TP3 - Exposição de Tecidos

Leandro Freitas de Souza - 2021037902

1 Introdução

Esse trabalho trata da resolução de um problema de programação dinâmica. Um vendedor de uma loja de tecidos recebe uma remessa de rolos e quer enfileirá-los em uma prateleira. No entanto, ao receber os rolos, ele possui uma série de limitações: os rolos precisam de ser enfileirados na ordem em que são recebidos e em ordem decrescente de preços. Além disso, o comerciante possui três alternativas para o posicionamento dos rolos: colocá-lo na pratelera pelo lado direito e empurrá-lo até encostar nos rolos já posicionados, realizar o mesmo procedimento colocando o rolo pelo lado esquerdo ou não colocá-lo.

O objetivo do trabalho é fornecer um algorítmo que determine o maior número de rolos que podem ser posicionados na prateleira dadas as regras impostas, tendo como entrada a lista de preços de cada rolo, na ordem em que são recebidos.

2 Modelagem

A partir da seqência preços fornecidas na entrada, os valores foram armazenados em um vetor em suas respectivas posições. Logo, temos que a partir de um rolo inicial, será escolhida uma subsequência de rolos com preço crescente para ser inserida à esquerda do rolo inicial e outra subsequência decrescente para ser inserida à direita. Dessa forma, o rolo cujo tamanho da subsequência crescente à sua direita somado com o tamanho da maior subsequência decrescente à sua direita for o maior possível será o rolo a ser colocado inicialmente na prateleira.

2.1 LIS e LDS

Para calcular as maiores subsequências crescentes e decrescentes que começam em um elemento específico do vetor, foi necessário invertê-lo para calcular os algorítmos de programação dinâmica LIS (Longest Increasing Subsequence) e LDS(Longest Decreasing Subsequence). Tais algorítmos são definidos por:

Seja v o vetor com os preços dos rolos em ordem fornecida. Temos que:

$$LIS(i) = \text{maior subsequência crescente que contenha } i \text{ em} \{1, \ldots, i\}$$

$$LIS(i) = \begin{cases} 1 & \text{se } i = 0 \\ 1 + \max_{0 \leq j < i} LIS(j) & \text{tal que } v[j] < v[i] \end{cases}$$

$$LDS(i) = \text{maior subsequência decrescente que contenha } i \text{ em} \{1, \ldots, i\}$$

$$LDS(i) = \begin{cases} 1 & \text{se } i = 0 \\ 1 + \max_{0 \leq j < i} LDS(j) & \text{tal que } v[j] > v[i] \end{cases}$$

A partir dessas definições, temos que $LIS(i)$ calcula a maior subsequência crescente que *termina* em i. Queremos encontrar a maior subsequência crescente que *começa* em i. Para que isso fosse possível, foi necessário inverter o vetor de entrada, de forma que $LIS(i)$ passe a calcular a maior subsequência decrescente que começa em i, e $LDS(i)$ passe a calcular a maior subsequência crescente que começa em i.

A partir disso, a resolução do problema se dá pela seguinte fórmula:

$$\text{Distribuição Ótima} = \max_{0 \leq i < n} LIS(i) + LDS(i) - 1$$

2.2 Análise de Complexidade

Para o cálculo dos algorítmos de programação dinâmica, foi necessário o aninhamento de dois loops. Tal aninhamento resulta na seguinte equação de recorrência:

$$f(n) = n\left(\frac{n+1}{2}\right) = \frac{n^2 + n}{2}$$

Dessa forma, como o termo polinomial de maior grau é n^2, temos que

$$f(n) \in \mathcal{O}(n^2).$$