Análise Knapsack 0/1

Lucas Carvalho, Letícia Americano, Rafael Dalli Soares

[1]Pontifícia Universidade Católica de Minas Gerais (PUC-MG)
Belo Horizonte – MG – Brasil

Abstract. *In this final project of the Algorithm Design and Analysis (PAA) course, we delve into the algorithm complexity, focusing on the performance of brute force, greedy method, and dynamic programming in the context of the 0/1 Knapsack problem. Detailed tests were conducted, analyzing metrics such as the number of books, total stars, average stars, and execution time. We observed that dynamic programming yields results comparable to brute force but with feasible execution time, while the greedy method provides quick but suboptimal solutions. We conclude that dynamic programming is ideal for the problem, while a greedy implementation can be viable with additional optimizations.*

Resumo. *Neste trabalho final da disciplina de Projeto e Análise de Algoritmos (PAA), exploramos a complexidade de algoritmos, destacando o desempenho de força bruta, método guloso e programação dinâmica no contexto do problema da Mochila 0/1. Realizamos testes detalhados, analisando métricas como quantidade de livros, total de estrelas, média de estrelas e tempo de execução. Observamos que a programação dinâmica apresenta resultados equiparáveis à força bruta, mas com tempo de execução viável, enquanto o método guloso oferece soluções rápidas, embora sub-ótimas. Concluímos que a programação dinâmica é ideal para o problema, enquanto uma implementação gulosa pode ser viável com otimizações adicionais.*

1. Informações gerais

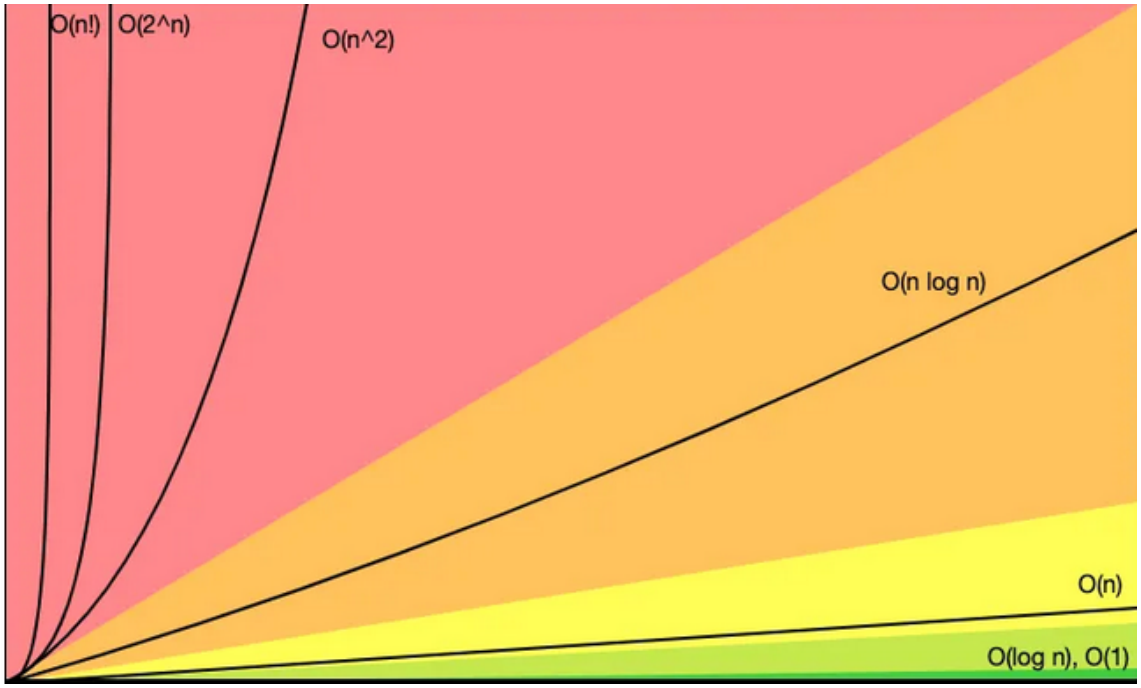
1.1. Hardware

Todos os testes aconteceram em um notebook gamer, Lenovo Ideapad Gaming 3i. O mesmo possui como processador um Intel Core i5 10300h, com 4 núcleos e 8 threads alem de 8GB de memória RAM a 2933MHZ.

1.2. Software

Todas as aplicações foram codificadas no VsCode, utilizando a linguagem Python3. Utilizamos a biblíoteca ”time” para medição de tempo, ”pandas” para tratamento de dados e ”tkinter” para a criação de interfaces graficas.

2. Introdução



2.1. Brute Force

Como já é senso comum dentro da disciplina, o algoritmo de força bruta é um algoritmo que garante solução ótima para todos os casos, mas a sua complexidade de $O(2^n)$, o torna inviável para entradas maiores, assim como veremos nos resultados.

2.2. Greedy

O método guloso, apesar de não garantir solução ótima, tem a menor complexidade dentre os testados. Sua complexidade $O(n * logn)$, faz com que o mesmo execute em tempos excelentes, até mesmo para entradas maiores. Neste trabalho utilizamos uma versão otimizada do método guloso, mantendo a mesma complexidade base, porém melhorando a precisão de suas respostas.

2.3. Dynamic Programing

A programação dinâmica possui uma complexidade especial, esta que não depende apenas do valor da entrada, mas também do tamanho da tabela de resultados que é gerada a fim de evitar redundância. Na prática, a mesma funciona como uma complexidade $O(n^2)$, que fica entre a força bruta e o método guloso.

3. Contextualização

Julia, estudante de Ciência de Dados, deseja uma coleção valiosa para seus estudos. Com orçamento limitado, busca otimizar suas escolhas, focando em livros altamente avaliados. Julia planeja usar seus conhecimentos em Ciência de Dados para criar um algoritmo que maximize a qualidade e quantidade da coleção dentro das restrições financeiras.

4. Metodologia

Os testes foram realizados em formas de bateria de execução com tamanhos de entradas e capacidade de mochila distintas.Neste caso, cada coluna representa um dado que contem informaçõe significativas quanto aos resultados obtidos por cada método.

A coluna QUANTIDADE LIVROS, indica quantos livros foram adicionados a solução. Não necessariamente é a melhor métrica para avaliar a qualidade de una resposta, porem é uma boa métrica para avaliar quantas permutações foram testadas por cada algorítmo

A coluna TOTAL ESTRELAS, é a coluna que melhor representa a qualidade de cada uma das respostas, uma vez que a mesma representa um equilibrio entre a quantidade de permutações testadas e a qualidade geral destas permutações

A coluna MÉDIA ESTRELAS, representa a avaliação média em estrelas de todos os livros dentre as opções. Sozinha não representa bem a qualidade da resposta, porem é um bom indicador de que os elementos da permutação da resposta possuem um bom nivel de qualidade proporcional.

E por fim a coluna TEMPO(S) representa o tempo de execução de cada método e nos da uma visão mais clara com valores reais das complexidades que acabamos de discutir.

4.1. Base de dados

Escolhemos a base ”Amazon Data Science Books” do site Kaggle.

O conjunto de dados contém 946 livros obtidos por meio da coleta de informações de livros relacionados a ciência de dados, estatísticas, análise de dados, Python, aprendizado profundo e aprendizado de máquina na Amazon.

5. Bateria 1

Como o esperado, o FORÇA BRUTA teve o maior tempo de execução,o MÉTODO GULOSO teve o menor. Porém o que podemos notar é que a resposta da PROGRAMAÇÃO DINÂMICA foi igual a da FORÇA BRUTA, porem com um tempo melhor.

	quantidade livros	total estrelas	média estrelas	tempo (s)
FB	3	13.3	4.43	0.002680
MG	2	8.3	4.15	0.001549
PD	3	13.3	4.43	0.002341

6. Bateria 2

Aqui ja conseguimos vizualizar a natureza de crescimento exponencial da FORÇA BRUTA. Esta bateria também consegue nos mostrar como o MÉTODO GULOSO pode

ser capaz de nos fornecer soluções sub-ótimas, com um tempo de execução extremamente chamativo. E por fim, pércebemos que a PROGRAMAÇÃO DINÂMICA iguala as soluções alcançadas pela FORÇA BRUTA novamenteo, porem a diferença de tempo esta ainda mais visivel desta vez.

	quantidade livros	total estrelas	média estrelas	tempo (s)
FB	6	27	4.53	50
MG	4	18.6	4.65	0.001145
PD	6	27	4.53	0.002964

7. Bateria 3

Aqui, temos a primeira situaçõa onde a FORÇA BRUTA resultou em um programa inviável devido ao tempo de execução absurdo. Porem podemos ter noção de qual seria a sua saída apenas olhando para a resposta dada pela PROGRAMAÇÃO DINÂMICA. Mais uma vez a execução do MÉTODO GULOSO foi extremamente rápida, porem muito menos precisa.

	quantidade livros	total estrelas	média estrelas	tempo (s)
FB	N/A	N/A	N/A	N/A
MG	25	114.0	4.56	0.007295
PD	56	269	4.42	0.077594

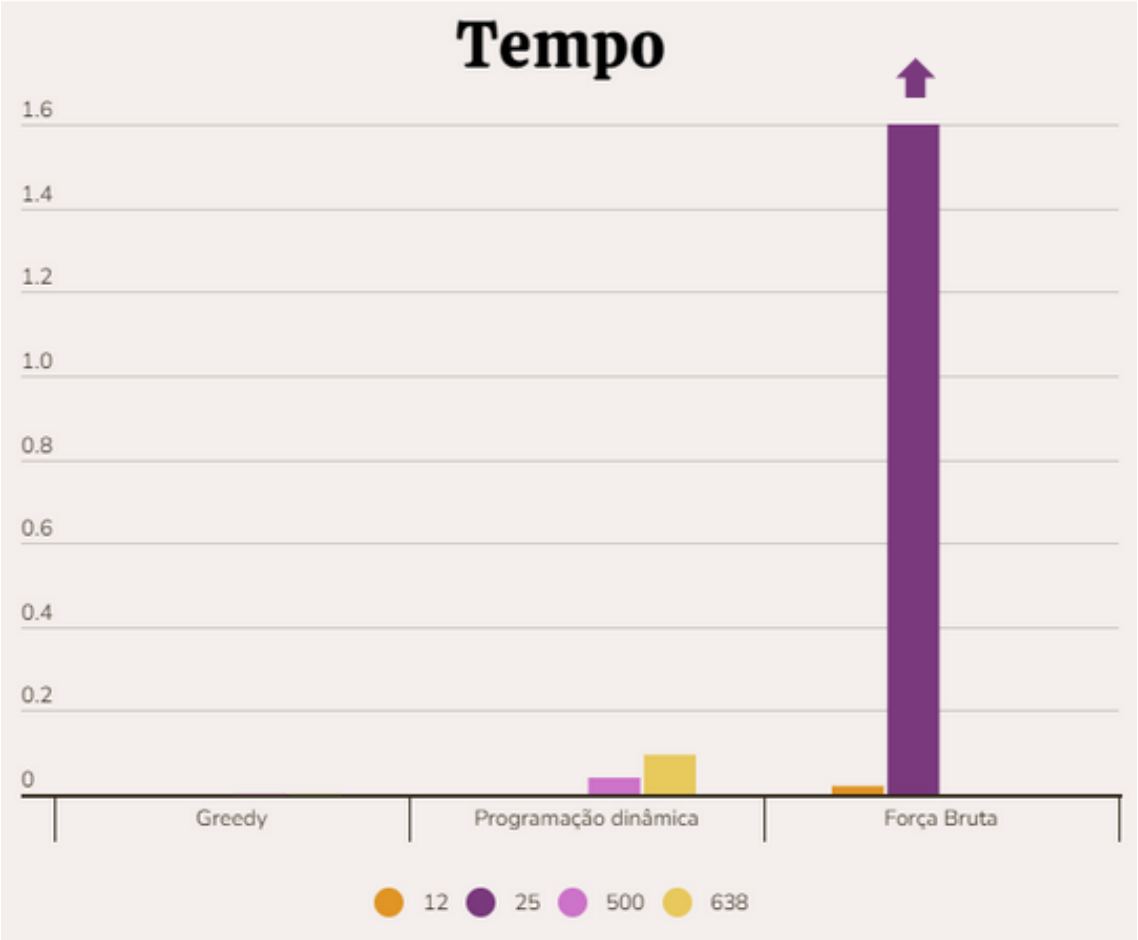
8. Bateria 4

Na nossa ultima bateria de tetes, mais uma vez a FORÇA BRUTA se mostra inviavel. Porem, os valores maiores tanto de orçamento quanto o maior numero de opções disponives, agravou a diferença de qualidade de soluçõa entre a PROGRAMAÇÃO DINÂMICA e o MÉTODO GULOSO. Porem, as entradas maiores também deixaram ainda mais clara a vantagem do MÉTODO GULOSO quando o assunto é tempo de execução

	quantidade livros	total estrelas	média estrelas	tempo (s)
FB	N/A	N/A	N/A	N/A
MG	34	150	4.33	0.010181
PD	60	472	4.41	0.199535

9. Comparação final

Pode-se dizer que o método da PROGRAMAÇÃO DINÂMICA é o método ideal para resolver o problema da Julia, uma vez que atinge os mesmos resultados da FORÇA BRUTA, porem em tempos de execução realista. Uma implementação GULOSA, também seria viavel, caso a fosse feita mais uma otimização com o intuíto de melhorar a precisão das sub-ótimas obtidas, sem afetar demais os tempos de execução.



Em última análise, a decisão entre esses métodos dependerá do compromisso desejado entre eficiência computacional e otimalidade da solução, considerando o contexto e as restrições do problema em questão.

https://github.com/Lucascluz/PAA-Trabalho-FINAL
https://www.kaggle.com/datasets/die9origephit/amazon-data-science-books