Laboratorio Nro. 4
Algoritmos Voraces

Kevin Alexander Herrera
Universidad Eafit
Medellín, Colombia
kaherrerag@eafit.edu.co

Jose Joab Romero
Universidad Eafit
Medellín, Colombia
jjromeroh@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

3.1 En la implementación de la solución del problema del agente viajero por medio de un algoritmo voraz se empleó como estructura de datos el grafo, esta cumple la función de relacionar los puntos a visitar por medio de arcos; el algoritmo consiste en buscar el camino mínimo hacia otro punto desde la posición actual hasta que se recorran todos lo puntos sin repetir y finalmente se regrese al origen.

3.2 La solución del problema del agente viajero por medio de un algoritmo voraz no es óptima en todos los casos, dado que es posible que exista una mejor solución que el algoritmo por naturaleza no ve (figura 1), además esta implementación posee limitantes que causan que en algunas ocasiones no arroje solución alguna (figura 1.2).

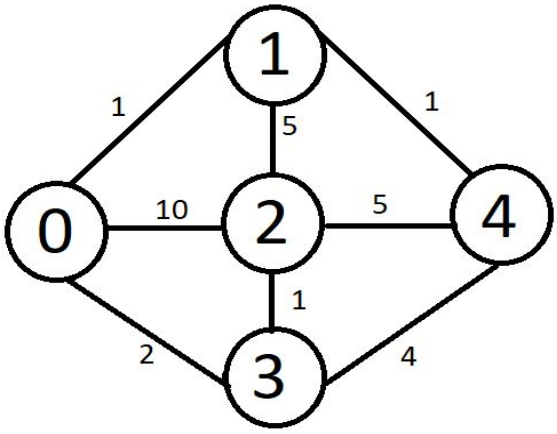

Figura 1: Camino voraz: 0 - 1 - 4 - 3 - 2 - 0
Camino óptimo: 0 - 3 - 2 - 4 - 1 - 0

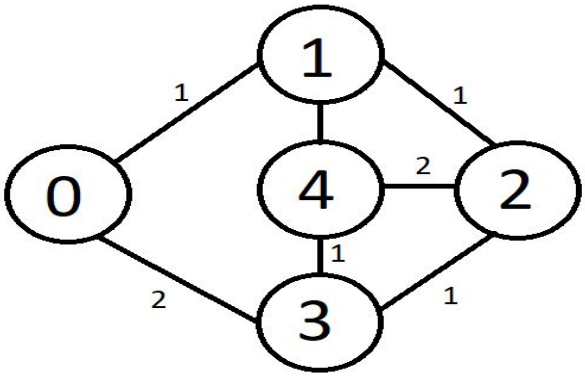

Figura 1.2: Camino Voraz:
0 - 1 - 2 - 3 - 4 ------ no puede continuar

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

3.3 Para adaptar la solución voraz del agente viajero para entregar domicilios en medellín es necesario conocer los lugares a los cuales se entregarán los pedidos y con estos puntos se realiza el grafo, el funcionamiento es el mismo, recorrer todos los puntos del grafo sin repetir y volver al origen, los arcos pueden ser dados en tiempo según la ruta más corta hasta ese punto, esta información puede ser obtenida por herramientas en internet.

3.4 la estructura de datos que utilizamos es el arreglo donde guardamos los valores de la horas de los conductores uno para la mañana y otro para la tarde, después verificamos las horas extras que hay y las multiplicamos cada una por el respectivo r

3.5 O(n*m)

3.6 n es la cantidad de casos de prueba
m es la cantidad de conductores de cada caso de prueba

4) Simulacro de Parcial

4.1 *i=j*
4.2 menor > adjacencyMatrix[element][i]
4.3.1

Paso	*A*	*B*	*C*	*D*	*E*	*F*	*G*	*H*
1	*A*	*20,A*	*infinito*	*80,A*	*infinito*	*infinito*	*90,A*	*infinito*
2	*B*	*20,A*	*infinito*	*80,A*	*infinito*	*30,B*	*90,A*	*infinito*
3	*G*	*20,A*	*infinito*	*80,A*	*infinito*	*30,B*	*90,A*	*infinito*
4	*D*	*20,A*	*infinito*	*80,A*	*infinito*	*30,B*	*90,A*	*infinito*
5	*F*	*20,A*	*40,F*	*70,F*	*infinito*	*30,B*	*90,A*	*infinito*
6	*C*	*20,A*	*40,F*	*70,F*	*infinito*	*30,B*	*90,A*	*60,C*
7	*H*	*20,A*	*40,F*	*70,F*	*infinito*	*30,B*	*90,A*	*60,C*
8	*E*	*20,A*	*40,F*	*70,F*	*infinito*	*30,B*	*90,A*	*60,C*

4.3.2 90,A
4.4
4.4.1 linea 10: temp/2
4.4.2 linea 11: temp + minimo
4.4.3 O(1)
4.5

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473

4.5.1 D
4.5.2 es posible implementar un algoritmo que trate el conjunto principal como un arreglo de elementos y lo ordene de menor a mayor para finalmente sumar las posiciones desde la primera hasta k donde k es el número de elementos del subconjunto
4.6
4.6.1 i+1
4.6.2 res + 1
4.6.3 i
4.6.4 2

PhD. Mauricio Toro Bermúdez
Docente | Escuela de Ingeniería | Informática y Sistemas
Correo: mtorobe@eafit.edu.co | Oficina: Bloque 19 – 627
Tel: (+57) (4) 261 95 00 Ext. 9473