Tutorial: Barras e Barras

1 Solução do Problema

O problema do corte de barras (*Rod Cutting*) pode ser resolvido de forma eficiente por meio de **programação dinâmica**. A ideia é decompor o problema em subproblemas menores: para cada comprimento L queremos saber qual é a melhor receita máxima que podemos obter cortando (ou não cortando) uma barra de comprimento L.

1.1 Definição do Subproblema

Seja $p[1 \dots n]$ o vetor de preços onde $p[i]$ é o preço de vender um pedaço de comprimento i. Definimos:

$$dp[L] = \text{o maior receita obtida cortando uma barra de comprimento } L.$$

Assim, $dp[L]$ representa a resposta ótima para uma barra de tamanho L.

1.2 Função de Transição

Para determinar $dp[L]$ consideramos a primeira peça que retiramos da barra: se escolhemos cortar uma peça de tamanho x (com $1 \leq x \leq L$), obtemos imediatamente o preço $p[x]$ e ficamos com o subproblema da barra restante de tamanho $L - x$, cujo melhor valor é $dp[L - x]$. Logo:

$$dp[L] = \max_{1 \leq x \leq L} \big(p[x] + dp[L - x]\big).$$

Em particular, se não fizermos cortes (i.e., vendermos a barra inteira), isso corresponde ao caso $x = L$ e é coberto pela fórmula acima.

1.3 Casos Base

O caso base é natural:

$$dp[0] = 0,$$

isto é, uma barra de comprimento zero tem receita zero.

1.4 Recuperação da Decomposição (pedaços)

Para além do valor máximo $dp[n]$, frequentemente queremos também uma decomposição concreta (lista de comprimentos dos pedaços) que atinja esse valor. Para isso mantemos, durante a computação de dp, um vetor auxiliar `cut[L]` que armazena o tamanho x do primeiro pedaço que produz o ótimo para $dp[L]$. Após preencher dp e `cut`, reconstruímos os pedaços repetindo a operação: tomar $x = \texttt{cut}[L]$, imprimir x, e reduzir $L \leftarrow L - x$, até $L = 0$.