

Facultad de Ciencias de las Ingenierías

Departamento de Ingeniería de Sistemas y Computación

Tarea 9 (T9):

Floyd Warshall

Por:

Ansël E. Corona Coste 2014-0031

Profesor:

J.R Felipe Nuñez

Asignatura:

Diseño y Análisis de Algoritmos

ISC-405-T-001

Santiago de los Caballeros

República Dominicana

**Marco teórico:**

El problema que intenta resolver este algoritmo es el de encontrar el camino más corto entre todos los pares de nodos o vértices de un grafo. Esto es semejante a construir una tabla con todas las distancias mínimas entre pares de ciudades de un mapa, indicando además la ruta a seguir para ir de la primera ciudad a la segunda. Este es uno de los problemas más interesantes que se pueden resolver con algoritmos de grafos.

Existen varias soluciones a este problema y los algoritmos a aplicar dependen también de la existencia de arcos con pesos o costes negativos en el grafo. En el caso de no existir pesos negativos, sería posible ejecutar V veces el algoritmo de Dijkstra para el cálculo del camino mínimo, donde V es el número de vértices o nodos del grafo. Esto conllevaría un tiempo de ejecución de O(V^3) (aunque se puede reducir). Si existen arcos con pesos negativos, se puede ejecutar también V veces el Algoritmo de Bellman-Ford, una vez para cada nodo del grafo. Para grafos densos (con muchas conexiones o arcos) esto conllevaría un tiempo de ejecución de O(V^4).

El algoritmo de Floyd-Warshall ('All-Pairs-Shortest-Path' - Todos los caminos mínimos) ideado por Floyd en 1962 basándose en un teorema de Warshall también de 1962, usa la metodología de Programación Dinámica para resolver el problema. Éste puede resolver el problema con pesos negativos sin embargo, para ciclos de peso negativo el algoritmo tiene problemas. A continuación se muestra el pseudocódigo del algoritmo:

Floyd-Warshall (G)

Inicializar

D = A ' matriz de distancias = matriz de arcos

si i=j o Dij= infinito entonces Pi,j = nulo sino Pi,j=i 'matriz de caminos

for k = 1 to V

for i = 1 to V

for j = 1 to V

Di,j = min(Di,j , Di,k + Dk,j )

si min = Di,k + Dk,j entonces

Pi,j = Pk,j

fin

**Problema:**

Ruta más corta entre todos los pares.

En este trabajo se materializará uno o varios algoritmos del problema de la ruta más corta entre todos los pares. Aquí están los archivos de datos que describen tres grafos: Grafos g1,g2,g3.txt

La primera línea indica el número de vértices y aristas, respectivamente. Cada línea subsiguiente describe una arista (a través de la cola y la cabeza, respectivamente) y su longitud. Nota: algunas de las longitudes de los bordes son negativas. NOTA: Estos grafos pueden o no pueden tener ciclos negativos costo. Su tarea consiste en calcular el "camino corto más corto". Precisamente, primero debe identificar cuál, en su caso, de los tres grafos no tienen ciclos negativos. Para cada grafo, debe calcular todos los pares de caminos más cortos y recordar el más pequeño (es decir, calcular min u, v Vd (u, v), donde d (u, ∈ v) denota la ruta más corta distancia de u a v ). Si cada uno de los tres grafos tiene un ciclo de costo negativo, indique "NULL" como respuesta. Si exactamente un grafo no tiene ciclos de costo negativos, a continuación, indique la longitud de la ruta corta más corta en la respuesta. Si dos o más de los grafos no tienen ciclos de costo negativos, a continuación, indique la menor de las longitudes de sus caminos cortos más cortos en la respuesta.

**Código:**

def Floyd\_WarshallFunction(NumeroNodos, Aristas):  
 inf = float('inf')  
 Connections = NumeroNodos + 1  
 testcase = [[inf for j in range(Connections)] for i in range(Connections)]  
 Ans = testcase  
 for x in range(Connections):  
 for y in range(Connections):  
 if x == y:  
 Ans[x][y] = 0  
 elif (x, y) in Aristas:  
 Ans[x][y] = Aristas[(x,y)]  
 for k in range(1, Connections):  
 for i in range(1, Connections):  
 for j in range(1, Connections):  
 testcase[i][j] = min(Ans[i][j], Ans[i][k] + Ans[k][j])  
 if i == j:  
 if testcase[i][j] < 0:  
 return "Ciclo negativo"  
  
 return str(min(min(testcase, key=lambda x:min(x))))  
  
def ejecutarFuncion(file):  
 print "Grafo: "+ file  
 Aristas = {}  
 with open(file) as file\_in:  
 next(file\_in)  
 for line in file\_in:  
 Cola, Cabeza, distancia = map(int, line.strip().split(' '))  
 Aristas[(Cola, Cabeza)] = distancia  
 NumeroNodos = int(open(file).readline().strip().split(' ')[0])  
 print "Longitud mas corta: " + Floyd\_WarshallFunction(NumeroNodos,Aristas)  
  
ejecutarFuncion('g1.txt')  
ejecutarFuncion('g2.txt')  
ejecutarFuncion('g3.txt')

**Corrida:**

Total retornado:

