.UBAfiuba
FACULTAD DE INGENIERÍA

Teoría de Algoritmos
(75.29) Curso Buchwald - Genender

Trabajo Práctico 0
Lineamientos sobre informes

30 de marzo de 2024

Zielonka Axel
110310

Petean Marina
110564

Romano Nicolas
110830

1. La Nacion del Fuego

1.1. Analisis del problema

1.2. Algoritmo Greedy propuesto

Aplicamos una regla sencilla que nos permita obtener el óptimo local a mi estado actual: Esto lo logramos ordenando nuestro arreglo de mayor a menor b/t, para que al recorrer el arreglo en cada iteracion podamos optener el óptimo local (que se corresponde al elemento actual)

Aplicamos iterativamente esa regla, esperando que nos lleve al óptimo general: Calculamos, iterando de manera ordenada el arreglo, la suma ponderada con los óptimos locales, que resulta ser la mínima y por lo tanto es el óptimo general.

1.3. Optimalidad

aca hablamos de la optimalidad del algoritmo y hacemos el analisis de si (y como) afecta la variabilidad de los valores de t_i y b_i al algoritmo planteado.

Definimos una inversión de dos elementos batalla[i] y batalla[j] dentro de un orden de batallas a toda $a_i < a_j$ tal que $i < j$. Sea $a_i = \frac{b_i}{t_i}$.

Supongamos que tenemos un orden de batallas tal que la suma ponderada $\sum_{i=0}^{n} f_i \cdot b_i$ sea la óptima (es decir, la mínima) y que ése órden posee inversiones como la anteriormente definida. Al revertir esas inversiones, es decir, reducir la cantidad de inversiones del orden de batalla, nosotros no podemos empeorar la suma ponderada final, solo podrá ser mejor (ver demostración). Entonces, al revertir todas las inversiones, nos queda un arreglo ordenado de mayor a menor por la relación $a = \frac{b}{t}$ que es más óptimo que el óptimo con inversiones que habíamos supuesto. Lo que nos lleva a que el orden óptimo con inversiones es un absurdo, y por lo tanto el orden óptimo es un arreglo sin inversiones.

¿Por qué al revertir la inversión no podemos empeorar la suma ponderada? Supongamos que tenemos solo dos batallas (t_0, b_0) y (t_1, b_1) ordenadas tal que $a_0 < a_1$. Queremos demostrar que $t_0 \cdot b_0 + b_1 \cdot (t_0 + t_1)$ es siempre mayor que $t_1 \cdot b_1 + b_0 \cdot (t_1 + t_0)$. En otras palabras, estamos poniendo a (t_1, b_1) como primera batalla y a (t_0, b_0) como segunda para demostrar que es mejor que el orden anterior. Para esto, nos basta demostrar con método directo que la inecuación cumple.

$$t_1 \cdot b_1 + b_0 \cdot (t_1 + t_0) < t_0 \cdot b_0 + b_1 \cdot (t_0 + t_1)$$

Aplicando distributiva

$$t_1 \cdot b_1 + b_0 \cdot t_1 + b_0 \cdot t_0 < t_0 \cdot b_0 + b_1 \cdot t_0 + b_1 \cdot t_1$$

Restando $t_1 \cdot b_1$ y $b_0 \cdot t_0$ de ambos lados

$$b_0 \cdot t_1 < b_1 \cdot t_0$$

$$\frac{b_0}{t_0} < \frac{b_1}{t_1}$$

Es decir: $a_0 < a_1$. Partimos de una hipótesis que supusimos como verdadera, lo que quiere decir que el que tenía mejor relación peso-duración era más óptimo que el otro.

¿Que pasa en el caso en que dos elementos i y j de la forma (ti, bi) y (tj, bj) respectivamente, tengan el mismo a? Ya demostramos que la solución óptima es con un arreglo ordenado de mayor a menor .ª", Planteamos ahora un caso genérico de dos elemenos consecutivos de nuestro arreglo: (supongo f al anterior a i y j)

Primer caso (ubicando i antes que j):

$$fi = f + ti$$

$$fj = fi + tj = f + tj + ti$$
$$sumaponderada(caso1) = ant + (f + ti) \cdot bi + (f + tj + ti) \cdot bj + sig$$

Segundo caso (ubicando j antes que i):

$$fj = f + tj$$

$$fi = fj + ti = f + tj + ti$$
$$Sumaponderada(caso2) = ant + (f + tj) \cdot bj + (f + tj + ti) \cdot bi + sig$$

Notar que el F del que ubico último en ambos casos es igual, y por lo tanto el orden de i y j no afecta a los siguientes F

Ahora planteo la igualdad de ambas sumas ponderadas, a ver si llego a un absurdo:

$$ant + (f + ti) \cdot bi + (f + tj + ti) \cdot bj + sig = ant + (f + tj) \cdot bj + (f + tj + ti) \cdot bi + sig$$

$$(f + ti) \cdot bi + (f + tj + ti) \cdot bj = (f + tj) \cdot bj + (f + tj + ti) \cdot bi$$
$$f \cdot bi + (ti \cdot bi) + f \cdot bj + tj \cdot bj + ti \cdot bj = f \cdot bj + tj \cdot bj + f \cdot bi + tj \cdot bi + ti \cdot bi$$

Cancelamos términos iguales:

$$ti \cdot bj = tj \cdot bi$$

$$\frac{b_j}{t_j} = \frac{b_i}{t_i}$$
$$a_j = a_i$$

Por lo tanto, vemos que se cumple que la suma ponderada es la misma al intercambiar dos elementos consecutivos, solamente si tienen el mismo a, por lo tanto nuestra solución es óptima también para este caso.

2. Minimizar la suma ponderada

En este trabajo de ejemplo realizaremos el análisis bla bla bla

2.1. Algoritmo

A continuación se muestra el código de solución del problema.

```
def minimizar_suma(batallas):
    batallas_indices = crear_arreglo_con_indice(batallas)
    batallas_ordenadas = sorted(batallas_indices, key=lambda batalla: -(batalla
    [0][1] / batalla[0][0]))
    f = []
    for i in range(len(batallas_ordenadas)):
        if i == 0:
            f.append(batallas_ordenadas[i][0][0])
        else:
            f.append(f[i - 1] + batallas_ordenadas[i][0][0])

    suma_ponderada = 0
    for i in range(len(batallas_ordenadas)):
        suma_ponderada += f[i] * batallas_ordenadas[i][0][1]

    return suma_ponderada, batallas_ordenadas
```

Ahora deberiamos explicar con palabras un poco que hacemos en codigo

2.2. Complejidad

aca hablamos de la complejidad del algoritmo y hacemos el analisis de si (y como) afecta la variabilidad de los valores de t_i y b_i a los tiempos del algoritmo planteado.

2.3. Mediciones

aca hablamos de las mediciones que hicimos y como se conectan con las complejidades que hablamos anteriormente. (graficos aca)

3. Testing

Aca hablamos un poco de los tests que hicimos, algunos casos borde que quizas generen duda y cualquier cosa que sea empirica.

4. Conclusiones

Acá irían las conclusiones de todo nuestro trabajo :)