LECCIÓN 12: ALG. VORACES

Problema: ¡CAMINOS MÍNIMOS

- Dado un grafo $G=\langle N,A\rangle$ N: nodos o vertices A: aristas dirigidas
- Cada arista se etiqueta por la longitud de arista, con un valor $>,0$.
- Se selecciona uno de los nodos entre N. El problema consiste en encontrar como ir desde un origen al resto de longitud mínima que va desde el nodo seleccionado, origen, y pasa por el resto de los nodos.
- Un algoritmo que resuelve este problema es el ALG. de Dijkstra.

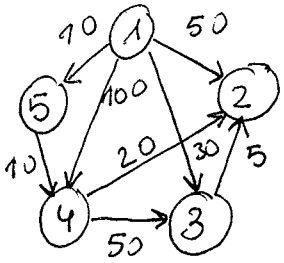
ALG. DIJKSTRA:

C: Todos los nodos — N = C U S en todo momento

S: Nodos seleccionados. — Si S=N el problema se ha resuelto.

- FUNCION de SELECCIÓN: Se selecciona aquel nodo en C cuya distancia al origen sea mínima
- Añadir a la Solucion: El nodo seleccionado se añade a la solucion
- FUNCION SOLUCION. Si S=N.
- FUNCION OBJETIVO: El camino obtenido sea de menor longitud.

EJEMPLO:



PASO	v	C	D (Distancia mínima)
Inicialización	—	{2,3,4,5}	[50, 30, 100, 10] D[2] D[3] D[4] D[5]
2	5	{2,3,4}	[50, 30, 20,]
3	4	{2,3 4	[40, 30]
4	3	{2 4	[35]

D = [35, 30, 20, 10]

P[2]=4 P[3]=1 P[4]=5 P[5]=1

LECCION 12.- ALGORITMOS VORACES

caminosMinimos (continuación)

```
FUNCION DIJKSTRA(L:matriz[1..n][1..n]
    vector D[2...n]
    C ← {2,3,...n} //candidatos
    para i=2...n
        D[i] ← L[1,i]

    repetir n-2 veces
        v ← el elemento de C que minimiza D[v]
        C ← C-{v}
        S ← S∪{v}
        para cada w en C
            Si D[w] > D[v] + L[v,w]
                D[w] ← D[v] + L[v,w]
                P[w] = v

        end
    end
    devolver D
end
```

EFICIENCIA

Tenemos un bucle interno que recorre todos los nodos de C

$$\sum_{j=1}^{n-2} \sum_{i=j}^{n} 1 = \sum_{j=1}^{n-2} n-j+1 =$$

$$\sum_{j=1}^{n-2} n - \sum_{j=1}^{n-2} j + \sum_{j=1}^{n-2} 1$$

$$\downarrow$$

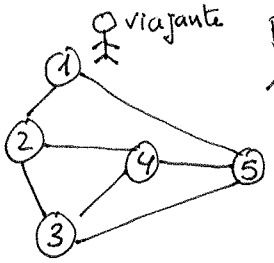
$$n(n-2) - \left(n-2\right)\left(\frac{n-1}{2}\right) + n-1$$

$$\in \Theta(n^2)$$

LECCION 12: ALG. VORACES

Problema: El viajante de comercio

Se conoce las distancias entre un cierto numero de ciudades. Un viajante debe, a partir de una de ellas, visitar cada ciudad exactamente una vez y regresar al pto de partida habiendo recorrido en total la menor distancia posible.



Formulación en términos de grafos

Dado un grafo conexo y ponderado y dado uno de sus vértices, v_0, encontrar el ciclo Hamiltoniano (se pasa una sola vez por cada vertice, excepto el v_0 que se vuelve a él), de coste mínimo que comienza y termina en v_0.

- ESTRATEGIA Greedy -

Sea C el camino construido hasta el momento que acaba en v y comienza en v_0. $(v_0, v_1 \dots, v)$

Iniciamos C a vacio. A continuación le añadimos v_0, en este caso $v = v_0$.

Caso final sería cuando C contiene todos los vertices de G, en el ejemplo C contendría 1, 2, 3, 4, 5 y simplemente le añadimos 1 para que vuelva al origen.

En el caso que C no contiene todos los vértices de G se busca aquel arco (v, w) con la condición de que $w \notin C$ que tenga peso mínimo y se añade w a C.

LECCION 12: ALG VORACES

El viajante de comercio (continuación)

```
typedef

void Viajante (Grafo &g, Grafo &sol){
    int n = g.num_vertices();
    bool yaesta[n];
    for (int i=0; i<n; i++)
        yaesta[i] = false;
    int verticesencurso = 0;
    for (int i=0; i<n; i++){
        int verticeanterior = verticeencurso;
        verticeencurso = Busca(g, verticeencurso, yaesta);
        sol.ponarista (verticeanterior, verticesencurso, true);
    }
}
```

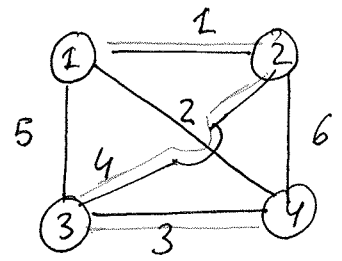
El proceso de selección de la técnica Greedy esta en la función Busca.

```
int Busca ( Grafo &g, int vertice, bool *yaesta) {
    int n = g.num_vertices();
    int mejorvertice = 0;
    int min = numeric_limits<int>::max();
    for (int i=0; i<n; i++)
    {   if (i<>vertice)
            if (!yaesta[i] && g.arista(vertice, i) < min)
            {
                min = g.arista(vertice, i);
                mejorvertice = i;
            }
    }
    return mejorvertice;
}.
```

EFICIENCIA $\Theta(n^2)$

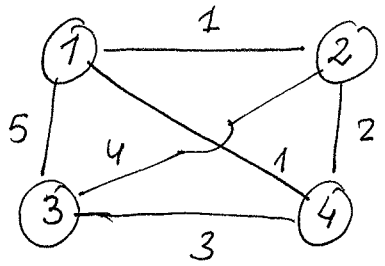
LECCION 12 : ALG. VORACES

Ejemplo: Encuentra la solucion optima



Paso	C	v	costo acumulado
1	(1)	1	0
2	(1, 2)	2	1
3	(1, 2, 3)	3	5
4	(1, 2, 3, 4)	4	8
5	(1, 2, 3, 4, 1)		10

Ejemplo 2.- No funciona.



Paso	C	v	costo acumulado
1	(1)	1	0
2	(1, 2)	2	1
3	(1, 2, 4)	4	3
4	(1, 2, 4, 3)	3	6
5	(1, 2, 4, 3, 1)		11

Sin embargo si hubiesemos escogido el camino

(1, 2) (2, 3) (3, 4) (4, 1) tiene un costo de 9.

LECCION 12: ALG Voraces

EJERCICIO (examen sept 2014)

Una compañía eléctrica debe elegir la ruta que deben seguir sus técnicos para arreglar una serie de averías que se han producido en la ciudad. Las averías están situadas en 5 ptos B, C, D, E, F. La central eléctrica está en A. La distancia entre los distintos puntos viene dada por la tabla

	A	B	C	D	E	F
B	6					
C	8.07	6				
D	17.55	12.70	15			
E	16.52	12.05	15.32	4		
F	19	15.52	19.38	8.6	5	

Encontrar usando una técnica greedy la mejor ruta posible para minimizar la distancia recorrida por los técnicos.

- Candidatos $\leftarrow \{B, C, D, E, F\}$
- Solucion: Se inicia $S = (A, -, -, -, -, -)$
- Matriz de distancias M. $M(i,j)$: distancia del pto i al pto j
- posicion-actual $\leftarrow$ ultima localizacion insertada en S
- OBJETIVO $\min \sum_{i=0}^{4} M(S(i), S(i+1))$

- ALGORITMO

LECCION 12. ALG VORACES

```
float Ruta (Matriz dM, vector<int> &S, vector<int> C){
  //INICIALIZACION
      S.push_back (C[0]);
      C.erase (C[0]);
      float dist_total=0; int posicion_actual=0;
      while (C.size()>0){
        float min_dis= numeric_limits <float>::max();
        int posicion_min;
        //SELECCION
        for(int i=0; i<C.size(); i++){
          if [M[S[posicion_actual], C[i]] <min_dis)
          { min_dis = M[S[posicion_actual], C[i]];
            posicion_min=i;
          }
        }
        S.push_back ( C[posicion_min]);
        posicion_actual ++;
        dist_total+= min_dis;
        C.erase ( C[posicion_min]);
      }
      return dis_total;
}
```