Informes de la Construcción
Vol. 71, 556, e320
octubre-diciembre 2019
ISSN-L: 0020-0883
https://doi.org/10.3989/ic.65988

Método de construcción de dígrafos a partir de redes viales reales en mapas digitales con aplicaciones en la búsqueda de rutas óptimas

Method to construct graphs from real street networks in digital maps with applications in the search of optimal routes

J. López Ortega (*), J. López-Sauceda (*, **), J.G. Carrillo (*, **), J. Sandoval (*)

RESUMEN

Actualmente las redes viales en zonas urbanas sufren obstrucciones, ya sea por manifestaciones, embotellamientos, u otro tipo de bloqueo, causando el cierre momentáneo o permanente de vías o zonas de tránsito que obligan a conductores a establecer nuevas rutas. Una solución es la creación de rutas alternativas en mapas digitales a partir de dígrafos relacionados con las características de una red vial real, y la aplicación de algoritmos de optimización de rutas. En este trabajo se propone un método para construir dígrafos con una aplicación en la API de *Google Maps* en la extracción visual de elementos como vértices (intersecciones), aristas (calles) y flechas de sentido (dirección vial), lo que permite la aplicación del algoritmo de Dijkstra en busca de rutas alternativas.

Palabras clave: Dígrafo, red vial, API *Google Maps*, algoritmo Dijkstra, ruta alternativa.

ABSTRACT

In urban areas, street networks are exposed to road obstructions since some traffic obstacles such as street protest, and traffic congestion are frequent. According to this, drivers are forced to choose alternative routes. Building alternative routes on digital maps represents a solution to these road problems. We determine alternative routes associating graphs to characteristics of real road networks using a API and incorporating optimization algorithms. In this work we propose a method to construct a graph from a zone with traffic blockages considering the network features. With an optimization algorithm (Dijkstra) the alternative routes are defined using digital maps. In addition, a computer program was developed (using Google Maps API) as a tool to create the graph, which is composed of: vertices (intersections) and directional edges (driving direction). This graph is used to establish alternative routes represented as polylines in digital maps.

Keywords: *Digraph, road network, Google Maps API, Dijkstra algorithm, alternative route.*

(*) Universidad Autónoma Metropolitana (México)
(**) Consejo Nacional de Ciencia y Tecnología CONACyT (México)
Persona de contacto/*Corresponding author*: jlo.lopez.ortega@gmail.com (J. López Ortega)
ORCID: https://orcid.org/0000-0002-2173-7146 (J. López Ortega); https://orcid.org/0000-0001-5174-6046 (J. López-Sauceda); https://orcid.org/0000-0002-4147-6764 (J.G. Carrillo); https://orcid.org/0000-0002-5578-4389 (J. Sandoval)

Cómo citar este artículo/*Citation*: López Ortega, J.; López-Sauceda, J.; Carrillo, J.G.; Sandoval, J. (2019). Método de construcción de dígrafos a partir de redes viales reales en mapas digitales con aplicaciones en la búsqueda de rutas óptimas. *Informes de la Construcción,* 71(556): e320. https://doi.org/10.3989/ic.65988

Recibido/*Received*: 08/06/2018
Aceptado/*Accepted*: 28/02/2019
Publicado on-line/*Published on-line*: 10/12/2019

1. INTRODUCCIÓN

Implementaciones como la ley de movilidad y vialidad en México (1) establecen permisos a la ciudadanía para realizar eventos y manifestaciones en vías públicas bajo la planeación de rutas específicas, reduciendo así la afectación vial. Esto permite a oficiales de tránsito, realizar una planificación adecuada de una o varias rutas alternativas que desvíen a conductores de zonas o rutas bloqueadas logrando un flujo continuo.

En México, el Instituto Nacional de Estadística y Geografía INEGI como unidad central de control del Sistema Nacional de Información Estadística y Geográfica SNIEG, norma la producción de objetos con datos geoespaciales requeridos por el Sistema de Información Geográfica SIG. Uno de estos objetos es la Red Nacional de caminos RNC, donde algunos de sus atributos son el ancho, velocidad máxima, nombre de las calles y número de carriles por calle, etc. (2) A pesar del valor de estos datos, no existen bases de datos en dicho instituto u otro encargado de la recopilación de información de redes viales en México que reúna las coordenadas geográficas de intersecciones o cruceros viales, y la relación que existe entre estas como son calles y vías de conexión junto a sus atributos de distancia y sentido. Esto es un inconveniente, ya que sin referencias geográficas de intersecciones como puntos de giro y conexión con sentido de dirección entre estas en atributos ya documentados, son inoperables en la construcción de dígrafos de redes viales y en la aplicación de algoritmos computacionales de optimización de rutas en redes.

El servicio de mapas viales web *Google Maps* permite a través de su extensa herramienta de desarrollo API, extraer, procesar y almacenar información vial actualizada y en tiempo real (3) como la vista aérea de una red vial, el tiempo de recorrido y distancia entre dos puntos y las coordenadas geográficas de intersecciones. No obstante, datos triviales requeridos en la construcción del dígrafo como la relación de las calles asociadas a dichas intersecciones y el tipo de sentido que le corresponde a cada calle, no existen como atributo en la API ya que no son de acceso libre o debe pagarse un costo excesivo por la base de datos. Por ello, una manera práctica de obtener esta información específica es a través del desarrollo de una aplicación que permita seleccionar visualmente los elementos característicos de una red y extraer los datos necesarios en la construcción de dígrafos conexos como una herramienta en la solución de problemas viales.

Actualmente la teoría de grafos se emplea en la construcción de redes como sistema de interconexión entre elementos de un conjunto que comparten una similitud, así el grafo es el medio por el cual se han aplicado numerosos métodos y técnicas en forma de algoritmos encauzados a resolver problemas de optimización. Los dígrafos son utilizados en la representación de redes viales como una de las aplicaciones más empleada (4) en áreas de estudio como ciudades inteligentes y sistemas SIG, resultando útiles en temas de logística y planeación en la búsqueda de rutas más cortas entre dos puntos, el recorrido más corto en puntos de parada, etc.

En gran parte de los trabajos sobre grafos en forma de redes viales encontrados en la literatura versa sobre el análisis de diversos algoritmos en la solución de rutas sobre grafos hipotéticos o creados de forma aleatoria para la construcción de redes viales (5), partiendo de expresiones como: *"Considere una red formada por un conjunto de N vértices conectados entre sí…"*, *"En esta idea hipotética, la red vial se ilustra como un grafo…"*, *"Por ejemplo, un mapa de ruta vial es modelado como grafo…"*, etc. Así mismo, se encuentran trabajos en donde la evaluación de estos algoritmos sobre grafos se aplica únicamente a datos viales reales extraídos de planos y dibujos (6) o de trabajo de campo (7), lo que limita su expansión a redes más grandes y complejas y la cantidad de datos a procesar.

En menor cantidad, encontramos trabajos dónde se aplican algoritmos en la búsqueda de rutas óptimas empleando base de datos limitadas de sistemas de información geográfica SIG de redes viales, por lo que los resultados se dedican únicamente a evaluar la eficiencia computacional y complejidad de los algoritmos a causa de la limitación o falta de propiedades en la red (8). En otros trabajos se utilizan las aplicaciones de servicio de mapas web para mostrar únicamente las rutas solución de una red de dígrafo con conexión de elementos, pero que no establece las indicaciones de las rutas a seguir como las intersecciones, calles y dirección de circulación, dejando la tarea de renderización a la aplicación para mostrar la conexión entre puntos sin considerar la ruta más corta (9).

El objetivo de este trabajo es desarrollar una aplicación con mapas que permita extraer las características geográficas de una red vial en la construcción ya sea de dígrafos o estructuras de datos equivalentes, permitiendo la aplicación de algoritmos de optimización como solución a problemas viales y de urbanización. En este sentido, la propuesta que aquí se expone trabaja con datos reales, actualizados y en tiempo real teniendo la capacidad de elegir cualquier zona vial de estudio, a diferencia de otros trabajos donde los mapas viales se cargan únicamente como imágenes de fondo (10).

2. EXTRACCIÓN DE CARACTERÍSTICAS VISUALES EN REDES VIALES REALES

Ya que un dígrafo $G = (V, E, \varphi)$ se representa como un conjunto de vértices conectados por aristas y un conjunto de aristas como pares ordenados con función de incidencia por $\varphi: V(2) \rightarrow E$, podemos comparar estos elementos del dígrafo con una red vial dónde cada vértice $x \in V$ corresponde a una intersección vial, y dos vértices adyacentes $[u, v] \in V$ como origen y destino forman una arista $e = (u, v)$ que representa la calle que une dichas intersecciones, ya sea con dirección en un solo sentido $(u, v) \neq (v, u)$ o en ambos sentidos $(u, v) = (v, u)$. La API de *Google Maps* permite cargar un mapa de una red vial donde se muestra el sentido de las calles de cualquier zona urbana del mundo, evitando así el trabajo de campo en la definición del sentido de las calles y el uso de planos y escalas. En la Figura 1 se muestra cómo obtener visualmente un dígrafo asociado a las características de una red vial en un mapa digital.

La representación de un dígrafo ponderado como red vial se realiza mediante una lista de incidencia l_i, esto se debe a su simplicidad en dígrafos dispersos y un mínimo uso de memoria requerida en comparación con la matriz de incidencia. Cada elemento a la entrada de la lista debe contener los siguientes apartados: un vértice origen u_i y un vértice destino v_i con coordenadas geográficas asociadas gu_i y gv_i respectivamente, el sentido de la arista formada por los elementos anteriores ya sea en un sentido con el alias o doble sentido con y dos pesos por cada arista con la distancia entre vértices $d(u_i,$

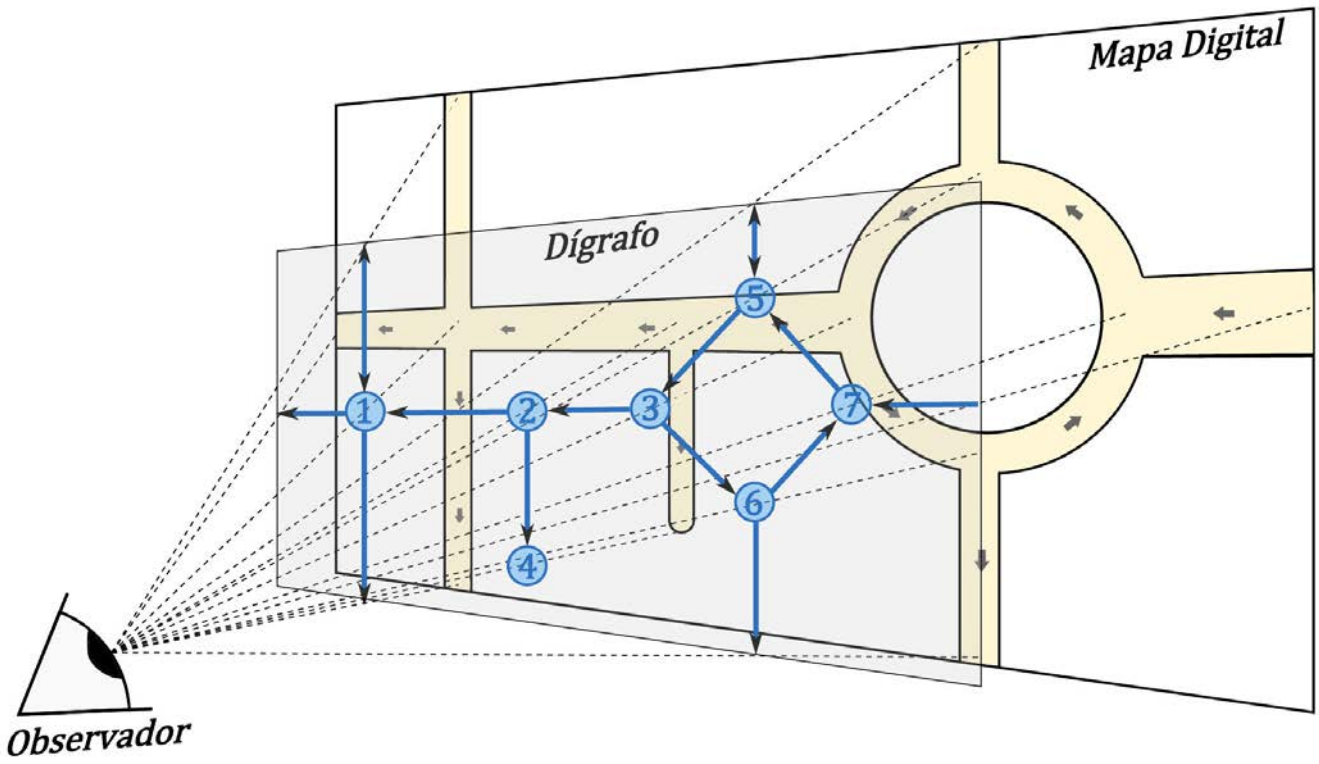


Figura 1. Relación de características entre red vial sobre un mapa y dígrafo ponderado detectados visualmente.

v_i) en metros y el tiempo de recorrido entre vértices $t(u_i, v_i)$ en segundos, ambos en el modo de manejo, quedando de la forma $l_i = [u_i, gu_i, v_i, gv_i, s_i, d(u_i, v_i), t(u_i, v_i)]$por cada uno de los elementos de la lista.

3. METODOLOGÍA

En la metodología de desarrollo, inicialmente se utiliza la API de *Google Maps* en lenguaje *JavaScript* para crear una aplicación que permita extraer de un mapa digital, información sobre las características de una red vial en forma de dígrafo en un archivo de texto *red.txt*.

En la Figura 2 se muestran en un mapa con red vial, dos elementos (intersecciones y flechas de sentido sobre calles) para su identificación y colocación manual de marcadores numerados consecutivamente; esto para poder relacionar intersecciones adyacentes dentro de una zona de estudio, donde el orden de colocación de los marcadores (origen y destino) señale el sentido de las calles mediante rectas que los conecten.

Con la información del dígrafo obtenida de una red vial y almacenada en el archivo *red.txt*, se lee dicho archivo con el software matemático *Matlab* para crear una lista de incidencia correspondiente, el cual contiene una serie de vectores con los vértices y aristas del dígrafo sin perder relación con las propiedades de la red; así mismo Matlab R2016b tiene un conjunto de funciones gráficas que permiten mostrar las listas o matrices de incidencia como dígrafos con etiquetas en aristas y vértices.

Con el dígrafo construido se aplica un algoritmo de optimización de rutas para validar los datos obtenidos por la aplicación, en este caso para obtener la ruta alternativa más corta entre otras rutas solución. Uno de estos algoritmos es el de Dijkstra, capaz de encontrar la ruta más corta desde un punto origen hasta un punto destino dentro de un dígrafo ponderado.

Finalmente, con las rutas obtenidas en forma de secuencia de vértices aplicando el algoritmo de Dijkstra programado en Matlab, se crea el archivo de salida *ruta.txt* el cual se lee nuevamente con la misma aplicación de *Google Maps* para mostrar la ruta alternativa solución sobre el mapa de red de caminos; esto con ayuda de la función de polilíneas e indicaciones textuales (dirección y calles) desde el punto origen al destino.

3.1. Construcción de una red como dígrafo sobre sobre un mapa digital con red vial

El desarrollo en la construcción del dígrafo como red vial se inicia con la creación de la red sobre el mapa digital de caminos. La aplicación desarrollada con la API de *Google Maps Java Script* contiene dos botones que precisan el tipo de sentido de las calles, ya sea en un solo sentido con la etiqueta "*one-way street*" en presencia de flechas sobre las calles o en doble sentido con la etiqueta "*two-way street*" en ausencia de flechas sobre las calles.

Con la selección previa del tipo de sentido por calle, la red de relación entre intersecciones viales se realiza colocando un primer marcador con el evento "*click*" sobre una intersección (vértice inicial), y posteriormente la colocación de un segundo marcador nuevamente con el evento "*click*" sobre una intersección adyacente (vértice final), generando así una línea de conexión entre vértices como tramo vial (arista) con dirección de inicio a fin. El color de las líneas de conexión indica el tipo de sentido, donde el color rojo representa un solo sentido vial y el color negro doble sentido.

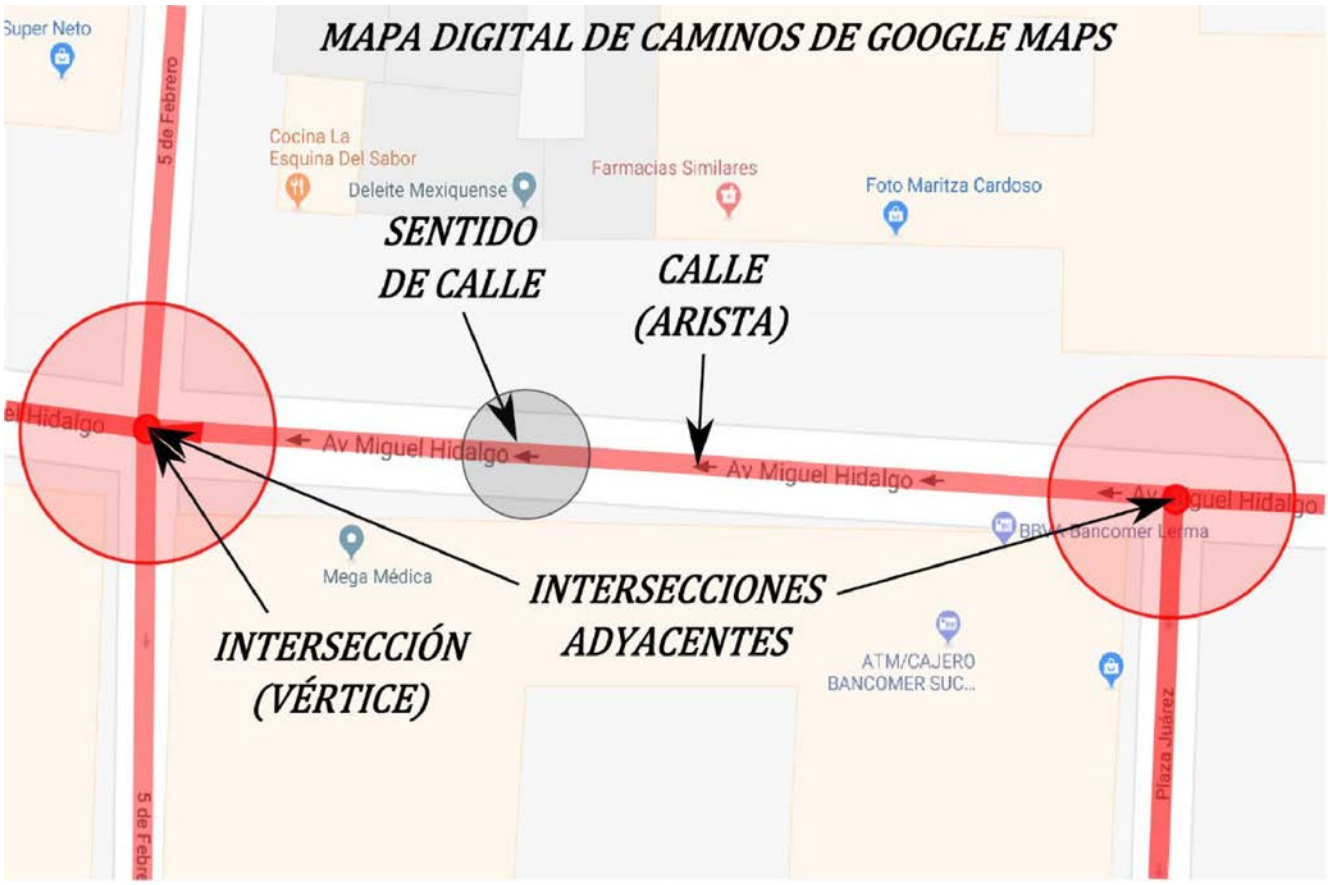


Figura 2. Visualización de elementos viales.

La selección de intersecciones ya identificadas con marcador se realiza con el evento "*right click*", y la colocación del segundo marcador en una nueva intersección adyacente con el evento "*click*", o de un marcador ya existente con el evento "*right click*", relacionando así cada una de las intersecciones a través de las vías de interconexión. Los datos de la red son guardados en un archivo de texto *red.txt* con la acción de un botón con etiqueta "*Save*". La Figura 3 muestra algunos elementos de la lista generados por la aplicación web con los datos extraídos separados por una coma (,).

La colocación de dos marcadores como origen y destino permiten realizar una llamada al servicio *Distance Matrix*, el cual calcula la distancia en metros y tiempo de recorrido en segundos en modo de navegación con vehículo "*Driving*". De esta forma, es posible aportar valores reales al peso de las aristas que ayuden a determinar la ruta más corta en función de la distancia o tiempo de recorrido total entre intersecciones adyacentes en una red vial, evitando el cálculo de distancias por coordenadas geográficas.

La Figura 4 muestra la aplicación desarrollada con la API de *Google Maps* en la construcción de una malla irregular sobre las características de la red vial de la ciudad de Lerma en Edo. Mex. México y una malla regular en la ciudad de Nueva York en N.Y. EUA como ejemplos de tipos de redes viales.

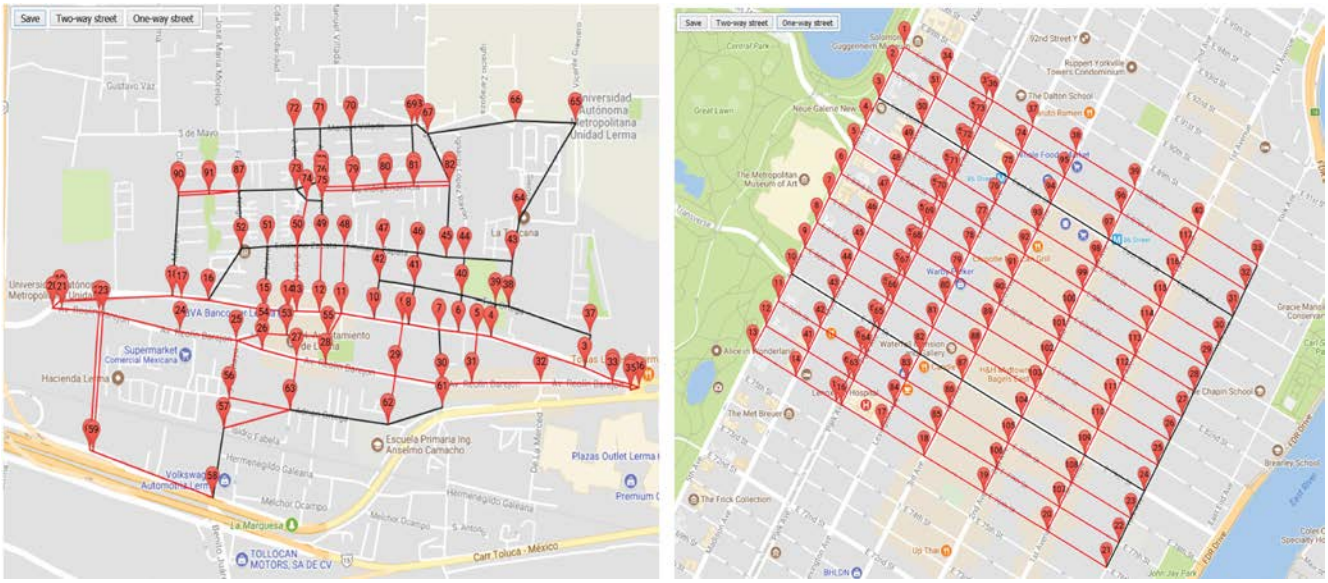
La aplicación inicia con la especificación de la zona geográfica donde se construirá la red del dígrafo, seguido del botón correspondiente al tipo de sentido de la calle y los eventos "*click*" o "*right click*" para colocar un nuevo marcador o tomar uno ya existente como vértices de origen y destino respectivamente. El Programa 1 carga la aplicación web con la API de *Google Maps*.

3.2. Creación de dígrafos con datos reales de Google Maps

El software matemático de desarrollo Matlab versión R2016b, tiene una extensa variedad de funciones de diseño y construc-

```
,1(19.284466351637175, -99.50093418359756),2(19.284587873752333, -99.50153768062592),One,159,38
,2(19.284587873752333, -99.50153768062592),3(19.284914463989953, -99.50243890285492),One,103,17
,3(19.284914463989953, -99.50243890285492),4(19.28572334165772, -99.50546577572823),One,329,48
,4(19.28572334165772, -99.50546577572823),5(19.285780304723158, -99.50591638684273),One,49,8
,5(19.285780304723158, -99.50591638684273),6(19.285856255446244, -99.50651988387108),One,64,14
,6(19.285856255446244, -99.50651988387108),7(19.285913218465453, -99.50713545084),One,65,12
,7(19.285913218465453, -99.50713545084),8(19.286046132099845, -99.50813323259354),One,105,19
,8(19.286046132099845, -99.50813323259354),9(19.28606891728348, -99.50832232832909),One,21,4
```

Figura 3. Elementos de la lista de incidencia extraídos de la red vial de un mapa web.



a) Red en mapa ciudad de Lerma b) Red en mapa ciudad de N.Y.

Figura 4. Redes como dígrafos en redes viales.

ción de dígrafos. La función $G = digraph\ (U, V, D)$ crea un grafo dirigido (dígrafo) con tres vectores U, V y D, como vértices de origen, destino y etiquetas de peso como distancia por cada una de las aristas formadas respectivamente.

Para crear los vectores U, V y D, necesarios en la construcción del dígrafo con información de la API de *Google Maps*, se realizó un algoritmo en Matlab de conversión de caracteres numéricos a números racionales, sin perder su posición por la fila y (,) de acuerdo con el archivo de la Figura 3. La Figura 5 muestra los dígrafos ponderados isomorfos de las redes construidas en la Figura 4, donde el ancho de las aristas es proporcional a su distancia en metros.

3.3. Validación de datos geográficos en redes viales como elementos de un grafo

Con la representación de los datos geográficos como dígrafo de una red vial real, se propone una solución al problema de bloqueos viales con la búsqueda de la ruta alternativa más corta (11), como una forma de validación de los datos geográficos obtenidos mediante la aplicación *Google Maps* correspondientes con los elementos de un dígrafo. Como primer paso se eligen los vértices que corresponden a las intersecciones que forman la ruta o zona a bloquear sobre de la red construida. La Figura 6 muestra el ejemplo de una ruta o zona bloqueada (color rojo) en la red vial del mapa de la Figura 4, así como en los vértices y aristas correspondientes con los dígrafos en la Figura 5. Adicionalmente, se incluye una ruta aleatoria (color verde) como recorrido de un vehículo especificando el vértice de origen (color naranja) y vértice destino (color magenta), que se interseca con la ruta o zona de bloqueo marcada para llegar de un punto origen a un punto destino . La función *highlight(h,path,'NodeColor','color','EdgeColor','color')* resalta los nodos y aristas especificados en el vector "*path*" dentro de un dígrafo con un color especifico .

Una solución que encuentra la ruta más corta S_p es la aplicación de algoritmos de optimización. El algoritmo de Dijkstra (12), (13), (14) es un algoritmo voraz que encuentra la ruta más corta desde un vértice origen $a \in V$, pasando por todos los vértices restantes $[x_2, x_3, \ldots, x_{n-1}] \in V$ hasta el vértice destino en el dígrafo , explorando todas las rutas posibles.

El algoritmo de búsqueda de la ruta alternativa más corta en dígrafos en Matlab se muestra en el Programa 2, iniciando con la eliminación del subconjunto de vértices $G' = (U', V', D')$ que forman la zona de bloqueo dentro de los vectores U, V, y D. Posteriormente si los vértices de origen y/o destino se encuentran dentro de dicha zona de bloqueo, se define uno o dos nuevos vértices respectivamente fuera de dicha zona.

Con el dígrafo sin los vértices de bloqueo, se aplica el algoritmo de Dijkstra desde el vértice origen $a = x_1$ hasta el vértice destino $b = x_n$, con aristas $e(x_i, x_{i+1})$ y pesos $p(x_i, x_{i+1})$ para $i = 1, 2, 3, \ldots n - 1$. La Tabla 1 lista la ruta alternativa más corta encontrada y tres soluciones más $R2$, $R3$ y $R4$, y en forma de lista de vértices secuenciales y distancia total recorrida en metros, la ruta se obtiene por la suma de pesos en aristas por la fórmula [1]. El Algoritmo de Dijkstra encontró 14,438 rutas para la red construida de la Ciudad de Lerma, mientras que para la red de N.Y. se encontraron 179,462 rutas, donde $R1$ es la ruta alternativa más corta de acuerdo con la Tabla 1.

$$S_p = min\left(\sum_{i=1}^{n-1} p(x_i, x_{i+1})\right) \quad [1]$$

En la Figura 7 se muestra la ruta alternativa más corta $R1$ y una ruta alternativa opcional $R2$ (color rojo) de la Tabla 1, así como los vértices y aristas de la zona de bloqueo (color gris) eliminados de la lista original en el dígrafo de la Figura 6 a).

En la Figura 8 se muestra la ruta alternativa más corta $R1$ y una ruta alternativa opcional $R2$ (color rojo) de la Tabla 1, así como los vértices y aristas de la zona de bloqueo (color gris) eliminados de la lista original en el dígrafo de la Figura 6 b).

En la Figura 9 se observa como el algoritmo de Dijkstra con el Programa 2 comienza a generar todas las rutas posibles encontradas dentro de los dígrafos construidos de la Figura 7 y 8, convirtiéndolo en un algoritmo de tipo voraz (*Gree-*

Programa 1. Programa de desarrollo de interfaz como herramienta de extracción de dígrafos con *API* de *Google Maps*.

Programa 1 Extracción visual de elementos viales

```
Head
    body height: 100%, floating-panel
end head
body
    bottom “Save” event click
    bottom “TwoWayStreet” event click
    bottom “OneWayStreet” event click
    i = 0; list = 1;
    node = 1;
    mark = 0;
    function initMap
        new mapa map { zoom: 17;
            center: {lat, lng}}
        event click on Map
            NewMark({lat, lng})
                if i >=2
                    list.push(No.Mark,position({lat, lng}));
                    list.push(way);
            service distanceMatrix{
                origins: line (0);
                destination: line (1);
                travelMode: ´driving´} response () {
                    list.push(distance);
                    list.push(duration)+’\n’}
                polyline(line) {
                    color: linecolor;}
                polyline.setMap(map);
            linea.length = 0; i = 0;
        else
            list.push(No.Mark,position ({lat, lng}));
        end if
    event right click i++;
        if i >=2
            line.push(marker.position); i++;
            Service distanceMatrix {
                origins: line (0);
                destination: line (1);
                travel Mode: ´driving´} response () {
                    list.push(distance);
                    list.push(duration)+’\n’}
                polyline(line) {
                    color: lineacolor;}
                polyline.setMap(map);
            linea.length = 0; i = 0;
        else
            lista.push(No.Mark,position ({lat, lng}));
        end right click
    end click
end function InitMap
function NewMark (position)
    marker: new marker;
    position: position;
    label: node++;
    map: map;
    line.push: position;
    mark++;
end function NewMark
function OneWayStreet () {
    way = “One”, color: red}
function TwoWayStreet () {
    way = “Two”, color: black}
function savetextAsfile () {
    filename: “myfile”,
    download&sabe: myfile}
end body
```

Programa 2. Algoritmo de Dijkstra en la búsqueda de la ruta alternativa más corta entre dos puntos.

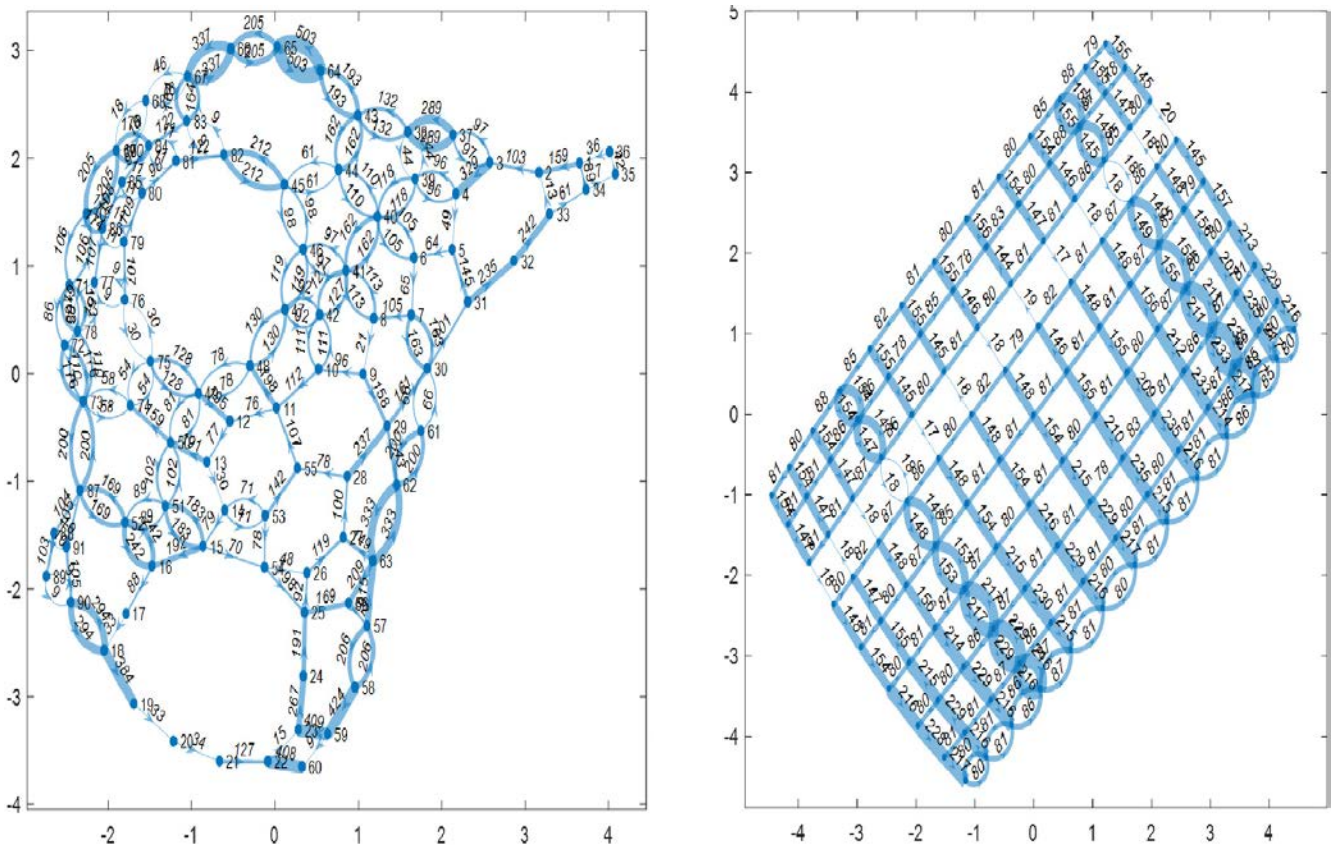
Programa 2 Búsqueda de rutas alternativas y ruta alternativa más corta con algoritmo de Dijkstra

```
Function Dijkstra () = (Vinicio, Vfin, distant, time)
    Norigen = vertice;
    Ndestino = vertice;
    path = [0, 0, Norigen, Norigen];
    rutaOptima = [0, 0, 0, 0];
    ColumnaD = 1;
    ColT = 1;
    filaR = 1;
    Recorrido (1) = 0;
    while ColumnaT >= 1
        noNodo = false;
        while noNodo ~= true
            nodos = [0, 0,0, 0], j = 0;
            For i = 1: NoAristas
                if Vinicio(i) == path (4)
                    NodoNo = false;
                    for j = 1: lgt (recorrido)
                        if Vfin(i) == recorrido(j)
                            NodoNo = true;
                    end, end
                        If NodoNo == false
j = j+1; nodos (j) = [distancia(i), Vinicial(i), Vfinal(i)]
                        end, end, end, end
                if nodos ~= 0
    path (filaR (1) +1, 1:4) = nodos (1, :);
    path(filaR (1) +1,2)=path(filaR(1),2)+nodos (1, 2);
    fila=fila+1;
    recorrido (fila (1),1) = nodos (1,3);
                for i = 2: j
ruta(1:fila(1)-1,(ColT+1)*4-3: (ColT+1)*4) =ruta(1:fila(1)-1,:);
recorrido (1: fila (1)-1, ColT+1) = recorrido (1: fila (1)-1,1);
ruta(fila(1), (ColT+1)*4-3:(ColT+1)*4)=nodos(i,:);
ruta(fila(1),(ColT+1)*4-2=nodos(i,2)+ruta(fila(1)-1,(ColT+1)*4-2
recorrido (fila (1), ColT+1) = nodos(i,3);
ColT = ColT+1;
fila(ColT) = fila (1);
    end, end, end
end function
```

dy) en busca de la ruta alternativa más corta. El algoritmo muestra cada una de las rutas posibles desde un punto origen y hasta un punto destino, pasando por todas las combinación posibles hasta encontrar la ruta de longitud mínima sin pasar por las zonas o rutas de bloqueo previamente establecidas.

4. ANÁLISIS DE RESULTADOS: REPRESENTACIÓN DE RUTAS SOLUCIÓN EN MAPAS DIGITALES

Con las rutas alternativas encontradas en forma de lista de vértices utilizando los dígrafos como representación de redes viales frente a problemas de obstrucción vial, es posible representar dichas soluciones en un mapa de caminos con la API de *Google Maps*. Con la aplicación desarrollada se incorpora un nuevo botón que lea un archivo de texto *ruta.txt* creado por Matlab con la lista de coordenadas geográficas relacionadas a los vértices de las rutas solución encontradas por el algoritmo de Dijkstra. Posteriormente con los puntos geográficos obtenidos, se ejecuta la herramienta de servicio de rutas por indicaciones con la lista de puntos (*waypoints)* sobre las intersecciones, dibujando una secuencia de polilí-



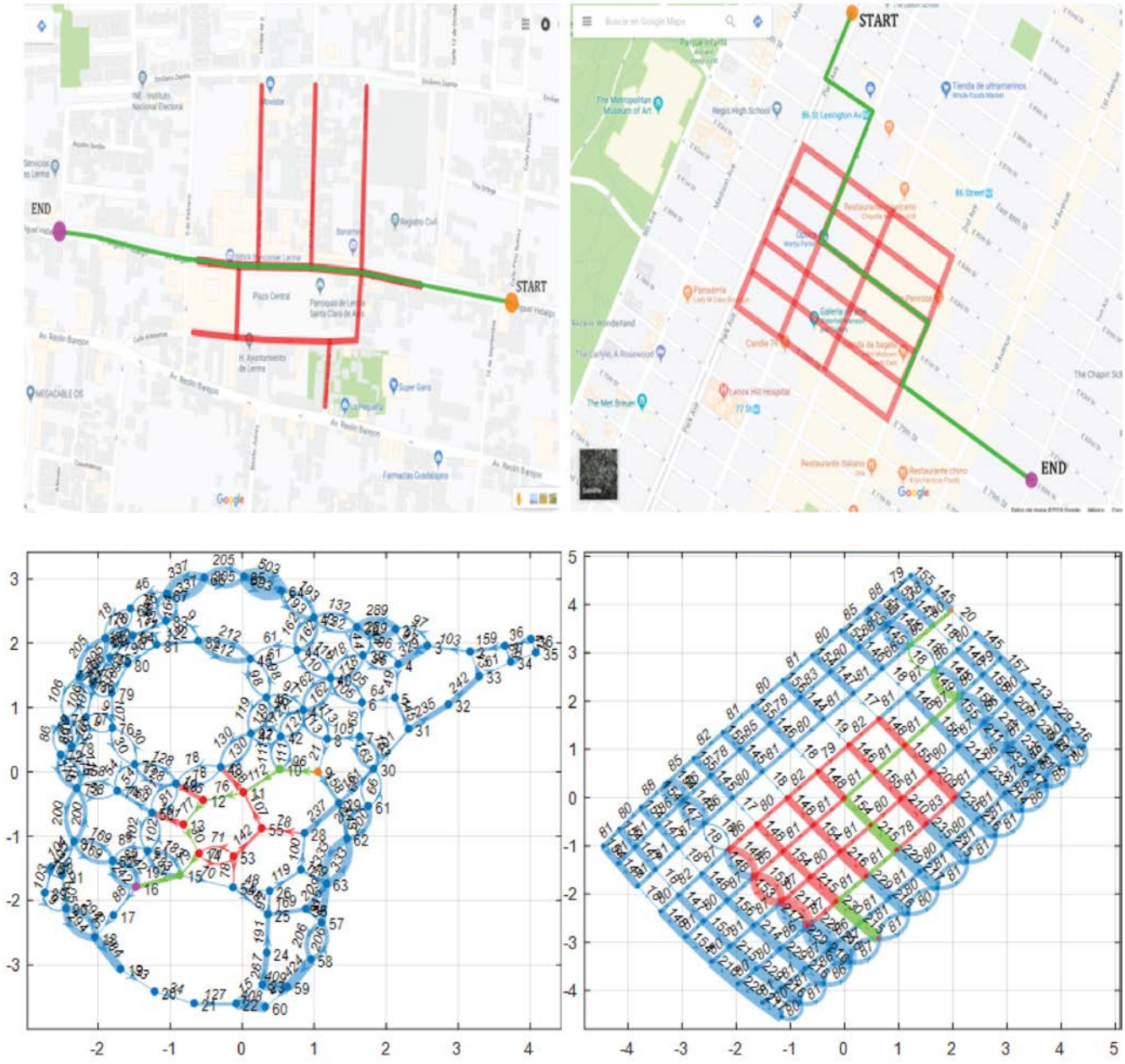
a) Dígrafo Ciudad de Lerma b) Dígrafo Ciudad de NY

Figura 5. Dígrafo a partir de una red vial.

Tabla 1. Rutas alternativas encontradas con algoritmo Dijkstra.

CIUDAD DE LERMA - MÉXICO NODO ORIGEN: 9 NODO DESTINO: 16				
x_i	R1	R2	R3	R4
1	9	9	9	9
2	10	10	10	10
3	42	42	42	42
4	47	47	41	47
5	48	46	40	46
6	49	45	39	45
7	50	82	38	44
8	51	83	43	43
9	52	84	44	64
10	16	85	45	65
11		86	46	66
12		77	47	67
13		76	48	68
14		75	49	69
15		49	50	70
16		50	51	71
17		51	15	72
18		52	16	73
19		16		74
20				75
21				49
22				50
23				51
24				52
25				16
26				
27				
DT	1021	1854	1996	3365

CIUDAD DE N.Y. - E.U.A. NODO ORIGEN: 35 NODO DESTINO: 25			
R1	R2	R3	R4
35	35	35	35
36	52	52	52
37	53	53	51
38	54	54	2
39	55	55	3
40	56	56	4
33	57	57	5
32	58	58	6
31	59	59	7
30	60	60	8
29	61	61	9
28	64	62	10
27	83	15	11
26	84	16	12
25	17	17	13
	18	18	14
	19	19	15
	20	20	16
	21	21	17
	22	22	18
	23	23	19
	24	24	20
	25	25	21
			22
			23
			24
			25
1635	2303	2304	2909



a) Dígrafo - red vial ciudad Lerma b) Dígrafo - red vial ciudad N.Y.

Figura 6. Ruta o Zona de bloque sobre la ruta de un vehiculo en una red vial y dígrafo.

neas en el mapa como secciones de líneas rectas interconectadas desde el punto origen pasando por todas las intersecciones restantes en la lista, hasta el punto de destino *B*. En la Figura 10 se muestra las rutas alternativas *R*1 y *R*2 impresas sobre redes viales formadas junto a los nodos de cada una de las intersecciones donde se realiza un movimiento de giro de un vehículo dentro de las rutas determinadas en la Tabla 1. Estas rutas son impresas en los mapas correspondientes a las zonas viales de la ciudad de Lerma en México y la ciudad de Nueva York en E.U.A.

Adicionalmente, la API de *Google Maps* cuenta con la herramienta de servicio de rutas por indicaciones, la cual permite que las rutas establecidas se indiquen a manera de instrucciones viales mediante una serie de descripciones textuales como: la distancia y tiempo de recorrido, indicaciones de sentido de giro sobre calles con nombre, y el número y calle de la posición de origen y destino de la ruta. La Figura 11 muestra las indicaciones textuales en forma de instrucciones con relación a las rutas *R*1 y *R*2 por polilíneas de los mapas en las zonas indicadas de Lerma y Nueva York respectivamente en la Figura 10.

El programa 3 obtiene la lista de indicaciones viales de manera textual y por el sentido de flechas en las rutas aplicadas como la ruta alternativa más corta encontrada por el algoritmo de Dijkstra, pasando por cada una de las intersecciones que forman la lista de vértices. Esta lista de instrucciones viales sigue las mismas rutas alternativas en forma de polilíneas en los mapas digitales con el servicio de direcciones viales renderizadas por la API de *Google Maps*.

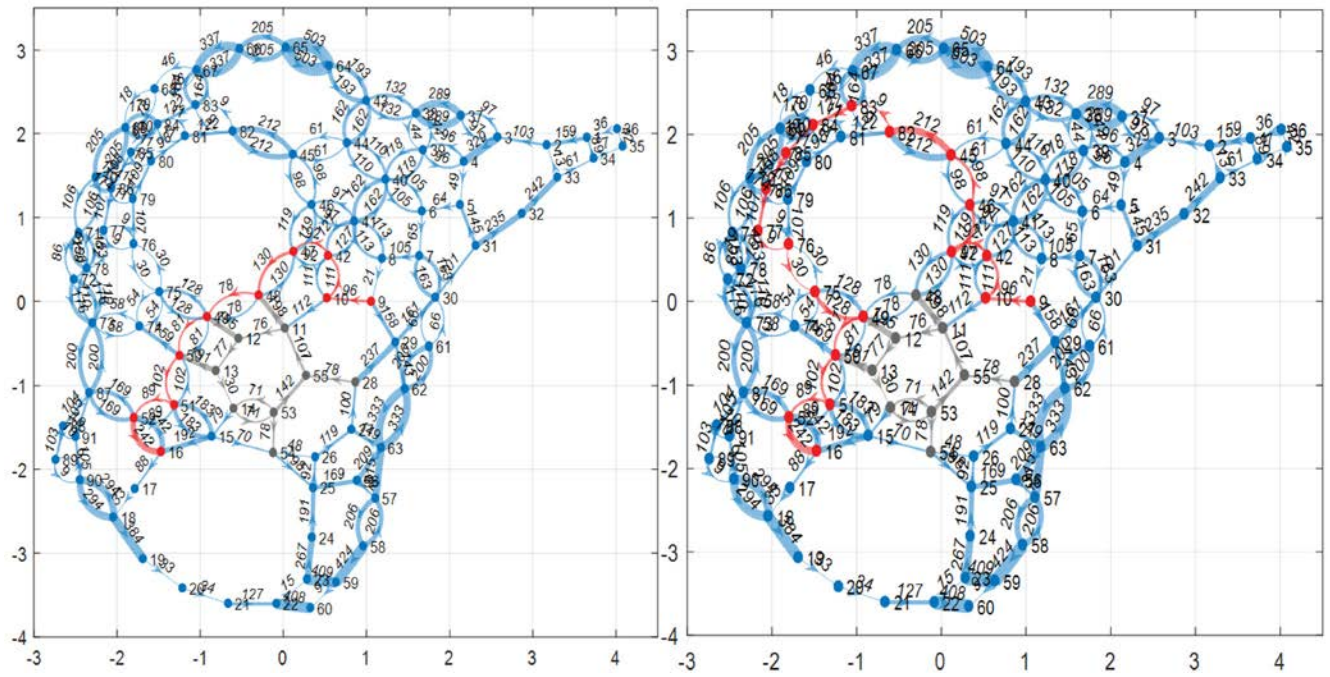


Figura 7. Ruta alternativa más corto y opcional a zona de bloqueo encontrado en dígrafo de red vial de Lerma – México.

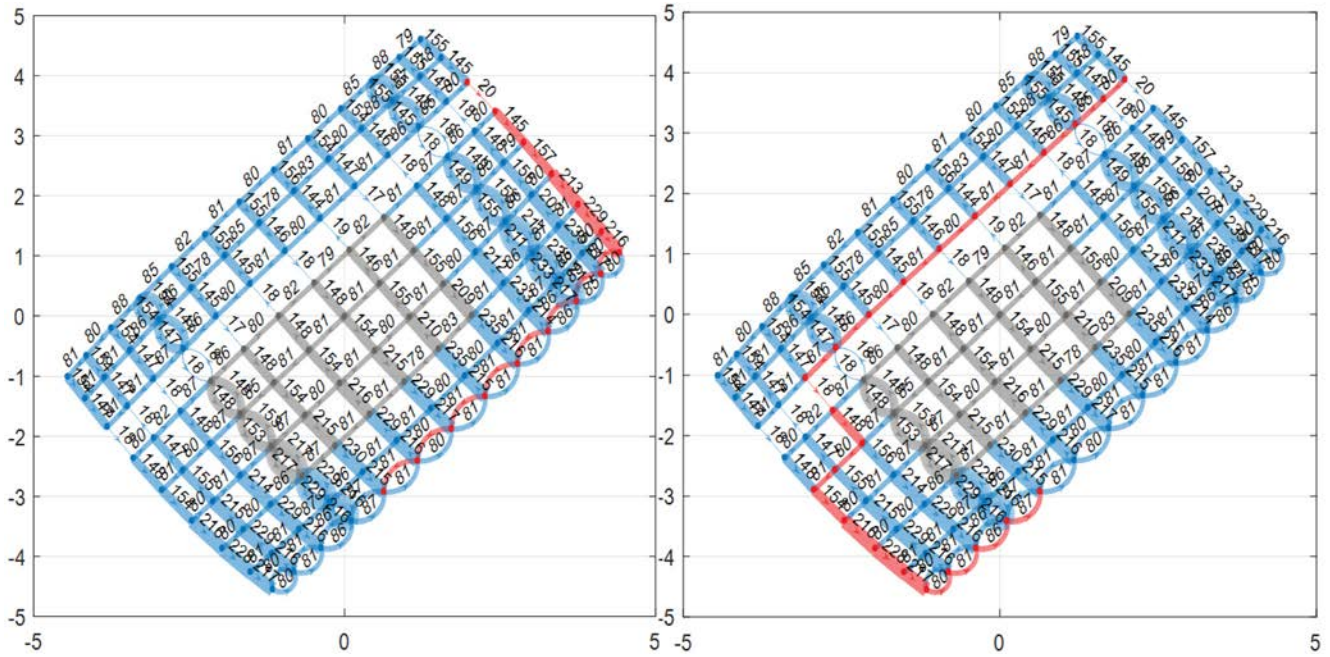


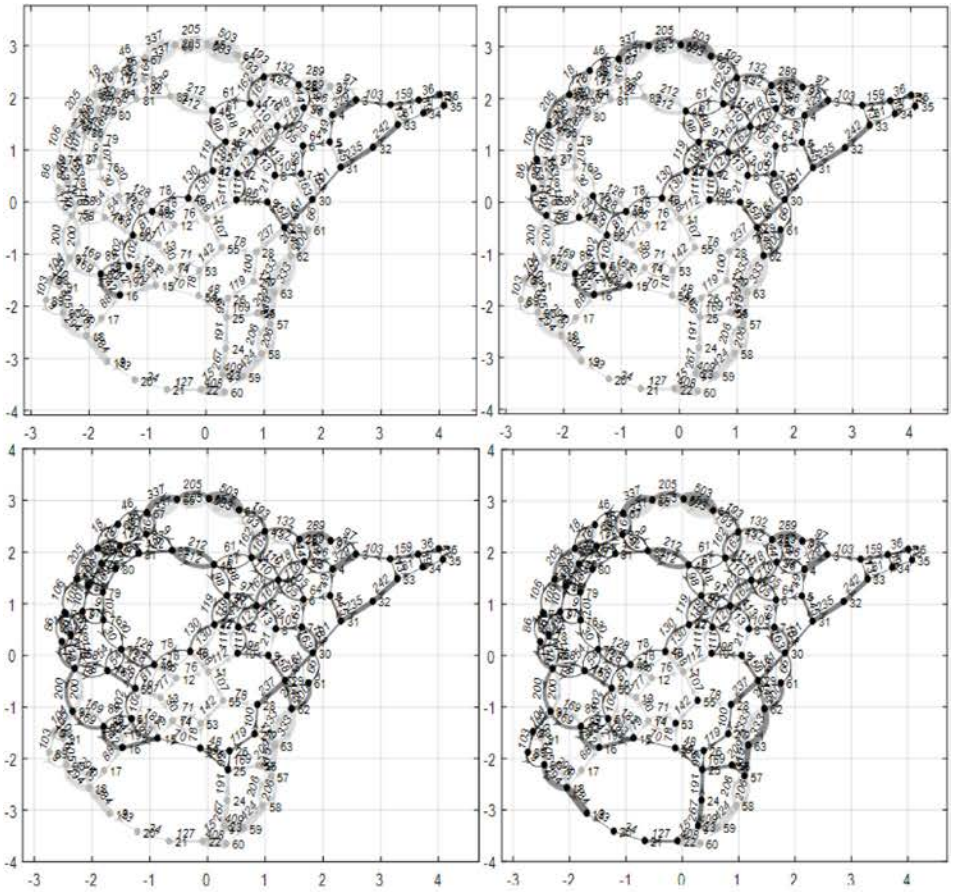
Figura 8. Camino alterno más corto y opcional a zona de bloqueo encontrado en dígrafo de red vial de N.Y. – E.U.A.

5. CONCLUSIONES

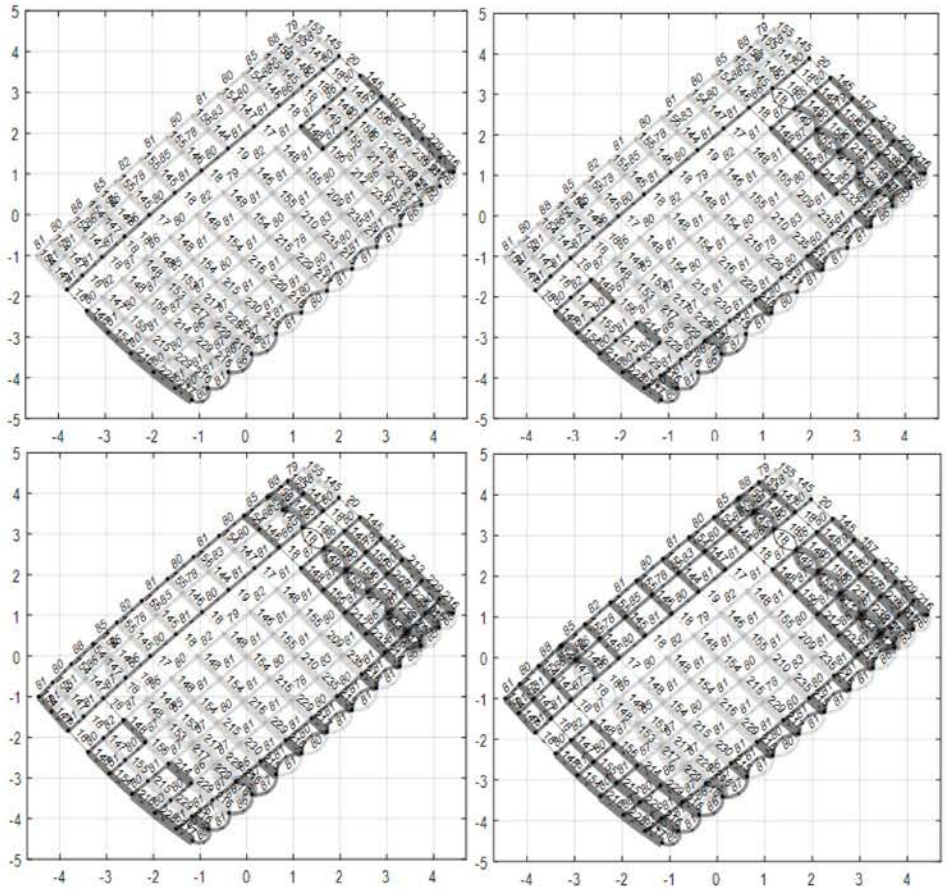
Es posible extraer y construir listas o matrices de incidencia o adyacencia en la aplicación de dígrafos relacionando sus elementos (vértices y aristas con dirección y pesos) con las características de una red vial. Esto se logra mediante la identificación visual de sus atributos (calles, flechas de dirección, intersecciones) en un mapa digital real con apoyo de aplicaciones de desarrollo en software en sistemas de información geográfica como: *Google Maps, Open Street Maps, Waze*, etc.

La selección de los atributos de una red vial en mapas web con información real y actualizada en la construcción de dígrafos evita el estudio y trabajo de campo en la recopilación de datos reales como: coordenadas de intersecciones, distancia entre intersecciones, sentido de las calles, cálculo de escalas de relación, etc. Así mismo permite definir nuevas rutas y/o puntos de interés sobre la red original como sub-dígrafos en la aplicación de algoritmos de optimización de rutas en redes viales.

La aplicación del algoritmo de Dijkstra en dígrafos construidos con datos extraídos de la aplicación desarrollada, permitió en-

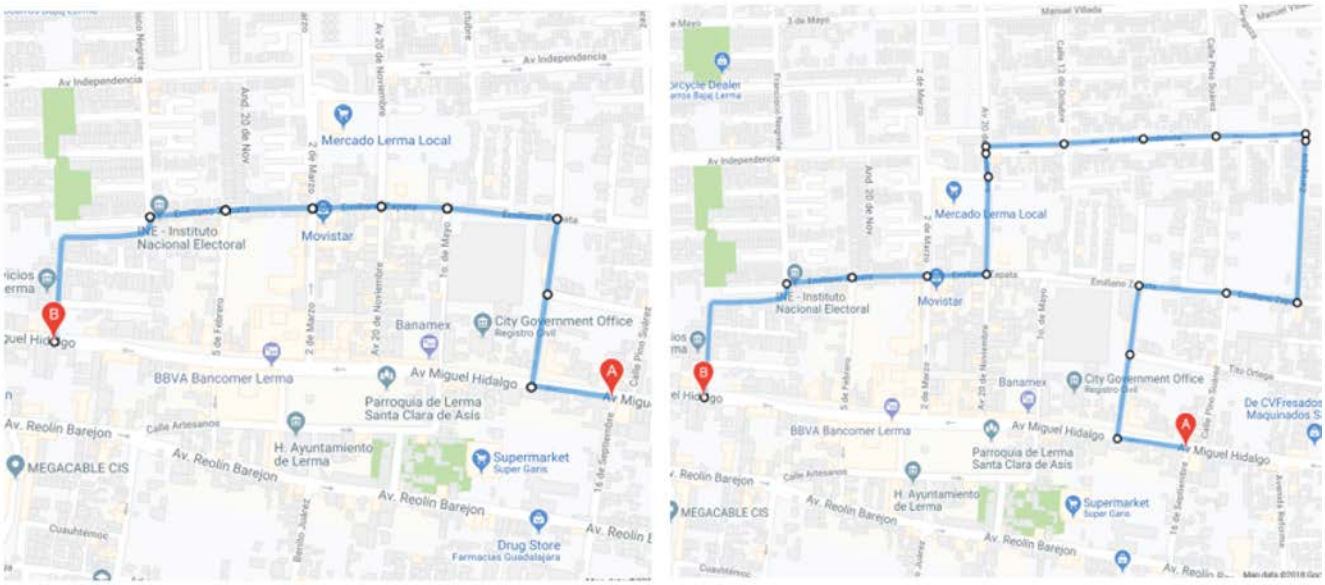


Exploración de rutas alternativas en red de Lerma.

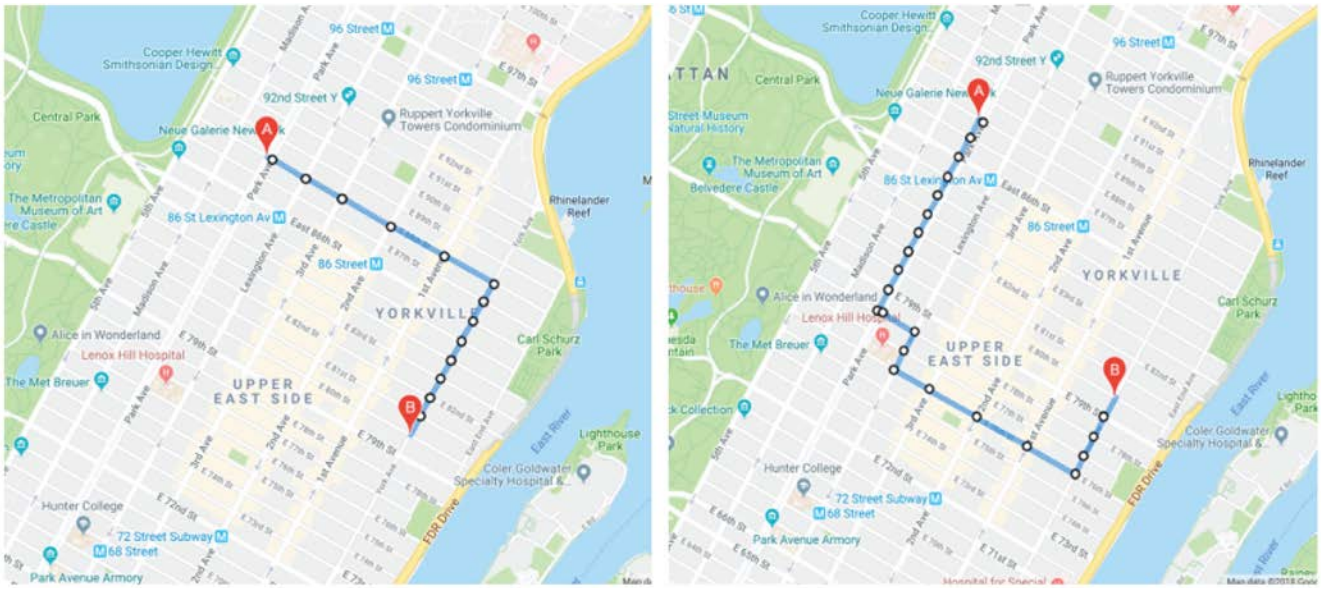


Exploración de rutas alternativas en red de N.Y.

Figura 9. Exploración en la búsqueda de la ruta alternativa más corta.



Ruta alternativa más corta *R*1 y Ruta opcional *R*2 en red vial de Lerma.



Ruta alternativa más corta *R*1 y Ruta opcional *R*2 en red vial de N.Y.

Figura 10. Rutas alternativas impesas en mapas con polilineas.

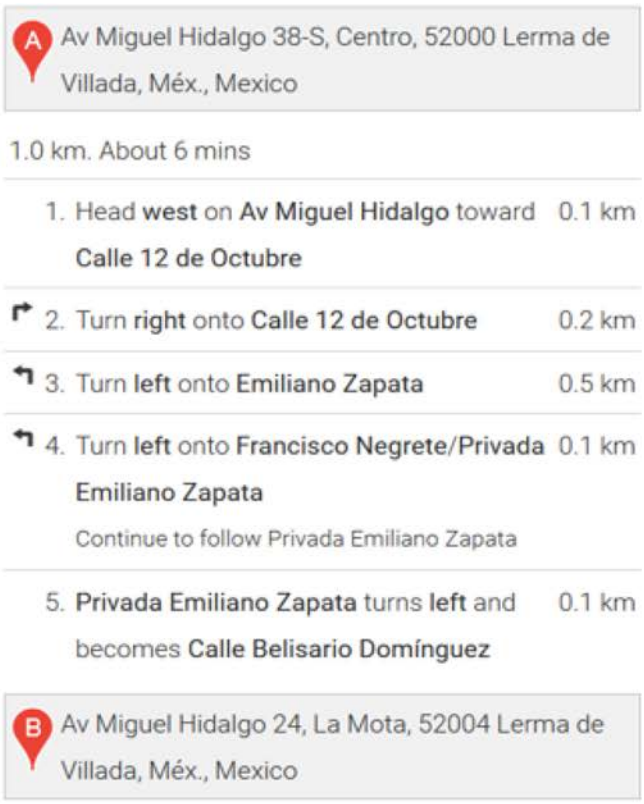
Programa 3. Programa de solicitud de rutas por lista de indicaciones textuales en API de *Google Maps*.

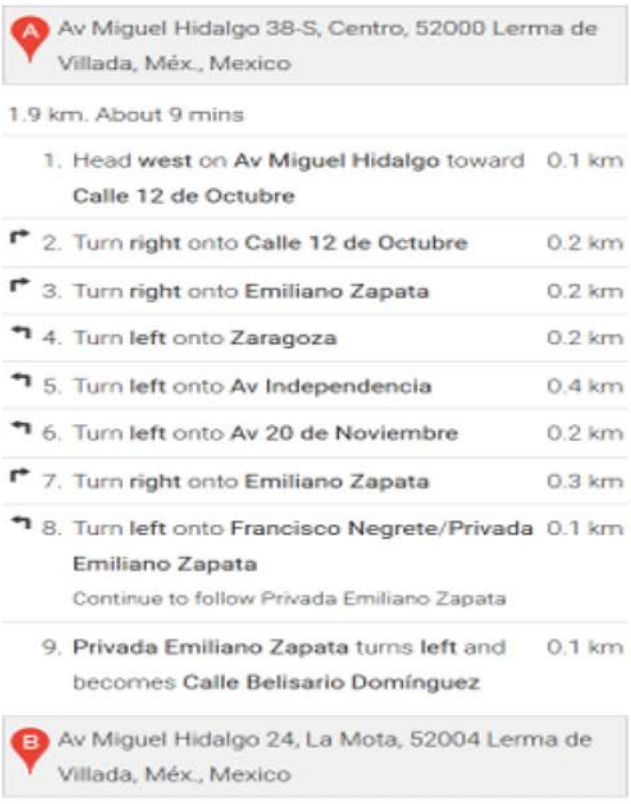
Programa 3 Regresión dígrafo – mapa

```
1:    Head
2:       body height: 100%;
3:       floating-panel;
4:    end head
5:    body
6:       function initMap {
7:    var direction service=new google.maps.directionService
8:    var directionDisplay=new google.maps.directionrenderer
9:       var newmap = new google.maps.map
10:      directiondisplay.setmap(newmap)
11:      Directiondisplay.setpanel(rightpanel)}
12:   function calculate&display(directionservice,directiondisplay) {
13:      origin (path [1]),
14:      destination([n]),
15:      waypoints ([2: n-1])
16:      travelMode: 'DRIVING'
17:      Map: Roadmap}
18:         Function (response) {
19:            directiondisplay.setdirection()}
20:   end body
```

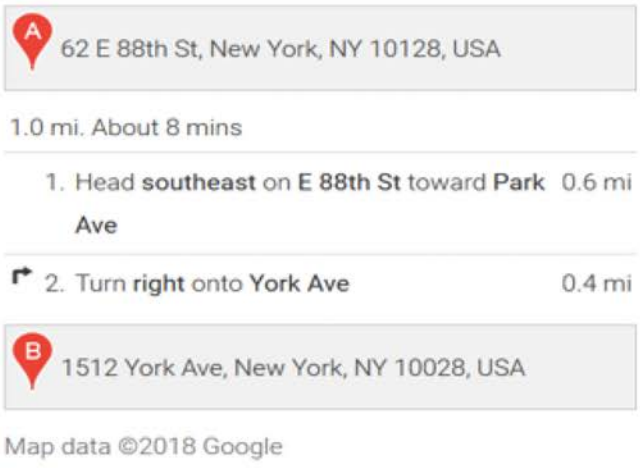
contrar la ruta alternativa más corta desde un origen a un destino, de entre todas las soluciones posibles. Esto permitió validar la información obtenida por la API de *Google Maps* para su aplicación y estudio con otros algoritmos de optimización de rutas con datos reales. Es importante considerar el uso de otros algoritmos de búsqueda de rutas como: Árbol de expansión, Floyd-Warshall, Bellman-Ford, búsqueda A*, Kruskal etc., que pueden encontrar la misma ruta en un menor tiempo. Sin embargo, el algoritmo de Dijkstra por la naturaleza de su desarrollo en dígrafos permite encontrar más de una ruta alternativa adicional a la ruta más corta óptima entre dichos dos puntos.

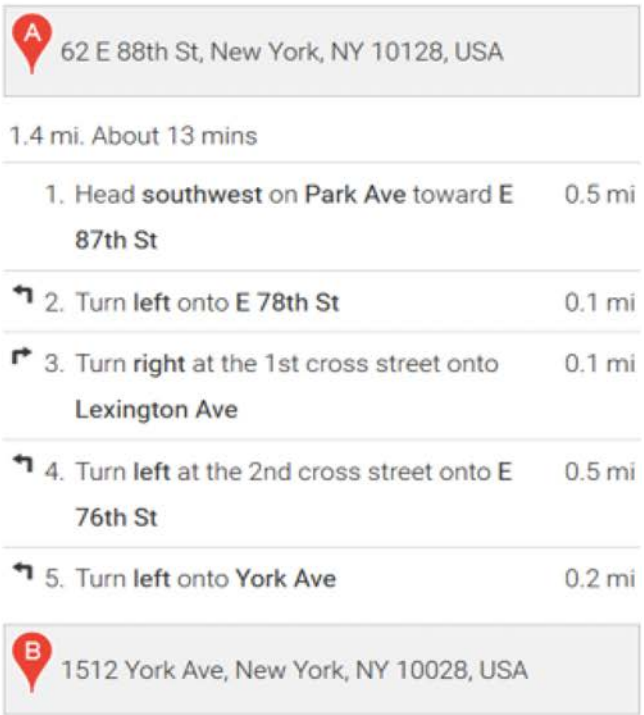
Con la ruta solución obtenida por el algoritmo de Dijkstra aplicado a la búsqueda de rutas alternativas en dígrafos, los resultados pueden ser expresados como recorridos en redes viales. Esto es, representar la solución como una serie de líneas (polilínea) impresas en la red vial al solicitar un cálculo de indicaciones sobre el mapa del cual se extrajo originalmente el dígrafo, así como por instrucciones en forma de indicaciones viales.





Indicaciones textuales de ruta alternativa *R*1 y *R*2 en Lerma.





Indicaciones textuales de ruta alternativa *R*1 y *R*2 en N.Y.

Figura 11. Servicio de rutas por indicaciones de *Google Maps*.

REFERENCIAS

(1) Gobierno de México (2016). *Ley de movilidad del distrito federal, Título tercero, Capitulo XIV, Artículo 212 de la infraestructura para la movilidad y su uso*. Gaceta oficial de la ciudad de México, CDMX, México.
(2) Instituto Nacional de Estadística y Geografía (INEGI) (2014). *Red nacional de caminos, ISO 14825:2011 Intelligent transport systems*. Recuperado de https://www.imt.mx/images/files/USIG/rnc/Documento_Tecnico.pdf

(3) Ardizzone, E. (2012, 25-29 de November). Extracting touristic information from online image collections, Conferencia. En *Eighth International Conference on Signal Image Technology and Internet Based Systems* (pp. 482-488). Nápoles, Italia: IEEE.
(4) Recuero, A. (1994). Aplicaciones de la teoría de grafos: búsqueda de caminos en una red y análisis de su conectividad. *Informes de la Construcción*, 46(433): 33-45. https://doi.org/10.3989/ic.1994.v46.i433.1115
(5) Chen, Y., Shen, S., Chen, T., Yang, R. (2014). Path Optimization Study for Vehicles Evacuation Based on Dijkstra algorithm. *Procedia Engineering*, 71(2014): 159-165. https://doi.org/10.1016/j.proeng.2014.04.023
(6) Angel Restrepo, P.L, Marín, L.F. (2011). A computational method to obtaint optimal paths in road networks. *Dyna*, 78(167): 112-121.
(7) Nuñez, M. (2016, 1 de septiembre). Knowledge Tier Platform for Graph Mining in (Smart) Cities, Ponencia. En *Symposium on Information Management and Big Data SIMBig* (pp. 110-113). Perú.
(8) Zhan, F., Noon, C. (1998). Shortest path algorithms: An evaluation using real road networks. *Transportation Science*, 32(1): 65-73. https://doi.org/10.1287/trsc.32.1.65
(9) Eneh, A., Arinze, U. (2017). Comparative analysis and implementation of dijkstra's shortest path algorithm for emergency response and logistic planning. *Nigerian Journal of Technology*, 36(3): 876-888.
(10) Villalobos, A. (2006, 7-8 de septiembre). Grafos: herramienta informática para el aprendizaje y resolución de problemas reales de teoría de grafos, ponencia. En *X Congreso de Ingeniería de Organización*, Valencia, España. Asociación para el Desarrollo de la Ingeniería de Organización (ADINGOR)
(11) Abraham, I., Delling, D., Goldberg, A. (2013). Alternative routes in road networks. *Journal of Experimental Algorithmics*, 18(1): 1-17. http://dx.doi.org/10.1145/2444016.2444019
(12) Peng, W. (2012). A Fast Algorithm to Find All-Pairs Shortest Paths in Complex Networks. *Procedia Computer Science*, 9: 557-566. https://doi.org/10.1016/j.procs.2012.04.060
(13) Restrepo, J., Sánchez, J. (2004). Aplicación de la teoría de grafos y el algoritmo de Dijkstra para determinar las distancias y las rutas más cortas en una ciudad. *Scientia et technica*, 10(26): 121-126.
(14) Nathaniel, O., Nsikan, A. (2017). Anapplication of Dijkstra's Algorithm to shortest route problem. *IOSR Journal of Mathematics*, 13(3): 20-32.

* * *