UN ALGORITMO PARA PLANIFICAR RUTAS MÁS RÁPIDAS CON ARCOS DEPENDIENTES DEL TIEMPO EN REDES URBANAS

Francisco A. Ortega
Departamento de Matemática Aplicada I, Universidad de Sevilla, España
Guido Marseglia
Departamento de Matemática Aplicada I, Universidad de Sevilla, España
Juan A. Mesa
Departamento de Matemática Aplicada II, Universidad de Sevilla, España
Ramón Piedra-de-la-Cuadra
Departamento de Matemática Aplicada I, Universidad de Sevilla, España

RESUMEN

Los sistemas de navegación implementados en los dispositivos móviles permiten a los usuarios buscar las rutas más cortas entre pares de puntos. Muchos de los productos comerciales existentes suponen de manera simplificada que el tiempo de viaje para atravesar cada arco de una red de carreteras es fijo, una vez establecida una hora de inicio. Sin embargo, el tiempo real de viaje a lo largo de un tramo de carretera dentro de las ciudades depende de muchos factores que están relacionados con la congestión del tráfico, las condiciones climáticas, posibles incidencias, etc. y, en consecuencia, depende del tiempo.

Como se puede mostrar fácilmente, la determinación de los itinerarios más cortos en un contexto dependiente del tiempo puede dar como resultado diferentes rutas óptimas desde el mismo origen según diferentes horarios de salida. Suponiendo la disponibilidad de los datos estimados del tiempo requerido para transitar a lo largo de cada tramo de la red de calles, una vez que se ha fijado previamente la hora de salida, proponemos en este trabajo un algoritmo eficiente de obtención de rutas más rápidas sobre arcos dependientes del tiempo, de tal modo que la suma de los tiempos de conducción se minimice, lo que en paralelo permite mejorar el consumo de combustible y reducir las emisiones contaminantes asociadas. Una evaluación experimental se lleva a cabo para mostrar la efectividad del algoritmo aportado.

1. INTRODUCCIÓN

La determinación de itinerarios óptimos en las redes de carreteras es un ingrediente básico en la planificación logística y la simulación del tráfico. En los problemas de planificación de rutas de acuerdo con un simple objetivo, se tiende a seleccionar un único mejor camino desde un origen a un destino.

Debido a su eficiencia y eficacia, el algoritmo de Dijkstra (Dijkstra, 1959) se suele utilizar preferentemente para obtener la solución al problema de la determinación del camino más corto entre dos nodos de un grafo conexo. Se puede afirmar por consiguiente que, desde un punto de vista teórico, el problema de encontrar un camino más corto desde un nodo a otro en un grafo con longitudes (o tiempos de viaje) fijas sobre sus arcos está satisfactoriamente resuelto. De hecho, la implementación de la estructura de datos conocida como montículo de Fibonacci en el algoritmo de Dijkstra requiere sólo un tiempo O (m + n log n), donde n es el número de nodos y m el número de arcos del grafo subyacente (Cormen et al., 1994).

Sin embargo, este algoritmo puede resultar impráctico en varios escenarios de aplicación real.

Uno de estas situaciones surge cuando las ponderaciones asociadas al tránsito entre dos nodos adyacentes son dependientes del tiempo. Esta circunstancia puede darse tanto en la determinación de rutas óptimas de transporte público (autobús, metro o redes densas de trenes de cercanía) como en las de transporte privado (vehículo motorizado propio, bicicleta o similar). Entre las variantes del problema de la determinación del camino más corto estudiadas en Dreyfus (1969) está aquella en la que los tiempos de viaje requeridos para las conexiones entre nodos son dependientes del tiempo de partida de los vehículos.

Esta específica perspectiva es la que se analiza en el presente trabajo, aunque no es la única que puede despertar interés en la literatura especializada en el transporte.

Por ejemplo, los usuarios del transporte público pueden mostrar diferentes preferencias desde una percepción personal (punto de vista del usuario), lo que implicaría una adaptación de los algoritmos iniciales para dar cabida a la existencia de otros tipos de preferencias. Entre estas adaptaciones está la consideración del uso de algoritmos que produzcan K-caminos más cortos (problema denominado KSP) para obtener un número razonable (K) de caminos factibles más cortos y jerarquizarlos a partir de la incorporación de las preferencias del usuario. Las preferencias de los usuarios del transporte público suelen ser varias. A modo de ejemplo:

1. Lograr un tiempo mínimo de recorrido, lo que significa llegar al destino en el menor tiempo posible a partir de la hora de salida establecida como inicio desde el punto origen.
2. Minimizar el número de transbordos: algunos viajeros prefieren viajar en un único vehículo (autobús o tren), antes que soportar los inconvenientes de los transbordos, a pesar de que la duración global del trayecto pudiera ser más larga.
3. Conseguir un recorrido de distancia mínima de desplazamiento: un viajero cargado con objetos pesados o incómodos podría preferir caminar hasta la parada de autobús más cercana, como primera etapa, en lugar de recorrer una distancia mayor hasta otra parada de autobús, aunque esta segunda opción le supusiera una ruta más rápida en tiempo.

La planificación de rutas para vehículos en general sobre mapas geográficos a escala ciudad, región o país, es un problema importante y bien conocido, debido a su amplia gama de aplicaciones para cualquier medio de transporte (Preuss y Syrbe, 1997). El cálculo rápido de las rutas más rápidas de punto a punto en redes de carreteras muy extensas y cuyos arcos dependan del tiempo debe llevarse a cabo mediante la participación de servicios centralizados de información basados en una web accesible, donde se gestionen tanto los patrones de congestión como datos actualizados de tráfico en tiempo real. Para un servidor central que tuviera que responder a un número potencialmente muy grande de peticiones on-line de clientes a través de una interfaz www, sería deseable que el tiempo máximo de respuesta del sistema, proporcionando una buena solución, estuviera limitado superiormente de una forma razonable (Delling y Wagner, 2009). Asimismo, debe tenerse en cuenta que las soluciones que quedarían registradas en estos servicios de información de viajes podrían proyectar su influencia hacia los patrones de congestión que fueran utilizados para asesorar futuras decisiones en la determinación de rutas óptimas sometidas a tráfico en tiempo real.

Se denomina enrutamiento al proceso de seleccionar las “mejores” rutas en un grafo $G = (V, A)$, donde V es un conjunto de nodos y A es un conjunto de arcos. La mayoría de los estudios sobre problemas de enrutamiento se han realizado bajo el supuesto de que toda la información necesaria para formular los problemas es invariante en el tiempo (Toth y Vigo, 2014). En muchas aplicaciones prácticas, esta suposición generalmente no se verifica dado que los tiempos de recorrido pueden variar de manera exógena debido a la congestión del tráfico, las condiciones climáticas, etc., o de manera endógena, en función las decisiones que libremente adopte en conductor modificando a su criterio la velocidad del vehículo (por ejemplo, para ajustar el consumo de combustible) o alterando el tiempo de viaje mediante la inclusión de periodos de descanso en la conducción. Una clasificación de los problemas de enrutamiento dependientes del tiempo con respecto a varios criterios puede verse en Pillac et al. (2013).

El cálculo de rutas que ofrece Google Maps, como navegador integrado en las prestaciones de un vehículo, permite que los usuarios puedan conocer el tiempo estimado de llegada al destino seleccionado. Dicho cálculo se supone que se realiza utilizando una serie de parámetros. Entre ellos, la velocidad máxima permitida y la velocidad recomendada en cada tramo. Además, estos tiempos se promedian teniendo en cuenta los datos históricos registrados de las velocidades medias a lo largo de esas carreteras y calles por las que pasan las rutas consideradas, así como los tiempos que han invertido otros usuarios en anteriores ocasiones (si tal información estuviera disponible). Otros datos que suponemos que deben intervenir en este cálculo son, por supuesto, los de tráfico, con la información en tiempo real que afecta especialmente al tiempo estimado. El éxito de las predicciones a futuro obtenidas sobre los tiempos estimados de llegada ofrecidas por Google Maps son muy dependientes de la posible concurrencia de tales incidencias.

En cualquier caso, el resultado de estas predicciones se basa en estimaciones calculadas mediante procedimientos sometidos a confidencialidad empresarial.

Actualmente, hay múltiples operadores que proporcionan información del estado del tráfico basándose en la velocidad observada de circulación de los vehículos a lo largo de los tramos. Sin embargo, esta información no es completa ya que no cubre la totalidad de vehículos circulantes, bien porque existe una porción de vehículos "no conectados" en el parque, o bien porque las fuentes de esta información no cubren la totalidad de navegadores, de los operadores de telefonía o de las flotas de vehículos. Por ejemplo, Google-Traffic e InfoTransit son dos servidores de información de tráfico en tiempo real que operan en España y que se basan en datos de operadores de telefonía, flotas específicas de vehículos (aseguradoras, transportistas, clubes de automovilistas, etc.), sensores fijos de tráfico fijos y sistemas de navegación GPS. Eglese et al. (2006) examinan los problemas relacionados con la construcción de una base de datos de tiempos de recorrido dependientes del tiempo para una red de carreteras y evalúan los beneficios de los sistemas de planificación y enrutamiento de vehículos dependientes del tiempo para una aplicación desarrollada en Inglaterra.

Asimismo, los problemas de enrutamiento dependientes del tiempo se han tratado en el ámbito del denominado Vehicle Routing Problem (VRP) como una variante del mismo en múltiples contextos. Entre ellos (ver Gendreau et al., 2015):

- Planificación de rutas aeronaves, barcos o submarinos en el espacio bidimensional o tridimensional, donde las decisiones incluyen no sólo la determinación del itinerario, sino también el ajuste de potencia. El factor de dependencia del tiempo puede ser causado por las corrientes de aire, oleaje en el océano o el flujo submarino. El objetivo a minimizar suele ser la duración del trayecto, el consumo de combustible (o, equivalentemente, las emisiones de CO2) o una combinación de ellos (Perakis y Papadakis, 1989; Norstad et al., 2011).
- Otras causas de dependencia del tiempo que pueden alterar la duración de recorrido en el VRP sería la necesidad de reabastecer unidades móviles (Helvig et al., 2003), o la necesaria interceptación de trayectorias de otros vehículos (Jiang et al., 2005).
- Finalmente, la presencia de obstáculos en movimiento podría ser también un factor determinante en la planificación del movimiento de robots que puede modificar el tiempo de recorrido de los tramos (Sutner y Maass, 1988; Latombe, 1990; Fujimura, 1995).

2. FORMALIZACIÓN DEL PROBLEMA

Sea el grafo $G = (V,A)$ donde V es un conjunto finito de n nodos o vértices y A es un conjunto finito de m aristas o arcos que conectan los nodos. Cada arco puede ser denotado como el par (i, j) cuando se corresponda con el par de vértices i y j.

En general, para este tipo de problema se considera que los pares (*i*, *j*) son ordenados. Sean *O* (vértice origen o inicial) y *D* (vértice destino o terminal) dos nodos dados de *G*; se define un camino *p* desde *O* hasta *D* en *G* como la secuencia alternada de vértices y arcos:

$$p = \{O = v_0, a_1, v_1, a_2, v_2, \ldots, a_k, v_k = D\} \tag{1}$$

tal que se cumple:

- $a_i \in A, \forall i = 1, \ldots k;\ v_i \in V, \forall i = 1, \ldots k-1.$
- $a_i = (v_{i-1}, v_i) \in A, \forall i = 1, \ldots k.$
- $O, D \notin \{v_1, v_2, \ldots, v_{k-1}\}.$

Se define el costo asociado al arco $a = (i, j)$ como el número real positivo $c(a) = c_{ij} \geq 0$. De esta forma, el costo de un camino *p* equivaldrá a la suma acumulada de los costos de los arcos que lo componen: $c(p) = \sum_{(i,j) \in p} c_{ij}$.

Sea P_{ij} el conjunto de todos los caminos de *i* a *j* en el grafo *G*. Siguiendo la notación de Dreyfus (1969), el problema de encontrar el camino más rápido entre los puntos *O* y *D* donde el tiempo de viaje entre el punto *i* y el punto *j* dependa del tiempo de partida del punto j se puede formular como sigue. Denotemos mediante los valores positivos $d_{ij}(t)$ dicho tiempo de viaje invertido y mediante $f_i(t)$ al mínimo tiempo de viaje invertido hasta alcanzar el destino *D* partiendo desde el punto *i*.

El esquema recurrente

$$\begin{cases} f_i(t) = \min_{j \neq i} \left[d_{ij}(t) + f_j(t + d_{ij}(t)) \right] \\ f_O(t) = 0 \end{cases} \tag{2}$$

nos permite diseñar un algoritmo iterativo que, a partir del procedimiento ideado por Dijkstra (1959), proporcione la ruta más rápida en este contexto de tiempos de recorrido en los arcos dependientes del tiempo. Dicho algoritmo se describe más abajo.

2.1 Algoritmo de Dijkstra (entorno estático)

1. **Read** grafo **G=(V,A),** matriz de adyacencia **Ady** y matriz de costes **D** de desplazamiento entre pares de puntos adyacentes.
2. **Construir** vector **f** que almacena los mínimos costes de desplazamiento al origen desde cada nodo de G (inicialmente, sólo los nodos adyacentes con el origen no tendrán valor infinito).

3. **Construir** vector **p** que almacena los nodos sucesores a cada nodo de G en el camino óptimo (inicialmente, sólo los nodos adyacentes con el origen podrán estar determinados).
4. **Inicializar** conjunto **S** de nodos explorados con el origen sólo. **Inicializar** conjunto **LIST** de nodos por explorar dándole el valor complementario LIST=G\S.
5. **While** LIST no sea el vacío:
   5.1 Identificar el índice **j***
   $$j^* = Arg\left[\min\left\{f(j) : j \in LIST\right\}\right]$$
   5.2 Remove índice j* de LIST
   5.3 For each sucesor **k** de j* incluido en LIST:
   If *f(k) > f(j*) + C(j*, k)* **then**
   a. *f(k):= f(j*) + C(j*, k)*
   b. *p(k):= j**
6. **End**.

A continuación, se expone el mismo ejemplo que se usa en Wen et al. (2014) para ilustrar el funcionamiento del algoritmo de Dijkstra y sus limitaciones cuando la red es dependiente del tiempo.

En la Figura 1 se considera una situación estática, donde los pesos de los arcos no varían con el tiempo.

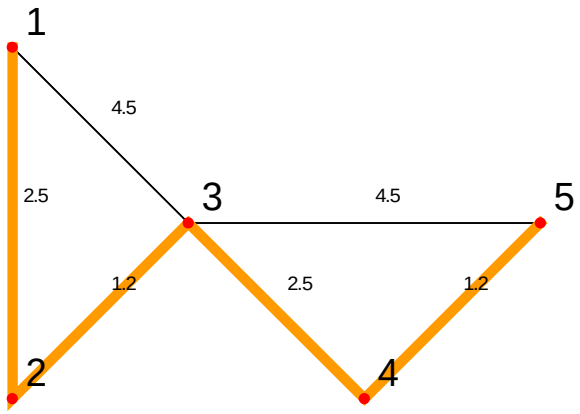

Fig. 1 –Camino más rápido entre nodos 1 y 5 usando el algoritmo de Dijkstra

Como se ilustra en la Figura 1, la solución al problema es el camino 1-2-3-4-5, y el tiempo invertido es 7.4 (2.5+1.2+2.5+1.2) minutos, como puede comprobarse aplicando el algoritmo paso a paso.

Inicio:

Vector f= (0, 2.5, 4.5, infinito, infinito)

Vector p= (- , 1, 1, - , -)

Conjunto S={1}. Conjunto LIST={2,3,4,5}

Paso 1:

j*= 2

Conjunto LIST={3,4,5}. Sucesores de j*= 2 en LIST= {3}

k=3. ¿Es f(3)=4.5 Mayor que la suma de f(2)=2.5 y C(2,3)=1.2? → Sí.

Entonces: f= (0, 2.5, 3.7, infinito, infinito); p= (- , 1, 2, - , -)

Paso 2:

j*= 3

Conjunto LIST={4,5}. Sucesores de j*= 3 en LIST= {4,5}

k=4. ¿Es f(4)=infinito Mayor que la suma de f(3)=3.7 y C(3,4)=2.5? → Sí.

Entonces: f= (0, 2.5, 3.7, 6.2, infinito); p= (- , 1, 2, 3 , -)

k=5. ¿Es f(5)=infinito Mayor que la suma de f(3)=3.7 y C(3,5)=4.5? → Sí.

Entonces: f= (0, 2.5, 3.7, 6.2, 8.2); p= (- , 1, 2, 3, 3)

Paso 3:

j*= 4

Conjunto LIST={5}. Sucesores de j*= 4 en LIST= {5}

k=5. ¿Es f(5)=8.2 Mayor que la suma de f(4)=6.2 y C(4,5)=1.2? → Sí.

Entonces: f= (0, 2.5, 3.7, 6.2, **7.4**); p= (- , 1, 2, 3 , 4)

Paso 3:

j*= 5

Conjunto LIST= Vacío. **END**

A partir del vector de nodos precedentes p= (- , 1, 2, 3 , 4) se puede reconstruir fácilmente el camino óptimo entre los nodos 1 y 5.

Supongamos ahora que el arco 3-5 dependa del tiempo de la forma siguiente:

- Para valores del tiempo entre 0 y 4, el tiempo necesario para recorrer el arco 3-5 es como en el caso anterior (estático) de 4.5 minutos.
- A partir de t=4, como novedad, el tiempo de recorrido del arco decae hasta valer 1,3 minutos.

En ese caso, como se expone en la Figura 2, el camino más rápido entre los nodos 1 y 5 sería la secuencia 1-3-5, siendo el tiempo invertido de 5.8 (4.5+1.3) minutos.

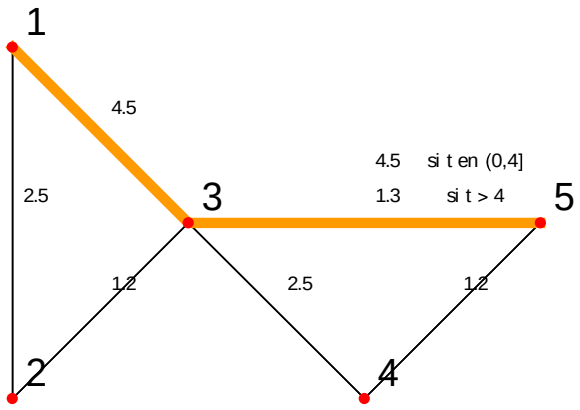

Fig. 2 –Camino más rápido entre nodos 1 y 5 en arcos dependientes del tiempo con prohibición de esperas en los nodos.

Para tratar de adaptar el algoritmo de Dijkstra a este nuevo contexto se debería incluir al menos una referencia temporal que recogiera la hora de inicio del recorrido de cada arco cuando se determina la ruta más rápida. Estas referencias temporales para cada uno de los nodos se deberían almacenar temporalmente como componentes de un vector. El nuevo punto 3 del algoritmo adaptado sería:

> **3'. Construir** vector **p** que almacena los nodos sucesores a cada nodo de G en el camino óptimo y, además, el vector **t** que almacena las horas de comienzo para la validez de esa sucesión.

Habría, asimismo, que modificar la instrucción **5.3** para incluir una actualización del vector t cada vez que se cumpliera la condición $f(k) > f(j^*) + C(j^*, k)$.

Sin embargo, se puede comprobar que la simple introducción de estas dos variantes no serviría para determinar la solución óptima. Aplicando el algoritmo modificado al ejemplo anterior, se puede apreciar que la dependencia en el tiempo del coste asociado al arco 3-5 no tiene efecto en el resultado final del algoritmo porque pasaría inadvertido.

Inicio:

Vector f= (0, 2.5, 4.5, infinito, infinito). Vector t= (- , 0, 0, - , -).

Vector p= (- , 1, 1, - , -).

Conjunto S={1}. Conjunto LIST={2,3,4,5}

Paso 1:

j*= 2

Conjunto LIST={3,4,5}. Sucesores de j*= 2 en LIST= {3}

k=3. Para t=0, ¿es f(3)=4.5 Mayor que la suma de f(2)=2.5 y C(2,3)=1.2? → Sí.

f= (0, 2.5, 3.7, infinito, infinito); t= (- , 0, 2.5, - , -); p= (- , 1, 2, - , -)

Paso 2:

j*= 3

Conjunto LIST={4,5}. Sucesores de j*= 3 en LIST= {4,5}

k=4. Para t=2.5, ¿es f(4)=infinito Mayor que la suma de f(3)=3.7 y C(3,4)=2.5? → Sí.
Entonces: f= (0, 2.5, 3.7, 6.2, infinito); t= (- , 0, 2.5, 3.7 , -); p= (- , 1, 2, 3 , -)

k=5. Para t=2.5, ¿es f(5)=infinito Mayor que la suma de f(3)=3.7 y C(3,5)=4.5? → Sí.
Entonces: f= (0, 2.5, 3.7, 6.2, 8.2); t= (- , 0, 2.5, 3.7 , -); p= (- , 1, 2, 3, 3)

Paso 3:

j*= 4

Conjunto LIST={5}. Sucesores de j*= 4 en LIST= {5}

k=5. Para t=3.7, ¿es f(5)=8.2 Mayor que la suma de f(4)=6.2 y C(4,5)=1.2? → Sí.
Entonces: f= (0, 2.5, 3.7, 6.2, **7.4**); t= (- , 0, 2.5, 3.7 , 6.5); p= (- , 1, 2, 3 , 4)

Paso 4:

j*= 5

Conjunto LIST= Vacío. END

Nótese que efectivamente el vector final p no ha variado (sigue valiendo (- , 1, 2, 3 , 4)), por lo que sería necesario introducir una nueva estrategia para el algoritmo. Proponemos la incorporación de una instrucción en la que se ampliara el grafo de conexiones, replicando cada nodo j* tantas veces como cambios a la baja se produzcan en los valores de los arcos salientes. En la Figura 2, el arco 3-5 toma dos valores, antes y después de t=4 minutos. El primer nodo de este arco que visita el algoritmo es el nodo 3 para t= 3.7 minutos. Este nodo se decide redefinirlo 3a y hereda las conexiones del nodo original (entre ellas, la conexión 3-5 valorada para t=3.7 minutos). Además, se replica un nuevo nodo 3b con las siguientes características:

- El nodo 3a se conecta de forma unidireccional al nuevo nodo 3b con el necesario tiempo de espera t=4-3.7=0.3 minutos.
- El nuevo nodo 3b replica todas las conexiones que tenía el nodo 3 en el grafo original salvo la conexión con el predecesor del nodo 3a en el camino que se está calculando. Los valores de estas conexiones se deberán aplicar en las respectivas ponderaciones a partir del tiempo referenciado de t=4 minutos.

En la Figura 3 se muestra el grafo ampliado según se ha descrito, así como el camino más rápido entre los nodos 1 y 5, que sería ahora la secuencia 1-2-3a (etiquetado como 3)-3b (etiquetado como 6)-5, lo cual implicaría una espera en el nodo 3 de 0.3 minutos, siendo el tiempo total invertido de 5.3 (4+1.3) minutos.

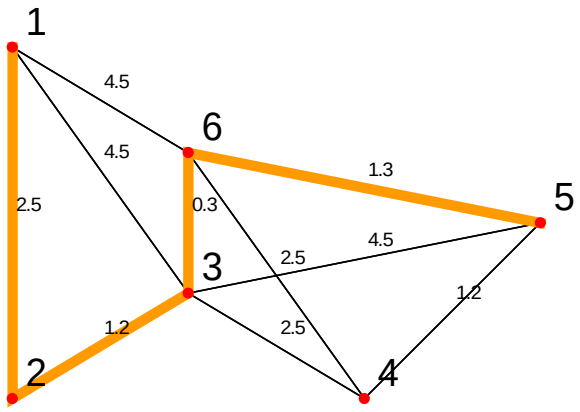

Fig. 3 –Camino más rápido entre nodos 1 y 5 en arcos dependientes del tiempo sin prohibición de esperas en los nodos.

3. UN ALGORITMO ADAPTADO

El algoritmo que se presenta a continuación incluye las modificaciones necesarias para poder determinar caminos más rápidos entre pares de nodos dentro de una red, estando las ponderaciones de sus arcos sometidos a variaciones según horario conocido o pronosticado con antelación.

3.1Un Algoritmo de Dijkstra adaptado a arcos dependientes del tiempo

1. **Read** grafo **G=(V,A),** matriz de adyacencia **Ady** y matriz de costes **D** de desplazamiento entre pares de puntos adyacentes.
2. **Construir** vector **f** que almacena los mínimos costes de desplazamiento al origen desde cada nodo de G (inicialmente, sólo los nodos adyacentes con el origen no tendrán valor infinito).
3. **Construir** vector **p** que almacena los nodos sucesores a cada nodo de G en el camino óptimo (inicialmente, sólo los nodos adyacentes con el origen podrán estar determinados).
4. **Inicializar** conjunto **S** de nodos explorados con el origen sólo. **Inicializar** conjunto **LIST** de nodos por explorar dándole el valor complementario LIST=G\S.
5. **While** LIST no sea el vacío:

   5.1 Identificar el índice **j***

   $$j^* = Arg\left[\min\left\{f(j) : j \in LIST\right\}\right]$$

   5.2 If los arcos de salida del nodo j* no cambian en el tiempo **then**

   5.2.1 **Remove** índice j* de LIST

   5.2.2 **For each** sucesor **k** de j* incluido en LIST:

   If *f(k) > f(j*) + C(j*, k)* **then**

   i. *f(k):= f(j*) + C(j*, k)*

   ii. *p(k):= j**

else

5.3 Por cada arco de salida nodo j* que cambie en el tiempo y por cada modificación de la ponderación del arco:

5.3.1 Replicar el nodo j* (sea j*'), actualizando el conjunto V.

5.3.2 Repetir para el nuevo nodo j*' las **conexiones** que tenía j* (con las mismas ponderaciones, si no fuera el arco de salida considerado), **excluyendo** la conexión de j* con su precedente en el camino.

5.3.3 Añadir un enlace dirigido de j* a j*', ponderado con el tiempo de espera requerido en el nodo j*.

5.3.4 Actualizar el conjunto A según 5.3.2 y 5.3.3.

5.3.5 Incluir en el conjunto LIST el nuevo nodo j*'.

6. End.

El bloque de programación que se añade está etiquetado como 5.3, y este se activa cuando se detecte la existencia de arcos de salida que dependan del tiempo. La complejidad del algoritmo depende precisamente del número de arcos cuyas ponderaciones están sometidas a cambios dependientes del tiempo, así como del número de cambios incorporados.

4. CONCLUSIONES

En este artículo se ha analizado el problema de la determinación de caminos más rápidos con arcos dependientes del tiempo en redes de transporte. Son numerosas las contribuciones bibliográficas existentes relativas a esta temática, debido principalmente a la relevancia que adquiere esta cuestión en la planificación logística y de viajes para todo tipo de usuarios.

Se ha estudiado específicamente la metodología presentada en Wen et al. (2014), donde se propusieron dos métodos heurísticos para resolver el problema de la ruta de costo mínimo entre un par de nodos con una red de carreteras que varía en el tiempo y existe además un cargo por congestión.

La herramienta desarrollada por estos autores se basaba en modificaciones del algoritmo de Dijkstra, donde se había establecido una prohibición de espera en los nodos.

La técnica de búsqueda algorítmica de caminos más rápidos con arcos dependientes del tiempo introducida en esta contribución sigue esa línea metodológica de adaptación del algoritmo de Dijkstra a este contexto, lo cual garantiza un alto nivel de eficiencia para el cálculo de soluciones.

En cambio, la contrapartida, como ha quedado constatado en la redacción del procedimiento, está en que el grafo original se deba ampliar convenientemente, tanto en número de nodos

como en nuevos arcos (algunos de ellos unidireccionales, que son los que indican el avance del tiempo). Sería posible, no obstante, limitar este crecimiento del grafo contenedor de soluciones, considerando sólo la incorporación de aquellos nodos y arcos que pudieran mejorar la solución actualmente calculada y obviando aquellas otras ampliaciones que claramente derivaran en empeoramientos.

AGRADECIMIENTOS

Este trabajo ha sido parcialmente subvencionado por el Ministerio de Economía y Competitividad/FEDER mediante el proyecto MTM2015-67706-P y por el Ministerio de Investigación (España) / FEDER con la subvención PID2019-106205GB-I00.

REFERENCIAS

CORMEN, T.H., LEISERSON, C.E., Y RIVEST, R.L. (1994). Introduction to Algorithms. MIT Press and McGraw-Hill.

DELLING, D. y WAGNER D. (2009). Time-dependent route planning. In: Robust and online large-scale optimization, pp.207–230. Berlin, Germany: Springer-Verlag.

DIJKSTRA, E. W. (1959). A note on two problems in connection with graphs. Numer. Math. 1, pp. 269-271.

DREYFUS, S.E. (1969). An Appraisal of Some Shortest-path Algorithms. Operations Research 17, pp. 395-412.

EGLESE, R., MADEN W. y SLATER, A. (2006). A road timetable TM to aid vehicle routing and scheduling. Comput. Oper. Res. 33, pp. 3508–3519.

FUJIMURA, K. (1995). Time-minimum routes in time-dependent networks. IEEE Trans Robot Autom 11, pp. 343–351.

GENDREAU, M., GHIANI, G. y GUERRIERO, E. (2015). Time-dependent routing problems: A review. Computers & Operations Research 64, pp. 189-197.

HELVIG, C., ROBINS, G. y ZELIKOVSKY. A. (2003). The moving-target traveling salesman problem. J Algorithms 49, pp.153–174.

JIANG, Q., SARKER, R., y ABBASS, H. (2005). Tracking moving targets and the non-stationary traveling salesman problem. Complex Int 11, pp. 171–179.

LATOMBE, J.-C. (1990). Robot motion planning. Berlin, Germany: Springer-Verlag.

NORSTAD, I., FAGERHOLT, K. y LAPORTE, G. (2011). Tramp ship routing and scheduling with speed optimization. Transp Res, Part C 19, pp. 853–865.

PERAKIS, A.N. y PAPADAKIS, N.A. (1989). Minimal time vessel routing in a time-dependent environment. Transp Sci 23: 266–276.

PILLAC, V., GENDREAU, M., GUÉRET, C. y MEDAGLIA, A.L. (2013). A review of dynamic vehicle routing problems. European Journal of Operational Research 225, pp.1–11.

PREUSS, T. y SYRBE, J.-H. (1997). An Integrated Traffic Information System. Proc. 6th Int. Conf. Appl. Computer Networking in Architecture, Construction, Design, Civil Eng., and Urban Planning (europIA '97).

SUTNER, K. y MAASS, W. (1988). Motion planning among time dependent obstacles. Acta Inf 26, pp. 93–122.

TOTH, P. y VIGO, D. (2014). Vehicle Routing: Problems, Methods, and Applications, Vol. 18, SIAM.

WEN, L., CATAY, B. y EGLESE, R. (2014). Finding a minimum cost path between a pair of nodes in a time-varying road network with a congestion charge. Eur J Oper Res 236, 915–923.