Departamento de
Informática e Ingeniería
de Sistemas

Universidad Zaragoza

Prácticas de Algoritmia para problemas difíciles
Especialidad en Computación, grado en Ingeniería Informática

Curso 2017-2018

Universidad de Zaragoza
Escuela de Ingeniería y Arquitectura
Departamento de Informática e Ingeniería de Sistemas
Area de Lenguajes y Sistemas Informáticos

2 de noviembre de 2017

Organización general de las prácticas.

Se formarán equipos de una o dos personas. Si una de las dos abandona la asignatura, la otra deberá terminar en solitario. Se debe enviar un email con el nombre de los miembros del equipo antes del 15 de noviembre de 2017.

Las prácticas se realizarán en el computador `hendrix`.

El lenguaje para la implementación puede ser Java o cualquier otro elegido por el alumno (incluyendo en la documentación todos los detalles necesarios sobre el mismo: versión, compilador, etc).

La entrega de la práctica X (X = 1, 2, ...) se realizará en `hendrix` mediante la ejecución de `someter apd_17 practicaX.tar`, siendo la fecha límite la indicada al final de cada enunciado. Además, habrá que concertar una cita con la profesora para explicar las prácticas: descripción general del programa, elaboración, funcionamiento, demostración, etc.

El fichero `practicaX.tar` contendrá **un directorio denominado practicaX** con los ficheros necesarios incluyendo:

- Una memoria en formato PDF en la que se incluya un informe completo con al menos:
  - Una descripción de qué se ha implementado y usando qué fuentes (web, libros, artículos, código propio, etc). Cualquier decisión tomada sobre implementación y sus razones. Si hay algo que decir sobre el lenguaje de programación utilizado.
  - Cómo se ha repartido el trabajo entre los dos integrantes del equipo.
  - Qué pruebas se han hecho, de dónde se han sacado los datos, resultados obtenidos (incluyendo eficiencia).
- Descripción general del programa en fichero de texto: cómo está organizado, qué se puede y qué no se puede hacer (tiene que llamarse **LEEME**). Contendrá en sus primeras líneas la lista de integrantes del grupo, con el siguiente formato:

  ```
  Apellido1 Apellido2, Nombre [tab] correo@electronico [tab] login en
  hendrix

  Apellido1 Apellido2, Nombre [tab] correo@electronico [tab] login en
  ```

```
hendrix
```

Donde [tab] representa el carácter tabulador.

- Listados del código debidamente comentados y dispuestos para ser compilados y utilizados.
- Un programa para el shell `ejecutarX.sh` que automatice la compilación y ejecución de algunos casos de prueba para los programas entregados. Deberá funcionar en hendrix.
- Los ficheros auxiliares de entrada necesarios para ejecutar las pruebas del punto anterior.
- Los ficheros de otras pruebas realizadas.

En la calificación se tendrán en cuenta los siguientes aspectos: documentación, funcionamiento e implementación.

El diseño ha de ser modular, basado en el uso de tipos abstractos de datos, con todas las funciones correctamente especificadas.

Las reglas generales de tratamiento de casos de plagio de la asignatura se aplicarán, en particular, a todas las prácticas.

Práctica 1

Por motivos fiscales amazon ha decidido partir su negocio en dos, pero quiere que esto perjudique lo menos posible a su volumen de ventas, ya que algunos clientes pueden decidir no realizar una compra cuando deben dividirla en dos compras a distintos proveedores. La partición va a realizarse seleccionando (de forma disjunta) algunos de los artículos para la marca *amazon* y otros para *amazonymas*.

En una primera aproximación se dispone de la información de qué parejas de productos se han comprado alguna vez juntos, es decir, contamos con una tabla de booleanos T de forma que $T[i, j]$ es `true` si los productos indexados por i y j han sido comprados juntos alguna vez y `false` en otro caso. Se pretende realizar una partición de los productos entre *amazon* y *amazonymas*, de forma que se minimice el número de pares de productos que han sido comprados juntos alguna vez y se asignan a distinto proveedor.

Hay que tener en cuenta lo siguiente

- Cualquier fragmento de código o idea para el que se utilicen fuentes externas debe ser identificado y dichas fuentes citadas.
- En la práctica hay que hacer como mínimo lo que se especifica a continuación, pero la tarea completa no está limitada, por lo tanto los más ambiciosos pueden necesitar fijarse un límite personal.
- En la evaluación de esta práctica se tendrán especialmente en cuenta dos aspectos: las prestaciones (cercanía a la solución óptima) y la eficiencia de la implementación realizada. Para evaluar ambos aspectos la profesora podrá realizar competiciones entre los distintos grupos de prácticas.

1.1. ¿Qué hay que hacer?

1. Diseñar e implementar una estructura de datos razonable que almacene los datos con los mínimos atributos necesarios para cada producto (nombre, unidades, precio, ...). El almacenamiento de los distintos productos debe hacerse con una estructura de datos adecuada, por ejemplo una tabla hash, existiendo un índice único para cada producto.
2. Identificar el problema propuesto como un problema de grafos, en concreto el problema de encontrar el corte mínimo ("minimum cut" o `min cut`) que consiste en partir los vértices

de un grafo en dos conjuntos disjuntos con un número de aristas mínimo entre los dos trozos.

3. Implementar un algoritmo que realice una partición cercana a la óptima de los productos entre *amazon* y *amazonymas*. Dicho algoritmo puede ser aleatorio (se aconseja el algoritmo de Karger de cálculo de `min cut`).

4. Probar el algoritmo implementado con un número razonable de datos y con distintos generadores aleatorios.

5. *Opcional.* Implementar el algoritmo de Karger-Stein, que es una mejora del de Karger.

6. Suponer que se amplía la información disponible, ahora contamos con una matriz de enteros $B[i, j]$ de forma que $B[i, j]$ es el número de veces que se han comprado juntos los productos indexados por i y j. Implementar un algoritmo que realice una partición cercana a la óptima de los productos entre *amazon* y *amazonymas*, de forma que se minimice la suma de los $B[i, j]$ para los pares i, j en los que i y j se asignan a distinto proveedor. Se puede probar una generalización del algoritmo de Karger en el que la probabilidad de elegir la conexión $[i, j]$ es **directamente proporcional** a $B[i, j]$.

7. Opcional. Justificar mínimamente las prestaciones alcanzadas con el algoritmo implementado en 6.

1.2. Entrega

Deberá entregarse **hasta el 7 de enero de 2018**.