Documentação do Algoritmo para o Cálculo do Maior Subarray Contíguo

Abordagem

O código fornecido implementa um algoritmo para calcular o Maior Subarray Contíguo em uma lista (array) de números inteiros. A abordagem usada é a técnica de "Divisão e Conquista", que divide o problema em subproblemas menores, resolve esses subproblemas e, em seguida, combina as soluções para obter a resposta final.

Funções Principais

`set_sum(init, final, interval, arr)`

Esta função calcula a soma máxima de um subarray dentro de um intervalo especificado. Ela percorre o array a partir do índice `init` até o índice `final - 1` com um passo especificado por `interval`. Durante a iteração, a função mantém um registro da soma atual (`current_sum`) e atualiza a soma máxima (`max_sum`) conforme necessário. A função retorna a soma máxima encontrada no subarray especificado.

`max_subarray_divide_conquer(arr, low, high)`

Esta é a função principal que implementa o algoritmo de Divisão e Conquista para encontrar o Maior Subarray Contíguo. Ela recebe como entrada o array `arr`, bem como os índices `low` e `high` que definem o intervalo atual que está sendo considerado. A função se divide em três etapas:

1. Caso Base: Se `low` e `high` são iguais, a função retorna o valor do único elemento no array, que é o caso base da recursão.
2. Divisão: A função divide o intervalo em duas partes, calculando o Maior Subarray Contíguo na metade esquerda (`left_max`) e na metade direita (`right_max`) do intervalo.
3. Combinação: A função calcula a soma máxima que cruza o ponto médio do intervalo (`cross_max`) usando a função `set_sum`. Em seguida, ela retorna o máximo entre `left_max`, `right_max` e `cross_max`, que representa o Maior Subarray Contíguo no intervalo considerado.

`find_max_subarray(arr)`

Esta é a função de entrada que chama `max_subarray_divide_conquer` com os parâmetros iniciais e retorna a resposta final, que é o Maior Subarray Contíguo em todo o array `arr`.