Universidad Rey Juan Carlos

ETSII Escuela Técnica Superior de Ingeniería Informática

Algoritmos Avanzados

-Práctica 1-

Algoritmos Voraces

Curso 2022/2023
GII

Alberto Radlowski Nova
Pablo Rodea Piornedo

Algoritmo Idealizado y Análisis

```java
public static int tareasOrdenadas(int [] ts, int n){
   //Algoritmo voraz que resuelve el problema de las tareas ordenadas
   //ts: array de enteros con los tiempos de ejecución de las tareas
   //n: número de procesadores que resuelven las tareas
   //Devuelve el tiempo mínimo que tardan en ejecutarse todas las tareas

   int tiempoTotal = 0;
   int [] tiempoProcesadores = new int[n];

   for(int i = 0; i < ts.length; i++){
       //Asigna el tiempo por separado a los procesadores
       int aux = tiempoProcesadores[i%n];
       tiempoProcesadores[i%n] += ts[i];
       //Lo suma al tiempo total de ejecución
       tiempoTotal = tiempoTotal + tiempoProcesadores[i%n];
   }

   return tiempoTotal;
}
```

Análisis en Tiempo de ejecución:

$$\sum_{i=0}^{n-1} 1 = 1+n-1 = n \Rightarrow O(n)$$

Análisis en Espacio:

E(n) = 1+1+1+1+1 = 5 = θ(1)

- 1 + 1 de variables de entrada “ts” y “n”
- 1 + 1 de variables locales “tiempoTotal” y “tiempoProcesadores”
- 1 de variable local del bucle “i”

Algoritmo Realista y Análisis

```
public static int tareasSinOrdenar(int [] ts, int n){
   //Ordenación por Burbuja
   for(int i=0; i<ts.length; i++){
       for(int j=i+1; j<ts.length; j++){
           if(ts[i] > ts[j]){
               int aux = ts[i];
               ts[i] = ts[j];
               ts[j] = aux;
           }
       }
   }
   int tiempoTotal = 0;
   int [] tiempoProcesadores = new int[n];

   for(int i = 0; i < ts.length; i++){
       tiempoProcesadores[i%n] += ts[i];
       tiempoTotal = tiempoTotal + tiempoProcesadores[i%n];
   }
   return tiempoTotal;
}
```

Análisis en Tiempo de ejecución:

$$\sum_{i=0}^{n-1} \sum_{j=i+1}^{n-1} 1 = - \frac{n^2-n}{2} - n + n * n = \frac{n^2-n}{2} \Rightarrow O(n^2)$$

Análisis en Espacio:

$$E(n) = 1+1+1+1+1+1+1+1+1 = 8 = \theta(1)$$

- 1 + 1 de variables de entrada “ts” y “n”
- 1 + 1 + 1 de variables del for anidado “i” , “j” y “aux”
- 1 + 1 de variables locales “tiempoTotal” y “tiempoProcesadores”
- 1 de variable local del bucle “i”

Conclusion Personal

Para la realización del algoritmo hemos seguido la heurística de búsqueda típica del algoritmo voraz.

Hemos terminado con un algoritmo no muy complicado (solamente unas pocas líneas de código) con el cual dividimos la lista de tiempos de tareas en función del procesador al que pertenecen, esta es la parte que más nos ha costado, el planteamiento inicial del algoritmo.

Por último, a la hora de realizar el algoritmo realista simplemente hemos expandido el algoritmo idealizado añadiendo una ordenación por burbuja previa a la ejecución del algoritmo.