PONTÍFICIA UNIVERSIDADE CATÓLICA DE MINAS GERAIS
Graduação em Engenharia de Software

Emmanuel Viglioni, Lucas Machado de Oliveira Andrade, Marcelo Aguilar Araújo D'Almeida, Paulo Victor Pimenta Rubinger

RELATÓRIO TRABALHO PRÁTICO

Belo Horizonte
2024

Emmanuel Viglioni, Lucas Machado de Oliveira Andrade, Marcelo Aguilar Araújo D'Almeida, Paulo Victor Pimenta Rubinger

RELATÓRIO TRABALHO PRÁTICO

Relatório apresentado para disciplina de Fundamentos de Projeto e Análise de Algoritmos, para conclusão do Trabalho Prático

Professor(a): João Caram Santos de Oliveira.

Belo Horizonte

2024

SUMÁRIO

1. INTRODUÇÃO 5

1.1. **Contextualização do Problema e Objetivos do Trabalho….....5**

1.2. **Problema.....6**

2. IMPLEMENTAÇÕES…….....7

2.1. **BackTracking……...7**

2.2. **Guloso……...7**

2.3. **Divisão E Conquista……7**

2.4. **Programação Dinâmica…….11**

3. COMPARAÇÕES DE RESULTADOS.....11

3.1. **Comparação dos Tempos de Execução……11**

3.2. **Comparação da Qualidade dos Resultados…….14**

3.3. **Discussões Sobre Vantagens e Desvantagens de Cada Abordagem……..14**

4. CONSIDERAÇÕES FINAIS.....15

1 INTRODUÇÃO

1.1 Contextualização do Problema e Objetivos do Trabalho

No contexto da disciplina de Fundamentos de Projeto e Análise de Algoritmos, o presente trabalho prático tem como objetivo a implementação e comparação de diferentes técnicas de resolução para um problema de otimização intratável, pertencente à classe NP. O problema em questão envolve a maximização do valor obtido na venda de energia por uma empresa produtora, através de um leilão em que diversas empresas interessadas fazem ofertas por lotes específicos de energia.

A partir deste problema, nosso grupo se propôs a desenvolver soluções utilizando quatro abordagens distintas: backtracking, algoritmos gulosos, divisão e conquista, e programação dinâmica. Cada abordagem foi implementada individualmente por um membro do grupo, permitindo uma avaliação detalhada e comparativa tanto em termos de eficiência temporal quanto na qualidade dos resultados obtidos.

Este relatório técnico está organizado da seguinte forma: inicialmente, apresentamos uma descrição detalhada do problema. Em seguida, cada abordagem de implementação é discutida em separado, com explicações sobre as decisões de projeto, estratégias adotadas e resultados dos testes realizados. Posteriormente, fazemos uma comparação entre os resultados das diferentes abordagens, analisando tanto o desempenho quanto a qualidade das soluções encontradas. Por fim, são apresentadas as considerações finais sobre o trabalho realizado, destacando as lições aprendidas e as possíveis melhorias para futuras implementações.

A divisão das tarefas foi feita da seguinte maneira: Emmanuel Viglioni ficou responsável pela implementação da solução utilizando divisão e conquista; Lucas Machado de Oliveira Andrade desenvolveu a abordagem de programação dinâmica; Marcelo Aguilar Araújo D'Almeida trabalhou nas estratégias gulosas; e Paulo Victor Pimenta Rubinger implementou a solução via backtracking. Essa divisão permitiu uma análise aprofundada e individualizada de cada técnica, contribuindo para uma compreensão mais ampla e detalhada do problema e das soluções possíveis.

1.2 Problema

No cenário atual, a gestão eficiente de recursos é crucial para a competitividade e sustentabilidade das empresas, especialmente aquelas envolvidas na produção e distribuição de energia. Empresas produtoras de energia frequentemente enfrentam o desafio de maximizar seus lucros ao vender sua produção a diversas empresas interessadas. Este problema de otimização, conhecido como o problema do leilão de energia, envolve a venda de uma quantidade fixa de energia a diferentes compradores, cada um fazendo ofertas específicas para lotes de tamanhos determinados.

A complexidade do problema reside na necessidade de escolher quais ofertas aceitar de forma que o valor total obtido seja maximizado, considerando que cada empresa interessada comprará apenas um lote do tamanho exato da oferta. Essa característica confere ao problema uma natureza combinatória intratável, tipicamente pertencente à classe NP. A dificuldade em encontrar a solução ótima aumenta exponencialmente com o número de empresas interessadas e suas respectivas ofertas, exigindo a aplicação de técnicas avançadas de projeto e análise de algoritmos. Abaixo segue um exemplo prático do problema que estamos lidando:

> "Uma empresa produtora de energia possui uma quantidade X de energia, medida em megawatts, para vender. Seu objetivo é vender sua energia produzida, obtendo o maior valor possível no conjunto de suas vendas. As vendas serão realizadas por leilão: cada empresa interessada dará um lance por um lote de K megawatts, oferecendo um valor V por este lote. As interessadas só comprarão um lote do tamanho exato da oferta. Por exemplo, suponha que a produtora tenha 1000 megawatts para venda e tenhamos os seguintes lances do leilão:
>
> - Interessada I1, 500 megawatts, 500 dinheiros
> - Interessada I2, 500 megawatts, 510 dinheiros
> - Interessada I3, 400 megawatts, 520 dinheiros
> - Interessada I4, 300 megawatts, 400 dinheiros
> - Interessada I5, 200 megawatts, 220 dinheiros
> - Interessada I6, 900 megawatts, 1.110 dinheiros
>
> Uma possível venda seria vender para as interessadas I1 e I2, com valor total de 1.010 dinheiros. Outra venda possível, com maior valor, seria para as interessadas I2, I4 e I5 com valor total de 1.130 dinheiros. Veja, por exemplo, que se for feita a venda para as interessadas I3, I4 e I5, o valor total seria de 1.140 dinheiros, mesmo sem vender toda a energia produzida."

2 IMPLEMENTAÇÕES

2.1 BackTracking

Preencher aqui com os dados do seu relátorio…… SE FOR COLAR DADOS, COLAR COM CRLT + SHIFT + V (win) ou Command + SHIFT + V (mac), FAVOR MANTER A FORMATAÇÃO.

Assim como todos os setores da sociedade, o setor de saúde vem se transformando por conta das novas tecnologias. O Hospital 4.0 já não é uma previsão futurística.

2.2 Guloso

Preencher aqui com os dados do seu relátorio…… SE FOR COLAR DADOS, COLAR COM CRLT + SHIFT + V (win) ou Command + SHIFT + V (mac), FAVOR MANTER A FORMATAÇÃO.

Assim como todos os setores da sociedade, o setor de saúde vem se transformando por conta das novas tecnologias. O Hospital 4.0 já não é uma previsão futurística.

2.3 Divisão e Conquista

Divisão e Conquista é uma técnica poderosa e amplamente utilizada na área de algoritmos e estrutura de dados. Para entender melhor, imagine que você tem uma tarefa complexa para realizar. Ao invés de tentar resolver tudo de uma vez, você a divide em partes menores e mais manejáveis. Você resolve cada parte individualmente e depois combina as soluções para resolver o problema original. Esse é o princípio básico da Divisão e Conquista.

Na nossa implementação, optamos por utilizar a técnica de memoização para otimizar a resolução do problema, evitando a recalculação de sub problemas já resolvidos e melhorando significativamente a eficiência do algoritmo. A memoização funciona como um bloco de notas: quando um subproblema é resolvido pela primeira vez, armazenamos o resultado nesse bloco de notas junto com uma chave única que representa esse subproblema. Essa chave, no nosso caso, é gerada a partir da

capacidade restante e do índice da oferta sendo analisada. Se o algoritmo se deparar com o mesmo subproblema novamente, ele simplesmente busca o resultado no bloco de notas usando a chave, sem precisar recalculá-lo. A estrutura de dados que utilizamos para implementar essa técnica foi o HashMap, por ser eficiente tanto na inserção quanto na busca de elementos através de chaves.

Além da memoização, a ordenação inicial das ofertas pelo valor por megawatt (de forma decrescente) também foi crucial para a otimização do algoritmo. Essa ordenação garante que as ofertas mais vantajosas sejam avaliadas primeiro, maximizando o uso da capacidade disponível e aumentando a chance de encontrar a solução ótima mais rapidamente. Imagine, por exemplo, que temos uma capacidade de 100 MW e duas ofertas: uma de 60 MW por R$100 e outra de 40 MW por R$90. Se avaliarmos a oferta de menor valor primeiro, podemos acabar escolhendo uma combinação de ofertas menos vantajosa. Ao ordenar pelo valor por megawatt, garantimos que a oferta de maior valor seja sempre considerada a primeira.

A implementação do algoritmo se divide em duas partes principais: o método "calcular", que atua como a porta de entrada, e o método "calcularRecursivo", responsável pela divisão do problema e combinação dos resultados. O método "calcular" inicia o cronômetro para medir o tempo de execução, ordena as ofertas pelo valor por megawatt e chama o método "calcularRecursivo" para obter o resultado final, que inclui as ofertas selecionadas, o tempo de execução e o valor máximo alcançado.

O método "calcularRecursivo" é o coração do algoritmo. Ele verifica primeiro se a capacidade restante é zero ou se todas as ofertas já foram processadas. Se uma dessas condições for verdadeira, significa que chegamos ao final da recursão e o método retorna 0. Caso contrário, o algoritmo gera a chave única para o subproblema atual e verifica se o resultado já está armazenado no HashMap. Se estiver, o resultado é recuperado e utilizado. Caso contrário, o algoritmo avalia a oferta atual e calcula o valor máximo que pode ser obtido incluindo ou não essa oferta na solução. Para isso, ele chama recursivamente o método "calcularRecursivo" para os subproblemas gerados a partir dessas duas possibilidades.

Após avaliar as duas opções, o algoritmo seleciona a que resulta no maior valor e armazena o resultado no HashMap, juntamente com a chave correspondente. Esse processo se repete recursivamente até que todos os sub problemas sejam resolvidos. A classe auxiliar "ResultadoParcial" é utilizada para armazenar o valor e as ofertas selecionadas para cada subproblema, facilitando a recuperação dos resultados memoizados.

Para ilustrar o funcionamento do algoritmo, considere um exemplo com capacidade total de 10 MW e as seguintes ofertas:

- Oferta 1: 5 MW por R$60
- Oferta 2: 3 MW por R$40
- Oferta 3: 4 MW por R$50

O algoritmo, após ordenar as ofertas, começaria avaliando a Oferta 1. A partir daí, ele se dividiria em dois subproblemas: um incluindo a Oferta 1 e outro não. O subproblema que inclui a Oferta 1 teria uma capacidade restante de 5 MW e avaliaria as Ofertas 2 e 3. Já o subproblema que não inclui a Oferta 1 teria uma capacidade restante de 10 MW e também avaliaria as Ofertas 2 e 3. Esse processo se repetiria recursivamente até que todas as combinações fossem avaliadas e a solução ótima fosse encontrada.

A análise dos resultados obtidos nos testes demonstra a eficiência do algoritmo de Divisão e Conquista com memoização. Para avaliar o desempenho do algoritmo, foram realizados testes com conjuntos de dados de tamanho crescente, variando de 10 a 33 empresas, e realizamos uma execução com 1000 empresas para testar o comportamento com grandes entradas. Para cada tamanho de conjunto, foram executados 10 testes, e o tempo médio de execução foi registrado. A tabela abaixo apresenta os resultados obtidos:

Número de Empresas	Tempo Médio de Execução (segundos)
10	0.001300845900
20	0.007489479300
30	0.025200595800
33	0.031487412500
1000	0.075831600000

Como podemos observar na tabela, o tempo de execução do algoritmo aumenta gradualmente conforme o número de empresas cresce. Esse comportamento é esperado, visto que a complexidade do algoritmo é influenciada pelo tamanho do conjunto de dados. No entanto, mesmo com o aumento do número de empresas, o tempo de execução se mantém dentro de limites aceitáveis, demonstrando a eficiência da implementação.

Para o problema base, do leilão que encontra-se na Seção "1.2 - Problema", o algoritmo de divisão e conquista encontrou os resultados: I4 comprou 300MW por 400 dinheiros, I3 comprou 400MW por 520 dinheiros e I5 comprou 200MW por 220 dinheiros. Obtendo-se assim o lucro máximo de 1140 dinheiros e essa solução foi encontrada em 0.000445250000 segundos

A capacidade de lidar com conjuntos de dados de tamanho considerável em tempo hábil, juntamente com a garantia de encontrar a solução ótima, torna essa implementação uma ferramenta poderosa para auxiliar na tomada de decisão em leilões de energia, maximizando o lucro da empresa produtora.

2.4 Programação Dinâmica

Preencher aqui com os dados do seu relátorio…… SE FOR COLAR DADOS, COLAR COM CRLT + SHIFT + V (win) ou Command + SHIFT + V (mac), FAVOR MANTER A FORMATAÇÃO.

Assim como todos os setores da sociedade, o setor de saúde vem se transformando por conta das novas tecnologias. O Hospital 4.0 já não é uma previsão futurística.

3 COMPARAÇÕES DE RESULTADOS

3.1 Comparação dos Tempos de Execução

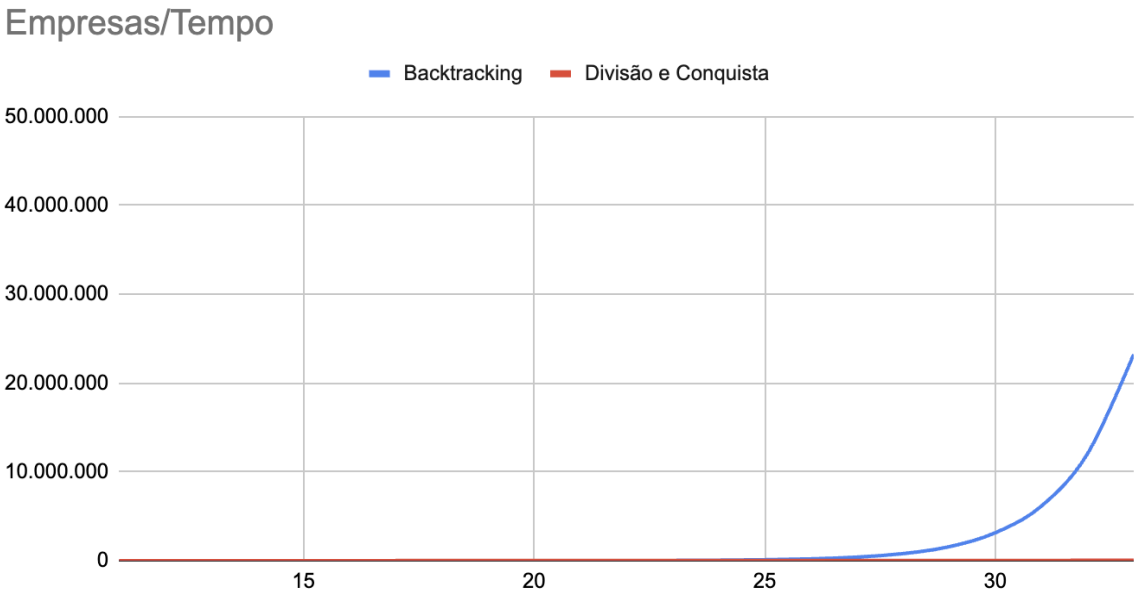
Os resultados obtidos a partir da execução de testes comparativos entre os algoritmos de Divisão e Conquista com memoização e Backtracking para resolver o problema de maximização de valor das ofertas revelam dados significativos sobre a eficiência e escalabilidade dessas abordagens em diferentes cenários.

Número de Empresas	Backtracking	Divisão e Conquista
10	0.000175	0.001673
11	0.000129	0.000497
12	0.000245	0.000864
13	0.000482	0.001016
14	0.000850	0.001345
15	0.001392	0.002032
16	0.000607	0.002418
17	0.001090	0.003346
18	0.001932	0.003639
19	0.003348	0.003975

20	0.007772	0.006831
21	0.012623	0.007726
22	0.025016	0.006257
23	0.049472	0.010349
24	0.098975	0.010740
25	0.197747	0.012061
26	0.395743	0.013120
27	0.789178	0.011348
28	1.566360	0.017066
29	3.138923	0.026012
30	6.141354	0.023905
31	12.039084	0.029358
32	23.218705	0.031570
33	44.563526	0.035229

Inicialmente, observamos que o algoritmo de Divisão e Conquista com memoização apresenta tempos médios de execução consistentemente menores em comparação com o algoritmo de Backtracking em todos os casos testados. Isso sugere que a estratégia de Divisão e Conquista, aliada à memoização para evitar recalculações, resulta em uma abordagem mais eficiente para resolver o problema em questão.

Ao analisar a evolução dos tempos de execução conforme o número de empresas aumenta, notamos uma diferença significativa entre os dois algoritmos. Enquanto o algoritmo de Backtracking demonstra um aumento exponencial no tempo de execução à medida que o problema se torna mais complexo, o algoritmo de Divisão e Conquista mantém um crescimento mais controlado, o que o torna mais escalável para problemas com um grande número de empresas e ofertas.



Com base no gráfico acima, onde os valores foram multiplicados proporcionalmente, a fins visuais e didáticos, podemos analisar e visualizar a complexidade dos algoritmos, enquanto o BackTracking se mostra exponencial, conforme a entrada. O algoritmo de divisão e conquista oferece uma menor complexidade e consequentemente uma linearidade no gráfico.

Uma análise mais aprofundada revela que a eficiência do algoritmo de Divisão e Conquista está intrinsecamente ligada à sua capacidade de resolver subproblemas de forma independente e reutilizar soluções já calculadas por meio da memoização. Isso reduz drasticamente o número de cálculos redundantes, resultando em um tempo de execução global mais rápido e um desempenho superior em comparação com o algoritmo de Backtracking.

Além disso, é importante destacar que o algoritmo de Divisão e Conquista, quando combinado com a técnica de memoização, não apenas oferece uma solução mais eficiente em termos de tempo de execução, mas também garante a obtenção da solução ótima para o problema de maximização de valor das ofertas dentro da capacidade total disponível. Isso é crucial em cenários onde a precisão e a qualidade da solução são requisitos essenciais.

Em resumo, com base nos resultados dos testes e na análise comparativa realizada, é recomendável utilizar o algoritmo de Divisão e Conquista com memoização para resolver o problema em questão. Essa abordagem oferece um desempenho superior, escalabilidade para problemas maiores e a garantia de obtenção da solução ótima, tornando-a a escolha mais eficiente e eficaz para resolver o problema de maximização de valor das ofertas.

3.2 Comparação da Qualidade dos Resultados

Ao se comparar os resultados obtidos através da execução do algoritmo de backtracking e do algoritmo de divisão e conquista, percebe-se que ambos os algoritmos convergiram para a mesma solução ótima, identificando o conjunto ideal de ofertas que resulta no maior valor possível para a empresa produtora de energia (28567). Essa convergência demonstra a corretude de ambas as implementações, garantindo que a solução encontrada é, de fato, a melhor possível.

No entanto, a semelhança nos resultados termina aqui. A diferença gritante reside no tempo de execução. Enquanto a Divisão e Conquista encontrou a solução ótima em míseros 0.10 segundos, o Backtracking demandou 46.39 segundos, uma diferença de duas ordens de grandeza.

Esses dados foram obtidos através dos dados de testes enviados pelo professor via Canvas no dia 20 de Junho de 2024

8000;E1;430;1043;E2;428;1188;E3;410;1565;E4;385;1333;E5;399;1214;E6;382;1498;E7;416;1540;E8;436;1172;E9;416;1386;E10;423;1097;E11;400;1463;E12;406;1353;E13;403;1568;E14;390;1228;E15;387;1542;E16;390;1206;E17;430;1175;E18;397;1492;E19;392;1293;E20;393;1533;E21;439;1149;E22;403;1277;E23;415;1624;E24;387;1280;E25;417;1330;

e outro conjunto de dados:

8000;E1;313;1496;E2;398;1768;E3;240;1210;E4;433;2327;E5;301;1263;E6;297;1499;E7;232;1209;E8;614;2342;E9;558;2983;E10;495;2259;E11;310;1381;E12;213;961;E13;213;1115;E14;346;1552;E15;385;2023;E16;240;1234;E17;483;2828;E18;487;2617;E19;709;2328;E20;358;1847;E21;467;2038;E22;363;2007;E23;279;1311;E24;589;3164;E25;476;2480

3.3 Discussões Sobre Vantagens e Desvantagens de Cada Abordagem

3.3.1 Divisão e Conquista:

A técnica de Divisão e Conquista, embora frequentemente associada à resolução de problemas matemáticos, demonstra grande versatilidade ao ser aplicada a desafios computacionais complexos, como o problema do leilão de energia. Sua essência reside em decompor um problema em subproblemas menores e independentes, resolvê-los recursivamente e combinar suas soluções para obter a resposta final.

A decomposição do problema em subproblemas menores e mais simples permite atacá-lo de forma estratégica, evitando a análise exaustiva de todas as combinações possíveis, como ocorre em algoritmos de força bruta. Essa abordagem reduz significativamente o espaço de busca e, consequentemente, o tempo de execução, especialmente em problemas com alto grau de complexidade.

A Divisão e Conquista se destaca pela escalabilidade. Diferentemente de algoritmos com complexidade exponencial, que se tornam inviáveis para conjuntos de dados maiores, a Divisão e Conquista mantém um crescimento do tempo de execução mais controlado, tornando-a adequada para lidar com cenários reais, onde o volume de dados pode ser considerável.

A implementação da Divisão e Conquista se beneficia enormemente da técnica de memoização. Ao armazenar e reutilizar resultados de subproblemas já calculados, a memoização elimina cálculos redundantes, impulsionando a eficiência do algoritmo, especialmente em problemas com subproblemas sobrepostos.

Sua natureza independente dos subproblemas na Divisão e Conquista abre portas para a paralelização. Em arquiteturas de hardware com múltiplos núcleos de processamento, os subproblemas podem ser resolvidos simultaneamente, reduzindo drasticamente o tempo total de execução e potencializando a eficiência da solução.

Porém, todos os algoritmos têm também seus pontos negativos. No caso a Divisão e Conquista exige uma análise mais profunda do problema para identificar a estrutura recursiva adequada e garantir que a decomposição em subproblemas seja efetivamente vantajosa. A implementação, por sua vez, pode ser mais complexa do que algoritmos iterativos, demandando atenção especial para evitar erros de lógica na recursão.

Sem contar também que a Divisão e Conquista, por natureza, se baseia em chamadas recursivas, o que pode gerar overhead de gerenciamento de memória, especialmente em problemas com profundidade de recursão muito grande. Em cenários com recursos de memória limitados, essa sobrecarga pode impactar o desempenho do algoritmo.

4 CONSIDERAÇÕES FINAIS