Soluções para problemas selecionados da apostila
Otimização Matemática e Pesquisa Operacional
de André R. Fioravanti e Matheus Souza

– Capítulo 7 –

Plínio Santini Dester (`p103806@dac.unicamp.br`)

18 de janeiro de 2022

*Em caso de dúvidas, sugestões ou correções, não hesite em mandar um e-mail.

7 Problemas

7.1. A Figura 7.14 mostra a localização dos principais terminais ferroviários nos Estados Unidos e as ferrovias existentes. O objetivo da autoridade de transporte ferroviário daquele país é decidir quais das ferrovias devem ser expandidas para ter um aumento de capacidade. Em particular, este programa de expansão deve revitalizar a ligação direta entre Los Angeles (LA) e Chicago (CH). Excluindo-se estes dois terminais, todos os outros terminais podem ser conectados entre si direta ou indiretamente de modo que o comprimento total das ferrovias revitalizadas seja minimizado. Quais trechos devem ser revitalizados? Qual é a distância total que deve ser expandida?

> **Solução:** Transforme os nós LA e CH em um único nó e aplique o algoritmo de Prim (por exemplo) normalmente. No fim, adicione a ligação entre LA e CH.

7.3. Suponha que $G = (V, E)$ seja um grafo não-direcionado com custos associados às arestas, que podem ser positivos ou negativos. Os algoritmos de Prim e de Kruskal ainda produzem a árvore geradora mínima para estes grafos? Justifique.

> **Solução:** Sim, pois se somarmos uma constante a todos os nós, ambos algoritmos não mudam seus resultados.

7.4. Considere o problema de *árvore geradora máxima*, ou seja, o problema de encontrar a árvore geradora que maximiza a soma dos custos (que agora podem ser vistos como utilidades) das suas arestas. Os algoritmos de Prim e de Kruskal, modificados para escolher a aresta mais pesada que cruza cada corte, ainda funcionam para este caso?

Solução: Sim, pois se multiplicarmos todos os nós por -1, os algoritmos originais farão exatamente o proposto.

7.6. Suponha que você esteja dirigindo de Campinas para Porto Alegre. O tanque de combustível do seu carro, quando cheio, tem autonomia de m quilômetros e você tem à sua disposição um mapa que fornece as distâncias entre postos de combustíveis ao longo da estrada. Suponha que $d_1 < d_2 < \cdots < d_n$ sejam as localizações dos postos de combustíveis ao logo da sua rota, sendo d_i a distância do posto a partir de Campinas. Assuma que a distância entre dois postos consecutivos não excede m. O seu objetivo é chegar a Porto alegre com o menor número de paradas possível. Construa um algoritmo guloso para resolver este problema e mostre que ele fornece a solução ótima.

Solução: O algoritmo guloso é abastecer no último posto antes de percorrer m quilômetros desde o último abastecimento.
Esse algoritmo é ótimo, pois para a primeira iteração, não seguir o algoritmo guloso coloca numa situação igual ou estritamente pior. A partir do novo ponto, caímos numa situação análoga e, assim, por indução o algoritmo guloso é ótimo.

7.12. Um técnico de manutenção tem à sua disposição quatro equipamentos de teste, os quais são usados em 25%, 30%, 55% e 15% das chamadas. Estes itens pesam 20, 30, 40 e 20 quilogramas, respectivamente. Este funcionário pode carregar até 60 quilogramas. Determine a combinação de equipamentos de maior utilidade que pode ser carregada pelo técnico.

Solução: Esse problema é análogo ao da mochila. Seja $T(i, p)$ a solução ótima usando até o i-ésimo item e permitindo carregar p de peso, então propomos a seguinte resolução por programação dinâmica:

$$T(i, p) = \max\{T(i-1, p), T(i-1, p-p_i) + u_i\},$$

onde p_i é o peso do item i e u_i é a utilidade do item i, $i \in \{1, 2, 3, 4\}$.

$i \backslash p$	0	20	30	40	50	60
0	0	0	0	0	0	0
1	0	25	25	25	25	25
2	0	25	30	30	55	55
3	0	25	30	55	55	80
4	0	25	30	55	55	80

Logo, a melhor solução é usar o primeiro e o terceiro item.

7.13. Considere um vetor de n números inteiros e não-negativos, $a_1, \cdots, a_n$. O seu objetivo é determinar o maior valor de uma combinação de somas e produtos destes elementos, sem parentizações ou reordenações, ou seja, apenas inserindo sinais de + ou × entre os elementos do vetor. Por exemplo, dado o vetor

$$1 \;\; 2 \;\; 3 \;\; 1,$$

o valor máximo que pode ser obtido com somas e produtos é

$$1 + 2 \times 3 + 1 = 8.$$

Resolva este problema com programação dinâmica.

Solução: Note que se não houver 1's entre números adjacentes, então é sempre melhor multiplicá-los; se houver 1's no começo ou no fim do vetor é sempre melhor somá-los. Usando isso, podemos supor que nosso problema é do tipo

$$b_1 \;\; k_2 \;\; b_2 \;\; k_3 \;\; b_3 \;\; \ldots \;\; k_m \;\; b_m,$$

onde b_i é um número inteiro diferente de 1 e pode ser interpretado como a multiplicação de todos os números adjacentes diferentes de 1 e k_i é o número de 1's adjacentes entre números maiores que 1. Nossa solução por programação dinâmica é dada por T_m, e pode ser encontrada através da seguinte equação para $i \in \{2, \ldots, m\}$

$$\begin{aligned} T_i = \max \big\{ & T_{i-1} + k_i + b_i,\; T_{i-2} + k_{i-1} + b_{i-1}b_i,\; \ldots, \\ & T_2 + k_3 + b_3 \cdots b_i,\; b_1 + k_2 + b_2 \cdots b_i,\; b_1 \cdots b_i \big\}. \end{aligned}$$

Vamos exemplificar com o vetor: $3 \;\; \underbrace{1 \;\; 1}_{k_2=2} \;\; 3 \;\; \underbrace{1 \;\; \ldots \;\; 1}_{k_3=9} \;\; 2,$

$$\begin{aligned} T_2 &= \max\{3 + 2 + 3,\; \underline{3 \cdot 3}\} = 9, \\ T_3 &= \max\{\underline{9 + 9 + 2},\; 3 + 2 + 3 \cdot 2,\; 3 \cdot 3 \cdot 2\} = 20. \end{aligned}$$

Logo, a melhor escolha é $3 \cdot \underbrace{1 \cdots 1}_{k_2=2} \cdot 3 + \underbrace{1 + \cdots + 1}_{k_3=9} + 2.$