Nome: Wayne Nascimento Souza
Matrícula: 0016183

Paradigma da Divisão e Conquista

A estratégia de Divisão e Conquista é amplamente utilizada na resolução de problemas computacionais. De acordo com Feofiloff (2020), essa abordagem consiste em dividir uma instância do problema em duas ou mais instâncias menores, resolver cada uma delas de forma independente e, em seguida, combinar as soluções obtidas para produzir uma solução para a instância original. Esse processo é implementado por meio de chamadas recursivas, sendo conhecido como a fase da conquista.

A etapa de divisão divide o problema original em subproblemas menores, enquanto a etapa de conquista resolve cada subproblema recursivamente. Por fim, na etapa de combinação, as soluções encontradas são combinadas para gerar o resultado final procurado.

A técnica de Divisão e Conquista apresenta diversas vantagens, segundo Toffolo e Carvalho (2011) esta técnica é excelente para buscar algoritmos eficientes, muitas vezes alcançando uma complexidade logarítmica. Além disso, é facilmente paralelizável na fase de conquista, o que permite explorar o poder de processamento de sistemas distribuídos.

No entanto, essa abordagem também possui algumas desvantagens. A recursão é uma delas, pois pode consumir recursos computacionais significativos, e também, a seleção adequada dos casos bases para a recursão pode ser um desafio.

A estratégia de Divisão e Conquista pode ser aplicada com sucesso em diferentes situações. Três condições indicam sua utilização adequada: é possível decompor uma instância em subinstâncias, a combinação dos resultados é eficiente e as sub-instâncias têm tamanhos semelhantes.

Existem dois casos genéricos em que a abordagem de Divisão e Conquista é adequada. De acordo com Figueiredo (2008) o primeiro é quando um grupo de operações é correlacionado ou repetido, como é o caso da multiplicação de matrizes. O segundo é quando uma decisão deve ser tomada e, uma vez tomada, o problema é dividido em peças distintas, sendo a abordagem de Divisão e Conquista interessante quando algumas dessas peças se tornam irrelevantes.

Conforme Manley (2012), dentre os diferentes tipos específicos de algoritmos de Divisão e Conquista, destacam-se o Binary Search, Merge Sort, Quick Sort, Multiplicação de Matrizes e Fourier Transform Algorithm. Esses algoritmos dividem os problemas em subproblemas para resolvê-los de maneira eficiente.

Modelagem da solução

Para realizar a modelagem de uma possível solução para o problema, primeiramente foi definido com base no enunciado de como seriam a entrada e a saída. A entrada ficou definida como uma lista de pontos, onde cada ponto é a representação de um par ordenado (x, y). Já a saída, retorna os dois pontos mais próximos e a distância entre eles. Com isso, foi possível determinar o formato dos candidatos e desenvolver um algoritmo baseado no paradigma de Divisão e Conquista para solucionar o problema para identificar os dois pontos mais próximos em um conjunto de pontos no plano 2D.

A aplicação do paradigma de Divisão e Conquista como parte da solução possui as seguintes etapas: verificação do tamanho da lista de pontos, uso da recursão para resolver cada metade do problema, identificação da menor distância entre os pontos nas duas metades, identificação da menor distância entre os pontos que estão em metades diferentes e retornar os dois pontos mais próximos juntamente com a distância entre eles.

Na etapa de verificação do tamanho da lista de pontos, temos o caso base em que se a lista estiver vazia ou contiver apenas um ponto não é possível calcular os dois pontos mais próximos, então o retorno será None com uma distância infinita. Caso a lista contenha apenas dois pontos, eles próprios serão a resposta juntamente com a distância entre eles. Por fim, se a lista for maior que dois pontos ela será dividida em duas partes com base na posição média e será resolvido com a chamada recursiva do algoritmo.

A próxima etapa é a de utilização da recursão para resolver cada metade do problema, que será obtido os pontos mais próximos e a distância mínima para cada metade.

Já na etapa da identificação da menor distância entre os pontos nas duas metades, é executada uma comparação entre as distâncias mínimas obtidas das duas metades e, permanece com o par de pontos mas faz a atribuição da distância mínima correspondente.

Na etapa da identificação da menor distância entre os pontos que estão em metades diferentes, é calculada a coordenada média entre o último ponto que está localizado na metade da esquerda com o primeiro ponto da metade da direita. Logo após, é criado uma faixa de pontos que estejam a uma distância menor que a distância mínima da coordenada e é aplicada uma ordenação nesta faixa pelo valor da coordenada y. Finalmente, percorre a faixa de pontos e compara as distâncias entre cada par de pontos adjacentes e caso uma distância menor seja encontrada, atualiza o par de pontos e a distância correspondente.

Com isso, é possível retornar o par de pontos mais próximos e a distância mínima encontrada. Lembrando que, existe uma função auxiliar para calcular a distância euclidiana entre dois pontos em um plano 2D usando exatamente a fórmula da distância entre dois pontos que segundo BRUNS (2023) a fórmula abaixo pode ser usada para calcular.

$$d = \sqrt{|y2 - y1|^2 + |x2 - x1|^2}$$

Com base nessa modelagem, foi possível resolver o problema para encontrar os dois pontos mais próximos em um conjunto de pontos no plano 2D, utilizando a técnica de Divisão e Conquista.

Análise do custo do algoritmo

Para analisar o custo do algoritmo será dividido em trechos de códigos, apontando a linha inicial e a final e consequentemente o custo associado.

- 05-06: é a função usada para calcular a distância entre dois pontos e tem custo constante (1).
- 11-13: apenas atribuição, comparação e retorno, logo o custo é constante (1).
- 14-15: retorno dos valores (1).
- 16-23: declarações e atribuições e, chamada recursiva da função nas metades da lista (1 + T(n/2) + T(n/2)).
- 24-30: comparações e atribuições (1).
- 31-33: definição do ponto médio e inicialização de uma lista vazia (1).
- 34-36: iteração sobre a lista de pontos e inclusão do ponto que acordo com a verificação na outra lista (n).
- 37: função para ordenar os elementos da lista, de acordo com o site Prepbytes (2023) o custo é logaritmo e no caso de já estar ordenado será o próprio n (nlogn).
- 38-44: iteração sobre a lista de faixas usando duas estruturas de repetição aninhadas para encontrar o par de pontos mais próximo dentro da faixa (n * n).
- 45: retorno da função (1).

A soma de todos esses custos resulta na equação recorrência abaixo:

$T(n) = 2T(\frac{n}{2}) + O(n) + O(n.logn) + O(n^2)$

Como o n^2 é a complexidade assintótica dominante dentre estas, podemos usar a equação simplificada $T(n) = 2T(\frac{n}{2}) + n^2$.

Ao resolver a equação de recorrência com o Teorema Mestre, temos:

$a = 2$

$b = 2$

$f(n) = n^2$

$n^{log_b^a} = n$

É necessário provar que $f(n) = \Omega(n^{(log_b^a + k)})$ para a constante $k > 0$,

$k = 1$

$n^2 = n^2$

Agora, é preciso verificar se a condição $af(\frac{n}{b}) \leq cf(n)$ é verdadeira.

$af(\frac{n}{b}) = 2 . (\frac{n}{b})^2 = \frac{n^2}{2}$

$cf(n) = c . n^2$

Para encontrar o valor de c que satisfaça a inequação $af(\frac{n}{b}) \leq cf(n)$

$\frac{n^2}{2} \leq c . n^2$

Ao dividir ambos os lados por n^2, temos

$\frac{1}{2} \leq c$

Com isso, ao escolher um valor para c tal que c seja maior ou igual a $\frac{1}{2}$, é possível satisfazer a condição $af(\frac{n}{b}) \leq cf(n)$ e aplicar o caso 3 do Teorema Mestre em que $T(n) = \Theta(f(n))$.

∴ A solução da equação de recorrência $T(n) = 2T(\frac{n}{2}) + n^2$ usando o Teorema Mestre é $\Theta(n^2)$.

Referências

FEOFILOFF, Paulo. Divisão e conquista, 2020. Disponível em: https://www.ime.usp.br/~pf/analise_de_algoritmos/aulas/divide-and-conquer.html. Acesso em: 08 de mai. de 2023.

TOFFOLO, Túlio; CARVALHO, Marco Antônio. Divisão e conquista, 2011. Disponível em: http://www3.decom.ufop.br/toffolo/site_media/uploads/2011-1/bcc402/slides/08._divisao_e_conquista.pdf. Acesso em: 08 de mai. de 2023.

FIGUEIREDO, Jorge. Análise e técnicas de algoritmos, 2008. Disponível em: http://www.dsc.ufcg.edu.br/~abrantes/CursosAnteriores/ATAL051/DivCong.pdf. Acesso em: 10 de mai. de 2023.

MANLEY, Melissa. Divide and conquer paradigm, 2012. Disponível em: http://www.csun.edu/~mam78887/MAMANLEY.pdf. Acesso em: 15 de mai. de 2023.

BRUNS, Kurt. Distance between two points 2D formula, 2023. Disponível em: https://wumbo.net/formulas/distance-between-two-points-2d. Acesso em: 01 de jun. de 2023.

PREPBYTES. Sort function in Python, 2023. Disponível em: https://www.prepbytes.com/blog/python/sort-function-in-python/. Acesso em: 18 jun. de 2023.