Universidad Simón Bolívar
Departamento de Computación
y Tecnología de la Información
CI-2693: Laboratorio de
Algoritmos III

Prof. Fernando Torre Mora

Examen 2 Septiembre – Diciembre 2018

Un Problema de Planificación

1 Introducción

Una constructora ha dividido su más reciente proyecto en un conjunto de tareas: $x_1, x_2, \dots, x_n$ y necesitan ser planificadas. Cada tarea toma un día en completar. Su misión es planificar las tareas para que se puedan terminar en un número mínimo de días. Hay dos tipos de restricciones: *Restricciones de Conflicto* y *Restricciones de Precedencia*.

- Las **restricciones de conflicto** indican que algunas tareas no se pueden realizar el mismo día. (Por ejemplo, la tarea x_i y la tarea x_j pueden requerir usar la misma máquina, por lo que deben efectuarse en días diferentes)
- Las **restricciones de precedencia** indican que, para algunos pares dados de tareas, una necesita ser completada antes de que la otra pueda iniciar. Por ejemplo, la tarea x_i no se puede iniciar ante de que la tarea x_j se complete.

La planificación debe satisfacer todas las restricciones.

2 Descripción del programa

Para grabar las restricciones, constrúyase un grafo G cuyos vértices son las tareas $x_1, x_2, \dots, x_n$. Conecte x_i y x_j con un lado no dirigido si x_i y x_j no se pueden realizar el mismo día. Conecte x_i y x_j por un lado dirigido de x_i a x_j si x_i necesita ser completada antes de que x_j inicie.

Si el grafo es complicado, el problema de planificación es muy difícil. Ahora supongamos que para nuestro problema, las restricciones no son muy complicadas: los grafos G que necesitamos considerar siempre son árboles (luego de omitir las direcciones de los lados). Su misión es averiguar el número de días requeridos en una programación óptima para tales entradas. Puede usar el siguiente resultado:

> *Si G es un árbol, entonces el número de días requeridos es, o bien,* k *ó* $k+1$ *donde* k *es el número máximo de vértices contenido en un camino directo de G. es decir, un camino* $C = (x_1, x_2, \dots, x_k)$ *donde, para cada* $i = 1,2, \dots k-1$ *hay un lado dirigido de* x_i *a* x_{i+1}.

La Figura 1 a la derecha es uno de tales ejemplos. Hay seis tareas: 1,2,3,4,5,6. A partir de esta figura, sabemos que la tarea 1 y la tarea 2 deben realizarse en fechas distintas. La tarea 1 necesita ser realizada antes de la tarea 3, la tarea 3 antes que la tarea 5, la tarea 2 antes que la tarea 4 y la tarea 4 antes que la tarea 6. Es fácil verificar que el número mínimo de días para terminar todos los trabajos es 4 días. En este ejemplo, el número máximo k de vértices contenidos en un camino directo es 3.

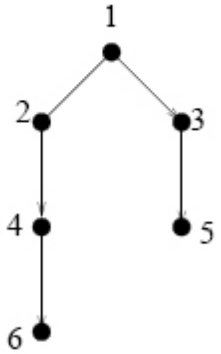

Figura 1

2.1 Requerimientos del programa

El programa se debe poder ejecutar desde la cónsola con el siguiente comando:

```
java Planificacion <archivo>
```

donde <archivo> es el archivo de casos de prueba. El archivo podrá contener varios casos de prueba.

2.2 Formato del Archivo de entrada

El archivo consiste de un número de árboles $A_1, A_2, \ldots A_m$, donde $m \leq 20$, cuyos lados pueden ser dirigidos o no dirigidos. Cada árbol tiene a lo sumo 200 vértices. Representamos cada árbol como un árbol con raíz (por conveniencia de presentación, la raíz es un vértice escogido al azar).

La información de los árboles está contenida en un número de líneas. Cada línea comienza con un vértice (el cual es un entero positivo) seguido de todos sus hijos (los cuales también son enteros positivos), luego seguido de un 0. Nótese que 0 no es un vértice, y que indica el fin de esa línea.

Algunos lados son dirigidos. La dirección de un lado puede ser de padre a hijo, pero también puede ser de hijo a padre. Si el lado es de padre a hijo, se coloca la letra 'd' luego del hijo (indicando una conexión de tipo "downward"). Si el lado es de hijo a padre, entonces colocamos la letra 'u' luego del hijo (indicando una conexión de tipo "upward"). Si el lado es no dirigido entonces no colocamos ninguna letra luego del hijo.

Los vértices consecutivos (números o números seguidos de letras) en una línea se separan con un solo espacio. Una línea que contenga solo un '0' indica el final de ese árbol. El siguiente árbol comienza en la siguiente línea. Dos líneas consecutivas de solo un '0' significa el final del archivo.

2.2.1 Ejemplo de Archivo

El primer caso de la entrada de ejemplo a continuación es el ejemplo de la Figura 1.

```
1 2 3d 0
2 4d 0
3 5d 0
4 6d 0
0
1 2d 3u 4 0
0
1 2d 3 0
2 4d 5d 10 0
3 6d 7d 11 0
6 8d 9 12 0
0
1 2 3 4 0
2 5d 0
3 6d 0
4 7d 0
5 8d 0
6 9d 0
7 10d 0
0
0
```

2.3 Formato de Salida

La salida contiene una línea por cada caso de prueba. Cada línea contiene un número, el cual es el número mínimo de días para terminar todas las tareas en ese caso de prueba.

2.3.1 Ejemplo de Salida

```
4
3
4
3
```

3 Requerimientos de la entrega

La entrega se realizará a través del Moodle antes de finalizar la hora de laboratorio

4 Evaluación

El examen tiene una ponderación de 10 puntos. Se asignarán

- 3 puntos por código
    - 1 punto por modificar un algoritmo de ordenamiento de grafos de precedencia para considerar ambas restricciones
    - 1 punto por su método de lectura del archivo
    - 1 punto por su programa principal
- 7 puntos por ejecución
    - 4 puntos por los casos del ejemplo dado
    - 3 puntos por poder hacerlo con otros casos

El programa debe correr sin errores.