Trabajo Práctico Especial
Estructura de Datos y Algoritmos
Primer Cuatrimestre 2013

1. Objetivos.

Implementar una clase para el manejo de **Hiper Grafos o "Grafos de tareas"**, donde se especifican las tareas necesarias para lograr un objetivo y la relación entre ellas.
Implementar algoritmos para encontrar el camino mínimo entre el origen y el destino de un Hiper Grafo con peso en los ejes (solución exacta y aproximada).

2. Descripción del problema.

Definición:

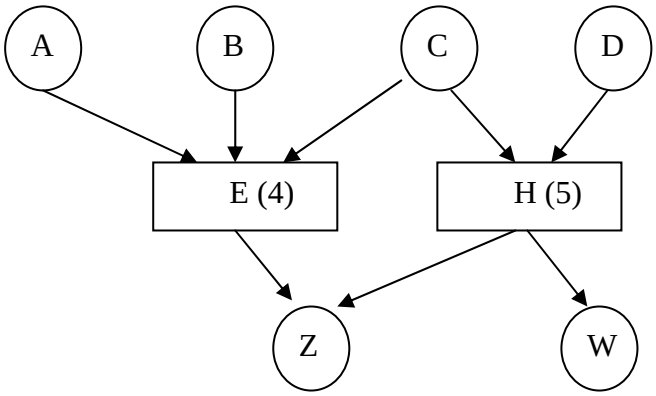
Un **hipergrafo** H es una tupla <V, E, w> donde V es un conjunto finito de nodos, E es un conjunto de hiperarcos y w es una función que le asigna a cada hiperarco un vector de pesos numéricos, a efectos de este TPE podemos asumir que cada hiperarco tiene un único peso numérico no negativo.

Un hipergrafo no tiene ciclos.

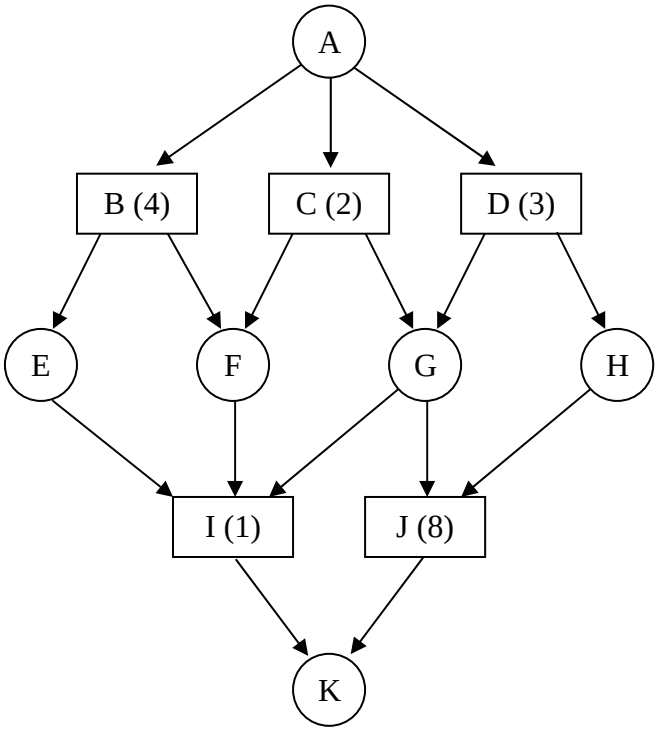
Un **hiperarco** es un par e = <T(e), h(e)>, donde T(e) $\subset$ V es la cola del hiperarco, h(e) $\subset$ V-T(e), es llamado la cabeza del hiperarco.

Cada nodo representa un “asset”. Un “asset” puede representar cualquier tarea que sea necesaria; es conocimiento que ya se obtuvo, relativo a un objeto real o propiedad de la realidad que se está modelando.

Ejemplo de hiperarcos, donde se puede llegar al nodo Z a través del eje E con peso 4 luego de haber visitado los nodos A, B y C, o bien se puede llegar a través del eje H con peso 5 luego de haber visitado los nodos C y D. Usando el eje H además de llegar al nodo Z se llega al nodo W.



A continuación se muestra un ejemplo de un hipergrafo, con los pesos indicados en cada hiper arco. El nodo de origen es “A” y el nodo destino “K”. El camino mínimo en este caso está conformado por los hiper arcos {B, C, I}, y tiene peso 7.



3. Descripción funcional.

El programa recibirá por línea de comandos el nombre de un archivo con extensión “hg” que contiene la información sobre el hipergrafo a analizar, y el modo en el que se ejecutará, dependiendo si se quiere encontrar la solución exacta o una solución aproximada:

- Para encontrar la solución exacta, el primer parámetro es el archivo con la descripción del grafo, y el segundo es el texto “exact”. Ejemplo:
  `java –jar tpe.jar file.hg exact`

- Para encontrar la solución aproximada, el primer parámetro es el archivo con la descripción del grafo, el segundo es el texto “approx”, y el tercero la cantidad máxima de segundos para que el algoritmo se ejecute antes de retornar una solución. El algoritmo a implementar debe ser de tipo local search. Ejemplo de uso:
  `java –jar tpe.jar file.hg approx 60`

En cualquiera de los dos casos, el programa debe imprimir por salida estándar el peso del camino mínimo encontrado, y además debe generar los siguientes archivos:

- con mismo nombre más “.min” y extensión “hg”: el subgrafo que representa el camino mínimo, con el mismo formato que el archivo de entrada
- con mismo nombre y extensión “dot”, representación del grafo en formato DOT para poder ser graficado usando Graphviz.
- con mismo nombre más “.min” y extensión “dot”: una representación del camino mínimo. Debe contener los mismos nodos y ejes que el hipergrafo pero resaltando en forma visual el camino mínimo

Los grafos que puede procesar el programa deben definirse en un archivo con extensión ".hg". Son archivos de texto plano que contiene dos tipos de líneas, las que comienzan con el caracter '#' son líneas de comentarios, las otras describe hiperejes.

Una línea que describe un hipereje tiene la siguiente estructura:

<nombre hipereje> <peso hipereje> <cantidad nodos cabeza> <nombre nodo 1> ... <nombre nodo N> <cantidad nodos cola> <nombre nodo 1> ... <nombre nodo M>

Donde el nombre es una secuencia alfanumérica de hasta 10 caracteres y el peso es un número entero positivo.

No es necesario definir los nodos previamente, a medida que aparecen nombrados en un hipereje se van insertando en el grafo.

La primera línea del archivo contiene el nodo de inicio y la segunda línea el nodo destino, a partir de la tercera línea contiene comentarios o información sobre los hiperejes.

4. Material a entregar

Cada grupo deberá dejar en el repositorio el siguiente material

- Códigos fuentes. Se debe incluir un *buildfile* de Ant que permita generar el archivo *jar* de la aplicación.
- Archivos de hipergrafos usados para las pruebas
- Un documento, en formato pdf, que explique en forma clara:
  - Las estructuras creadas para soportar el hipergrafo
  - Algoritmos creados para encontrar el camino mínimo (exacto y aproximado)
  - Problemas encontrados durante el desarrollo y decisiones tomadas
  - Tablas de comparación de tiempos
  - Cálculo de complejidad del camino mínimo exacto

5. Fecha de entrega

La entrega se hará mediante el repositorio antes del viernes 7 de junio a las 18 hs.

6. Criterios de Evaluación y Calificación

Para la evaluación y calificación del trabajo especial se considerarán:

- el correcto funcionamiento
- el cumplimiento de las reglas de estilo de programación dadas en clase
- la claridad del código
- la presentación del trabajo en todos los aspectos (documentación, facilidad de uso, etc.)

7. Consultas

Cualquier consulta que se desee realizar sobre el enunciado o la implementación del trabajo especial deberá hacerse en el horario propio de consulta de la materia o por mail a la cuenta de la cátedra.