Relatório 1º projeto ASA 2024/2025

Grupo: AL077

Aluno(s): Alexandre Delgado (109441) e Madalena Yang (110206)

Descrição do Problema e da Solução

O objetivo do problema é encontrar a parentização mais à esquerda da sequência de operações de forma que o resultado final seja o valor desejado.

A solução proposta consiste em preencher uma tabela de programação dinâmica que guarda valores necessários para o cálculo dos próximos.

Solução e Pseudocódigo

A tabela dp[i][j] armazena, para cada subsequência i até j, os vários valores possíveis que podem ser obtidos a partir desta. Para além dos valores, também são armazenados o índice em que se separa a subsequência em duas outras subsequências, o valor em que a subsequência da esquerda resulta e o valor em que a subsequência da direita resulta.

Para criar a string da solução, caso exista, usa-se uma função recursiva cujo caso base é i = j, ou seja, quando se chega à diagonal principal da tabela. Neste caso apenas se retorna o valor correspondente. Caso contrário, percorre-se todos os vetores que se encontram em dp[i][j] e assim que se encontra o resultado pretendido, chama-se recursivamente a função para o lado esquerdo e o lado direito da sequência.

$$D[i,j] = \begin{cases} sequência[i], & se\ i = j \\ \underset{para\ i < k \leq j}{D[i,k] \oplus D[k+1,j]}, & c.c. \end{cases}$$

for *i = 1 to m* *// O(m)*

    dp[i][i] := [sequence[i], i, -1, -1]

for *diagonal = 1 to m – 1* *// O(m-1) ≈ O(m)*

    for *i = 1 to m – diagonal* *// O(m-diagonal) ≈ O(m)*

        j = i + diagonal

        for *k = j – 1 to i* *// O(m-1) ≈ O(m)*

            for *o = 0 to x where x = | dp[i][k] |* *// O(n/2) ≈ O(n)*

                for *p = 0 to y where y = | dp[k+1][j] |* *// O(n/2) ≈ O(n)*

                    dp[i][j] += [dp[i][k] ⊕ dp[k+1][j], k, dp[i][k][o][0], dp[k+1][j][p][0]]

Análise teórica

- **Leitura dos dados de entrada:** ciclo a depender quadraticamente de *n* (tamanho do conjunto de inteiros sobre os quais a operação é aplicada) e linearmente de *m* (tamanho da sequência de inteiros).
  Complexidade: $O(n^2 + m) \approx O(n^2)$. Logo, $\mathbf{O(n^2)}$.

Relatório 1º projeto ASA 2024/2025

Grupo: AL077

Aluno(s): Alexandre Delgado (109441) e Madalena Yang (110206)

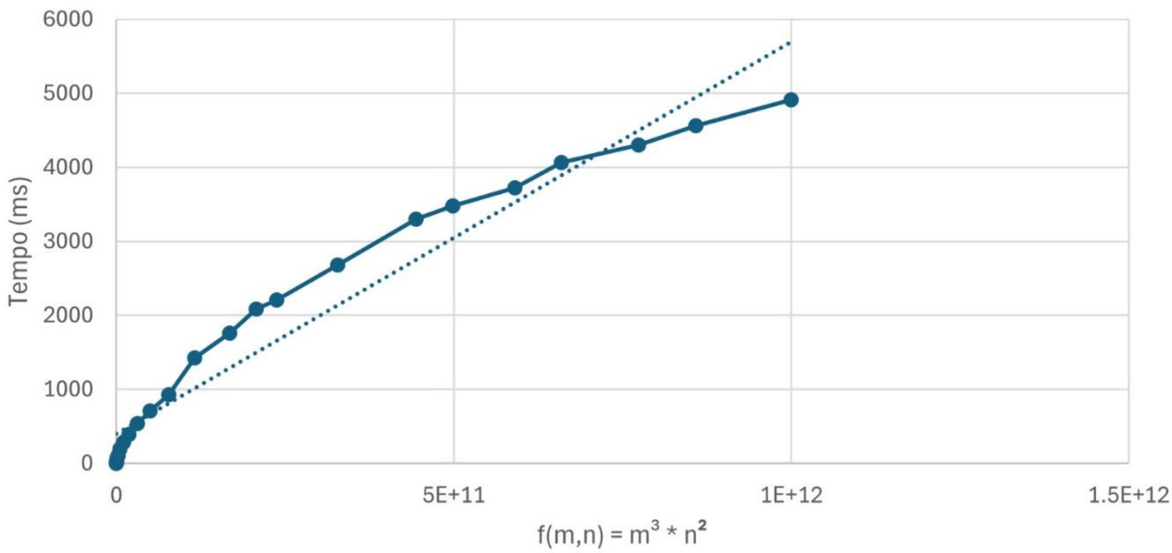
- **Preenchimento da tabela:** olhando para o pseudocódigo acima apresentado, a complexidade do preenchimento da diagonal principal é O(m), que quando comparado com a complexidade do preenchimento da tabela é insignificante. A complexidade do preenchimento da tabela é:
  Dimensões da tabela X Custo de preenchimento de cada célula
  $O(m^2)$ X $O(mn^2)$ = $\mathbf{O(m^3n^2)}$
- **Apresentação dos dados:** no pior caso, a função divide a sequência em duas partes a cada chamada, formando uma árvore binária de chamadas recursivas proporcional à profundidade da dp (O(m)). Para cada chamada, tem-se um loop for cuja complexidade é O(n). Assim, a complexidade total da reconstrução da solução é:
  Nº de chamadas recursivas X Custo de cada chamada
  O(m) X O(n) = **O(mn)**

Como a complexidade dominante é o preenchimento da tabela, conclui-se que a complexidade global da solução é $\mathbf{O(m^3n^2)}$

Avaliação Experimental dos Resultados

Após gerar várias instâncias de tamanho incremental, obteve-se o seguinte gráfico: no eixo das abcissas, está a função correspondente à complexidade prevista; no eixo das ordenadas, o tempo em milissegundos.

Observa-se que, devido a otimizações implementadas, o comportamento do gráfico apresenta desvios em relação à regressão linear esperada, aproximando-se de uma função logarítmica em alguns casos.



Assim, conclui-se que a implementação não só está de acordo com a análise teórica, mas também se beneficia de otimizações que reduzem o impacto prático da complexidade prevista.