RELATÓRIO – ATIVIDADE 02

Diogo de Lima Menezes, Marcos Vinicius Medeiros
Universidade Federal do Mato Grosso Do Sul
Implementação Algorítmica; T01 – 2023-2

1 - Introdução

Diversos problemas do cotidiano envolvem a necessidade de tomada de sucessivas decisões, a fim de se obter uma solução ótima. A partir de certa análise, é possível remodelar algum desses problemas maiores em subconjuntos de pequenos problemas, interligados entre si e que se sobrepõem. A programação dinâmica é um método de modelagem e solução matemática para problemas que apresentam essa natureza – sejam esses problemas determinísticos ou estocásticos [1] – a partir da segmentação de um problema em subproblemas onde se possa encontrar a solução ótima; e posteriormente calcular o valor de uma solução ótima para o problema geral, a partir das soluções ótimas de seus subproblemas. Na computação, a partir da utilização desse método, é possível obter uma solução ótima global em um tempo de execução eficiente; e em uma complexidade assintótica menor, em relação a outras abordagens exaustivas, como as 'meramente' recursivas (que não utilizam memoização).

Já os algoritmos gulosos, se concentram em obter o melhor retorno imediato a partir dos parâmetros do problema; ou seja, é um método que busca encontrar de maneira eficiente a melhor solução local. Não sendo essa, algumas vezes, igual a uma solução ótima global. Mas por sua estratégia simplista (não muito recomendável em alguns problemas), tendem a ser mais eficientes que algoritmos que utilizam programação dinâmica.

Este relatório tem como objetivo descrever e analisar os resultados obtidos a partir de experimentações realizadas a partir de dois algoritmos, um criado a partir de estratégias de programação dinâmica, e o segundo sendo um algoritmo guloso (*greedy*). Ambos foram aplicados ao mesmo problema (*Rod Cutting Problem)*, a fim de se comparar suas eficiências em tempo, e suas capacidades de obter uma solução ótima global, quando utilizados.

2 - Metodologia

O *Rod Cutting Problem* apresenta a seguinte situação: tendo uma tora de madeira de tamanho ***T***, é possível realizar a segmentação desse pedaço de madeira em ***I*** pedaços menores, de tamanhos variados, cuja soma dos ***I*** comprimentos dos segmentos deve ser igual a ***T***. Assim, a partir de uma tabela (*TP)* que enumera o preço de venda de segmentos de tora de tamanhos **1** até ***T***, deve-se determinar o melhor preço de venda a partir da tora de madeira, e suas possíveis segmentações de tamanho inteiro.

Como pode-se supor, os dois parâmetros determinantes do problema são o tamanho da tora de madeira (*T*) e a tabela de preços dos segmentos (*TP*). O principal fator de interesse dos experimentos realizados foi a porcentagem de similaridade dos resultados dos diferentes algoritmos (dinâmico e guloso) quando expostos às mesmas instâncias do problema. A fim de facilitar o processo de entrada de parâmetros, resolveu-se por realizar a determinação da tabela de preços de forma pseudo-aleatória, definindo os preços para cada tamanho ***Ti*** de segmentação em um intervalo de 1 até 10. Portanto, os parâmetros de entrada da aplicação foram reduzidos para:

inc - tamanho inicial de toras

fim - tamanho final de toras

stp - intervalo entre os tamanhos

Para cada tamanho de tora (***T***) entre ***inc*** e ***fim***, com um intervalo de ***stp*** entre si, realizou-se a determinação de uma tabela de preço ***TP*** para os segmentos de tamanho 1 até ***T***. E a partir do tamanho corrente da tora e da mesma tabela de preços gerada, realizou-se a execução dos dois algoritmos anteriormente citados – um algoritmo baseado em programação dinâmica, com uma abordagem *top-down*, e um algoritmo guloso.

O software responsável pela execução dos experimentos, assim como os algoritmos propostos, foi realizado a partir da linguagem de programação Python. A aplicação é responsável por receber as entradas de configuração de experimento do usuário, chamar os métodos, organizar os dados e apresentá-los ao usuário em forma de tabela impressa no console da aplicação.

Para geração dos gráficos, utilizados para facilitar a apresentação dos resultados obtidos, e comparações entre os desempenhos dos algoritmos, utilizou-se a ferramenta *gnuplot*.

3 - Resultados de experimentações

Para a execução dos algoritmos, que produziram os resultados demonstrados na tabela e gráficos deste relatório, informou-se os seguintes parâmetros de execução:

Tamanho inicial de toras: 100; **Tamanho final de toras:** 2000;

Intervalo entre os tamanhos: 100;

A seguir, são apresentados os resultados de execução, em forma de tabela. Os valores ***vDP*** e ***tDP*** indicam, respectivamente, o valor total da venda obtido pelo algoritmo de programação dinâmica e o tempo de execução do algoritmo, em segundos, quando aplicado a uma tora de tamanho ***n*** (primeira coluna). Os valores ***vGreedy*** e ***tGreedy*** fazem as mesmas referências ao algoritmo guloso. Na última coluna foram dispostas as porcentagens de similaridade que a solução gulosa obteve em relação à solução da programação dinâmica (*vGreedy / vDP * 100*) ao aplicadas a uma tora de tamanho n.

n	vDP	tDP	vGreedy	tGreedy	%
100	800	0.001039	800	0.000037	100.00
200	1200	0.003533	1200	0.000046	100.00
300	2700	0.008154	2700	0.000067	100.00
400	3200	0.015588	3200	0.000103	100.00
500	4500	0.024056	4500	0.000166	100.00
600	1800	0.034925	1800	0.000125	100.00
700	2800	0.047346	2800	0.000154	100.00
800	8000	0.063227	8000	0.000828	100.00
900	5400	0.081143	5400	0.000205	100.00
1000	6000	0.101009	6000	0.000233	100.00
1100	6600	0.123414	6600	0.000251	100.00
1200	3600	0.147096	3600	0.000269	100.00
1300	3465	0.176011	3465	0.000501	100.00
1400	8400	0.206880	8400	0.000342	100.00
1500	4500	0.237893	4500	0.000300	100.00
1600	11200	0.269418	11200	0.000393	100.00
1700	8500	0.305486	8500	0.000357	100.00
1800	12600	0.339065	12600	0.000424	100.00
1900	11400	0.377438	11400	0.000454	100.00
2000	8000	0.416988	8000	0.000467	100.00

4 - Conclusão

Nesta seção, serão discutidos os resultados apresentados brevemente na seção anterior (3). Para isso, resolveu-se separar os resultados a partir de duas subseções, uma focada para as discussões relacionadas aos resultados de **valor máximo de venda**, e outra voltada ao **tempo de execução** desempenhado pelos algoritmos utilizados, durante os experimentos.

Em cada subseção a seguir são apresentados três gráficos. Um gráfico foi gerado para representar a variação dos atributos analisados nas subseções (valor e tempo) em função do tamanho das toras de madeira (n), para cada um dos dois algoritmos. Por fim, um último foi realizado a fim de comparar os resultados dos dois algoritmos – sendo, portanto, o produto dos dois primeiros gráficos apresentados.

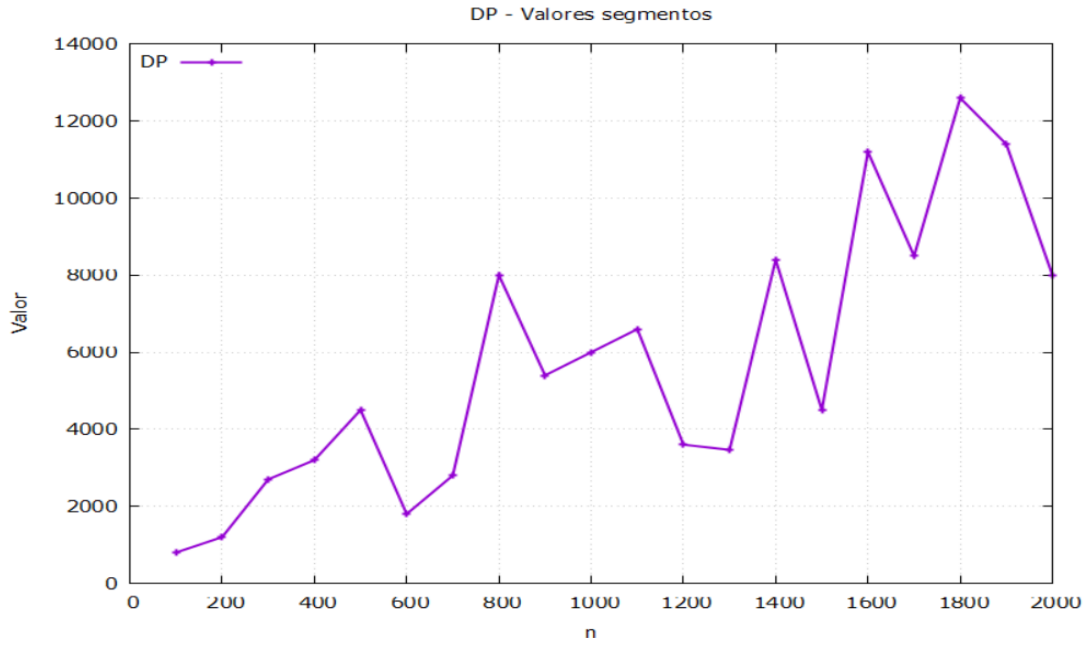
4.1 - Valor máximo de venda

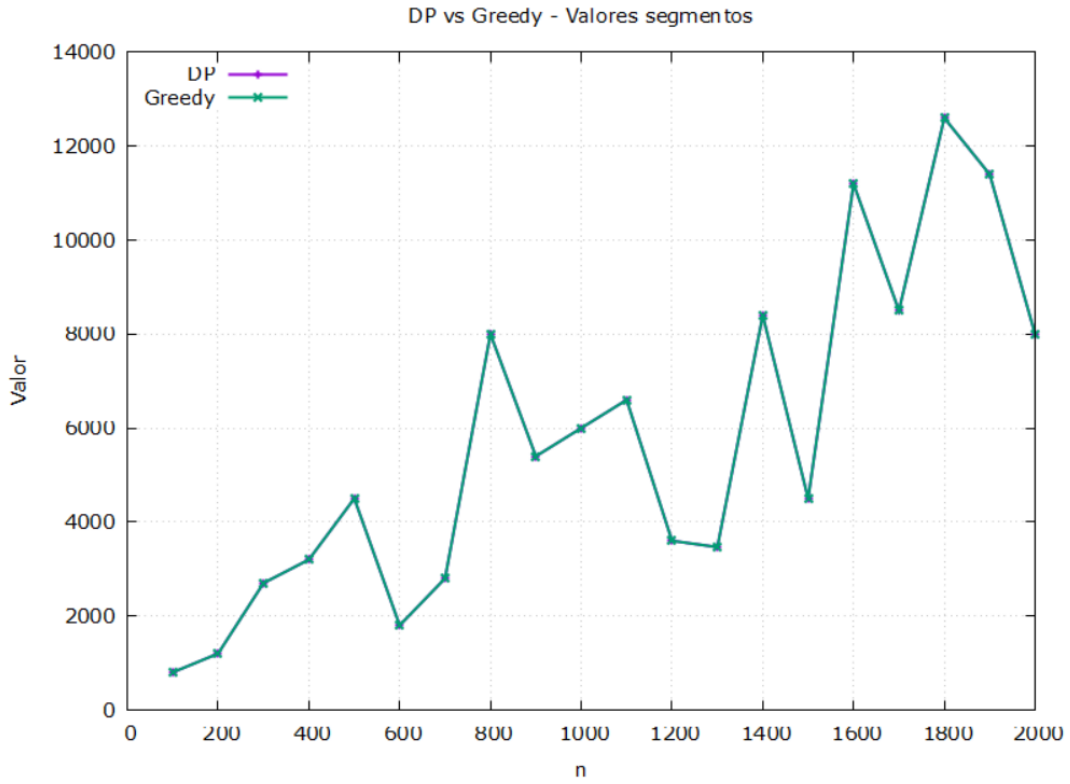
Diferente do que normalmente se esperaria, os experimentos realizados resultaram todos em valores exatamente iguais em ambos os algoritmos. Algoritmos gulosos têm a tendência de se desviar do resultado ótimo global em alguns casos, onde a solução ótima local não corresponde, em conjunto à solução de outros subproblemas, à solução ótima global. Já algoritmos baseados em programação dinâmica, buscam sempre obter uma solução ótima global para o problema.

Muito provavelmente, a coincidência dos resultados se deve a dois fatores principais, a geração aleatória não ordenada de valores e ao intervalo de preços gerados. Como descrito anteriormente (seção 2), a tabela de preços de tamanhos de tora foi gerada de forma pseudo-aleatória, em intervalo de preços de 1 até 10. Pois essa metodologia pode resultar em casos onde tamanhos pequenos de tora possuam um valor máximo, ou próximo do mesmo. Ou seja, pode ser, por exemplo, que um segmento de tora de tamanho 1 possua o preço de venda máximo 10; enquanto um segmento de tamanho 1000 possua o valor de 1.

Sob essa perspectiva, se torna mais compreensível o porquê de, em casos de entrada onde o comprimento da tora é minimamente grande, os valores obtidos pela soluções gulosas correspondam a uma solução ótima global, encontrada pelo algoritmo que utiliza programação dinâmica.

Não obstante, mesmo nesses casos não é possível garantir que o algoritmo guloso sempre encontrará a solução ótima global. Contudo, definitivamente alcançou uma cobertura interessante dos problemas produzidos a partir da metodologia empregada nos experimentos.





4.2 - Tempo de Execução

O algoritmo guloso produzido para a realização dos experimentos utiliza uma extratégia extremamente simples. Primeiramente realiza o cálculo da densidade (*valor/preço*) para todos os tamanhos possíveis de segmentação em uma tora de tamanho ***T*** (1,..,*T*) a partir da tabela de preços utilizada como entrada (***TP*** = *[P1,Pi,...,Pt],* sendo ***Pi*** o preço de um segmento de tamanho ***i*** ente 1 e ***T,*** e ***Pt*** o preço do segmento de comprimento máximo da tora, sendo esse o tamanho da própria tora de madeira, sem cortes). Posteriormente, com os valores da densidade armazenados em um vetor de tamanho ***T***, realiza a obtenção do valor máximo de densidade no vetor. Posteriormente, realiza a subtração do comprimento do segmento de densidade máxima encontrado no vetor, e a adição do valor do segmento a uma variável de valor máximo reproduzível. Esse processo se repete até que o comprimento restante da tora de madeira seja menor que o do segmento máximo, anteriormente encontrado.

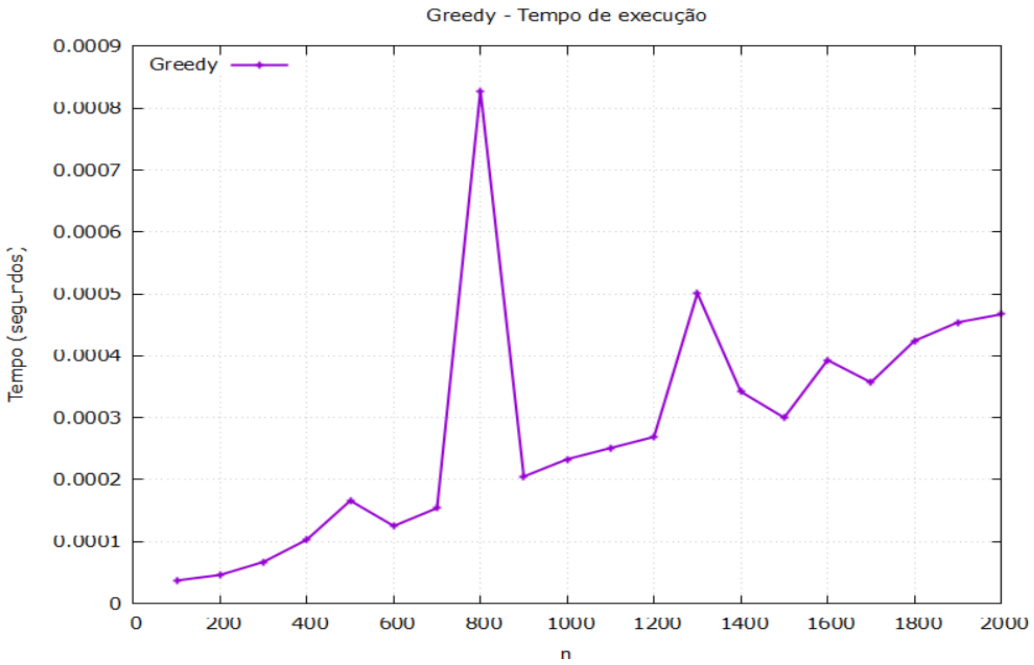
Após isso, caso a tora ainda possua comprimento (*comprimento > 0*), um próximo comprimento de segmento, com densidade máxima, que possa ser subtraído do valor do

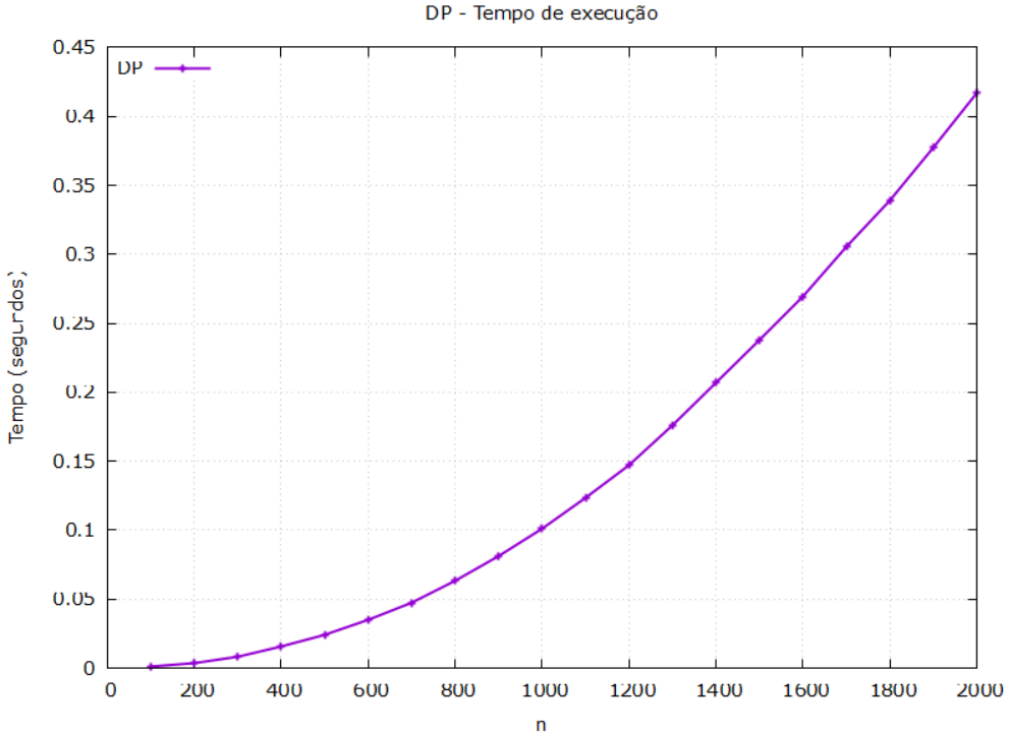
comprimento de tora restante é utilizado. O processo descrito se repete até que o comprimento restante da tora seja igual a zero.

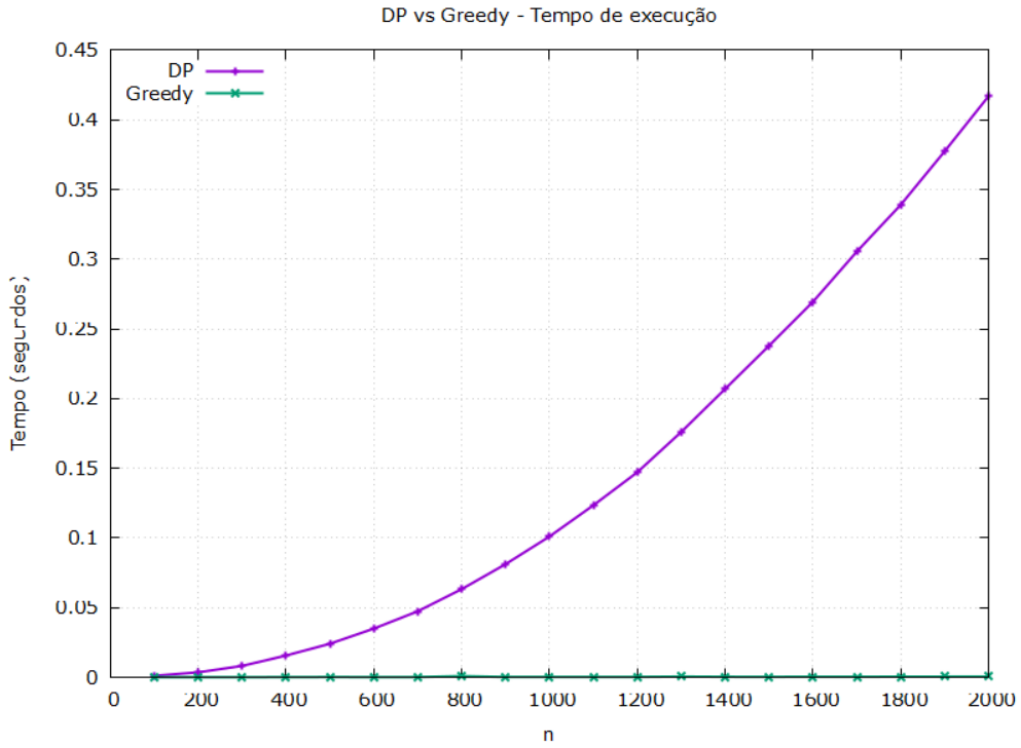
Assumindo que a complexidade assintótica do processo de cálculo de densidade possa ser definida como $\Theta(n)$, e a do processo de cálculo de um valor de venda máximo seja de $O(m*n)$ – onde m representa a quantidade de vezes que um valor máximo de densidade é buscado, e $m < n$ – pode-se inferir que o algoritmo guloso utilizado alcança uma complexidade assintótica linear ($O(n)$).

Já para o algoritmo de programação dinâmico, em sua forma *up-down*, é muito mais difícil de se inferir a complexidade. Contudo, ao perceber o gráfico gerado a partir de seus resultados de tempo de execução, fica claro que há pouca tendência à uma formação semelhante à esperada de um algoritmo linear. Na verdade, possui uma perceptível angulação ascendente, típica de funções polinomiais com exponenciação. Mas felizmente, consultando o material oferecido pelo livro "*Introduction to Algorithms*" [2] , foi possível confirmar a natureza quadrática ($\Theta(n^2)$)do algoritmo utilizado.

Os gráficos a seguir ilustram bem a diferença entre os dois algoritmos, em relação aos seus tempos de execução. O gráfico de tempo do algoritmo *greedy* (guloso) parece ter sofrido com algumas instabilidades durante a execução dos experimentos, muito provavelmente relacionados a fatores ambientais. Mas a sua formação ainda é perceptivelmente linear; e o gráfico de comparação entre os dois algoritmos representa bem o abismo de diferença entre os dois métodos. De forma que o gráfico do algoritmo guloso se torna uma linha quase imperceptível na borda inferior do gráfico, enquanto a linha que representa o desenho do gráfico do algoritmo de programação dinâmica domina todo o restante do espaço, com sua formação ascendente e angular.


Greedy - Tempo de execução
Greedy
Tempo (segundos)
n
0
0.0001
0.0002
0.0003
0.0004
0.0005
0.0006
0.0007
0.0008
0.0009
0
200
400
600
800
1000
1200
1400
1600
1800
2000


DP - Tempo de execução
DP
Tempo (segundos)
n
0
0.05
0.1
0.15
0.2
0.25
0.3
0.35
0.4
0.45
0
200
400
600
800
1000
1200
1400
1600
1800
2000



5- Considerações finais

Ao observar a tendência de comportamento demonstrada nesse relatório, ignorando os conhecidos casos onde os algoritmos gulosos produzem falhas, pode ser que se suponha que o algoritmo guloso deva ser o escolhido, em uma situação onde se busca pela aplicação mais eficiente. Pois, realmente, observando a similaridade de cem por cento entre os valores máximos de venda oferecidos pelos dois algoritmos, não parece haver um porquê de se escolher o mais lento.

Contudo, embora os algoritmos gulosos atendam uma vasta quantidade de problemas semelhantes, a sua utilização deve ser muito bem fundamentada, e não se limitar a testes experimentais. Isso porque experimentos muito dificilmente cobrirão o conjunto de todas as possibilidades de ocorrências de problemas. Afinal, em sentido literal, para isso podem ser necessários testes infinitos, com todas as possibilidades de instâncias do problema analisado. Para evitar vieses, deve-se sempre provar, ou buscar provas, a partir de demonstrações lógicas e matemáticas sobre a cobertura das técnicas utilizadas em relação a uma problemática.

6 - Referências

1. BELLMAN, Richard; KALABA, Robert. Dynamic programming and statistical communication theory. **Proceedings of the National Academy of Sciences**, v. 43, n. 8, p. 749-751, 1957.
2. CORMEN, Thomas H. et al. **Introduction to algorithms**. 4. ed. MIT press, 2022. p. 482 - 491.