

S T Q Q S S D

— / — /

Prava 2 - CANA

José Douglas Gondim Soares, 4853417

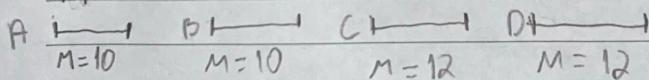
2) O primeiro passo é ordenar os tarefas em ordem decrescente de multa. A lógica é: Essas tarefas possuem uma prioridade de execução maior para diminuir a multa total.

Agora, precisamos iterar o vetor de tarefas e para cada tarefa T_i , queremos alocá-la (se possível) em algum dia entre 1 e PT (é a praça daquela tarefa). Vamos escolher a última horária disponível da última dia ainda disponível entre 1 e PT . (Podemos fazer isso usando um array de alocação de horas e procurando nela um espaço disponível de acordo com a condição de escolha mencionada). Ao final, retornaremos o array de horas.

Argumento: Esse algoritmo funciona porque sempre estamos tentando alocar uma tarefa que geraria uma multa maior caso atrasada, primeira. A reportagem está também na forma como a alocamos (na última horária disponível entre 1.. PT). Isso porque, dessa forma, ela vai atingir a mínima possível a execução de outras tarefas com multa menor que também não desejamos atingir.

Para exemplificar: Em uma situação que podemos alocar 2 tarefas por dia:

$$PT=1 \quad P+=1 \quad P+=2 \quad PT=2$$



Se escolhessermos alocar as tarefas no primeiro horário disponível, iríamos executar C e D e servirmos feriados a atender A e B. V que não acontece se alocarmos C e D no último horário disponível. Assim, conseguimos executar todos os tarefas.

CONTINUA...

spiral®

(2) CONTINUAÇÃO

O exemplo mostra a importância da ordem de alocação após a ordenação inicial do vetor.

Para concluir: A nossa escolha é a última porque se existisse alguma tarifa T' que tornasse a multa total maior e ela tivesse sido cadastrada, então o vetor não estaria ordenado em ordem crescente de multa em vez estariamos escolhendo a intimação horária ainda disponível para aloca-la. E podíamos então tirar alguma tarifa T_i da nossa seleção por T' .

(3-) a) No nosso problema inicial i) cortar um tronco de comprimento L em k partes, mas só podemos fazer um corte por vez. Logo, sempre queremos achar a melhor corte entre um ponto inicial e final da tronco. Percebemos que a cada corte, nós ficamos com 2 troncos que podem ou não serem cortados, e então acabamos com um problema semelhante ao original (Achar uma forma ideal de cortar esses 2 troncos). Isso é a propriedade de subestrutura ótima (dividindo o problema em instâncias menores). Para concluir, somamos o custo do corte inicial com o custo mínimo das 2 novas troncos e ao final teremos o custo mínimo para cortar o tronco.

$$b) C(n) = C(p - \text{início}) + C(\text{fim} - p) + (j\text{im} - \text{início})$$

Explicação: P é onde o corte ocorre, restando cortar os troncos idealmente na primeira parte ($p - \text{início}$) e na segunda parte ($\text{fim} - p$). O custo C para cortar troncos, início e fim não variável que mudam.

CL VERSO

c) int custoMinimo(int inicio, int fim, int p) {
 int c = mem[inicio][fim];
 Se (c != ∞) retorna c;
 int custo = custoMinimo(inicio, p) + custoMinimo(p, fim) +
 (fim - inicio);
 mem[inicio][fim] = custo;
 retorna custo;
}

1- a) Temos 2 caixas A e B (contra-exemplo)

A $\boxed{ } \begin{matrix} P=4 \\ k=100 \end{matrix}$ B $\boxed{ } \begin{matrix} P=2 \\ k=1 \end{matrix}$

Ordenando por peso, a caixa A ficaria encima da da caixa B, o que gera esforço $100 \cdot (4+2) = 600$.

Mas se colacarmos a caixa B encima da A, temos esforço $1 \cdot (4+2) = 6$, o esforço é menor e está mostrado no contra-exemplo.

b) O esforço é calculado pelo número de objetos da caixa mais superior multiplicado pela somatória dos pesos das caixas que estão embaladas, incluindo a caixa atual. logo, a única forma de diminuir o esforço é diminuir a constante do número de objetos da caixa mais superior. É como os caixas estão em ordem decrescente de K, e menor K está encima o esforço já é o mínimo.

Nbs: Eu estou achando essa questão muito extensa e não estou vendo a semelhança com o problema da mochila do jeito como foi explicado na aula.

spiral® CONTINUA a Obs...

CONTINUAÇÃO do Obs:

A minha interpretação é diferente, mas decidi fazer como foi explicado na aula. Eu acredite que o numero de objetos deveria influenciar no peso da caixa, de forma que a solução seria diferente. A não ser que o problema da mochila falada na sala seja a fracionária.

De qualquer forma, eu decidi fazer como foi explicado na aula de exercícios, onde o peso é a quantidade de objetos na caixa mais a parte negativa da somatória dos pesos das caixas inferiores, incluindo a caixa do topo.

