DEPARTAMENTO DE CIENCIAS DE LA COMPUTACIÓN
UNIVERSIDAD DE CHILE

Tarea 2: Planificación de Trabajos

CC4102 - Diseño y Análisis de Algoritmos

Profesor:	Gonzalo Navarro
Auxiliar:	Teresa Bracamonte
Alumnos:	Cristián Carreño
	Sergio Maass
Fecha:	17 de Noviembre de 2013

1. Introducción

Los algoritmos online cumplen un rol muy importante al permitir resolver problemas a medida que su entrada se va recibiendo, o sea, sin conocer el total de los datos previamente. Pero además, son muy útiles para aproximar soluciones a problemas difíciles en tiempo razonablemente corto.

Diversos problemas de optimización pertenecientes a la clase NP admiten soluciones aproximadas mediante algoritmos online p-competitivos. Un algoritmo de estas características permite resolver instancias arbitrarias del problema en tiempo polinomial, garantizando que la solución estará dentro de un radio p del óptimo. Esto es muy importante, ya que si $P \neq NP$ -lo que parece ser el caso- no es posible encontrar una solución óptima en tiempo polinomial.

En este informe se presentarán dos versiones del problema de planificación de trabajos ($NP - completo$[1]) y se diseñará un algoritmo online 2-competitivo para cada una. Se demostrará formalmente la competitividad de cada algoritmo y luego se comprobará experimentalmente.

2. Trabajos de una etapa

El problema consiste en un conjunto de n trabajos que deben asignarse a m máquinas idénticas. Cada trabajo j requiere una cantidad de tiempo T_j para ser finalizado. Además, cada trabajo j sólo puede ser asignado a una máquina A_j, y debe ejecutarse por completo y sin interrupciones en esta.

Sea A una asignación de trabajos a máquinas, y sea el *makespan* definido por: $makespan(A) = max_i \sum_{A_j=i} T_j$. El objetivo del problema es determinar la asignación A que minimiza el *makespan*.

2.1. Algoritmo 2-competitivo propuesto

El algoritmo consiste simplemente en asignar cada trabajo j a la máquina que posea la menor carga. En particular, para cada j:

1. Obtener la máquina M_i de menor carga L_i.
2. Asignar j a M_i
3. $L_i := L_i + T_j$

La complejidad del algoritmo depende del paso 1. Usando un *heap*, o un algoritmo como quicksort, podemos obtener la máquina de menor carga en tiempo $O(\log m)$. Por lo tanto, para n trabajos tenemos $O(n \log m)$.

[1] Garey, M.R. (1976). "The Complexity of Flowshop and Jobshop Scheduling"

2.2. Demostración de competitividad

Sea L_i la carga de la máquina con mayor tiempo asignado, y sea j su último trabajo. La carga de la máquina antes de esa asignación era $L_i - T_j$. Debido a que el algoritmo asigna un trabajo a la máquina de menor carga tenemos que $L_i - T_j \leq L_k, \forall k \in [1, m]$.

Luego, sumando sobre todas las máquinas obtenemos

$$\begin{aligned} L_i - T_j &\leq \frac{1}{m}\sum_{k=1}^{m} L_k \\ &= \frac{1}{m}\sum_{k=1}^{n} T_k \\ &\leq L^* \end{aligned} \tag{1}$$

Donde L^* corresponde al *makespan* del caso óptimo.

Finalmente, considerando (1) y el hecho de que $T_j \leq L^*$, ya que el trabajo más largo igual debe ser realizado por alguna máquina en el caso óptimo, tenemos que

$$\begin{aligned} L_i &= L_i - T_j + T_j \\ &\leq L^* + T_j \\ &\leq 2L^* \end{aligned} \tag{2}$$

Con lo que se demuestra que el *makespan* obtenido por el algoritmo online está acotado por el doble del *makespan* en el caso óptimo. O sea, el algoritmo es 2-competitivo.

2.3. Comprobación empírica de competitividad

2.3.1. Implementación del algoritmo online

2.3.2. Implementación del algoritmo óptimo

2.3.3. Experimentos

2.3.4. Resultados

3. Trabajos con múltiples etapas

3.1. Algoritmo 2-competitivo propuesto

3.2. Demostración de competitividad

3.3. Comprobación empírica de competitividad

3.3.1. Implementación del algoritmo online

3.3.2. Implementación del algoritmo óptimo

3.3.3. Experimentos

3.3.4. Resultados

4. Conclusiones