Trabalho Prático 3

DCC215 - Algoritmos 1

Igor Joaquim da Silva Costa

1. Introdução

O problema proposto foi descobrir qual a maior sequência de rolos que um vendedor de tecidos poderia colocar em sua prateleira. Mais precisamente, é apresentada uma sequência de rolos e seus pesos. Para montar a prateleira, o vendedor opta por organizar os rolôs em ordem decrescente de peso, a partir da ordem com que eles são apresentados, colocando um novo rolo à direita ou à esquerda da sequência já formada. Dessa forma, a prateleira ideal é aquela onde o maior número de rolos é disposto.

Para resolver o problema citado, cada rolo foi tratado como um número em um vetor, onde, a partir dele, devia-se calcular sub-sequências crescentes e decrescentes máximas que representam a prateleira montada, para cada elemento. Assim, a prateleira ideal inicialmente é formada pelo elemento que maximiza tais subsequências, com as próprias subsequências sendo a configuração da prateleira. Nesse sentido, o problema apresentado é reduzido à encontrar as maiores subsequências crescentes e decrescentes possíveis de serem formadas para todo rolo, após, encontrar qual rolo possui maior soma de subsequências. Diante disso, foi implementado um algoritmo polinomial do tipo Programação Dinâmica capaz de resolver o problema.

Diante do exposto, a documentação presente possui como objetivo detalhar como o sistema foi modelado (Seção 2), o quão eficiente ele pode ser (Seção 3) . Por fim, o projeto é sumarizado junto com os aprendizados gerados durante a produção do trabalho(Seção 4).

2. Modelagem

Esta seção tem como objetivo discutir as decisões que levaram à atual modelagem do programa.

2.1 LIS e LDS

Como elucidado na seção 1, o problema apresentado pode ser reduzido ao problema à encontrar sequências decrescentes e crescentes. O motivo disso se dá pela natureza do

problema. Ao observar como o vendedor monta a prateleira após inserir o primeiro elemento, fica nítido que os elementos a direita do primeiro elemento na prateleira formam uma sequência decrescente, e os elementos a direita, uma crescente - considerando a ordem com que os elementos são selecionados a partir da entrada.

Assim, no contexto apresentado, é calculado o LIS(Longest Increasing Subsequence) e LDS(Longest Decreasign Subsequence) para cada elemento do array, considerando as seguintes equações de recorrência, onde N = tamanho da entrada:

LIS(I) = 1 + max{LIS(J)}, tal que I<J<=N e ARR[I] < ARR[J], se J existir
LIS(I) = 1, caso contrário

LDS(I) = 1 + max{LDS(J)}, tal que I<J<=N e ARR[I] > ARR[J], se J existir
LDS(I) = 1, caso contrário

2.2 Estrutura de Dados

Para representar o array de entrada, o array de LDS e o array de LIS, foi usado o tipo padrão vector do c++.

3. Análise de complexidade

3.1 Espaço

Seja N o tamanho da entrada, inicialmente, são criados dois vetores de N posições para representar o LDS e o LIS, além do vetor de entrada. Assim, a complexidade de espaço se torna $O(N)$ na quantidade de rolos.

3.2 Tempo

Para análise de tempo, considere N o número de rolos.

3.2.1 Função de recorrência

Para se resolver o LIS e o LDS, são feitas operações iterativas usando o conceito de programação dinâmcia. Tomando as equações de recorrência:

LIS(I) = 1 + max{LIS(J)}, tal que I<J<=N e ARR[I] < ARR[J], se J existir
LIS(I) = 1, caso contrário

Para cada elemento i da entrada, é procurado iterativamente o valor de j, que pode ser qualquer valor entre j e N. Dessa forma, o número de operações cresce como uma progressão aritmética da forma:

$$N(N + 1)/2$$

Ou seja, O(N^2) no tamanho da entrada.

No que tange o LDS, o tempo de execução é equivalente ao cálculo do LIS. Além disso, é necessário procurar qual i que maximiza a soma LDS(i) + LIS(i). Essa busca é feita de forma iterativa, gastando O(N) passos. Logo, o algoritmo para resolver o problema cresce quadraticamente no número de rolos, visto que

$$O(N^2) + O(N^2) + O(N) = O(N^2) \ .$$

4. Conclusões

Com o intuito de descobrir qual a maior sequência de rolos que um vendedor de tecidos poderia colocar em sua prateleira, foi implementado um programa que utiliza algoritmos de programação dinâmica para resolver o problema.

Durante o projeto do sistema foram levadas em consideração não só aspectos práticos da implementação de uma modelagem computacional, mas também como a linguagem de programação escolhida poderia ser uma ferramenta útil para chegar no objetivo esperado. Toda a questão de mapear um mini-mundo de interesse em um modelo computacional robusto se mostrou bastante produtiva, levando o aluno a pensar em formas criativas de se resolver e entender o problema, tendo como resultado um extenso aprendizado sobre como pensar, questionar e implementar um algoritmo de programação dinâmica na prática. Por fim, o tempo extra usado para projetar o sistema trouxe várias recompensas no sentido da implementação, sendo um aspecto a ser levado para trabalhos futuros.

Nesse sentido, todo o fluxo de trabalho foi essencial para a consolidação de conteúdos aprendidos em sala, além de apresentar, de forma prática, como softwares maiores, mais consistentes, robustos e inteligentes são projetados e implementados.

5. Instruções para compilação e execução:

5.1 Compilação

Existem partes do programa que são compatíveis apenas às versões mais recentes da linguagem c++, dito isso, deve-se seguir as seguintes configurações para a compilação:

Linguagem: C++
Compilador: Gnu g++
Flags de compilação: -std=c++11 -g
Versão da linguagem: standard C++11
Sistema operacional (preferência): distribuições baseadas no kernel Linux 5.15.

O comando para compilar o programa automaticamente está presente no arquivo **“Makefile”** e sua execução é chamada pelo comando **“make all”**. Deste modo, o executável “tp03” estará compilado e pronto para ser utilizado.

5.2 Execução

Seguem as instruções para a execução manual:

1. Certifique-se que o compilável foi gerado de maneira correta, se algum problema ocorrer, execute o comando “make all” presente no “Makefile”.

2. Uma vez que os passos anteriores foram cumpridos, execute o programa com o comando: ./tp03 < Teste01.txt)
3. A saída será impressa no terminal.