Ma47a Optimización Combinatorial	**7 de Julio, 2006**

Pauta Examen

Prof. Cátedra: M. Kiwi — *Prof. Auxiliar: G. Sanchez*

Problema 1: Ver apuntes de cátedra.

Problema 2: Primero construiremos una red $G = (V, E)$. El conjunto de nodos V corresponde a:

- Un par de nodos fuente y destino, denotados s y t respectivamente.
- Un nodo por cada una de las tareas $T_1, \ldots, T_n$ y que denotaremos por $v_1, \ldots, v_n$ respectivamente.
- Un nodo por cada uno de los intervalos de tiempo $I_1, \ldots, I_m$ y que denotaremos por $w_1, \ldots, w_m$ respectivamente.

El conjunto de arcos E consiste en

- Para todo $j \in \{1, \ldots, n\}$, arcos sv_i de capacidad p_i.
- Para todo i y j tales que la tarea T_i pueda ser procesada en el intervalo I_j (o sea, tal que $r_i \leq t_j < t_{j+1} \leq d_i$), arcos v_iw_j de capacidad el largo del intervalo I_j, i.e. de capacidad $t_{j+1} - t_j + 1$.
- Para todo $j \in \{1, \ldots, m\}$, arcos w_jt de capacidad igual a k veces el largo del intervalo I_j, i.e. de capacidad $k(t_{j+1} - t_j + 1)$.

Claramente, el flujo máximo en G no puede exceder $\sum_{i=1}^{n} p_i$. Afirmamos que existe un flujo máximo integral que alcanza dicho valor si y sólo si existe un calendario factible de ejecución de las tareas. Una vez establecida la afirmación, tendremos que para calcular el calendario factible de ejecución de las tareas sólo basta resolver un problema de flujo máximo, algo que sabemos se puede hacer eficientemente.

Supongamos entonces que tenemos una asignación factible de tareas a máquinas y definamos el (s, t)-flujo x por:

- $x_{sv_i} = p_i$.
- $x_{v_iw_j}$ igual al tiempo de procesamiento total de la tarea T_i durante el intervalo I_j (independiente del número de máquinas que hayan procesado la tarea).
- $x_{w_jt} = \sum_{i:v_iw_j \in E} x_{v_iw_j}$.

Claramente, x es integral. Como la asignación de tareas a máquinas es factible, se conserva el flujo en $v_1, \ldots, v_n$. Además, por definición de x se tiene conservación de flujo en $w_1, \ldots, w_m$. Hemos concluido que x es flujo. Para verificar que x es factible notar que el tiempo de procesamiento de la tarea T_i durante el intervalo I_j es a lo más $t_{j+1} - t_j + 1$, por lo que $x_{v_i w_j} \leq (t_{j+1} - t_j + 1) = u(v_i w_j)$. Además, el trabajo total realizado por las k máquinas en el intervalo de tiempo I_j es a lo más $k(t_{j+1} - t_j + 1)$, por lo que $x_{w_j t} \leq k(t_{j+1} - t_j + 1) = u(w_j t)$. Hemos concluido que x es factible. Claramente, el valor del (s,t)-flujo x es $\sum_{i=1}^n p_i$.

Supongamos ahora que x es un (s,t)-flujo factible integral de valor $\sum_{i=1}^n p_i$. Por integralidad de la función capacidad podemos asumir que x es integral. Definimos la asignación de tareas a máquinas como sigue. La tarea T_i será procesada en alguna máquina durante el intervalo I_j durante un lapso igual a $x_{v_i w_j}$. Como el valor de x es $\sum_{i=1}^n p_i$ y la capacidad del corte $\delta(s)$ es también $\sum_{i=1}^n p_i$, se debe tener que todos los arcos sv_i están saturados, i.e. $x_{sv_i} = p_i$. Por conservación de flujo en v_i sigue que el tiempo de procesamiento asignado a la tarea T_i será en total p_i. Por otro lado, la cantidad total de trabajo realizado en el intervalo de tiempo I_j es

$$\sum_{i:v_i w_j \in E} x_{v_i w_j} = x_{w_j t} \leq u(w_j t) \leq k(t_{j+1} - t_j + 1)\,.$$

Debemos garantizar que dicho trabajo pueda ser realizado por las k máquina de manera que ninguna tarea requiera ser realizada simltáneamente en dos máquinas. En efecto, esto es posible dado que el tiempo de procesamiento requerido por la tarea T_i durante el intervalo I_j es $x_{v_i w_j} \leq (t_{j+1} - t_j + 1)$. Una asignación glotona de máquinas a tareas hace que ninguna tarea requiera ser realizada en dos máquinas de manera simultánea (detalles omitidos).

Problema 3:

(i).- Al comienzo de cualquier iteración, digamos la k-ésima, del Algoritmo Set Cover Glotón los elementos que no están en C pueden ser recubiertos a un costo de OPT (de hecho todo Ω puede ser recubierto a un costo de OPT). En la solución óptima debe existir algún conjunto S tal que

$$\frac{c(S)}{|S \setminus C|} \leq \frac{OPT}{|\Omega \setminus C|}\,,$$

porque de lo contrario, para cubrir $\Omega \setminus C$ se requeriría incurrir en un costo mayor que OPT, una contradicción. Sigue que el conjunto S' elegido en la k-ésima iteración es tal que

$$\frac{c(S')}{|S' \setminus C|} \leq \frac{OPT}{|\Omega \setminus C|}\,.$$

Obviamente, al comienzo de la iteración en que ω_k entró a C, éste último conjunto debió tener menos de k elementos. Sigue que

$$precio(\omega_k) = \min\left\{\frac{c(S)}{|S \setminus C|} \,:\, S \in \mathcal{S}\right\} \leq \frac{OPT}{|\Omega \setminus C|} \leq \frac{OPT}{n-k+1}\,.$$

(ii).- Basta observar que si $\mathcal{C}$ es la colección que retorna el Algortimo Set Cover Glotón, entonces

$$c(\mathcal{C}) = \sum_{\omega \in \Omega} precio(\omega) \leq \sum_{k=1}^{n} \frac{OPT}{n-k+1} = OPT \cdot H_n\,.$$

Problema 4:

(i).- Claramente $G \setminus \emptyset$ tiene el mismo número de componentes conexas que G. Luego, $\emptyset \in \mathcal{I}$, i.e. se cumple (M0).

Veamos ahora que se tiene (M1). Para ello, notar que al remover arcos de G el número de componentes conexas se mantiene o aumenta, pero nunca disminuye. Sea entonces $J \in \mathcal{I}$ y $J' \subseteq J$. Como el número de componentes conexas de $G \setminus J$ es al menos el número de componentes conexas de $G \setminus J'$, sigue que $G \setminus J'$ tiene a lo más tantas componentes como G. Ciertamente no puede tener menos. Concluimos que el número de componentes conexas de $G \setminus J'$ y G son iguales, i.e. $J' \in \mathcal{I}$.

Finalmente, probaremos que se cumple (M2). Sea $A \subseteq S$ y $J \subseteq A$ tal que $J \in \mathcal{I}$ es maximal. Necesariamente, $A \setminus J$ es un bosque maximal del grafo $G' = (V, A)$. Luego, si los conjuntos de nodos de las componentes conexas de G' son $V_1, \ldots, V_k$, entonces

$$|A \setminus J| = \sum_{i=1}^{k} (|V_i| - 1) \ .$$

Como la expresión de la derecha no depende de J, sigue que se tiene (M2).

(ii).- Consideremos la entrada del Algoritmo de Kruskal dada por $G = (V, E)$ grafo y $\omega : E \to \mathbb{R}$ función de costo sobre los arcos. El conjunto S corresponderá al conjunto de arcos E del grafo y la función de costo $c : S \to \mathbb{Q}$ será $c(e) = \omega(e)$. La colección de conjuntos independiente corresponderá a los J tales que J es un bosque de G.

Para ver que $M = (S, \mathcal{I})$ es un matroide, notar que $\emptyset$ es claramente un bosque, i.e. se cumple (M0). Además, si $J \in \mathcal{I}$ es un bosque, en particular no posee circuitos. Inmediatamente se tiene que $J' \subseteq J$ no posee circuitos y por lo tanto también es un bosque, i.e. se cumple (M1). Finalmente, consideremos $A \subseteq S = E$ y $J \in \mathcal{I}$ maximal y tal que $J \subseteq A$. Sigue que J es un bosque del grafo $G' = (V, A)$. Luego, al igual que en la parte (i) se puede concluir que $|J|$ no depende de J, por lo que se cumple (M2).