Optimización de Recorridos

Isnardo Arenas Navarro[*], Nicolas Rey Villamizar[**]

14 de mayo de 2009

Resumen

En este proyecto tratamos el problema de encontrar el costo mínimo en un recorrido desde un lugar A a un destino B, en un principio consideraremos únicamente la distancia tal como lo plantea el problema original y finalizando haremos una propuesta para considerar otro importante factor que interviene a la hora de transportarnos como lo es el trafico en las vías.

Índice

1. Introducción **2**

2. Objetivos **3**

3. Problema y Modelado **3**

3.1. Algoritmo de Floyd's 3

3.2. Algoritmo de Floyd's y Ruta entre los nodos 6

4. Otras posibles aplicaciones **7**

5. Conclusiones **9**

Referencias **9**

[*]Estudiante Graduado de Ms. Matemática Aplicada Universidad de Puerto Rico Recinto Mayagüez

[**]Estudiante Graduado de Ms. Ingeniería Eléctrica Universidad de Puerto Rico Recinto Mayagüez

1. Introducción

Nos es interesante estudiar el caso optimización en el que debemos decidir que recorrido nos es más conveniente tomar a la hora de desplazarnos en medio de una ciudad ya que suele ser un sitio complicado para conducir. Debido a calles estrechas, vías cerradas por mantenimiento, peatonales o de una sola dirección. En este pequeño ejemplo en la figura 1 se muestra el problema a la hora de planificar una ruta realizando el recorrido más corto posible.

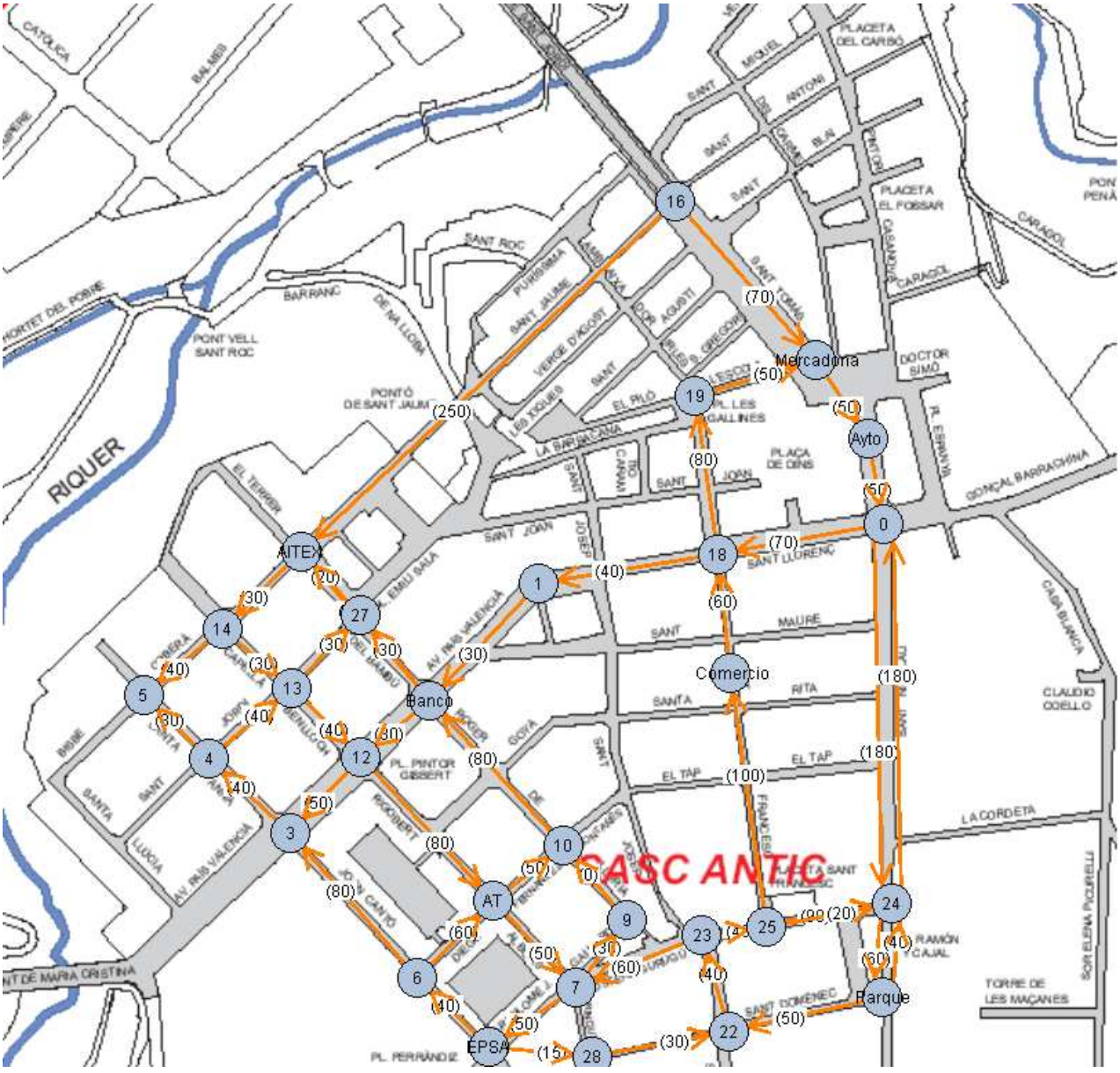

Figura 1: Centro de una ciudad[1]

Notemos que es un problema interesante de la vida real sobre todo si no se conoce bien la ciudad y en donde el mapa se puede construir a partir de información conocida de imágenes de las vías incluso se puede extender agregando un valor de peso según el trafico que registre la vía que conecta los lugares por los que debemos desplazarnos. Hoy en día esta información no es un

problema, o difícil de obtener, gracias a herramientas como sistemas de posicionamiento global o un simple mapa que podemos adquirir previamente en tiendas.

2. Objetivos

- Discutir un modelo matemático que nos permita tratar el problema.
- Estudiar algoritmos que nos permitan resolver el problema basado en el modelo conseguido.
- Proponer un posible tratamiento del problema al vincular el trafico en la ruta.

3. Problema y Modelado

Primero asumiremos el interés de resolver la distancia mas corta para ir del origen A al destino B, esto en la vida real difiere bastante del concepto intuitivo de viajar en linea recta, ya que en una ciudad esto es imposible en la mayoría de los casos, también se podría llegar a pensar en tomar la mejor aproximación a la linea recta pero esto no se puede conseguir ya que algunos caminos no pueden ser tomados (caminos peatonales o vías en reparación) o simplemente por que la dirección en la que necesitamos ir no esta permitida, es decir,ni si quiera podemos considerar las vías como una maya cuadriculada vasta con ver el mapa de cualquier ciudad para ver que es complejo.

En la figura 2 mostramos nuevamente la información que se tenia de la figura 1 sin considerar el mapa, solo la información vial de la distancia y dirección en la que se puede viajar, esto nos induce a pensar en el problema como un grafo orientado o Di-grafo, donde los nodos están representados por los lugares de intersección de las vías y las aristas llevarían el peso de la distancia, usualmente un Di-grafo tiene una representación matricial, $D \in \mathbb{R}_{+}^{n \times n}$, donde n es el numero de nodos y $d_{ij} = w$ indica que hay una arista que sale del nodo i hacia el nodo j y que tiene un peso w, cuando no hay arista desde i hacia j usualmente se asigna un valor de 0, pero para nuestro problema este modelo no nos sirve, así que consideraremos las no conexiones o ausencia de aristas desde i hacia j con un valor de ∞ para que esta ruta no sea tomada y así cualquier otra ruta se considere mas apropiada.

Fácilmente se puede escribir la matriz con estas condiciones del grafo en un lenguaje como Matlab, ya que nos permite definir matrices donde las entradas pueden tomar valor de ∞, pero si nos enfrentáramos a un lenguaje que no considere o considere un valor infinito, entonces podríamos usar un valor suficientemente grande, o pensar en sumar todas las distancias presentes en el grafo y aumentar esta en uno, lo cual nos garantiza que cualquier otro camino sería mejor.

3.1. Algoritmo de Floyd's

El algoritmo de Floyd compara todos los posibles caminos a través del grafo entre cada par de vértices, lo hace comparando la ruta conocida hasta el momento de forma "directa" con la posible tomando un nodo intermedio. El algoritmo es capaz de hacer esto en orden n^3 (esto es notable considerando que pueden haber hasta n^2 aristas en el grafo, y que cada combinación de aristas se prueba). Lo hace mejorando paulatinamente una estimación del camino más corto entre dos vértices, hasta que se sabe que la estimación es óptima, es decir cuando ya ha terminado de comparar todas las rutas.[2]

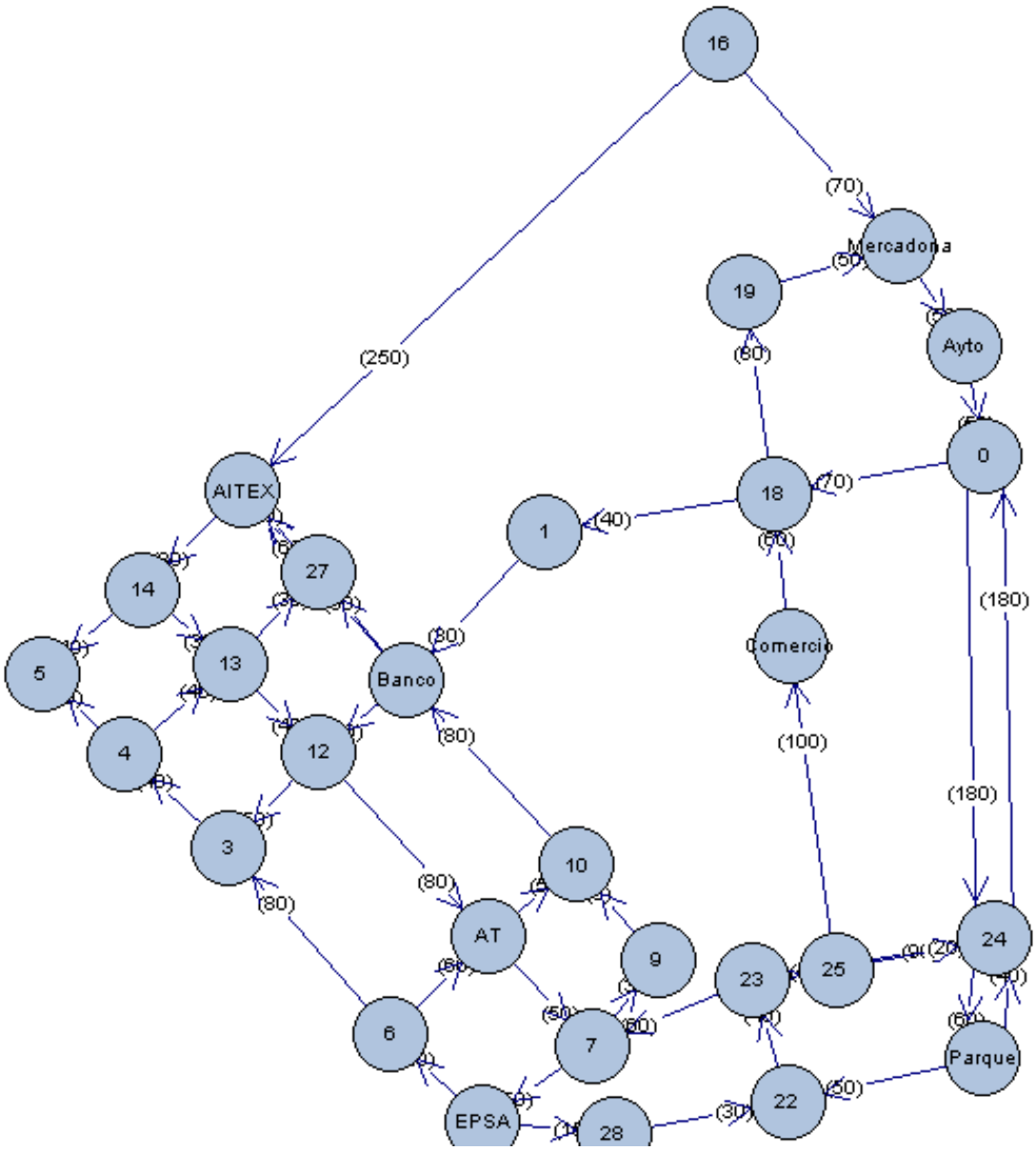

Figura 2: Centro de una ciudad sin mapa[1]

```
function [ A ] = Floyd( A )
%Encontrar el camino minimo más corto
n=size(A,1);
for k=1:n
    for i=1:n
        for j=1:n
            r=A(i,k)+A(k,j);
            A(i,j)=min(A(i,j),r);
        end
    end
end
end
```

Ejemplo. 3.1. *Consideremos un sistema de vías dadas en el grafo de la figura 3, a partir de este se puede asociar la siguiente matriz*

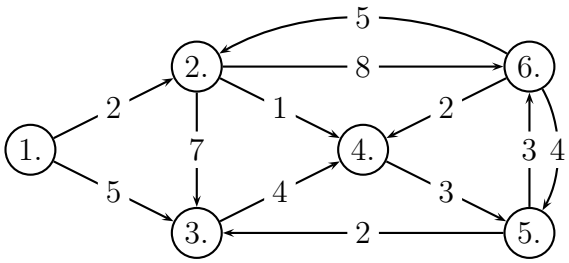

Figura 3: Representación del Grafo de distancia

$$\begin{bmatrix} 0 & 2 & 5 & \infty & \infty & \infty \\ \infty & 0 & 7 & 1 & \infty & 8 \\ \infty & \infty & 0 & 4 & \infty & \infty \\ \infty & \infty & \infty & 0 & 3 & \infty \\ \infty & \infty & 2 & \infty & 0 & 3 \\ \infty & 5 & \infty & 2 & 4 & 0 \end{bmatrix}$$

Ahora con los datos del ejemplo 3.1 podemos aplicar el algoritmo de Floyd's y en Matlab tedríamos el siguiente resultado

```
A =

     0     2     5   Inf   Inf   Inf
   Inf     0     7     1   Inf     8
   Inf   Inf     0     4   Inf   Inf
   Inf   Inf   Inf     0     3   Inf
   Inf   Inf     2   Inf     0     3
   Inf     5   Inf     2     4     0

>> Floyd(A)

ans =
```

```
  0     2     5     3     6     9
Inf     0     6     1     4     7
Inf    15     0     4     7    10
Inf    11     5     0     3     6
Inf     8     2     5     0     3
Inf     5     6     2     4     0
```

Entonces a partir de la matriz asociada al grafo para las orientaciones de las vías y sus respectivas distancias se tiene con el algoritmo de floyd's como resultado una nueva matriz con los valores o costos mínimos que hay, es decir, cada entrada de la matriz resultante en la posición ij determina la distancia mínima que hay desde el origen i al destino j. En esto observamos una deficiencia del algoritmo, ya que mas que podría ser de mayor interés tener la ruta que se debe tomar para lograr esta distancia mínima. En la sección 3.2 mostraremos una modificación para encontrar la ruta haciendo una modificación del algoritmo de Floyd's.

3.2. Algoritmo de Floyd's y Ruta entre los nodos

Retomando la idea del algoritmo que se trato en la sección 3.1, donde cada interacción se pretende comparar si es mejor la ruta obtenida hasta el momento o si se consigue un mejor resultado visitando un lugar intermedio, entonces almacenaremos este lugar intermedio por el que se consigue un mejor resultado para viajar desde i hasta j y luego reconstruimos la ruta en un proceso repetitivo chequeando si hay otros intermedios que tomar, ya que seguramente antes de llegar a ese lugar intermedio se deba frecuentar otro y esto lo haremos hasta no tener mas lugares intermedios que visitar proponiendo así el siguiente algoritmo.

```
function [ A,R ] = Floyds( A,li,lj )
%Encontrar el camino minimo más corto
n=size(A,1);
rut(n,n)={[]};
for k=1:n
    for i=1:n
        for j=1:n
            r=A(i,k)+A(k,j);
            if(r<A(i,j))
                rut(i,j)={k};
            end
            A(i,j)=min(A(i,j),r);
        end
    end
end
cam={[rut{li,lj}]};
R={[li,cam{1,1},lj]};
while(size(cam{1,1},2)>0)
idx=R{1,1};
t=size(idx,2);
R={[]};
cam={[]};
for i=1:t-1
```

```
    R={[R{1,1},idx(i),rut{idx(i),idx(i+1)}]};
    cam={[cam{1,1},rut{idx(i),idx(i+1)}]};
end
R={[R{1,1},lj]};
end
R=R{1,1};
end
```

Ejemplo. 3.2. *Consideremos nuevamente los datos del ejemplo 3.1 y descubramos por medio del algoritmo cual es la mejor ruta que se tiene para ir desde* 1 *hasta* 6.

Dando la siguiente orden sobre Matlab obtenemos.

```
[B,C]=Floyds(A,1,6)

B =

     0     2     5     3     6     9
   Inf     0     6     1     4     7
   Inf    15     0     4     7    10
   Inf    11     5     0     3     6
   Inf     8     2     5     0     3
   Inf     5     6     2     4     0


C =

     1     2     4     5     6
```

Basado en esto podemos interpretar que para ir desde 1 hasta 6 se debe tomar la ruta $1 \rightarrow 2 \rightarrow 4 \rightarrow 5 \rightarrow 6$ donde recorre una distancia de apenas 9

4. Otras posibles aplicaciones

Si consideramos algo más que la simple distancia el concepto de linea recta como mínimo muere totalmente, por ejemplo si tenemos que para cierta hora del día se conoce el estado de trafico en cada arista o vía, y que este trafico induce una velocidad promedio sobre la vía, podemos darle una ponderación o valor directamente proporcional a la distancia ya que no es lo mismo recorrer $20m$ de trafico pesado a $300m$ con el mismo trafico pesado, entonces se podría considerar un factor $\mu \geq 1$ que depende de la velocidad promedio que se tiene en esa vía y que este solo sera 1 si no se presenta problema de trafico.

Consideremos a modo de idea la siguiente función en donde tenemos el valor $\mu(v)$

$$\mu(v) = \begin{cases} 1 & \text{si } v \geq v_0 \\ 1 - ln\left(\frac{v}{v_0}\right) & \text{si } v < v_0 \end{cases}$$

Donde v_0 lo consideramos un valor mínimo ideal de velocidad promedio, esta es solo una primera elaboración, la cual se podría luego ajustar a condiciones más apropiadas o reales, pero se ajusta a lo que queremos mostrar y dejar abierto para la discusión.

Ejemplo. 4.1. *Nuevamente tomemos como referencia el ejemplo 3.1 y que ahora a demás tenemos que la velocidad mínima ideal es* $v_0 = 40\frac{millas}{hora}$ *y ademas los siguientes informes de trafico(velocidad promedio en* $\frac{millas}{hora}$*) sobre las vías las cuales para simplificar también daremos sobre un Di-grafo mostrado en la figura 4.*

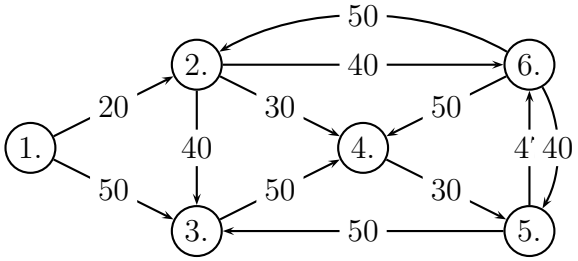

Figura 4: Representación de La velocidad media sobre cada vía

Luego de esta información tenemos que $\mu_{12} \approx 1{,}51$, $\mu_{24} \approx 1{,}29$ y $\mu_{45} \approx 1{,}29$ en las demás vías tendríamos que el factor es 1, entonces ahora la matriz de costos esta dada por

$$\begin{bmatrix} 0 & 3{,}02 & 5 & \infty & \infty & \infty \\ \infty & 0 & 7 & 1{,}29 & \infty & 8 \\ \infty & \infty & 0 & 4 & \infty & \infty \\ \infty & \infty & \infty & 0 & 3{,}87 & \infty \\ \infty & \infty & 2 & \infty & 0 & 3 \\ \infty & 5 & \infty & 2 & 4 & 0 \end{bmatrix}$$

Ahora nuevamente veamos cual es la mejor ruta para ir desde 1 hasta 6 damos la orden en Matlab y tenemos

```
A =

         0    3.0200    5.0000       Inf       Inf       Inf
       Inf         0    7.0000    1.2900       Inf    8.0000
       Inf       Inf         0    4.0000       Inf       Inf
       Inf       Inf       Inf         0    3.8700       Inf
       Inf       Inf    2.0000       Inf         0    3.0000
       Inf    5.0000       Inf    2.0000    4.0000         0

>> [B,C]=Floyds(A,1,6)

B =

         0    3.0200    5.0000    4.3100    8.1800   11.0200
       Inf         0    7.0000    1.2900    5.1600    8.0000
       Inf   15.8700         0    4.0000    7.8700   10.8700
       Inf   11.8700    5.8700         0    3.8700    6.8700
       Inf    8.0000    2.0000    5.0000         0    3.0000
       Inf    5.0000    6.0000    2.0000    4.0000         0
```

```
C =

     1     2     6
```

Como se ve en el ejemplo 4.1 ya la ruta más conveniente no esta asociada a la mera distancia más corta, por que ahora se debe ir por la ruta $1 \rightarrow 2 \rightarrow 6$, de forma similar como consideramos el trafico se podría considerar otras variables que influyan en el recorrido, como si hay que pagar algún tipo de impuesto las condiciones de la vía, entre otras.

5. Conclusiones

Durante el desarrollo de este proyecto hemos dado algunas cortas conclusiones que agruparemos en esta sección junto a otras nuevas.

Aunque el algoritmo de Floyd's fue creado para encontrar la distancia más cortas se pueden lograr múltiples aplicaciones, como por ejemplo haciendo modificaciones que nos indiquen la ruta o pasos a seguir,además de tratar otros problemas que se puedan representar con Di-grafos, ejemplo si se conoce como se incrementa el costo de un articulo a partir de unos procesos y se usaría para encontrar los que me disminuyen los costos.

Dado cualquier otra variable que intervenga en el fenómeno a estudiar si esta es significativa el problema se puede ampliar modificando los pesos del grafo de la forma más apropiada al fenómeno estudiado.

Referencias

[1] Caso práctico: Moviéndose en coche por el casco antiguo
`http://personales.upv.es/arodrigu/grafos/ayuda/practico2.htm`.

[2] Michael J. Quinn Parallel Programming with MPI and OpenMP, Mc. Graw Hill