Resumo Heurísticas Construtivas

Pedro Tavares de Carvalho

6 de janeiro de 2022

Resumo

Nesse documento resumirei os capítulo 15, 16 e 35 do Cormen [**?**] além do capítulo 8 de Arts[**?**].

1 Capítulo 15

Nesse capítulo exploraremos um pouco sobre programação dinâmica, descorrendo sobre alguns algoritmos, incluindo *Rod Cutting* e multiplicações de matrizes em cadeia. No final, também discutimos sobre as partes principais de programação dinâmica e seus conceitos mais explorados.

A principal caracterização do uso comum programação dinâmica é a do compartilhamento de subproblemas ao se dividir o problema principal, o que normalmente se gera repetição de trabalho, que pode ser minimizado ao se utilizar desse método, que consiste, em base, em:

1. Caracterizar a estrutura de uma solução ótima
2. Definir uma solução ótima recursivamente
3. Computar o valor de uma solução ótima, geralmente de forma *bottom up*
4. Contruir uma solução ótima a partir da informação computada anteriormente

1.1 15.1

Nessa sessão, exploraremos o conceito do problema de *rod cutting*.

Definição 1.1 Problema *Rod Cutting*

A partir de uma barra de metal de tamanho r, podemos fazer cortes discretos na mesma. Cada tamanho de barra é vendido por um valor diferente conhecido. O objetivo do problema é maximizar o valor total da barra a partir dos cortes feitos nela.

O problema pode ser resolvido de forma recursiva, se imaginarmos cada corte como dois subproblemas, onde temos que maximizar os cortes na barra da esqueda e na barra da direita. Um algoritmo recursivo para resolver esse problema é escrito a seguir.

Ao construirmos a árvore de chamads do Algoritmo 1, podemos perceber que subproblemas são resolvidos diversas vezes, o que gera um trabalho desnecessário, aumentando a complexidade do algoritmo, que pode ser deduzida em $O(2^n)$.

Podemos modificar esse algoritmo para aplicar a programação dinâmica no mesmo, tanto de forma *bottom up* quanto *top down*. Para a foma *top down*, utilizamos o conceito de memoização para guardar os resultados dos subproblemas, consultando se esse resultado já existe antes de realizar realmente a operação. Para a forma *bottom up*, utilizamos uma

Algoritmo 1 Cut-Rod(p, n)

if then return 0
end if
$q \leftarrow -\infty$
for $i \leftarrow 1$ **to** n **do**
    $q \leftarrow \max(q, p[i] + \text{Cut-Rod}(p, n - i))$
end for

tabela para armazenar os resultados parciais dos subproblemas antes de fazer o problema principal.

Em ambas as maneiras, utilizamos fazemos um *tradeoff* entre memória e tempo de execução, armazenando informação a fim de diminuir o tempo total de execução do algoritmo.

A complexidade de tempo de ambos os algoritmos, sendo ele memoizado ou por tabela, resulta em $O(n^2)$, melhorando muito da solução recursiva ingênua.

1.1.1 Reconstrução de solução

Referências

[1] Cormen, Thomas: *Introduction to Algorithms*. MIT Press, 3ª edição, 2009.