Tarea 3

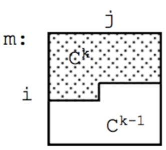
Nombre: Juan Sebastian Vargas
Fecha: 4/05/16

Punto 1

Floyd-Warshall

Para Solucionar este problema se utilizaria el mismo algoritmodescrito en 1, donde D^* es la matriz que contiene el costo minimo de los viajes.

```
proc FW (var m:array[1..n,1..n] of R*)
{Pre:     m = D}
{Pos:     m = D*}
⌈  k:= 0;
   {Inv P: 0≤k≤n ∧ m = C^k }
   do k≠n →    i,j,k:= 1,1,k+1;
                {Inv P1: 1≤i≤n+1 ∧ 1≤j≤n ∧ m:
```




```
                                                          }
               do   i≠n+1
                    →   m[i,j]:= min {m[i,j], m[i,k]+m[k,j]} ;
                        if   j≠n  →   j:= j+1
                        []   j=n  →   i,j:= i+1,1
                        fi
               od
   od
⌋
```

Figura 1: Floyd-Warshall ejemplo libro

si se necesitara encontrar y reconstruir ese camino, se haria el mismo algoritmo, solo que actualizando una matriz tal que:

Esta idea se plasma al mantener una matriz `intmd`, de manera que, para $1 \le i, j \le n$:

$$
intmd_{ij} = \begin{cases} v & \text{, si existe una ruta que pasa por } v \in V\text{, con costo mínimo} \\ 0 & \text{, si la ruta más corta de } i \text{ a } j \text{ es el arco } (i,j) \in E \\ -1 & \text{, si no hay camino de } i \text{ a } j. \end{cases}
$$

Figura 2: Matriz para reconstruir el camino de costo minimo

Para la complejidad temporal podemos encontrarla facilmente, como vemos en el algoritmo variamos i, j y k hasta n, donde n es el numero total de Nodos, y $T_{FW} = O(n^3)$, para la comlejidad Espacial tenemos guardamos una matriz de nxn y es por esto que $S_{FW} = O(n^2)$

Dijkstra

el algoritmo que me encuentra la ruta optima de un nodo f hasta el resto de nodos es:

```
proc Dijkstra (G(V,E,c): grafo, f: V; var df:array[1..n] of R* )
{Pos:     df = D*f. }
⌈ M:= {f};
  for v∈V     →    df[v]:= c[f,v] rof;
  {Inv P:  ⟨∀v: v∈M : df[v]=D*fv ⟩  ∧  ⟨∀v: v∈V\M : df[v]=C^M fv ⟩ }
  do M≠V →    w:= Escoja_min(df,V\M);
              {Q1: P ∧ df[w]≤df[V\M]  ∧ w∈V\M }
              M:= M∪{w};
              for v ∈ V\M
                   →    df[v]:= min{df[v],df[w]+c[w,v]}
              rof
  od
⌋
```

Figura 3: Algoritmo de Dijkstra

donde la complejidad temporal depende de dos partes, la inicializacion y las iteraciones, la inicializacion es el primer for que llena el df, y las iteraciones son los ciclos que siguen.

$$T_D = T(Ini) + T(ciclos) = O(n) + O(n)[T(Escoja_min) + T(eliminar)] + O(e) \quad (1)$$

Donde $T(Escoja_min) = O(n)$, ya que existen n diferentes nodos, teniendo todo en cuenta se obtiene que

$$T_D = O(n)[O(n)] + O(e) = O(n^2 + e) \quad (2)$$

pero esta seria la complejidad temporal para encontrar el camino minimo de un nodo dado f a el resto de los nodos del grafo, para obtener la totalidad de caminos minimos de todos los nodos hacia todos los nodos, debemos hacer este proceso para todos los nodos, en este casi la complejidad seria entonces:

$$T_{TD} = O(n(n^2 + e)) = O(n^3) \tag{3}$$

donde se asumio que $e \ll n^3$, y para la espacial se guardara n vectores de la manera $di : i \in 1..f..n$, y de esta manera $S_D = O(n^2)$

Punto 2

2.a

Para modelar este problema de esta manera tenemos un grafo $G(V, E, c)$debemos y definir cuando se llega a un estado de solucion, y esto pasa cuando:

$$v \in V : P = \sum_{i=1}^{n} c_i^v \tag{4}$$

donde:

$$c_i^v = \begin{cases} c_i & si \quad v_i estallenaenv \\ \\ 0 & si \quad v_i estavaciaenv \end{cases} \tag{5}$$

2.b

Punto 3