UN ALGORITMO APROXIMADO PARA EL PROBLEMA DEL CARTERO RURAL MIXTO

Antonio Hervás, Francisco J. Sanchis, David Soler
Departamento de Matemática Aplicada.
Universidad Politécnica de Valencia.

Abstract

En este trabajo se presenta un algoritmo heurístico para resolver el problema del Cartero Rural Mixto, del que no conocemos ningún algoritmo exacto o heurístico.

1 Introducción

Muchos problemas reales en los que calles o carreteras deben ser recorridos por vehículos, se modelizan a través de la Teoría de Grafos. Así sucede, por ejemplo, en la recogida de residuos sólidos, limpieza de calles, reparto de correos, y en general, distribución de cualquier tipo de producto a sus puntos de venta. El problema real se transforma en un problema sobre un grafo, donde los arcos o aristas se corresponden con las calles o carreteras, y los costes asociados a aquellos, con los costes (generalmente distancias) de atravesar la calle o carretera que representan. Dicho problema sobre el grafo tendrá por objetivo minimizar el coste total recorrido por el vehículo o vehículos, sujetos en general a determinadas restricciones (capacidad, horario, etc.)

Dentro de este contexto se enmarcan los Problemas Rurales, llamados así por su similitud a un entorno rural, donde las calles de los pueblos son aristas o arcos requeridos para realizar el reparto, recogida, etc. y las carreteras que unen dichos pueblos son aristas o arcos no requeridos pero sí susceptibles de ser atravesados.

2 Definición y casos particulares

El caso más general dentro de los Problemas Rurales para un solo vehículo, es el conocido como Problema del Cartero Rural Mixto, en adelante PCRM, que se define como sigue:

Definición 1 *Sea $G = (V, A \cup E)$ un grafo mixto, donde A es el conjunto de arcos y E el conjunto de aristas y cuyos costes son no negativos. Sea $R \subseteq A \cup E$, $R \neq \emptyset$ el conjunto de arcos y aristas requeridas, encontrar un tour de coste mínimo que pase al menos una vez por cada uno de los elementos del subconjunto R.*

Los casos particulares del PCRM (en que $R = A \cup E$), se corresponden con el conocido Problema del Cartero Chino en sus tres versiones: no dirigido ($A = \emptyset$), dirigido ($E = \emptyset$) y mixto ($A \neq \emptyset \neq E$). Los dos primeros se resuelven en tiempo polinomial, mientras que el tercero no (es NP-duro), existiendo algoritmos exactos para su resolución y varios algoritmos aproximados de tiempo polinomial.

Los casos particulares en que $R \neq A \cup E$ pero $A = \emptyset$ o $E = \emptyset$, son así mismo NP-duros, habiéndose desarrollado también para ellos algoritmos exactos y aproximados. En [1] y [3]se presentan sendos exhaustivos y recientes resúmenes de lo publicado hasta la fecha sobre los cinco casos particulares antes mencionados.

Para el PCRM (en su caso más general) no conocemos en la literatura científica ni algoritmos exactos ni aproximados para su resolución.

Presentamos un algoritmo aproximado de tiempo polinomial para el PCRM, el cual hemos implementado en **Mathematica** y que revisa ideas desarrolladas por otros autores al estudiar casos particulares antes mencionados, destacando el algoritmo aproximado MIXED2 de Frederickson [4] para el Problema del Cartero Chino Mixto y el algoritmo aproximado de Christofides et al. [2] para el Problema del Cartero Rural Dirigido ($E = \emptyset$).

3 Estudio del Algoritmo

Supondremos que $R \neq A \cup E$ ya que en caso de igualdad, como ya hemos mencionado, se sabe resolver tanto de forma exacta como heurística.

En el caso en que $G_R = (V_R, R)$ es conexo. (Diremos que el grafo mixto es conexo si para cada par de vértices existe un camino formado por aristas y por arcos que los une). El problema se resuelve de forma similar al caso en que $R = A\ cupB$.

Aplicamos un algoritmo **MIXED** "ESPECIAL" donde para cada vértice de G_R miramos si es simétrico (el número de arcos de entrada es igual al número de arcos de salida) y si es par (inciden en él un número par de aristas). Si no cumple estas condiciones transformaremos el grafo añadiendo arcos y aristas resolviendo su asimetría y su imparidad. Para convertir los vértices en simétricos aplicamos el procedimiento INOUTDEGREE (definido por Frederickson) consistente en transformar G_R en un grafo dirigido $G_1 = (V_R, A_R \cup E_1 \cup E_2)$ donde A_R es el conjunto de arcos de **R** y que tienen costes originales y capacidad infinita, E_1 contiene dos arcos (de direcciones opuestas) por cada arista de R con costes iguales al coste de la arista y capacidad infinita y E_2 contiene dos arcos (de direcciones opuestas) por cada arista de R pero de coste 0 y de capacidad 1. Debemos resolver el problema de flujo de coste mínimo (f.c.m.) que satisfaga todos los requerimientos de las fuentes y los sumideros en G_1 con la variante de que calculamos el f.c.m. en el grafo original **G**. Así obtenemos un nuevo grafo G' como sigue: si en la solución óptima al problema del flujo anterior, una unidad del flujo atraviesa un arco de E_2, dirigir la correspondiente arista en la misma dirección. Después añadir al original tantas copias de A_R y de E_1 como unidades del flujo lo hayan atravesado. El resultado será un grafo simétrico $G' = (V_R, A' \cup E')$ donde A' es el conjunto de todos los arcos y donde E' representa el conjunto de aristas que todavía no se han asignado. Para convertir los vértices impares en pares aplicamos el procedimiento llamado **LARGECYCLES** donde primeramente identificamos vértices de grado impar en **G'** y debemos encontrar los caminos más cortos entre estos vértices usando únicamente elementos del conjunto E (sólo aristas). Insertamos las aristas del emparejamiento en nuestro grafo. Así conseguimos un grafo Euleriano donde calculamos el tour solución.

Si G_R no es conexo, que es el caso más general, nuestro algoritmo tiene dos etapas: una de transformación y otra de resolución.

ALGORITMO 1 *TRANSFORMACION DE UN GRAFO PCRM*

Paso 1. *Sea $G_R = (V_R, R)$, añadir un arco entre cada par de vértices de V_R con coste igual al camino mixto más corto entre ellos (calculándolo* *desde el grafo original* **G**). *Denotamos el nuevo grafo completo por* $G_c = (V_R, A')$ *donde* $A' = R \cup A_c$ *y donde* A_c *es el conjunto de arcos añadidos al grafo para que sea completo. Así nos habrá surgido al menos un arco en cada dirección por cada uno de* **R**. *(Cualquier circuito en* G_c *donde cada arco de* **R** *es atravesado al menos una vez, forma un circuito solución al* **PCRM** *en* **G**. *Una solución óptima en* G_c *será solución óptima en* **G** *con el mismo coste).*

Paso 2. *Simplificamos y obtenemos* $G_T = (V_R, A_T)$ *eliminando:*

> *(a)- Todos los arcos* $(i,j) \in A_c$ *tal que* $c_{ij} = c_{ik} + c_{kj}$ *para algún* $k \in V_R$.
>
> *(b)- Los arcos paralelos añadidos (y misma dirección) si tienen el mismo coste.*

Esta transformación es análoga a las realizadas en los estudios en que $E = \emptyset$ *o* $A = \emptyset$.

ALGORITMO 2 *RESOLUCION DEL PCRM EN EL GRAFO TRANSFORMADO*

Paso 1. *Construir un nuevo grafo completo y condensado* G_{CC} *a partir de* G_T *(grafo transformado), de tal forma que cada vértice de* G_{CC} *representa una componente conexa (conexa según caminos mixtos) inducida por* R *en* G_T*, y cada arco de* $G_{CC}(i,j)$ *tiene por coste* $c_{ij} = \min\{c.m.c.(a,b)\}$ *donde* $a \in C_i$, $b \in C_j$*, siendo* C_k *los vértices de la componente conexa* k *inducido por* R *en* G_T. *En* G_{CC} *se calcula una arborescencia generadora de mínimo peso* S *comenzando por un vértice raíz cualquiera y conectando todas las componentes conexas. Añadimos como requeridos a* G_R *(grafo requerido) aquellos arcos que representan los c.m.c. que definen los arcos expresados por la arborescencia. Obtenemos* $G_C = (V_R, R \cup S)$ *que es conexo.*

Paso 2. *Como ya es conexo aplicamos a* G_C *un algoritmo heurístico* **MIXED "ESPECIAL"** *definido anteriormente para cuando el grafo requerido era conexo. Según la aplicación desarrollada usamos el* **MIXED2 "ESPECIAL" (INOUTDEGREE Y LARGECYCLES)**... *obteniendo* G_F *un grafo par y simétrico.*

Paso 3. *Si quedan aristas aplicamos a G_F el procedimiento de Ford y Fulkerson (ver siguiente sección) para orientar un grafo simétrico, consistente en ir asignando direcciones a las aristas por ciclos. Aplicamos por último el algoritmo "ENDPARING" (ver siguiente sección) calculando el tour solución.*

4 Algoritmos complementarios

ALGORITMO 3 *ALGORITMO DE FORD Y FULKERSON PARA ORIENTAR COMPLETAMENTE UN GRAFO SIMETRICO*

Paso 1. *Si no hay ninguna arista en el grafo,* **FIN**

Paso 2. *Sea v un vértice que es incidente con al menos una arista del grafo y sea la arista (v,w). Tomamos v=v1 y w=v2.*

Paso 3. *Orientar la arista (v1,v2) desde v1 a v2. Si v2=v, ir a Paso 1.*

Paso 4. *Asignar a v1 el valor de v2 e identificar una nueva arista (v1,v2) incidente con v1. Ir a Paso 3.*

ALGORITMO 4 *ALGORITMO DE EDMONDS Y JOHNSON (END PARING)*

Paso 1. *Definir un ciclo cualquiera en el grafo. Si incluye todas las aristas,* **FIN**.

Paso 2. *Considerar un vértice v incidente con una arista que no es del ciclo. Formar un segundo ciclo desde v y que no se solape con ninguna arista del primer ciclo.*

Paso 3. *Sean e_1, e_2 dos aristas incidentes con v por el primer ciclo; y e_3, e_4 aristas incidentes con v por el segundo ciclo. Juntar los dos ciclos en uno. Empezar desde v por la arista e_3 recorriendo el segundo ciclo hasta e_4 y luego continuar por e_2 recorriendo el primero hasta e_1. Si el nuevo ciclo incluye todas las aristas del grafo* **FIN**. *En caso contrario volver a Paso 2.*

Referencias

[1] Assad, A.L.; Golden, B.L.: "Arc Routing Methods and Applications", Network Routing. North-Holland (1995).

[2] Christofides, N; Campos, V.; Corberán, A y Mota, E. "An Algorithm for the Rural Postman Problem on a Directed Graph", Mathematical Programming Study. 26, 155-166 (1986).

[3] Eiselt, H.A.; Gendreau, M. ; Laporte, G. "Arc routing: Problems, applications and algorithms". Pub. 872. Centre de recherche sur les transports. Université de Montreal. Canada (1992).

[4] Frederickson, G.N. "Approximation Algorithms for Some Postman Problems", Journal of the Association for Computing Machinery. 26, 538-554 (1979).