Algoritmos Voraces

Problema 5 - Problema del electricista

A. Herrera, A. Moya, I. Sevillano, J.L. Suarez

12 de mayo de 2015

Introducción

- En esta presentación se proporciona una solución para el ejercicio 5.
- El código, los resultados de las ejecuciones, las gráficas y los pdf asociados se puede encontrar en GitHub.

Explicación del problema

A. Herrera, A. Moya, I. Sevillano, J.L. Suarez

Resumen del enunciado

Un **electricista** necesita hacer **n reparaciones urgentes**, y sabe de antemano el tiempo que le va a llevar cada una de ellas: en la tarea i-ésima tardará t_i minutos.

Como en su empresa le pagan dependiendo de la satisfacción del cliente y esta es inversamente proporcional al tiempo que tardan en atenderles, necesita **decidir el orden** en el que atenderá las reparaciones para **minimizar el tiempo medio de espera** de los clientes (desde el inicio hasta que su reparación es efectuada).

Explicación del problema

Ejemplo:

Supongamos que tenemos 4 personas, Andrés, Antonio, Iván y Juanlu, y que el tiempo que se tarda en atender sus reparaciones es de 1, 4, 2, 4 segundos respectivamente.

Si el electricista los atiende por orden alfabético, los tiempos de espera son:

- Andrés: 1
- Antonio: $1+4 = 5$
- Iván: $1+4+2 = 7$
- Juanlu: $1+4+2+4 = 11$

El tiempo medio de espera es: $\frac{1+5+7+11}{4} = 6$

Explicación del problema

Desarrollamos:

- Un algoritmo voraz que obtiene el óptimo en $\theta(n \log n)$.
- Una modificación del algoritmo para cuando hay varios electricistas trabajando.

Explicación del problema

Dados los tiempos de las tareas $\{t_i : i = 1, \ldots, n\}$, buscamos una permutación $x = (i_1, \ldots, i_n)$ de los n primeros números naturales que minimice

$$f(x) = \frac{1}{n} \sum_{k=1}^{n} T_k \text{ donde } T_k = \sum_{j=1}^{k} t_{i_j}$$

T_k es el tiempo de espera del cliente i_k.

Algoritmo voraz óptimo

En cada momento, el electricista elige un trabajo de los que le quedan pendientes y lo realiza.

- **Estrategia voraz:** Elegir el más corto (pues si elegimos uno de mayor duración mucha gente tendrá que esperar a su finalización).

Algoritmo voraz óptimo

En cada iteración se elige el más corto no realizado:

```
# t es el vector con los tiempos de los trabajos
sol = []
for i in range(0, n):
    siguiente_trabajo = t.index(t.min())
    sol.append(siguiente_trabajo)
    t[siguiente_trabajo] = math.inf
```

Algoritmo voraz óptimo

Una mejor implementación es la siguiente:

1. Ordenar los tiempos $\{t_i : i = 1, \ldots, n\}$ de menor a mayor.
2. Tomar como solución la permutación que los ordena $(i_1, \ldots, i_n)$. Esto es equivalente a tomar como solución $(1, \ldots, n)$ para los tiempos ordenados.

Hemos conseguido una eficiencia $\theta(n \log n)$.

Algoritmo voraz óptimo

Proposición. El algoritmo voraz es óptimo.

Demostración. Ordenamos el vector con los tiempos de menor a mayor. La solución dada por el algoritmo para el vector de tiempos ordenado es $x = (1, \ldots, n)$. Veamos que cualquier otra permutación $x' = (i_1, \ldots, i_n)$ tiene mayor o igual tiempo medio de espera.

1. Sea j tal que $t_{i_j} > t_{i_{j+1}}$.

   - Si este índice no exíste, entonces el tiempo medio de x' es el mismo de x.
   - En caso de que exista, transponemos i_j con i_{j+1}.

Algoritmo voraz óptimo

Proposición. El algoritmo voraz es óptimo.

Esta nueva permutación x'' tiene menor tiempo medio de espera que x':

$$f(x') - f(x'') = \frac{1}{n}(t_{i_j} - t_{i_{j+1}}) > 0$$

2. Remetimos el proceso hasta que no exista j.

Por la transitividad del orden, la primera permutación x' tiene mayor tiempo medio de espera que la final, cuyo tiempo medio de espera es el de x.

Algoritmo voraz óptimo

Ejemplo (revisitado)

Supongamos que tenemos 4 personas, Andrés, Antonio, Iván y Juanlu, y que el tiempo que se tarda en atender sus reparaciones es de 1, 4, 2, 4 segundos respectivamente.

Si el electricista los atiende usando el algoritmo voraz, el orden de atención es: Andrés, Iván, Antonio, Juanlu. Los tiempos de espera son:

- Andrés: 1
- Iván: $1+2 = 3$
- Antonio: $1+2+4 = 7$
- Juanlu: $1+2+4+4 = 11$

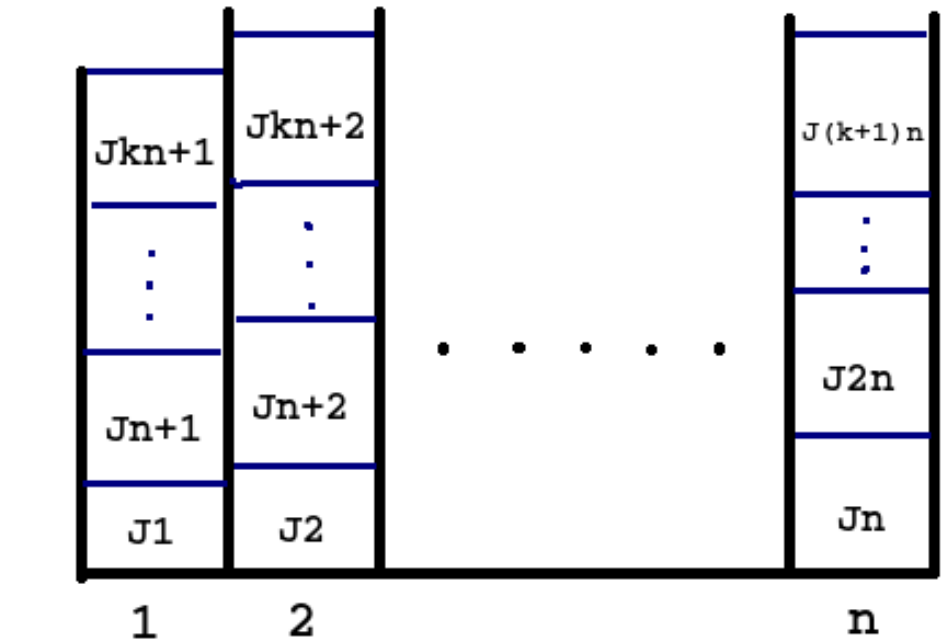
El tiempo medio de espera es: $\frac{1+3+7+11}{4} = 5.5$

Algoritmo óptimo para más de un electricista

- Mantenemos la idea anterior.
- Cada vez que un electricista termina un trabajo elige el siguiente por hacer que menos tiempo requiera.
- El orden en el que se efectuarán los trabajos es el mismo que en el apartado anterior. Cambia el tiempo medio de espera pues varios trabajos se ejecutan en paralelo.

Algoritmo óptimo para más de un electricista

A. Herrera, A. Moya, I. Sevillano, J.L. Suarez



Distribución de los trabajos entre los operarios

Fin de la presentación

¡Gracias por su atención!