27 Congreso Nacional de Estadística e Investigación Operativa
Lleida, 8-11 de abril de 2003

ADAPTACIÓN DE LA HEURÍSTICA LIN-KERNINGHAM PARA LA RESOLUCIÓN DE PROBLEMAS DE DISEÑO DE ITINERARIOS

J. Bautista[1], J. Pereira[2]

[1]Departamento de Organización de Empresas
Universitat Politècnica de Catalunya
E-mail: joaquin.bautista@upc.es

[2] Departamento de Organización de Empresas
Universitat Politècnica de Catalunya
E-mail: jorge.pereira@upc.es

RESUMEN

La heurística Lin-Kerninghan es conocida por su eficiencia para la resolución del problema del viajante de comercio (simétrico). En los últimos años, la heurística ha sido modificada, incluyendo nuevos tipos de intercambios, definición de vecindarios y procedimientos de reinicio que amplían su espacio de búsqueda. Actualmente, esta heurística se considera el procedimiento más eficaz para resolver el problema. En el presente trabajo, se muestra una implementación de dicha heurística adaptada para resolver el problema de diseño de itinerarios (VRP), y se compara con otros procedimientos existentes. Se realiza una experiencia computacional apoyada en una colección de ejemplares de referencia.

Palabras y frases clave: TSP, Heurísticas, VRP.
Clasificación AMS: 90C26, 90C59, 90B06

1. Introducción

Aunque la heurística de Lin-Kernighan, Lin y Kernighan (1973), para el problema del viajante de comercio (TSP) data del año 1973, es considerada, todavía, la mejor heurística disponible para el problema. A pesar de que la heurística fue diseñada para la resolución de problemas que ahora se consideran de tamaño reducido, el incremento de velocidad de computación en los ordenadores ha permitido la implementación de refinamientos en la heurística y la inclusión de nuevos elementos hasta tal punto que las mejores implementaciones existentes obtienen de forma regular el óptimo para todos

aquellos problemas con solución optima conocida. En la actualidad, el problema de mayor tamaño resuelto óptimamente mediante el método más eficiente disponible, basado en branch and cut, Applegate et al. (1998), Applegate et al. (2001), es un problema de 15112 ciudades, con una solución idéntica a la ofrecida por una de las últimas implementaciones aparecidas de la heurística, Helsgaun (2000).

Desafortunadamente, las mejoras realizadas para el problema del viajante de comercio simétrico no han sido trasladadas, o se han trasladado con poco éxito, a otros problemas de diseño de itinerarios, véase por ejemplo Kanellakis y Papadimitriou (1980) para aplicaciones de la heurística al problema de viajante de comercio asimétrico. Esto se debe a las diferencias existentes en el espacio de soluciones del viajante de comercio simétrico y otros problemas de diseño de itinerarios, diferente estructura de distancias y restricciones adicionales principalmente, que impiden una conversión directa de los procedimientos de resolución para el problema del viajante a otros problemas.

El presente trabajo muestra una implementación del algoritmo de Lin-Kerninghan específicamente adaptada a la resolución del problema de diseño de itinerarios (VRP) con limitaciones de capacidad. El problema puede definirse como un problema teórico en grafos con la siguiente formulación - véase Toth,Vigo (2002)-. Sea $G=(V,A)$ un grafo completo dónde $V=\{0,...,n\}$ es el conjunto de vértices y A es el conjunto de arcos. Los vértices $i=1,...,n$ corresponden a los clientes, mientras que el vértice 0 corresponde al depósito. Cada arco $(i,j)\in A$ tiene asociado un coste no negativo c_{ij}, que representa el coste de viajar desde el vértice i al vértice j. Cada cliente, vértice, tiene asociado una demanda conocida no negativa d_i, que debe ser transportada desde, o transportada a, el vértice 0. Para realizar esta tarea se dispone de K vehículos idénticos con capacidad C (superior a la demanda de cualquier vértice) y cada vehículo puede realizar como máximo una ruta de entrega, recogida. El objetivo del problema es encontrar un conjunto de exactamente K circuitos de coste mínimo, cuyo valor de la solución se define como la suma de los costes de los circuitos; además, cada circuito debe pasar por el vértice depósito, cada cliente debe estar en un circuito y la suma de las demandas cubiertas por un circuito no debe superar la capacidad del vehículo.

El contenido de lo que resta de trabajo está ordenado así: en el apartado dos se muestra el algoritmo original; en apartado tres se comentan las modificaciones más importantes a este algoritmo realizadas durante los últimos años; el apartado cuatro se centra en la implementación realizada, así como en las modificaciones necesarias para tratar las características especiales del problema de diseño de itinerarios (VRP); en el apartado cinco se presentan los resultados de una experiencia computacional con nuestra propuesta y se comparan los resultados obtenidos por el algoritmo con otras heurísticas para el problema; finalmente, en el apartado seis, se detallan las conclusiones del trabajo.

2. La heurística de Lin y Kernighan

La heurística de Lin y Kernighan se basa en la aplicación concatenada de intercambios simples que permite una exploración más amplia del espacio de soluciones que la que se realiza con la aplicación iterativa de dichos intercambios.

Los primeros antecedentes de esta heurística provienen de los trabajos sobre procedimientos de búsqueda local aplicados al problema del viajante de comercio. En los años 50 y 60, se desarrollan los primeros heurísticos para la resolución del problema, los cuales aportarán diversos conceptos utilizados en posteriores heurísticos para el problema. Entre ellos cabe destacar el intercambio 2-opt, debido a Croes, Croes (1958), y el concepto de n-optimalidad.

En intercambio 2-opt se eliminan dos arcos del problema y se sustituyen por otros dos arcos de tal manera que se forme un único circuito cerrado cuyo coste sea inferior al coste de la solución original. En la figura 1 se muestra un esquema de lo dicho.

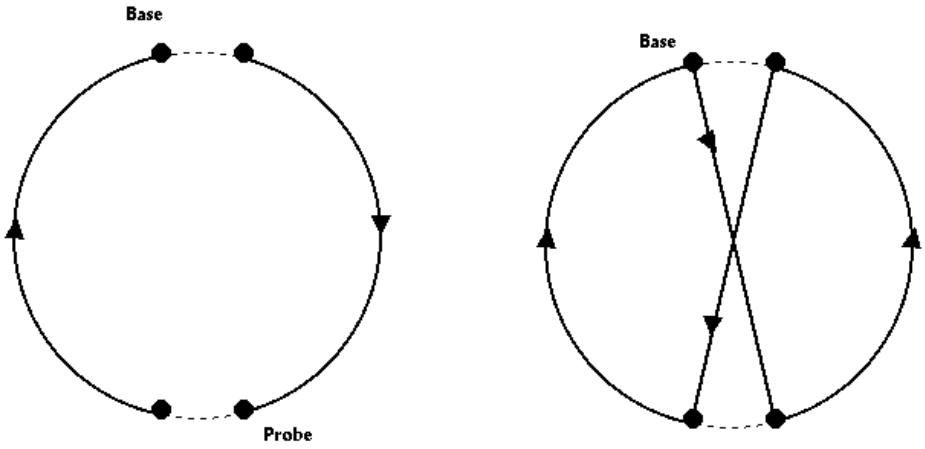

Figura 1: Un intercambio 2-opt. En el intercambio se eliminan los dos arcos mostrados en línea discontinua y se substituye por los dos nuevos arcos creando un único circuito.

Para realizar un intercambio 2-opt que mejore la solución inicial, se hallan dos arcos del ciclo y se sustituyen por otros dos arcos cuya suma de costes sea inferior a la de los originales. Un movimiento “provechoso” 2-opt a partir de un vértice “base” consiste en encontrar un vértice “probe” tal que:

$$c(Base, Next(Base)) + c(\mathrm{Pr}obe, Next(\mathrm{Pr}oble)) \geq$$
$$c(Base, \mathrm{Pr}obe)) + c(Next(Base), Next(\mathrm{Pr}obe))$$

Dónde *c(vértice1,vértice2)* equivale al coste asociado a un arco representado por sus vértices.

El procedimiento de mejora local resultante de la aplicación de intercambios 2-opt a una solución inicial del problema (hasta que ningún intercambio 2-opt mejore la solución en curso) genera una solución 2-óptima; ésta es una solución tal que la aplicación de cualquier intercambio 2-opt no produce mejoras y por tanto es localmente óptima.

Otro procedimiento, denominado Or-Opt -propuesto por diversos autores, entre ellos véase Bock (1958)-, consiste en extraer una subsecuencia de ciudades de la solución e insertarla en medio de dos ciudades de la subsolución restante en busca de la reducción del coste. Estos intercambios se emplean en varias heurísticas de inserción, tal como la de Geni, Gendreau, Hertz y Laporte (1992), aparecida posteriormente, y producen soluciones no alcanzables mediante un procedimiento basado en 2-opt.

Lin (1965) desarrolla intercambios de tipo 3-opt, consistentes en la sucesión de dos intercambios 2-opt que sí son capaces de obtener las mismas soluciones halladas con intercambios Or-opt. En la figura 2 se muestra un esquema de este tipo de intercambio.

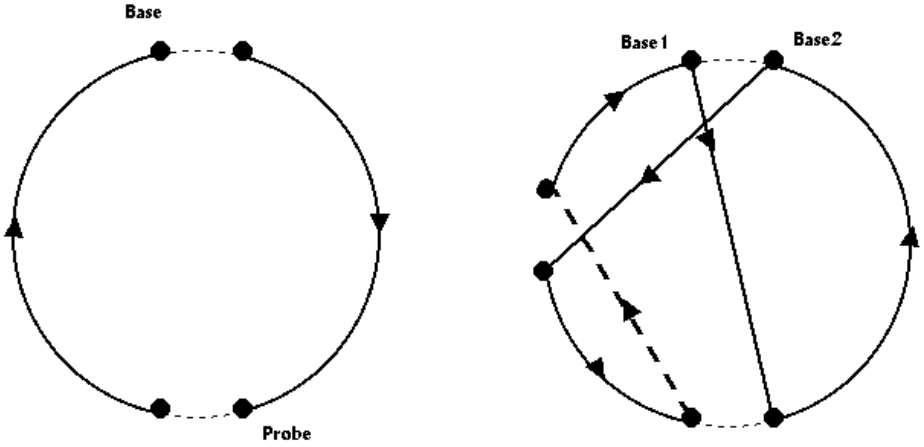

Figura 2: Un intercambio 3-opt. En el intercambio se eliminan tres arcos y se substituye por tres nuevos arcos creando un circuito cerrado.

Al igual que en el caso anterior, se denomina solución 3-óptima a aquella que no se puede mejorar por un intercambio 3-opt. Es preciso advertir que una solución 3-óptima no necesariamente es 2-óptima; por ello, en parte, la heurística de Lin y Kernighan extiende y generaliza ambos intercambios proponiendo el uso de intercambios λ-opt que generan soluciones λ-óptimas, donde λ es variable. La heurística resultante es conocida como LK, y así se denominará en el resto del presente trabajo.

La heurística LK realiza una secuencia de intercambios 2-opt encadenados de manera eficiente, sin investigar aquellos intercambios posibles que no tengan posibilidades de

mejorar la solución de partida. Hacemos notar que cualquier movimiento λ-opt puede realizarse como una secuencia de movimientos 2-opt, en la que algunas soluciones intermedias pueden ser infactibles y/o tener un coste superior al de la solución inicial (aunque para aceptar dicha secuencia de intercambios, la solución final debe tener un valor inferior al de la solución inicial). Si la solución final es mejor que la de partida, ésta pasa a ser la solución inicial y se inicia el proceso de búsqueda; en caso contrario, estos intercambios son rechazados y se procura que no se repita la misma secuencia de movimientos no provechosos. El procedimiento termina cuando no existe ningún intercambio que mejore la solución.

El método utilizado para evitar búsquedas no provechosas consiste en considerar un criterio de ganancia que determina de forma heurística si la solución construida puede mejorar o no la solución de partida. Dicho método consiste en calcular el coste de los arcos eliminados e insertados de tal manera que su suma-resta siempre sea positiva (como el algoritmo siempre tiene un arco menos insertado que eliminado menos cuando se acaba de implementar la secuencia de intercambios, se acepta un valor constante de "empeoramiento" durante una secuencia de intercambios). La ganancia *g*, por tanto, es igual a la suma de los costes asociados a los arcos eliminados menos la suma de los costes asociados a los arcos insertados.

A continuación se describe el algoritmo tal como fue propuesto por los autores:

1. Generar una solución inicial T al azar o mediante una heurística constructiva.
2. Sea G*=0 [G* es la reducción de coste mejor encontrada hasta el momento]. Escoger cualquier nodo t_1 y sea x_1 uno de los dos arcos de T adyacentes a t_1. Sea i=1
3. Desde el otro vértice del arco, t_2, de x_1, escoger y_1 que une t_2 con t_3 con $g_1>0$. Si no existe, ir al paso 6(d). [Aquí se aplica el criterio de ganancia].
4. Sea i=i+1. Escoger x_i [que en la actualidad une t_{2i-1} con t_{2i}] e y_i como sigue:
    a. x_i se escoge de tal manera que si t_{2i} se reúne con t_1, la configuración resultante es un camino. Este elemento permite la aplicación del criterio de factibilidad de la solución que garantiza que siempre se puede "cerrar" un camino si se desea, simplemente uniendo t_{2i} con t_1 para cualquier $i\geq 2$. La elección de y_{i-1} condicionada por el paso 4(e) asegura que siempre existe un x_i que lo cumpla.
    b. Y_i es un arco en el final de t_{2i} compartido con x_i, sujeto a (c), (d) y (e). Si no existe y_i, ir al paso 5.
    c. Para garantizar que las x's e y's son dos conjuntos disjuntos, x_i no puede ser un arco anteriormente unido (eg. A y_i, $j<i$), y similarmente y_i no puede ser un enlace anteriormente roto.
    d. $G_i=\Sigma g_j>0$
    e. Para asegurar que el criterio de factibilidad de (a) puede ser satisfecho en i+1, y_i debe ser escogido de tal manera que permita la ruptura de un x_{i+1}.

f. Antes de que y_i sea construido, miramos si cerrar el camino uniendo t_{2i} con t_1 resulta en un valor de ganancia positivo mejor que el que se tiene guardado. Sea y_i^* el arco conectando t_{2i} con t_1, y sea $g_i^* = |y_i^*| - |x_i|$. Si $G_{i-1}+g_i^* > G^*$, entonces $G^* = G_{i-1}+g_i^*$ y k=i. [G^* siempre es la mejor mejora de T realizada hasta el momento].

5. Terminar la construcción de x_i y y_i en los pasos 2 a 4 cuando no existe ningún otro enlace x_i e y_i que satisfagan 4(c)-(e) o cuando $G_i \leq G^*$ y repetir el proceso completo desde el paso 2, usando T' como el camino inicial.
6. Si $G^*=0$, se permite un cierto nivel de backtracking como sigue:
   a. Repetir los pasos 4 y 5, escogiendo y_2 en orden según longitud mayor, mientras se mantenga el criterio de ganancia $g_1+g_2>0$. [Cualquier mejora hacer volver al paso 2].
   b. Si todas las elecciones de y_2 en el paso 4(b) se prueban sin resultados, volver al paso 4(a) e intentar un nuevo valor de x_2.
   c. Si esto también no consigue mejorar, se vuelve al paso 3 para probar diferentes y_1 en orden según longitud mayor.
   d. Si y_1 también es agotado sin provecho, probar un x_1 alternativo
   e. Si todo esto falla, escoger una nueva t_1 y repetir desde el paso 2.
7. El procedimiento termina cuando todos los n posibles valores de t_1 han sido examinados sin provecho. En este momento podemos iniciar la búsqueda desde otro punto de partida en el paso 1.

En definitiva, la heurística LK puede verse como un procedimiento que realiza, a la vez, operaciones de búsqueda en anchura (el número de candidatas tratadas para cada ciudad) y en profundidad (las operaciones de backtracking). Con esta misma idea, se han propuesto modificaciones tanto en la forma de tratar aspectos relacionados con la anchura, la composición de la lista de candidatas para cada ciudad y el número de intercambios a probar en cada ocasión, con otros aspectos relacionados con la profundidad de la búsqueda, así como formas de reiniciar la búsqueda una vez se ha alcanzado la optimalidad local. La siguiente sección trata las modificaciones más relevantes sobre la heurística original.

3. Modificaciones de la heurística original

3.1 Definición de ciudades candidatas

Lin y Kernighan definen el conjunto de arcos candidatos para cada ciudad mediante una lista de arcos compuesta por la ciudad y sus más cercanas utilizando como medida de cercanía la distancia entre aquella y éstas. Desafortunadamente, se ha podido apreciar que este procedimiento dejaba algunos arcos de las soluciones óptimas fuera de la lista de arcos candidatos. Por ello se han propuesto otros procedimientos para determinar la lista de candidatos según el coste marginal asociado a la penalización en la solución en que se incurre por sustituir un arco por otro.

Entre las diferentes medidas adoptadas, se han propuesto algunas basadas en penalizar la incorporación de un arco a la solución óptima de un problema relajado del viajante de comercio, que a su vez es más sencillo de resolver, como el del árbol parcial mínimo o el *1-tree* de un grafo, y otros basados en cercanía geométrica, como los basados en las triangulaciones de Delauny - ver Lawler (1985)-.

3.2 Amplitud de la búsqueda

El algoritmo original propone un tipo de intercambio 2-opt para el problema, así como un intercambio denominado “double-bridge” (que se comenta posteriormente) aplicado como post-optimización. Adicionalmente se consideran aquellos intercambios 2-opt que, a pesar de generar dos ciclos, se pueda aplicar un segundo intercambio 2-opt que genere un único ciclo. Este tipo de intercambio se denomina no-secuencial, ya que no pueden representarse mediante un número finito de 2-opt intercambios que mantengan una solución factible durante todo el proceso. En la figura 4 se muestra un ejemplo de un intercambio de este tipo.

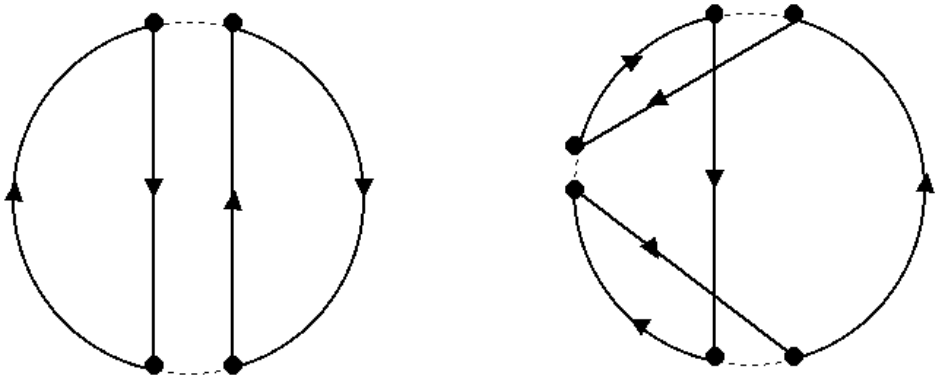

Figura 3: 3-opt con un primer 2-opt que genera una solución no factible.

Mak y Morton (1993), en un intento de simplificar la implementación de la heurística, eliminando los intercambios 2-opt no secuenciales, y de añadir amplitud en cada uno de los pasos, proponen un nuevo tipo de 2-opt que permite alcanzar soluciones generadas por intercambios 3-opt no secuenciales a través de dos intercambios 2-opt del nuevo tipo. El intercambio propuesto se basa en no considerar la conexión entre base y probe para examinar la calidad del intercambio, sino considerar la conexión existente entre Next(base) y Prev(probe), transformándose Next(probe) en la base para el siguiente intercambio. Este intercambio se puede aplicar conjuntamente con el original, y duplicar así el espacio de búsqueda en cada paso. Este procedimiento ha sido utilizado desde entonces en la mayoría de implementaciones, aunque se utilice también el movimiento

3-opt con solución intermedia no factible, ya que el procedimiento puede converger más rápidamente con su presencia. En la figura 3 se muestra el intercambio.

Otros intercambios interesantes son los de tipo “double-bridge” que se han citado con anterioridad. Un intercambio tipo “double-bridge” es un caso especial de 4-intercambio no secuencial, puede verse un ejemplo en la figura 4. La propuesta original los utilizaba como post-proceso de mejora, mientras que algunas propuestas actuales los consideran una vez han agotado los intercambios secuenciales partiendo de una ciudad. Esta misma idea también ha sido utilizada en la aplicación encadenada del algoritmo que se muestra posteriormente.

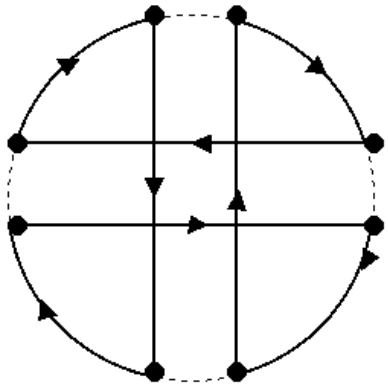

Figura 4: Ejemplo de un intercambio “double-bridge”.

3.3 Profundidad de la búsqueda

La propuesta original comprobaba un máximo de cinco candidatas para cada ciudad en un 2-opt intercambio, cinco para los 3-opt intercambios y un máximo de uno para cada λ-opt intercambios posteriores. Con los aumentos de velocidad de computación, se han probado amplitudes mayores para profundidades de 4-opt y 5-opt intercambios, mientras que se han ido abandonando las pruebas de λ-opt intercambios en las que sólo se consideraba un candidato para el intercambio.

3.4 Aplicación iterativa del procedimiento

Una de las primeras mejoras que puede aplicarse al procedimiento es el uso iterativo de éste, utilizando soluciones diferentes como solución de partida. Este procedimiento, fue utilizado como una forma de ampliar la búsqueda una vez se llegaba al óptimo local hasta la aparición de la variante encadenada que se describe a continuación.

3.5 Aplicación encadenada del procedimiento

Esta idea se debe a Martin, Otto y Felten -ver Martin, Otto y Felten (1991) y Martin, Otto y Felten (1992)-. La idea de reiniciar la búsqueda desde una solución inicial diferente cada vez que se obtenía el óptimo local no era una buena forma de explorar las soluciones óptimas locales producidas por el algoritmo. La estrategia propuesta como alternativa era el uso de movimientos denominados "kicks", los cuales introducían perturbaciones en la solución difícilmente reparables por los intercambios locales secuenciales y volver a resolver el problema mediante la heurística. El "kick" propuesto era un intercambio tipo "double-bridge", ver figura 4, que aunque empeorase la solución de partida, permitía una nueva aplicación del algoritmo LK, generando soluciones diferentes a la original.

4. Implementación propuesta de la heurística

La descripción de la implementación propuesta se ha dividido en dos partes. En una primera parte se tratan las modificaciones implementadas sobre la heurística básica y propuestas con anterioridad por otros autores, mientras que en una segunda fase se estudian las modificaciones necesarias para adaptar la heurística al problema de diseño de itinerarios.

4.1 Modificaciones respecto la heurística original

La versión implementada hace uso de los intercambios de Mak-Morton así como de los originales, pero no utiliza ni los intercambios "double-bridge" ni el intercambio 3-opt formado por un 2-opt no secuencial y un 2-opt de "reparación", denominado así porque vuelve a transformar la solución en una solución factible. Con el uso de ambos intercambios se consigue una mayor amplitud en la búsqueda que el algoritmo original, aunque inferior a la mayoría de implementaciones actuales, que sí hacen uso de los intercambios 2-opt no secuenciales. El algoritmo utiliza una definición de vecindarios mediante la distancia entre dos ciudades.

La profundidad de la búsqueda como en otras implementaciones se deja como un parámetro a elección del usuario, aunque se utiliza la propuesta de mantener constante el número de candidatas en cada paso de la búsqueda en profundidad y no explorar con un ancho reducido una vez se llega al final de esta profundidad. Como elemento de reinicio de la búsqueda se han implementado las dos propuestas existentes en la literatura, la realización de un movimiento de "kick" y la generación de otra solución de partida inicial generada mediante el algoritmo greedy de ciudades más cercanas en que la primera ciudad se escoge de forma aleatoria. Finalmente se ha optado por el uso de una técnica de reducción de intercambios, en cada iteración del algoritmo, basada en examinar un conjunto reducido de ciudades como base de los intercambios. La implementación mantiene, para cada vértice, una variable donde se guarda si se ha obtenido una mejora o no partiendo del vértice en la llamada al procedimiento. Si no es

así, no se vuelve a tratar de mejorar la ruta desde este vértice, salvo en circunstancias especiales.

4.2 Modificaciones de la heurística para la resolución del problema de diseño de itinerarios

Las principales diferencias existentes entre el problema del viajante de comercio simétrico y el problema de diseño de itinerarios (VRP) es la existencia de más de un ciclo en las soluciones, así como la existencia de restricciones adicionales asociadas a la capacidad de cada vehículo que realiza el servicio.

La primera diferencia, la existencia de más de un ciclo, puede subsanarse con la inserción de un número adicional de vértices depósito igual al número de vehículos existentes para el reparto menos uno; se construye así un grafo ampliado. Los costes asociados a los arcos en el grafo ampliado son los costes asociados a los arcos del grafo original, mientras que aquellos arcos que unen los vértices ficticios con los vértices cliente tendrán un coste igual al existente entre los arcos asociados al depósito original con dicho cliente, y los arcos asociados a dos vértices depósito tendrán un coste asociado infinito. Dado un circuito que recorra cada uno de los vértices en el grafo ampliado, las rutas asociadas a cada vehículo se compondrán por todos los arcos pertenecientes al circuito entre dos parejas de vértices depósito que, para ser una solución válida del problema, deberán respetar las restricciones de carga. Esta representación de las soluciones del VRP permite codificar todas las soluciones factibles del problema fácilmente.

La segunda diferencia, restricciones de capacidad, debe tenerse en cuenta durante el procedimiento de construcción de la solución inicial, así como durante el procedimiento de búsqueda. La propuesta implementada para la generación de una solución inicial resuelve heurísticamente un problema de empaquetado de clientes las rutas, asegurando así una solución inicial factible con dichas restricciones. Este procedimiento, en un primer paso, realiza una asignación al azar de los clientes a cada vehículo, seguido por un procedimiento de mejora local basado en una serie intercambios que asigne los clientes a las rutas de tal manera que no violen las restricciones de capacidad. En caso de no poder generar una solución factible, se realiza una nueva asignación al azar de los clientes, seguida por la mejora local, hasta que se halla una solución factible. Una vez asignados los clientes a las rutas, éstas se generan siguiendo un procedimiento greedy en que se visitan las ciudades más cercanas partiendo del vértice depósito. Una vez conocida la composición de las rutas, se unen en un único circuito que permite, a partir de entonces, a la versión adaptada del algoritmo LK trabajar con un único circuito.

La propuesta adoptada en la implementación permite un cierto nivel de infactibilidad en las soluciones, aunque no se basa en la penalización de dichas soluciones en la función objetivo. En vez de ello se permite que las secuencias de 2-opt con ganancias positivas continúen siendo estudiadas hasta la profundidad máxima que se permita a la heurística

LK sin tener en cuenta las restricciones de factibilidad, aunque la solución final después de aplicar la secuencia completa de 2-opt deba ser factible. Cuando una secuencia de intercambios 2-opt genera una secuencia con coste inferior pero no factible por restricciones de capacidad y no se ha llegado a la profundidad máxima de exploración, se permite continuar la secuencia de intercambios 2-opt.

5. Experiencia computacional

Para comparar los resultados ofrecidos por el algoritmo con otros procedimientos se ha utilizado un conjunto de 19 ejemplares de referencia con un número de clientes entre 50 y 385 y un número de rutas entre 5 y 47.

Parámetros	Desviación (10 iteraciones)	Desviación (50 iteraciones)
(0,0,5)	0.866 (0)	0.931 (1)
(0,0,6)	0.894 (0)	0.945 (1)
(0,0,7)	0.920 (1)	0.953 (3)
(0,0,8)	0.930 (1)	0.960 (3)
(0,0,9)	0.937 (0)	0.961 (2)
(0,0,10)	0.936 (2)	0.964 (4)
(0,1,5)	0.940 (0)	0.970 (4)
(0,1,6)	0.950 (2)	0.970 (4)
(0,1,7)	0.950 (1)	0.970 (5)
(0,1,8)	0.950 (0)	0.970 (4)
(0,1,9)	0.952 (1)	0.974 (6)
(0,1,10)	0.951 (1)	0.973 (4)
(1,0,5)	0.885 (0)	0.920 (0)
(1,0,6)	0.900 (0)	0.934 (0)
(1,0,7)	0.906 (0)	0.934 (0)
(1,0,8)	0.922 (0)	0.945 (0)
(1,0,9)	0.929 (1)	0.950 (1)
(1,0,10)	0.932 (1)	0.958 (1)
(1,1,5)	0.933 (0)	0.964 (2)
(1,1,6)	0.931 (0)	0.961 (3)
(1,1,7)	0.932 (0)	0.965 (2)
(1,1,8)	0.939 (1)	0.965 (3)
(1,1,9)	0.942 (0)	0.961 (1)
(1,1,10)	0.940 (1)	0.964 (2)

Tabla 1: Resultados ofrecidos por el procedimiento con una profundidad máxima de 3 intercambios

Para todos los ejempalres, se ha considerado la diferencia ofrecida, en tanto por uno, por la mejor solución obtenida en dos lanzamientos del procedimiento con cada combinación de parámetros propuestos (número de candidatas entre 5 y 10, profundidad

de los intercambios entre 3 y 5, procedimiento de cierre fuerte o no y reinicio iterado o encadenado) y la mejor solución conocida por cada problema con distancias reales sin redondeo y con dos decimales de exactitud, tal como se reporta en Taillard (1993), y consideradas quasi-óptimas.

Para definir cada configuración de parámetros se ha utilizado una terna (*X*,*Y*,*Z*) dónde *X* representa el encadenamiento de las llamadas al procedimiento LK (0) ó el uso iterativo del procedimiento (1), *Y* representa el uso de una lista restringida de candidatos para los depósitos (0) o el uso de todas las ciudades como lista de candidatos de los vértices depósito (1), y *Z* representa el tamaño de la lista de candidatos en cada ciudad. La tabla 1 muestra los resultados obtenidos por cada uno de las combinaciones de los parámetros permitiendo una profundidad de búsqueda de tres intercambios, así como el número de ejemplares en que se haya una solución a menos de un 1% de la mejor solución conocida (entre paréntesis).

La tabla 2 muestra el mejor valor obtenido por cualquiera de los procedimientos 3 intercambio y la mejor solución conocida para cada problema.

Problema	Mejor	Obtenida	%	Problema	Mejor	Obtenida	%
c50	524.61	524.61	1.000	tai100a	2041.336	2084.94	0.979
c75	835.26	850.04	0.983	tai100b	1940.61	1950.19	0.995
c100	826.14	827.39	0.998	tai100c	1406.202	1412.22	0.996
c120	1042.11	1060.33	0.983	tai100d	1581.25	1610.45	0.982
c150	1028.42	1048.33	0.981	tai150a	3055.23	3155.96	0.968
c199	1291.45	1332.90	0.969	tai150b	2727.77	2755.15	0.990
tai75a	1618.36	1626.38	0.995	tai150c	2341.84	2392.15	0.979
tai75b	1344.64	1409.16	0.954	tai150d	2645.39	2677.39	0.988
tai75c	1291.01	1293.86	0.998	tai385	24431.44	25461.7	0.960
tai75d	1365.42	1365.42	1.000				

Tabla 2: Mejores soluciones obtenidas por el procedimiento con una profundidad máxima de 3 intercambios

La tabla 3 muestra los resultados obtenidos por cada uno de las combinaciones de los parámetros permitiendo una profundidad de búsqueda de 4 intercambios, así como el número de ejemplares en que se haya una solución a menos de un 1% de la mejor solución conocida (entre paréntesis).

Los tiempos de ejecución de cada ejecución no superan en ningún caso los 10 minutos, todos ellos para el ejemplar más grande tai385 con 385 ciudades y 47 rutas) no superando el minuto en ninguno de los otros ejemplares.

Parámetros	Desviación (10 iteraciones)	Desviación (50 iteraciones)
(0,0,5)	0.919 (0)	0.954 (3)
(0,0,6)	0.939 (3)	0.966 (6)
(0,0,7)	0.945 (1)	0.975 (6)
(0,0,8)	0.957 (2)	0.977 (7)
(0,0,9)	0.964 (3)	0.983 (7)
(0,0,10)	0.966 (5)	0.984 (8)
(0,1,5)	0.968 (0)	0.989 (10)
(0,1,6)	0.977 (5)	0.989 (10)
(0,1,7)	0.966 (1)	0.985 (10)
(0,1,8)	0.968 (5)	0.987 (8)
(0,1,9)	0.978 (4)	0.988 (10)
(0,1,10)	0.977 (6)	0.987 (10)
(1,0,5)	0.927 (1)	0.947 (3)
(1,0,6)	0.941 (2)	0.960 (3)
(1,0,7)	0.949 (2)	0.973 (5)
(1,0,8)	0.950 (2)	0.974 (6)
(1,0,9)	0.967 (3)	0.976 (6)
(1,0,10)	0.963 (1)	0.977 (5)
(1,1,5)	0.961 (2)	0.982 (7)
(1,1,6)	0.963 (3)	0.983 (6)
(1,1,7)	0.971 (5)	0.984 (9)
(1,1,8)	0.967 (3)	0.984 (8)
(1,1,9)	0.972 (5)	0.985 (7)
(1,1,10)	0.967 (3)	0.985 (8)

Tabla 4: Resultados ofrecidos por el procedimiento con una profundidad máxima de 4 intercambios

La tabla 4 muestra el mejor valor obtenido por cualquiera de los procedimientos 4 intercambio y la mejor solución conocida para cada ejemplar.

Problema	Mejor	Obtenida	%	Problema	Mejor	Obtenida	%
c50	524.61	524.611	1.000	tai100a	2041.336	2051.2	0.995
c75	835.26	839.011	0.996	tai100b	1940.61	1940.7	1.000
c100	826.14	828.599	0.997	tai100c	1406.202	1406.89	1.000
c120	1042.11	1043.71	0.998	tai100d	1581.25	1592.66	0.993
c150	1028.42	1040.23	0.989	tai150a	3055.23	3078	0.993
c199	1291.45	1323.54	0.976	tai150b	2727.77	2737.45	0.996
tai75a	1618.36	1619.28	0.999	tai150c	2341.84	2370.89	0.988
tai75b	1344.64	1360.06	0.989	tai150d	2645.39	2663.32	0.993
tai75c	1291.01	1291.02	1.000	tai385	24431.44	25159	0.971
tai75d	1365.42	1365.42	1.000				

Tabla 4: Mejores soluciones obtenidas por el procedimiento con una profundidad máxima de 4 intercambios

Los tiempos de ejecución de cada ejecución no superan los 10 minutos, menos en el ejemplar tai385, con unos tiempos de ejecución que llegaron a una hora para la configuración (0,1,10) y mínimo de 15 minutos para la configuración (1,1,5), siendo inferior el tiempo de ejecución de los procedimientos encadenados que sus versiones iteradas.

En conjunto, el algoritmo resulta competitivo en tiempos de ejecución y resultados obtenidos, aunque sufre de problemas de escalabilidad en aquellas instancias con un número elevado de rutas.

6. Conclusiones

El procedimiento presentado puede considerarse competitivo con otros procedimientos existentes en la literatura tanto a nivel de tiempo de ejecución como calidad de los resultados. Si bien el algoritmo se muestra menos eficiente en aquellos ejempalres con un mayor número de rutas, la solución obtenida por la mejor combinación de parámetros para esta instancia está a menos de un tres por ciento de la mejor solución conocida. Aún así, la definición de vecindarios y el tratamiento de las rutas son puntos que podrían mejorar la eficiencia del algoritmo.

Referencias

D. Applegate, R. Bixby, V. Chvátal, W. Cook (1998) On the solution of traveling salesman problems. *Documenta Mathematica Journal der Deutschen Mathematiker-Vereinigung International Congress of Mathematicians* 645-656.

D. Applegate, R. Bixby, V. Chvátal, W. Cook (2001) ” TSP cuts which do not conform to the template paradigm. *In Computational Combinatorial Optimization, M. Junger and D. Naddef, editors* (2001), pp. 261-304.

F.Bock (1958). Research report, Armour Research Foundation. *Operations Research Society of America Fourteenth National Meeting, St. Louis*, 24-10-58

G. Clarke y J.V. Wright (1964) Scheduling of vehicles from a central depot to a number of delivery points. *Opers. Res.* 12, 568-581.

G.A. Croes (1958) A method for solving traveling-salesman problems, *Opers. Res.* 5, 791-812

M. Gendreau, A. Hertz y G. Laporte (1992) New insertion and post-optimization procedures for the traveling salesman problem, *Opers. Res.* 40, 1086-1094.

M. Gendreau, A. Hertz y G. Laporte (1994) A tabu search heuristic for the vehicle routing problem. *Management Science*, 40, 1276-1290.

K. Helsgaun, (2000) An Effective Implementation of the Lin-Kernighan Traveling Salesman Heuristic, *European Journal of Operational Research* 126 (1), 106-130.

P.C. Kanellakis y C.H. Papadimitriou (1980) Local Search for the Asymmetric traveling salesman problem. *Oper. Res.* 28. 1066-1099.

E.L.Lawler ed. (1985) The Traveling Salesman Problem: A Guided Tour of Combinatorial Optimization (Wiley-Interscience Series in Discrete Mathematics), John Wiley & Sons
S.Lin (1965) Computer solutions of the traveling salesman problem, *The Bell System Technical Journal* 44, 2245-2269.
S.Lin y B.W.Kernighan, (1973) An effective heuristic algorithm for the traveling-salesman problem, *Opers. Res.*, 21. 498-516.
K.-T. Mak y A.J.Morton (1993) "A modified Lin-Kernighan traveling-salesman heuristic", *Opers. Res. Lett.* 13, 127-132.
O.Martin, S.W. Otto y E.W.Felten (1991) Large-step Markov chains for the traveling salesman problem, *Complex Systems* 5, 299-326.
O.Martin, S.W. Otto y E.W.Felten (1992) Large-step Markov chains for the TSP incorporationg local search heuristics, *Opers. Res. Lett.* 11, 219-224.
I.H. Osman (1993) Metastrategy simulated annealing and tabu search algorithms in combinatorial optimization. *Parallel Computing*, 7 65-85.
E.D. Taillard (1993) Parallel iterative search methods for vehicle routing problems. *Networks*, 23 661-673, 1993