FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Trabajo Práctico 2 — Algoritmos D&C y Programacion Dinamica

[75.29/95.06] Teoria de Algoritmos I
Primer cuatrimestre de 2022

Alumno	Padron	Email
BENITO, Agustin	108100	abenito@fi.uba.ar
BLÁZQUEZ, Sebastián	99673	sblazquez@fi.uba.ar
DEALBERA, Pablo Andres	106585	pdealbera@fi.uba.ar
DUARTE, Luciano	105604	lduarte@fi.uba.ar
PICCO, Martín	99289	mpicco@fi.uba.ar

Entrega:	Primera
Fecha:	Miercoles 27 de Abril del 2022

Índice

1. Parte 1: Un evento exclusivo

2. Parte 2: Ciclos negativos

2.1. Algoritmo de Bellman-Ford

El algoritmo de Bellman-Ford es de tipo programacion dinamica y genera el camino mas corto en un grafo dirigido ponderado a partir de un nodo origen y a diferencia de Dijkstra permite hacerlo con grafos que tienen aristas con pesos negativos y detecta ciclos negativos.

Este algoritmo se puede describir como los siguientes pasos:

1. Iniciamos:
   - un diccionario de distancias con clave `nodo/vertice` y valor infinito.
   - un diccionario de predecesores con clave `nodo/vertice` y valor nulo.
   - la distancia de clave `nodo_origen` se cambia a 0.
2. Iterar por la cantidad de nodos del grafo:
   - por cada arista, si la distancia guardada para llegar al origen de la arista mas el peso de moverse al nodo destino de la arista es menor a la distancia guardada para llegar al nodo destino de la arista, reemplazar la distancia guardada del nodo destino.
3. Verificar que no haya ciclos negativos
   - por cada arista, si se sigue cumpliendo la condición del punto anterior, entonces hay un ciclo negativo

2.1.1. Relación de Recurrencia

Definiendo:

- s: nodo origen o *source node*
- n_i: otro nodo distinto al origen
- j: longitud maxima para llegar de s a n_i
- $minPath(n_i, j)$: función recursiva para llegar al camino mínimo de s a n_i
- n_x: predecesores a n_i
- $w(n_x, n_i)$: peso de la arista n_x y n_i

$$minPath(S, j) = 0$$

$$minPath(n_i, 0) = +\infty \text{ con } n_i \neq S$$

$$minPath(n_i, j) = min \left\{ \begin{array}{l} minPath(n_i, j-1) \\ min\ \{\ minPath(n_x, j-1) + w(n_x, n_i)\ \} \end{array} \right\}$$

2.1.2. Pseudo-codigo

```
funcion Bellman_Ford(Grafo grafo, Nodo origen)
    siendo distancias un diccionario
    siendo predecesores un diccionario

    por cada vertice v del grafo:
        distancias[v] = infinito

    distancias[origen] = 0
    padres[origen] = None

    por cada vertice:
        por cada arista de origen v y destino w y peso:
            si distancias[v] + peso < distancias[w]:
                predecesores[w] = v
                distancias[w] = distintas[v] + peso

    por cada arista de origen v y destino w y peso:
        si distancias[v] + peso < distancias[w]:
            tirar error "Hay un ciclo negativo"
```

2.2. Detalles de implementación

El algoritmo fue implementado en Python y no tiene dependencias aparte de tener instalado cualquier versión de `python3`.

2.2.1. Ejecución del programa

El programa contiene un `shebang` para ser ejecutado en una terminal de la siguiente forma:

```
./src/parte_2.py <filename>
```

El comprimido entregado incluye un archivo ejemplo en `assets/grafo.txt` con grafos ejemplos, por ejemplo:

```
B
D,A,-2
B,A,3
D,C,2
C,D,-1
B,E,2
E,D,-2
A,E,3
```

```
./src/parte_2.py ./assets/grafo.txt
```

```
Existen al menos un ciclo negativo en el grafo. A,E,D → costo: -1
```

3. Parte 3: Un poco de teoría