo fue validado experimentalmente. Se generaron algunas instancias aleatorias con diferentes valores de F y n y se resolvieron con Cplex v.9.0.

Se observa, como era de esperarse, que cuando n aumenta, el solver se vuelve más lento, especialmente cuando $F = 2$ ó $F = 3$. Cuando $F = 4$ ó 5 se identifica que el problema es un VRP sin restricciones adicionales y se resuelve más rápidamente.

III. El método de solución

El modelo que se propone en este trabajo es una generalización del CVRP, el cual se sabe que es NP–completo [9], por lo que se diseñó un método de solución basado en la metaheurística GRASP, que puede encontrar soluciones cercanas a la óptima en tiempos de cómputo razonables.

GRASP [3] es un procedimiento iterativo. Cada iteración consiste básicamente en dos fases: construcción y búsqueda local. La fase de construcción trata de construir una solución factible paso a paso, en la búsqueda local se explora una vecindad de la solución construida.

La fase de construcción está caracterizada por una función miope adaptativa, la cual estima el beneficio de incluir un elemento candidato en la solución parcial. La selección se hace aleatoriamente de una lista restringida de candidatos formada por los elementos con los mejores valores de la función miope.

Un bosquejo del método propuesto es el siguiente.

Fase constructiva

- Construir k grupos de nodos como sigue:
  - Aproximar el problema, siguiendo las ideas de Fisher y Jaikumar, por un problema de asignación generalizada, (GAP).
  - Resolver el GAP.
- Diseñar la ruta en cada grupo de nodos.

Búsqueda local

Realizar intercambios intra–rutas y/o inter–rutas de cadenas de vértices.

A continuación describimos a detalle las fases de construcción y búsqueda local.

A. Fase constructiva

La fase constructiva consiste en diseñar k grupos de nodos, uno para cada vehículo, y posteriormente resolver el Problema del Agente Viajero para los nodos que pertenecen a cada grupo. La fase de agrupamiento está basada en la idea de Fisher y Jaikumar de aproximar el problema por un problema de asignación generalizada (GAP) [4]. Para resolver el GAP utilizamos una modificación del algoritmo de Martello y Toth [6].

Sea a_{il} el costo de asignar el nodo i al grupo l, obtenido por la aproximación de Fisher y Jaikumar. La modificación que se propone al algoritmo de Martello y Toth es la siguiente.

$S = \emptyset$, $P = \{1, 2, \ldots, n\}$
Repetir
Calcular Δ_i, $\forall i \in P \backslash S$
Calcular $\Delta_{\text{mín}}$, $\Delta_{\text{máx}}$
$L = \{i \in P \backslash S \ / \Delta_i \geq \alpha \Delta_{\text{máx}} - (1-\alpha)\Delta_{\text{mín}}\}$
Escoger $i^* \in L$ aleatoriamente
$S = S \cup \{i^*\}$ y asignar i^* al mejor grupo
Hasta que $|S| = n$

Donde

$$\Delta_i = a_{il_{\text{mín }1(i)}} - a_{il_{\text{mín }2(i)}}$$

$a_{il_{\text{mín }1(i)}}$ = mín$\{a_{il}$ tal que la asignación i a l sea factible para $l = 1, 2, \ldots, k\}$

$a_{il_{\text{mín }2(i)}}$ = mín$\{a_{il}, l \neq l_{\text{mín }1(i)}$, tal que la asignación i a l sea factible para $l = 1, 2, \ldots, k\}$

Una vez que han quedado conformados los grupos de nodos, el diseño de la ruta en cada grupo se realiza utilizando la heurística de inserción generalizada GENI [5].

B. Fase de búsqueda local

La solución obtenida en la fase constructiva se mejora con un procedimiento de búsqueda local que en cada iteración busca la mejor solución vecina de la solución actual y si ésta mejora la solución actual la reemplaza. El procedimiento finaliza cuando la solución actual no tiene ninguna solución vecina mejor que ella.

En este caso las soluciones vecinas se obtienen por cambios de cadenas de nodos de una ruta a otra (cambios entre-rutas) o en la misma ruta (cambios intra-rutas). Se consideran 3 tipos de cambios, concretamente uno intra-rutas y dos entre-rutas.

C_1) Cambio de una cadena dentro de una ruta.
C_2) Cambio de una cadena de una ruta a otra.
C_3) Intercambios de dos cadenas entre dos rutas diferentes.

Estos tres tipos de cambios están basados en dos operaciones, usadas de forma diferente en cada caso:
1) Eliminación de una cadena de una ruta.
2) Inserción de una cadena en una ruta.

En las siguientes tres figuras, sin pérdida de generalidad y para simplificar la notación, se identifican los puntos de visita con el orden que ocupan en la ruta. Los arcos que se eliminan se muestran con líneas de puntos, los que se añaden con guiones, y en los tramos que cambian de sentido se añade la nueva orientación debajo de la original.

Específicamente se tienen 3 tipos de eliminaciones:
i) Eliminación usual ó clásica, mostrada en la figura 1. La eliminación de la ruta de la cadena de nodos que se inicia en el punto i y contiene r puntos, supone la eliminación de los arcos $(i-1, i)$ y $(i+r-1, i+r)$, y la incorporación del arco $(i-1, i+r)$.

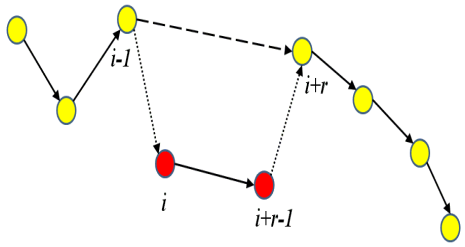

Fig. 1. Eliminación usual

ii) Eliminación tipo GENI-I, mostrada en la figura 2. La eliminación de la ruta de la cadena de puntos que se inicia en el punto i y contiene r puntos supone la eliminación de los arcos $(i-1, i)$, $(i+r-1, i+r)$, $(k, k+1)$ y $(j, j+1)$; la incorporación de los arcos $(i-1, k)$, $(i+r, j)$, $(k+1, j+1)$; y el cambio de sentido de los tramos de $i+r$ a k, y de $k+1$ a j. Por tanto, además de los parámetros i y r que definen la cadena a eliminar, este tipo de eliminaciones depende de otros 2 parámetros j y k, $(k \geq i+r, j \geq k+1)$.

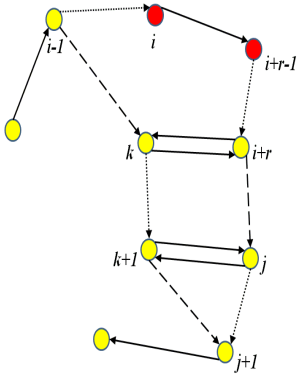

Fig. 2. Eliminación tipo GENI-I

iii) Eliminación tipo GENI-II, mostrada en la figura 3. La eliminación de la ruta de la cadena de puntos que se inicia en el punto i y contiene r puntos supone la eliminación de los arcos $(i-1, i)$, $(i+r-1, i+r)$, $(j-1, j)$, $(l, l+1)$ y $(k, k+1)$; la incorporación de los arcos $(i-1, k)$, $(l+1, j-1)$, $(i+r, j)$, $(l, k+1)$; y el cambio de sentido de los tramos de $l+1$ a k, y de $i+r$ a $j-1$. Por tanto, además de i y r, este tipo de eliminaciones depende de 3 parámetros j, l y k, $(j \geq i+r+1, l \geq j, k \geq l+1)$.

De manera similar, se consideran tres tipos de inserciones. En las tres siguientes figuras, igual que antes, se identifican los puntos de la ruta con el orden que ocupan. El primer y último punto de la cadena se identifican con x e y respectivamente. Los arcos que se eliminan se muestran con líneas de puntos, los que se añaden con guiones, y en los tramos que cambian de sentido se añade la nueva orientación debajo de la original.

i) Inserción usual ó clásica, mostrada en la figura 4. La inserción entre los puntos i e $i+1$, $(i \geq 1)$

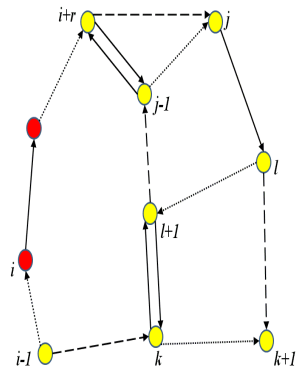

Fig. 3. Eliminación tipo GENI-II

de la cadena de vértices entre el x y y, supone la eliminación del arco $(i, i+1)$, y la incorporación de los arcos (i, x) e $(y, i+1)$.

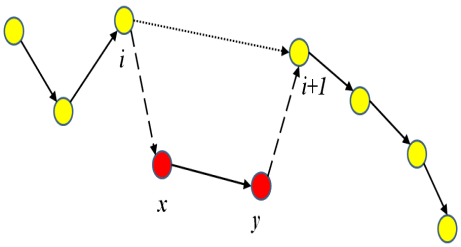

Fig. 4. Inserción usual

ii) Inserción tipo GENI-I, que se muestra en la figura 5. La inserción en la ruta de la cadena de x a y supone la eliminación de los arcos $(i, i+1)$, $(j, j+1)$ y $(k, k+1)$; la incorporación de los arcos (i, x), (y, j), $(i+1, k)$ y $(j+1, k+1)$; y el cambio de sentido de los tramos de $i+1$ a j, y de $j+1$ a k. Por tanto, además de la cadena a insertar, este tipo de inserciones depende de 3 parámetros i, j y k, $(i \geq 1, j \geq i+1, k \geq j)$.

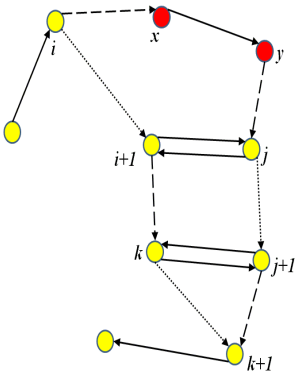

Fig. 5. Inserción tipo GENI-I

iii) Inserción tipo GENI-II
La figura 6 ilustra dicha inserción que supone: la eliminación de los arcos $(i, i+1)$, $(l-1, l)$, $(j, j+1)$ y $(k-1, k)$; la incorporación de los arcos (i, x), (y, j), $(l, j+1)$, $(k-1, l-1)$ y $(i+1, k)$; y el cambio de sentido de los tramos de l a j, y de $i+1$ a $l-1$. Por tanto, (además de la cadena a insertar) este tipo de inserciones dependen de 4 parámetros i, l, j y k, $(i \geq 1, l \geq i+2, j \geq l, k \geq j+2)$.

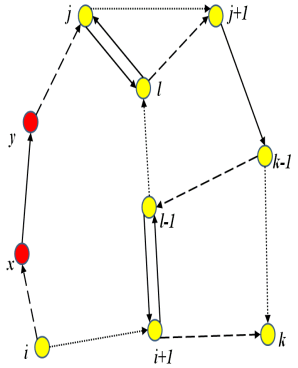

Fig. 6. Inserción tipo GENI-II

Hay que indicar que tanto para las eliminaciones como para las inserciones se consideran ambas orientaciones de las rutas implicadas.

A continuación se explican los tres tipos de cambios que dan lugar a las diferentes soluciones vecinas.

C_1) Cambio de una cadena dentro de una ruta
Supone la eliminación de una cadena de una ruta y su reinserción en la misma ruta. Para ello, la cadena seleccionada se elimina mendiante la eliminación clásica y a continuación se evalúan los tres tipos de inserciones: usual, GENI-I y GENI-II, en la ruta modificada.

C_2) Cambio de una cadena de una ruta a otra diferente
Supone la eliminación de una cadena de una ruta y su inserción en otra distinta. Dada una cadena se evalúan los tres tipos de eliminaciones (usual, GENI-I y GENI-II) y se considera la mejor de todas. A continuación se evalúan los tres tipos de inserciones en las otras rutas.

C_3) Intercambios de dos cadenas entre dos rutas diferentes
Supone la eliminación de las cadenas C_A y C_B de las rutas R_A y R_B respectivamente y la inserción de C_A en R_B y C_B en R_A. Sólo se considera la eliminación usual, pero sí se consideran los tres tipos de inserciones posibles.

El chequeo de la factibilidad de cada cambio con respecto a las restricciones del problema se ha incluido dentro de la evaluación del cambio.

Finalmente, es conveniente apuntar lo siguiente.

- Las eliminaciones tipo GENI-I incluyen a algunas de las eliminaciones usuales: concretamente cuando $k = i+r$ y $j = k+1$. Sin embargo no abarca a todas: cuando la cadena a eliminar finaliza en el último o penúltimo punto antes de la vuelta al origen, la eliminación usual no puede ser incluida dentro del tipo GENI-I ya que ésta al menos necesita 2 puntos distintos (j y k) entre el final de la cadena y la vuelta al origen. Se deben incluir las adaptaciones necesarias

en la correspondiente implementación para evitar duplicar cálculos.

- La misma reflexión anterior se hace con las inserciones.
- En el caso de los cambios C_1) y C_3) las rutas de donde se eliminan cadenas van a ser modificadas posteriormente con alguna inserción. Por tanto no tiene mucho sentido buscar y ejecutar la mejor eliminación de una cadena de una ruta si después esa ruta va a ser modificada. Además, fijando la cadena (cambio C_1) o par de cadenas a cambiar (tipo C_3) y considerando cada ruta concreta, hay una dependencia de la inserción en esa ruta de la eliminación que ha habido antes en la misma: la mejor combinación (inserción-eliminación) puede no corresponder a la mejor eliminación inicial. Sin embargo, el tiempo de cálculo para hallar esta mejor combinación puede ser excesivo dado la cantidad de parámetros que se manejan. Por tanto una solución razonable es considerar la eliminación usual.
- No ocurre lo mismo con el cambio tipo C_2), una vez que una cadena se elimina de su ruta, ésta no se modifica posteriormente. Por tanto para cada cadena, los procesos de eliminación e inserción son independientes. Entendemos que en este caso sí tiene sentido, y no supone un tiempo excesivo, elegir la mejor eliminación.
- Los cambios que se proponen generalizan las ideas expuestas en el trabajo de Gendreau, Hert & Laporte [5], donde se propone una heurística (GENIUS) para el TSP simétrico con buenos resultados.

Para evitar tiempos de cómputo excesivos, se ha utilizado la estrategia denominada Búsqueda Local Rápida expuesta con detalle en [2] y [10].

IV. Algunos resultados computacionales

El experimento computacional se divide en dos partes, la primera sobre instancias completamente aleatorias y la segunda sobre instancias similares al caso real.

Mostramos enseguida en la tabla I algunos resultados computacionales, obtenidos en 16 instancias aleatorias. Los nodos fueron generados aleatoriamente en la región $[-10, 10] \times [-10, 10]$ y la matriz de costos se forma con la distancia euclideana redondeada, las demandas se generaron entre 1 y 25 para cada nodo y la capacidad de los vehículos se fijó en 100, 250 ó 300. Se resolvió el modelo de forma exacta utilizando Cplex v.9.0 en una Sun Fire V440 Ultra Sparc III, 1062 GHz y el GRASP se ejecutó en una PC, 1GHz. Dado que el algoritmo se encuentra en fase de pruebas y ajustes para valorar su efectividad, y se requieren resultados en tiempos de cómputo razonables, el criterio de paro utilizado fue de 10 iteraciones sin mejora.

Se puede observar que el GRASP logra llegar a la solución óptima en un tiempo de cómputo muy corto en estas instancias pequeñas, incluso cuando Cplex no termina de ejecutarse durante varias horas, la cota superior obtenida es mejorada por el GRASP, que utiliza sólo algunos segundos. Estas pruebas fueron realizadas para validar que el software comercial tiene dificultades para resolver aún estas instancias pequeñas, mientras que el GRASP requiere pocos segundos para llegar a la solución óptima.

TABLA I
Resultados computacionales sobre instancias completamente aleatorias

				CPLEX		GRASP	
n	k	Q	F	Costo	Tiempo	Costo	Tiempo
10	3	100	1	157	1.84	157	0.201
	2	100	2	136	4.61	136	0.130
	2	100	3	147	37.58	147	0.621
	2	100	4	132	53.00	132	3.245
10	3	250	1	157	2.97	157	0.230
	2	250	2	119	8.35	119	0.150
	2	250	3	116	46.35	116	0.720
	1	250	4	78	20.90	78	1.230
12	3	300	1	184	20.39	184	0.070
	2	300	2	139	120.50	139	0.221
	2	300	3	135	1345.60	135	0.841
	1	300	4	92	23.80	92	4.640
15	3	300	1	253	2651	253	0.131
	2	300	2	183	11802	183	0.801
	2	300	3	183	19840*	182	3.024
	1	300	4	126	350	126	26.488

* truncado

En la segunda parte del experimento computacional se generaron instancias de la siguiente manera:

- Las localizaciones se sitúan geográficamente en diferentes poblaciones del norte de España, concretamente el depósito central se ubica en Briviesca, Burgos, donde la empresa tiene su fábrica. Las distancias entre localizaciones son las distancias por carreteras.
- La capacidad de los vehículos se fija en 30.
- La demanda de cada pedido se genera entre 1 y $Q_{\text{máx}}$, donde $Q_{\text{máx}} = 15, 22$.
- El número de pedidos, n, toma los valores $n = 42, 102, 184$.
- Para cada combinación de parámetros $Q_{\text{máx}}$ y n se generan 5 instancias y cada instancia se resuelve considerando $F = 0$ (ruteo día a día) y $F = 1$ (adelanto máximo de un día).

Se presentan los resultados en la tabla II.

TABLA II
Resultados medios del % de mejora de $F = 1$ sobre $F = 0$

n	$Q_{\text{máx}}$	medio	mínimo	máximo
42	15	17.4	16.0	18.6
42	22	12.9	8.1	17.2
102	15	15.0	12.3	17.9
102	22	10.0	8.6	11.2
185	15	13.0	11.4	13.7
185	22	10.0	8.4	11.6

Se observa de los experimentos que del ruteo día a día a considerar $F = 1$, que es permitir el adelanto

de un día en la entrega, se logra una reducción en la distancia a recorrer de alrededor del 10 % al 20 %.

En la tabla III se observa el desempeño de la búsqueda local. Se muestra el porcentaje de mejora obtenido para las soluciones del procedimiento constructivo y para las obtenidas después de realizada la búsqueda local, sólo para el caso de adelantar un día la fecha de entrega, esto es, $F = 1$.

TABLA III

Porcentaje de mejora de constructivo más búsqueda local sobre constructivo y tiempos de computación

			Tiempo	
n	$Q_{\text{máx}}$	% mejora	Const	Const+BL
42	15	3.10	0.02	1.63
42	22	2.18	0.04	2.52
102	15	3.47	0.27	83.54
102	22	2.75	0.46	62.88
185	15	3.29	2.00	874.18
185	30	4.09	3.60	539.39

V. Conclusiones

Se mostró una aplicación del problema de ruteo de vehículos donde se permite flexibilidad en las fechas de entrega de productos. Se presentó un modelo matemático, un método de solución apropiado basado en la metaheurísica GRASP y los primeros resultados computacionales para este método de solución. Los resultados preliminares muestran que el procedimiento obtiene buenas soluciones, alcanzando el óptimo en instancias pequeñas. Sin embargo, observando los porcentajes de mejora obtenidos por la búsqueda local, en comparación con el tiempo que consume, sugiere que es necesario profundizar en este aspecto.

En trabajo futuro se pretende abordar el problema desde el punto de vista multiobjetivo al considerar también los costos de almacenamiento al adelantar los pedidos.

Agradecimientos

Este trabajo fue parcialmente apoyado por el proyecto 61903 de CONACYT (México), por los proyectos SEJ2005-08923/ECON y ECO2008-06159/ECON del Ministerio Español de Ciencia y Tecnología y por la Universidad Autónoma de Coahuila (México). Los autores agradecen estos apoyos.

Referencias

[1] N. Christofides, J. Beasley, The Period Routing Problem, *Networks*, **14**, 237–256, 1984.

[2] C. Delgado, J. Pacheco, Resultados de diferentes experiencias con Búsqueda Local aplicadas a problemas de rutas, *Rect@ ASEPUMA*, **2**, 54–81, 2000.

[3] T. Feo and M. Resende, Greedy Randomized Adaptive Search Procedures, *Journal of Global Optimization*, **6**, 109–133, 1995.

[4] M. Fisher and R. Jaikumar, A Generalized Assignment Heuristics for Vehicle Routing Problem, *Networks*, **11**, 109–124, 1981.

[5] M. Gendreau, A. Hertz and G. Laporte New insertion and postoptimization procedures for the traveling salesman problem, *Operations Research*, **40**, 1086–1094, 1992.

[6] S. Martello and P. Toth, *Knapsack Problems Algorithms and Computer Implementations*, John Wiley & Sons, 1990.

[7] I. Or, *Traveling Salesman Type Combinatorial Problems and their relations to the logistics of blood banking.* Ph. Tesis, Department of Industrial Engineering and Management Sciences, Northwestern University, 1976.

[8] E. Taillard, P. Badeau, M. Gendreau and J.Y. Potvin, A Tabu Search heuristic for the Vehicle Routing Problem with Time Windows, *Transportation Science*, **31**, 170–186, 1997.

[9] P. Toth and D. Vigo, *The Vehicle Routing Problem*, SIAM, 2002.

[10] C. Voudouris and E. Tsang, Guided Local Search for the Traveling Salesman Problem, *European Journal of Operations Research*, **113**, 469–499, 1999.