Laboratorio Nro. 4: Algoritmos voraces

Eduard Damiam Londoño
Universidad Eafit
Medellín, Colombia
edlondonog@eafit.edu.co

Gonzalo Garcia
Universidad Eafit
Medellín, Colombia
ggarcia@eafit.edu.co

3) Simulacro de preguntas de sustentación de Proyectos

1. para el punto 1.1 usamos un grafo y ArrayList, lo que se hizo fue una implementación del vecino mas cercano, el grafo se para en el vértice 0, mira la distancia entre el vértice y sus sucesores, luego se va para el sucesor que tenga menos distancia, entonces repite el mismo proceso hasta volver al vértice 0.
2. Depende del algoritmo voraz, en nuestro caso el algoritmo debe ser completamente conexo (si no podría devolver que no existe un camino), otro caso es por ejemplo el algoritmo Dijkstra el cual no funciona en caso de que algún peso sea negativo ya que se puede confundir y quedarse en un loop infinito.
3. En el punto 2.1 usamos arreglos, lo que hace el algoritmo es que coge las rutas de la mañana y las ordena de menor a mayor y luego ordena las rutas de la noche de menor a mayor, luego a cada conductor le asigna la menor de las rutas de la mañana con la mayor de la noche luego las horas totales de la ruta de la noche y de la tarde las resta con la cantidad de horas de trabajo normal, y el resultado lo multiplica por lo que vale cada hora extra.
4. (no se tomará en cuenta la complejidad de leer la entrada del usuario ya que lo consideramos irrelevante para este ejercicio)

   la complejidad de Arrays.sort fue sacada de :
   https://docs.oracle.com/javase/7/docs/api/java/util/Arrays.html#sort(int[])

```
public int HoraExtra (){
  Arrays.sort(mañana);// O(n log(n))
  Arrays.sort(noche);//O(n log(n))
  int extra =0; // C1
  int horas = 0; //C2
  int x = 0; //C3
  for(int i =0; i < mañana.length;++i){ //O(n)
    x = mañana[i] + noche[noche.length-1-i]; //C4
    horas = x-duracion; //C5
    extra = horas*tarifa; //C6
  }
  return extra; //C7
}
```

   T(n) = (n log (n))+(n log(n)) + n + C

T(n)= 2(n log(n))+n+C
T(n) = O(n log(n))

5. En el cálculo de complejidad anterior la variable es la cantidad de elementos del arreglo mañana que es la misma cantidad de elementos del arreglo noche

4) Simulacro de Parcial

1. d) i = j;
2. a) min > adjacencyMatrix[element][i]
3. a)

PASO	a	B	C	D	E	F	G	H
1	A	20, A	∞	80, A	∞	∞	90, A	∞
2	B	20, A	∞	80, A	∞	30, B	90, A	∞
3	F	20, A	40, F	70, F	∞	30, B	90, A	∞
4	C	20, A	40, F	50, C	∞	30, B	90, A	60, C
5	D	20, A	40, F	50, C	∞	30, B	90, A	60, C
6	H	20, A	40, F	50, C	∞	30, B	90, A	60, C
7	G	20, A	40, F	50, C	∞	30, B	90, A	60, C
8	E	∞	∞	∞	∞	∞	∞	∞

b) El camino más corto de A a G, es directamente tomar la arista que conecta a A con G que tiene un costo de 90.