Entrega Ejercicios Algoritmos Voraces

Iñigo Biedma Ramos

Ejercicio 1 Tenemos n peticiones de uso de un laboratorio, cada uno con un tiempo de comienzo si y un tiempo de finalización ti para cada petición i. Asumimos que todos los tiempos de comienzo y final son diferentes. Dos peticiones entran en conflicto si se solapan (es decir si el tiempo de comienzo de una petición es anterior al tiempo de finalización de otra). Nuestro objetivo es seleccionar un mayor número de peticiones que no contengan solapamientos. (Por ejemplo si tenemos estas tres peticiones [0, 3], [2, 5], [4 7] entonces seleccionaremos la primera y la tercera). Diseña un algoritmo voraz que calcule la solución óptima.

Necesitamos pensar en la acción de selección del algoritmo voraz que elaboraremos. Como la elección de una petición depende de si se solapa o no con el resto, y no queremos revisar todas cada vez, las ordenaremos por tiempo de comienzo. La última tarea la vamos a tomar siempre, ya que no condiciona al resto. Después escogeremos el resto teniendo en cuenta que no se solapen, de la que mayor tiempo de comienzo a la de menor.
El pseudocódigo para la selección quedará de la siguiente manera:

```
tipo
    matriz = tabla[1..N,1..2];
ftipo
accion seleccion(ent tord:matriz, ent n, sal sel:matriz, sal nsel);
    var
        i:entero;
    fvar
    sel[1] := tord[n];
    nsel := 1;
    para i = n-2 hasta 1 hacer
        si tord[i][2] < sel[nsel][1] -->
            nsel := nsel + 1;
            sel[nsel] := tord[i]
        [] tord[i][2] > sel[nsel][1] --> // por el enunciado no pueden ser iguales
            continuar;
        fsi
    fpara
faccion
```

Se puede ver que la eficiencia de la selección es O(n). Como antes de ello realizamos la ordenación (merge/quick/heap/intro sort) de coste $O(nlog_2n)$, el coste total será $O(nlog_2n)$.
He codificado el algoritmo en C. Adjunto el código, que trabaja con una tabla de 62 tareas (generadas al azar con numeros positivos todos distintos). Se utiliza qsort (stdlib) para ordenar la tabla. Ejemplo de ejecución del programa:

```
La tabla original:
{132,152} {57,150} {23,52} {15,124} {16,179} {161,187} {19,119}
    {26,39} {50,198} {10,75} {88,200} {96,166} {90,109} {14,138}
    {17,101} {46,63} {68,92} {73,87} {99,193} {44,158} {3,37} {48,157}
     {125,128} {29,135} {49,185} {25,108} {131,194} {66,70} {100,159}
    {165,199} {4,103} {51,123} {81,141} {79,111} {8,147} {13,60}
    {20,148} {12,30} {35,91} {94,98} {105,114} {34,47} {62,129}
    {133,163} {1,45} {153,170} {149,183} {107,118} {69,196} {65,86}
    {38,169} {9,192} {11,130} {78,164} {95,156} {84,155} {67,175}
    {115,145} {178,184} {162,188} {72,122} {31,120}
Tras ordenar:
{1,45} {3,37} {4,103} {8,147} {9,192} {10,75} {11,130} {12,30}
    {13,60} {14,138} {15,124} {16,179} {17,101} {19,119} {20,148}
    {23,52} {25,108} {26,39} {29,135} {31,120} {34,47} {35,91}
    {38,169} {44,158} {46,63} {48,157} {49,185} {50,198} {51,123}
    {57,150} {62,129} {65,86} {66,70} {67,175} {68,92} {69,196}
    {72,122} {73,87} {78,164} {79,111} {81,141} {84,155} {88,200}
    {90,109} {94,98} {95,156} {96,166} {99,193} {100,159} {105,114}
    {107,118} {115,145} {125,128} {131,194} {132,152} {133,163}
    {149,183} {153,170} {161,187} {162,188} {165,199} {178,184}
Se han elegido 10 tareas:
{26,39} {46,63} {66,70} {73,87} {94,98} {107,118} {125,128} {132,152}
     {153,170} {178,184}
```

Ejercicio 3 Consideramos un grafo no dirigido G=(V,A), donde cada arco a in A tiene un coste c_a. Suponemos que todos los arcos tienen un coste positivo y diferente entre ellos. Sea T el árbol recubridor mínimo de G y P el camino más corto entre vértice s al vértice t. Ahora vamos a suponer que incrementamos el coste de cada arco en 1 unidad, es decir $c_a + 1$. Llamamos a este nuevo grafo G'. ¿Cuál de las siguientes es cierta?

a. T es siempre el árbol recubridor mínimo y P es siempre el camino más corto entre s y t.

b. T puede que no sea el árbol recubridor mínimo pero P si es el camino más corto entre s y t.

c. T es el árbol recubridor mínimo y P puede que no sea el camino más corto entre s y t.

d. T puede que no sea el árbol recubridor mínimo y P puede que no sea el camino más corto entre s y t.

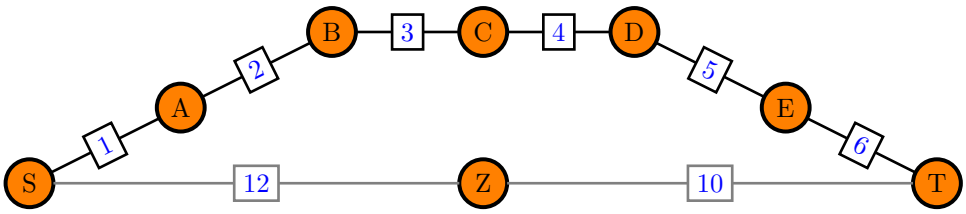
Debemos ver:

- **T sigue siendo árbol recubridor mínimo del nuevo grafo G'**

  Al tener todos los arcos un coste diferente, sólo existirá un único árbol recubridor mínimo.
  Demostracion: Supongamos que tenemos dos ARM: T y T*. Si son diferentes, difieren en algún arco. Tomemos el arco a_1, que tiene el menor peso de aquellos en los que difieren. Asumimos que pertenece a T pero no a T*. Como T* es ARM, $T \cup \{a_1\}$ formará un ciclo. En ese ciclo habrá otro arco a_2 con peso mayor que a_1. Cambiando a_1 y a_2 en B obtenemos otro ARM, contradiciendo la suposición inicial.
  Un árbol con n vértices tendrá n-1 aristas. Al aumentar en 1 los pesos de todas las aristas, entonces el coste de los posibles árboles generadores de un grafo aumentará en n-1. Como todos los árboles generadores aumentan su peso en la misma cantidad, el que de ellos sea mínimo lo seguirá siendo.
  Por lo tanto en nuestro caso T seguirá siendo árbol recubridor (o generador) mínimo de G'.

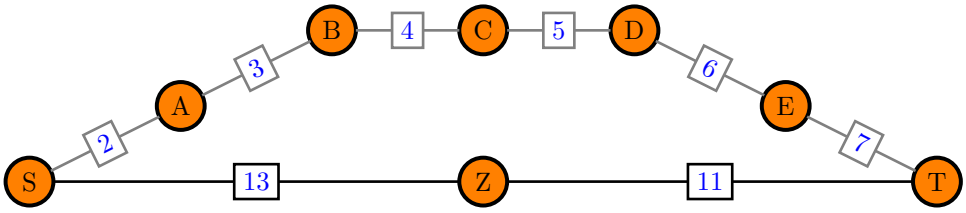
- **P sigue siendo el camino más corto entre s y t**

  Podemos encontrar un contraejemplo. El siguiente grafo:



El camino superior tiene longitud 21, mientras que el inferior 22.

Entre S y T el camino más corto es {S,A,B,C,D,E,T}, pero al aumentar el peso de cada arista en 1 deja de serlo.



El camino superior tiene longitud 27, mientras que el inferior 24.

El camino más corto es ahora {S,Z,T}, por lo que P puede que no sea el camino más corto.

La única respuesta correcta es la c: T es el árbol recubridor mínimo y P puede que no sea el camino más corto entre s y t.