Departamento de
Ciencias e Ingeniería de la Computación
Universidad Nacional del Sur

Algoritmos y Complejidad
Actividad 15
18 de junio de 2019

- **Objetivos:** Grafos. Algoritmo Kruskal. Implementaciones y análisis empírico.
- **Preliminares:** Se dispone de un web-service (WS) que posibilita la construcción de un grafo no-dirigido y con pesos. El WS requiere de la especificación de dos parámetros, `nodos=N` y `arcos=A`, para construir el grafo de `N` nodos y `A` arcos. Por ejemplo:

  `http://cs.uns.edu.ar/~mom/AyC2019/grafo.php?nodos=500&arcos=65000`

  invoca al WS para generar un grafo no-dirigido y con pesos, de 500 nodos y 65000 arcos. Notar que el grafo resultado podrá o no, ser conexo. Por ello, en caso de ser necesario, el WS admitirá el uso de un tercer parámetro, `conexo=1`, para determinar si restringimos la construcción a un grafo que satisfaga tal propiedad. Es decir:

  `http://cs.uns.edu.ar/~mom/AyC2019/grafo.php?nodos=500&arcos=65000&conexo=1`

  El resultado del WS será un string en formato JSON describiendo un grafo no-dirigido y con pesos, generado de forma aleatoria, que satisface la especificación de parámetros según su invocación previa.

  Nota: El WS aceptará invocaciones para la construcción de grafos de N *nodos y* A *arcos, con* $2 \leq N \leq 500$ *y* $N-1 \leq A \leq N(N-1)/2$.

- **Problemas:** Implementar los siguientes dos problemas en un *lenguaje a elección*:

  1. Determinar si un grafo no-dirigido y pesado es conexo, según dos variantes:
     a) Implementado un recorrido por niveles o *breadth-first search*(BFS)
     b) Mediante un uso apropiado de la estructura disjoint-set, evitando el recorrido del grafo.
  2. A partir de un grafo *conexo*, no-dirigido y pesado, construir el árbol de cubrimiento minimal implementando cuatro variantes del algoritmo de Kruskal, mediante:
     a) Un conjunto de arcos:
        1) ordenados por su peso
        2) implementado con un heap invertido (min-heap).
     b) Estructura disjoint-set implementada:
        1) con heurísticas de *compresión de caminos* y *union-by-rank*.
        2) sin heurísticas.

  Para ambos problemas se deberá trabajar con grafos construidos a partir del uso apropiado del WS descripto. Seleccionar una *estructura adecuada* para el manejo del grafo.

- **Metodología:** Desarrollar un informe que conste de las siguientes partes:

  1. ***Análisis teórico espacio-temporal*** de *todas* las implementaciones desarrolladas. Es decir, desarrollar el análisis teórico de los seis algoritmos implementados: dos variantes del problema 1 más cuatro correspondientes al problema 2.
  2. ***Tablas de resultados empíricos*** detallando el tiempo demorado para la terminación de cada variante implementada, para cada uno de los dos problemas dados. Para ello, se deberá utilizar *timestamps* de forma adecuada, antes y después de cada invocación a cada variante implementada. Así, mediante el uso del WS:

     a) Construir al menos seis grafos diferentes, sin hacer uso del parámetro `conexo`, para contrastar el rendimiento de ambas variantes implementadas para el problema 1. Los resultados empíricos deberán ser plasmados en una tabla:

Grafo		BFS	Disjoint-set
N	A		
500	65000	...	...
...	...	...	...

     b) Construir otros seis grafos diferentes –haciendo uso del parámetro `conexo`– para contrastar el rendimiento de las cuatro variantes implementadas para el problema 2. Los resultados empíricos deberán ser plasmados en una tabla:

Grafo		Ordenado		Heap	
N	A	C/heurísticas	S/heurísticas	C/heurísticas	S/heurísticas
500	65000	...	...	...	...

  3. ***Conlusiones*** vinculando conceptos teóricos y resultados empíricos –según la tabla.
  4. ***Listado*** de los códigos implementados, comentados e informando el lenguaje elegido.

- **Resultados:** El trabajo deberá organizarse en comisiones de hasta ***tres alumnos***. La comisión deberá enviar un archivo ZIP conteniendo dos archivos:

  1. un PDF con el informe organizado en las cuatro secciones indicadas, y
  2. un ZIP conteniendo los códigos fuentes implementados.

  El archivo deberá ser enviado al profesor de la materia, con copia a todos los participantes del grupo, hasta las 23:59 hs del martes 18/6/2019.

  Nota 1: A modo de ayuda, en el sitio de la materia, quedan a disposición dos modelos diferentes, uno en `JAVA` *y otro en* `Python`*, con implementaciones triviales para la invocación del WS y generación de objetos a partir del string JSON resultante. Los objetos generados en estos modelos no constituyen estructuras adecuadas para la representación de grafos.*

  Nota 2: No podrá utilizarse ninguna librería adicional sin ser consultada con la cátedra.

- **Evaluación:** Esta actividad otorgará hasta 6 créditos: 1 en CT1 (estrategias), 1 en CT2 (estructuras de datos), 1 en CT3 (performance) y 3 en CT5 (consecuencias).

- **Observación:** La actividad es optativa.