Programação Dinâmica

Questão 1. (CLRS) Exercícios: 15.2-1, 15.2-2, 15.2-3, 15.3-1, 15.3-2, 15.3-3, 15.3-4 (3ed), 15.3-5 (2ed), 15.4-1, 15.4-2, 15.4-4, 15.5-2,

Questão 2. (CLRS) Problemas: 15-3 (2ed), 15-5 (3ed), 15-4 (2ed), 15-6 (3ed),

Questão 3. Relembre o problema da mochila: Dado um inteiro K e n itens de pesos diferentes, $S = \{p_1, p_2, \ldots, p_n\}$, encontrar um subconjunto $S' \subseteq S$ cuja soma dos pesos é exatamente K, ou determine que tal conjunto não existe.

Na tabela seguinte, temos uma tabela de programação dinâmica para o problema da mochila parcialmente preenchida. O itens têm pesos $p_1 = 2$, $p_2 = 7$, $p_3 = 5$ e $p_4 = 2$. Preencha os dados que faltam para os itens p_3 e p_4. Obs.: O símbolo **O** significa existe solução para o subproblema da mochila correspondente ao item, sem usar o item. O símbolo **I** significa que existe solução para a respectiva mochila, usando o respectivo item. O símbolo **-** significa que não existe solução até o momento para a respectiva mochila.

Tam. Mochila $\rightarrow$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14
$p_1 = 2$	O	-	I	-	-	-	-	-	-	-	-	-	-	-	-
$p_2 = 7$	O	-	O	-	-	-	-	I	-	I	-	-	-	-	-
$p_3 = 5$															
$p_4 = 2$															

Questão 4. Melhore o uso de espaço do algoritmo Mochila, apresentado em sala. Há uma necessidade de usar uma matriz completa $n \times K$? Qual a complexidade de espaço do algoritmo melhorado?

Questão 5. O problema da mochila inteira é como a mochila binária, mas neste caso cada item pode ser repetido várias vezes, i.e., um item e pode não ser colocado na solução, ou ser colocado uma ou várias vezes. Resolva este problema por programação dinâmica.

Questão 6. Considere a seguinte versão bidimensional do problema da subsequência consecutiva máxima. Considere uma matriz M, de dimensões $m \times n$ com números inteiros (positivos ou negativos). O objetivo é encontrar uma submatriz N cuja soma dos elementos é máxima. Uma submatriz é exatamente isso que você está pensando.

Algoritmos Gulosos

Questão 7. (CLRS) Exercícios: 16.1-1, 16.1-2, 16.1-3, 16.1-4, 16.2-1, 16.2-2, 16.2-4, 16.2-5, 16.2-7, 16.3-1 (2ed), 16.3-2 (3ed), 16.3-2 (2ed), 16.3-3 (3ed), 16.3-3 (2ed), 16.3-4 (3ed), 16.3-7 (2ed), 16.3-8 (3ed),

Questão 8. (CLRS) Problemas: 16-1, 16-4a,

Questão 9. Uma caixa d-dimensional com lados $(x_1, \ldots, x_d)$ cabe numa caixa $(y_1, \ldots, y_d)$ se existe uma permutação π de $1, \ldots, d$ tal que

$$x_{\pi_1} < y_1, \ \cdots, \ x_{\pi_d} < y_d.$$

Dê um algoritmo eficiente para determinar se $(x_1, \ldots, x_d)$ cabe em $(y_1, \ldots, y_d)$. Prove que este algoritmo está correto.

[1] Esta lista deve ser feita logo após as aulas do conteúdo correspondente e serve para fixar o conteúdo, confirmar ou identificar as dúvidas. Anote suas dúvidas e procure atendimento! Os exercícios são referências ou transcrições de exercícios dos livros-textos (CLRS/Manber), ou foram gentilmente cedidos por outros professores, particularmente por Flávio Keidi Miyazawa (FKM), Cid Carvalho de Souza e Orlando Lee (CID/OL).

Questão 10. São dados n livros, $1, 2, \ldots, n$ com pesos $p_1, p_2, \ldots, p_n$, respectivamente. Os pesos satisfazem a condição $0 < p_i < 1$, para $i = 1, \ldots, n$. Deseja-se acondicionar os livros em um número mínimo de envelopes satisfazendo as condições abaixo:

1. Cada envelope contém no máximo dois livros.
2. Em nenhum envelope o peso dos livros ultrapassa 1. Descreva um algoritmo com *número de comparações* $O(n \log n)$ que acha um acondicionamento ótimo de n livros dados. Demonstre que o acondicionamento encontrado por seu algoritmo é ótimo, i.e., ele usa o menor número possível de envelopes.

Questão 11. Seja $N = 2^k$ e $S = \{x_1, \ldots, x_s\}$, onde x_i é potência de 2, $x_i \leq N$ e $\sum_{i=1}^{s} x_i \geq N$. Então existe um conjunto $S' \subseteq S$ tal que $\sum_{x' \in S'} x' = N$. Prove este resultado e apresente um algoritmo para encontrar tal S'.

Questão 12. Uma empresa de esquadrias metálicas precisa de n_i vigas de tamanho 2^i, $i = 0, \ldots, k$. Mas a metalúrgica que vende as vigas para a empresa de esquadrias, só vende vigas de tamanho M, M é um inteiro positivo e $2^i \leq M$, $i = 0, \ldots, k$. Assim, a empresa de esquadrias precisa comprar o menor número de vigas de tamanho M. Projete um algoritmo para resolver este problema de forma ótima, i.e., usando o menor número de vigas grandes. Prove que ele devolve a solução ótima. O valor dos n_i e M são dados. (Você pode usar o algoritmo da questão anterior como subrotina).