Instituto Tecnológico de Aeronáutica - ITA
CTC-12 - Projeto e Análise de Algoritmos
Aluno: Ulisses Lopes da Silva

Relatório do Laboratório 4 - Paradigmas de Programação (parte 2)

Objetivo: Implementar algoritmos para a resolução do Problema de Soma de Subconjuntos (SSP), cuja abordagem foi simplificada a partir do *Knapsack* geral, mediante o paradigma de Programação Dinâmica e de outra qualquer, à escolha do aluno, e que, para este relatório, será o *Meet-in-the-Middle*.

1 Introdução

O Problema da Mochila (Knapsack Problem) é um clássico problema de otimização combinatória que possui várias versões e aplicações práticas. Em sua forma mais simples, o problema consiste em determinar quais itens de um conjunto devem ser incluídos em uma mochila de capacidade fixa para maximizar o valor total dos itens, sem ultrapassar a capacidade da mochila. Para este relatório, abordou-se uma versão ainda mais simplificada do problema, nomeada de Subset Sum Problem (SSP), cujo objetivo é determinar se existe um subconjunto de itens cuja soma dos pesos é exatamente igual à capacidade da mochila.

1.1 Algoritmos utilizados

Dois algoritmos foram implementados para resolver o SSP:

- **Programação Dinâmica** (PD): Um método baseado em construir uma tabela que armazena subproblemas menores até chegar à solução do problema original.
- **Meet in the Middle** (MM): Um método que divide o conjunto de itens em duas partes, calcula todas as somas possíveis de cada parte e utiliza uma abordagem de busca para encontrar a combinação de somas que resulta no valor desejado.

2 Comparação entre os algoritmos e análise dos resultados

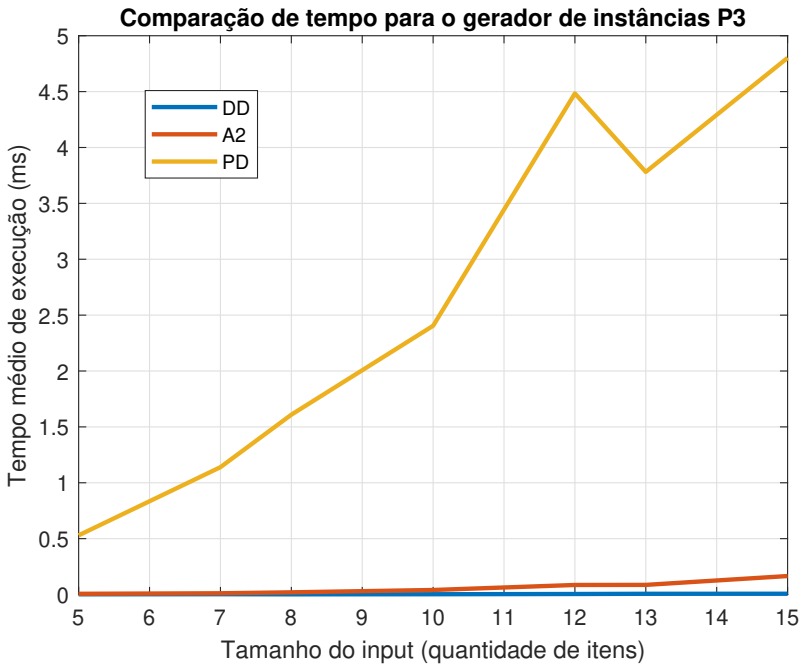

Fig. 1: Comparação de tempo para o gerador de instâncias P3

Neste gráfico, observa-se que o algoritmo PD (Programação Dinâmica) apresenta uma taxa de crescimento alta no tempo de execução à medida que o tamanho do input aumenta. Isso se deve à complexidade pseudopolinomial do PD, que é influenciada diretamente pelo valor dos itens. Em contraste, o algoritmo A2 (Meet in the Middle) demonstra um desempenho significativamente melhor, com tempo de execução quase constante para diferentes tamanhos de input, considerando o tipo de instância. A diferença de desempenho entre PD e A2 pode ser atribuída à eficiência do MM em dividir o problema em partes menores e resolver subproblemas de maneira mais eficaz. Todavia, como esperado, ainda é mais lento que o gabarito.

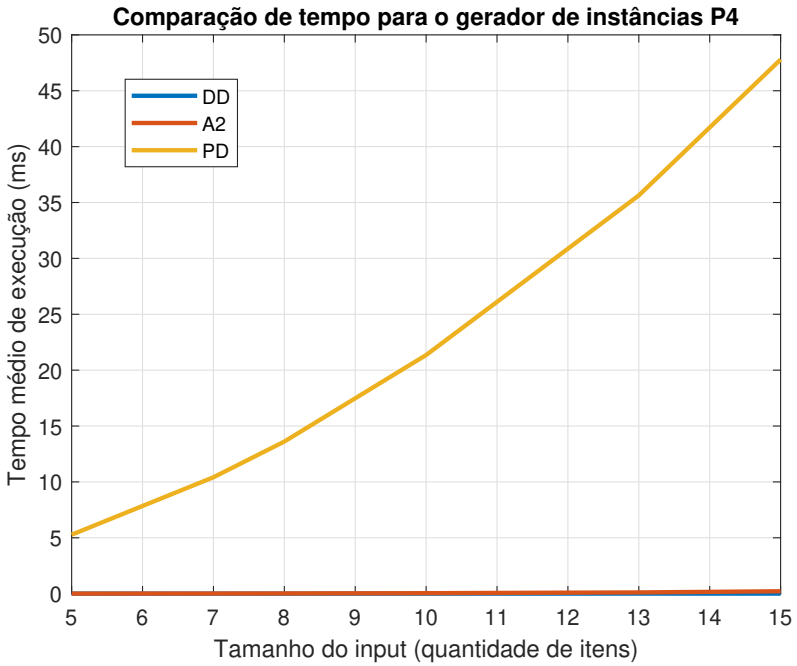

Fig. 2: Comparação de tempo para o gerador de instâncias P4

Neste gráfico, a tendência observada é semelhante à do gráfico anterior, com o algoritmo PD exibindo um crescimento significativo no tempo de execução conforme o tamanho do input aumenta. O algoritmo A2 continua a apresentar tempos de execução baixos e aproximadamente constantes em relação ao número de inputs. Essa estabilidade no tempo de execução do A2 reforça a superioridade do MM em detrimento do PD, que está se mostrando não tão bom para a resolução desse tipo de problema, pelo menos para a quantidade considerada (até 15).

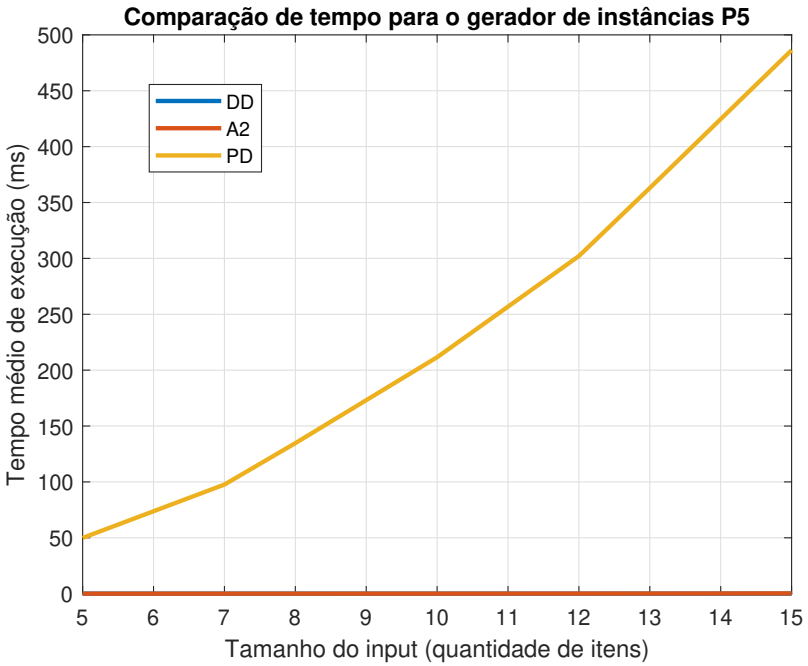

Fig. 3: Comparação de tempo para o gerador de instâncias P5

O gráfico para o gerador de instâncias P5 mostra um padrão consistente com os gráficos anteriores. O tempo de execução do PD aumenta de forma acentuada com o tamanho do input, enquanto o A2 mantém um tempo de execução constante e baixo. Note que a linha de gabarito (DD) quase não aparece, o que significa que este foi ainda mais rápido que o A2, o que era esperado.

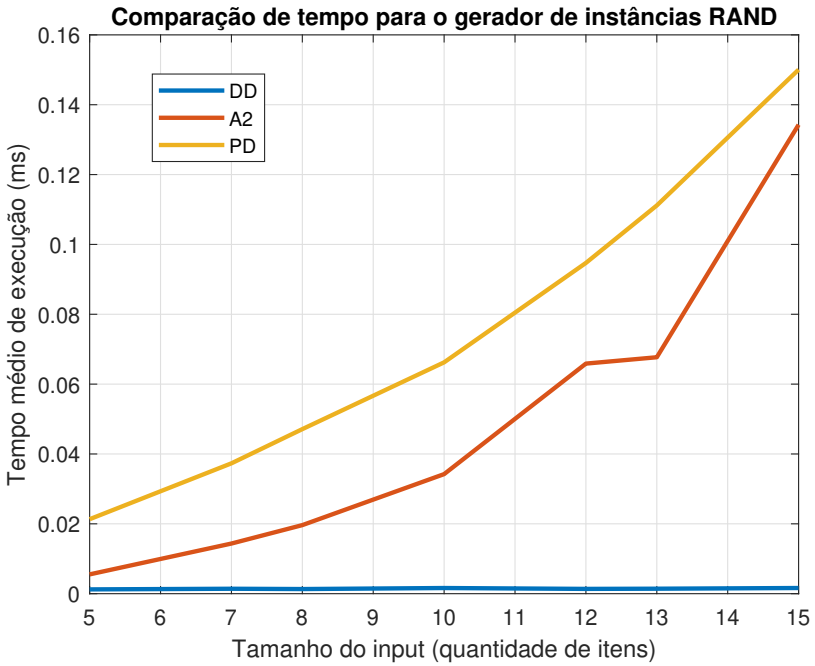

Fig. 4: Comparação de tempo para o gerador de instâncias RAND

Para o gerador de instâncias RAND, o algoritmo A2 começa a apresentar um aumento no tempo de execução conforme o tamanho do input cresce, mas ainda assim, mantém-se inferior ao tempo de execução do PD. Este gráfico ilustra que, mesmo em instâncias aleatórias (para o limite de inputs considerado), o MM é mais eficiente que o PD. A variação no tempo de execução do A2 pode ser explicada pela natureza aleatória das instâncias, que pode ocasionalmente aumentar a complexidade das combinações de somas.

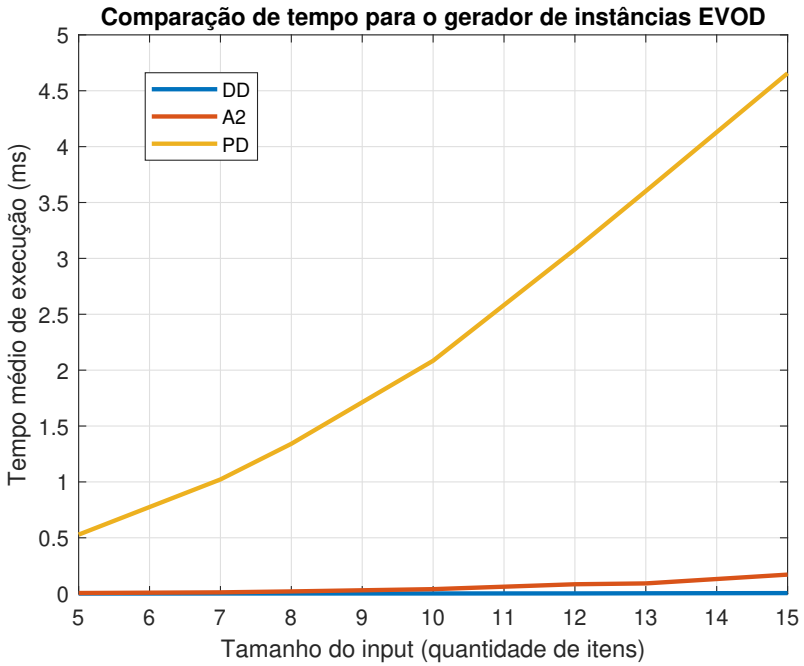

Fig. 5: Comparação de tempo para o gerador de instâncias EVOD

No gerador de instâncias EVOD, observa-se que o tempo de execução do PD continua a crescer sensivelmente com o tamanho do input. O algoritmo A2, por outro lado, mantém sua eficiência, apresentando um tempo de execução quase constante. Este gráfico reforça ainda mais a superioridade do MM em termos de eficiência de tempo, mesmo em instâncias que podem ter uma estrutura mais complexa devido à distribuição dos valores.

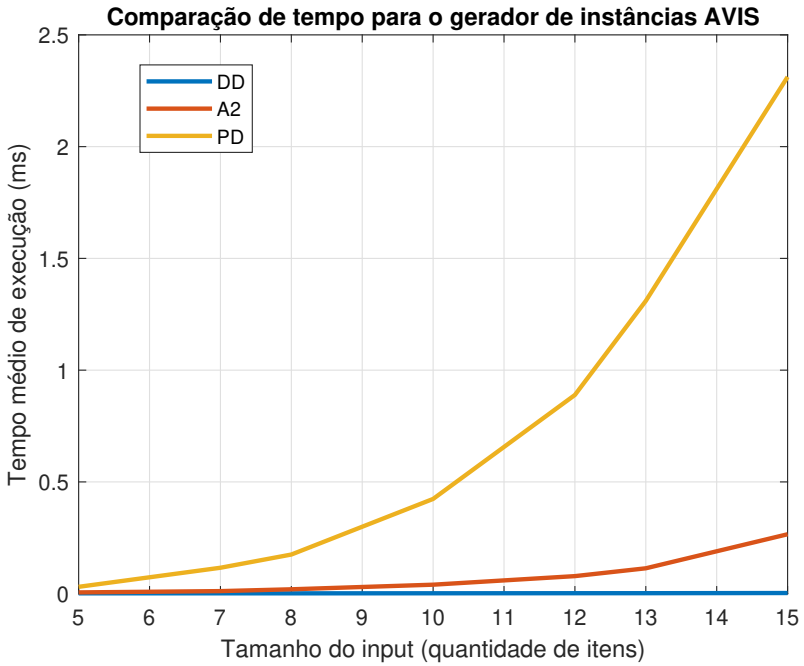

Fig. 6: Comparação de tempo para o gerador de instâncias AVIS

Para o gerador de instâncias AVIS, o comportamento dos algoritmos é igualmente consistente com os gráficos anteriores. O tempo de execução do PD aumenta significativamente com o tamanho do input, enquanto o A2 mantém um tempo de execução baixo, apesar de começar a crescer a partir de 12, aparentemente de forma exponencial.

3 Conclusão

Os gráficos e as análises demonstram que, para a quantidade de inputs testados, o algoritmo Meet in the Middle (MM) demonstrou ser superior à abordagem de Programação Dinâmica em termos de eficiência de tempo na resolução do Subset Sum Problem (SSP). Enquanto o PD enfrenta um aumento aparentemente exponencial no tempo de execução conforme o tamanho do input cresce, o MM mantém um tempo de execução relativamente baixo e, em boa parte dos gráficos, constante, dados os tamanhos de inputs. Esta eficiência se deve à capacidade do MM de dividir o problema em subproblemas menores e resolver essas partes de maneira mais eficaz.

É possível que, para a resolução desse problema, que possui caráter pseudopolinomial, como já comprovado pela teoria, a abordagem PD se sobressaia para instâncias com inputs maiores, nas quais o MM já não será mais tão eficiente, de modo semelhante ao que acontece com o Quick Sort (que apresenta extrema eficiência em seu caso médio, mas em situações específicas, seu tempo pode ser muito ruim). Todavia, para esse número de inputs, o algoritmo não foi testado para o escopo deste relatório, de modo que, para entradas de até 15 inputs, o MM se saiu sensivelmente melhor.

Cabe ressaltar, também, que o caráter pseudopolinomial, apesar de comprovado, não pôde ser verificado claramente nos gráficos, cuja escala ainda era pequena devido ao baixo número de

inputs. Finalmente, o laboratório em questão foi de grande valia para uma melhor familiarização com o tema da Programação Dinâmica na construção de algoritmos eficientes, bem como no melhor entendimento de problemas que possuem diversas aplicações práticas.

4 Referências Bibliográficas

[1]. **CORMEN**, Thomas M. Algoritmos - Teoria e Prática. 3a edição, Editora ELSEVIER, Rio de Janeiro, RJ, 2012.

[2]. **CAETANO, SOUZA e BERNARDINI**, A. F. M., E. S. S e M. S. - Um Estudo Empírico de Algoritmos para o Problema da Soma de Subconjuntos. Disponível em: $https$: $//drive.google.com/file/d/1NS9IEUzO4u1fQdUlqWx6N_Wdvrj OnIPm/view$. Acessado em 09 de junho de 2024.