Modelagem de problemas de relação de recorrência linear usando matrizes

Enoque Alves de Castro Neto[1], **Wladimir Araújo Tavares**[1]

[1]Universidade Federal do Ceará (UFC) – Campus Quixadá

{enoquealvesufc, wladimirufc}@gmail.com

Abstract. *In this paper, a linear recurrence modeling technique with matrices and rapid matrix exponentiation is presented, as well as an efficient and n-th term of a linear recurrence. It will also contain an efficient way to get the result of the graph problem "How many paths of size k exists from a vertex to a vertex v".*

Resumo. *Neste artigo, é apresentado a técnica de modelagem de recorrência linear com matrizes e exponenciação matricial rápida, conseguindo assim obter de forma eficiente o n-ésimo termo de uma recorrência linear. Também conterá uma forma eficiente de como obter o resultado do problema de grafos "Quantos caminhos de tamanho k existe de um vértice u para um vértice v".*

1. Introdução

As matrizes são estruturas matemáticas com várias aplicações na computação, entre elas temos: Modelagem de grafos e relações, modelagem de sistema de equações lineares, matrizes de rotação em computação gráfica, modelagem de recorrência lineares e de problemas de programação dinâmica. Neste artigo, apresentaremos um problemas envolvendo recorrência linear e um problema envolvendo grafos que podem ser resolvidos usando exponenciação matricial, que são eles, respectivamente:

1. Encontrar o n-ésimo termo de uma recorrência linear com complexidade de $\mathcal{O}(log\ n)$, onde k é a ordem da relação de recorrência.
2. Contar o números de caminhos de tamanho L existentes de um vértice u para um vértice v em grafo G com a complexidade $\mathcal{O}(log\ L)$.

   Os dois problemas acima podem ser resolvidos em dois passos:

1. Encontrar a matriz de transformação T da relação de recorrência linear do problema.
2. Calcular $T^n_{k\times k}$ com a complexidade $\mathcal{O}(log\ n)$ para obter n-ésimo termo da relação de recorrência.

2. Relação de Recorrência Linear

Uma relação de recorrência é uma função que permite calcular n-ésimo termo a partir dos termos anteriores e casos mais simples chamados de caso base. Por exemplo:

$$g(n) = \begin{cases} n^2 g(n-1) & \text{se } n \geq 1 \\ 1 & \text{se } n = 1 \end{cases} \tag{1}$$

Uma relação de recorrência linear é uma equação que define o n-ésimo termo de uma sequência em termos dos k termos anteriores da sequência. A relação de recorrência linear é da seguinte forma:

$$g(n) = c_1 g(n-1) + c_2 g(n-2) + \ldots + c_k g(n-k) \tag{2}$$

Por exemplo, a sequência de Fibonacci é um exemplo de relação de recorrência linear.

$$g(n) = \begin{cases} g(n-1) + g(n-2) & \text{se } n \geq 3 \\ 1 & \text{se } n = 1 \vee n = 2 \end{cases} \tag{3}$$

O processo de modelagem de uma relação de recorrência como uma matriz de transformação[Fuadi 2011] pode ser realizado em quatro passos:

1. Determinar o número de termos que n-ésimo termo depende.
2. Determinar o vetor f_1.
3. Determina a matriz de transformação T.
4. Encontrar o vetor f_n.

2.1. Determinar o número de termos que n-ésimo termo depende

O número de termos que n-ésimo termo depende é o menor inteiro k tal que $g(n)$ dependa de $g(n-c)$, para todo $c \leq k$. Para a seguinte relação de recorrência linear, temos que $k = 4$:

$$g(n) = 2g(n-2) + g(n-4)$$

2.2. Determinar o vetor f_1, os casos base

Após determinar o valor de k, precisamos encontrar os casos bases para a relação de recorrência. Para o exemplo do Problema Fibonacci, o vetor f_1 fica:

$$\begin{bmatrix} g(1) \\ g(2) \end{bmatrix}$$

No caso geral, o vetor f_1 tem a dimensão $k \times 1$. Ficando da seguinte forma:

$$\begin{bmatrix} g(1) \\ g(2) \\ \vdots \\ g(k) \end{bmatrix}$$

2.3. Determinar a matriz de transformação

A obtenção da matriz de transformação é o passo mais importante para se obter sucesso nesse método. Ela pode ser obtida resolvendo a seguinte equação:

$$T f_n = f_{n+1} \tag{4}$$

Suponha que $g(n) = \sum_{j=1}^{k} c_j g(n-j)$. A partir disso podemos definir a matriz T da seguinte forma:

$$T = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots & 0 \\ 0 & 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\ c_k & c_{k-1} & c_{k-2} & c_{k-3} & \cdots & c_1 \end{bmatrix}$$

A matriz de transformação para o Problema de Fibonacci é:

$$T = \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}$$

2.4. Encontrando f_n

O cálculo do vetor f_n pode ser obtido através da seguinte equação:

$$f_n = T^{n-1} f_1 \tag{5}$$

Note que este método pode ser usado para modelar relações de recorrências mais complexas como $g(n) = g(n-1) + g(n-2) + n^2$:

$$\begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 2 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} g(n-1) \\ g(n-2) \\ n^2 \\ n \\ 1 \end{bmatrix} = \begin{bmatrix} g(n) \\ g(n-1) \\ (n+1)^2 \\ n+1 \\ 1 \end{bmatrix}$$

3. Exponenciação de matrizes

O algoritmo mais simples de exponenciação matricial tem complexidade $O(n)$, onde n é o expoente da matriz. Podemos adaptar o algoritmo de divisão e conquista de exponenciação rápida para o cálculo da exponenciação matricial com complexidade $\mathcal{O}(log\ n)$. Considere que I_n denota a matriz identidade $n \times n$.

Algoritmo 1: EXPMAT

```
Entrada: T, k
Saída: M
início
    M ← I_n
    repita
        se k%2 == 1 então
            M ← M × T
        fim
        T ← T × T
        k ← k/2
    até k == 0;
fim
retorna resposta
```

Esse algoritmo se aproveita da seguinte propriedade de exponenciação:

$$M^n = \begin{cases} I_n & \text{se n = 0} \\ (M^{n/2})^2 & \text{se n é par} \\ (M^{\lfloor n/2 \rfloor})^2 M & \text{se n é ímpar} \end{cases} \tag{6}$$

Assim, fazendo apenas $log\ n$ multiplicações matriciais. Considerando constante, a complexidade de realizar a multiplicação matricial, a complexidade final da exponenciação matricial é $\mathcal{O}(log\ n)$.

4. Problema Ônibus

O problema consiste em contar de quantas maneiras diferentes podemos formar uma fila de tamanho n de uma empresa que possui ônibus e micro-ônibus. Cada ônibus e micro-ônibus tem tamanho 1 e 2, respectivamente. Cada ônibus e cada micro-onibus possui k e l cores disponíveis, respectivamente. A pergunta final do problema é: Dado dois inteiros k e l, onde k é o número de cores disponíveis para os micro-ônibus e l o número disponíveis para os ônibus, você deve responder de quantas maneiras diferentes é possível posicionar ônibus e micro-ônibus no estacionamento em uma fila de tamanho n. [1]

A função de recorrência presente neste problema $g(n) = kg(n-1) + lg(n-2)$. Aplicando as técnicas vistas neste artigo, obtemos a seguinte seguinte matriz T:

$$T = \begin{bmatrix} 0 & 1 \\ l & k \end{bmatrix}$$

E o seu vetor f_1 é:

$$\begin{bmatrix} k \\ (k^2 + l) \end{bmatrix}$$

Agora, basta aplicar o algoritmo de exponenciação rápida, para achar o vetor f_n, obtendo a resolução desse problema na primeira linha deste vetor.

Para um n, k e l = 5, a matriz T^4 é:

$$T^4 = \begin{bmatrix} 150 & 175 \\ 875 & 1025 \end{bmatrix}$$

O seu vetor f_1 é:

$$\begin{bmatrix} 5 \\ 30 \end{bmatrix}$$

Com isso, temos que:

$$T^4 f_1 = \begin{bmatrix} 6000 \\ 35125 \end{bmatrix}$$

Onde $g(5)$ é exatamente 6000.

Como a matriz tem um tamanho constante, temos que sua multiplicação é dada como $O(1)$, então a complexidade final ficará somente a complexidade da exponenciação, logo sua complexidade é $O(logn)$. Uma outra forma de resolver esse problema, seria usando programação dinâmica, porém sua complexidade seria $O(n)$.

[1]Para mais informações sobre o problema, acesse `https://www.urionlinejudge.com.br/judge/pt/problems/view/1474`

5. Problema Teletransporte

Um outro problema que pode ser respondido usando exponenciação matricial é um problema de grafos que tem como pergunta ”Quantos caminhos de tamanho K existem do vértice u ao vértice v?”.[2] Observe que o número de caminhos caminhos de tamanho 1 entre o vértice u e o vertice v será igual a 1 se o vértice u e v são adjacentes e 0, caso contrário. Logo, a matriz de transição será:

$$M = \begin{bmatrix} m_{11} & m_{12} & \dots & m_{1n} \\ m_{21} & m_{22} & \dots & m_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ m_{n1} & m_{n2} & \dots & m_{nn} \end{bmatrix}$$

onde

$$m_{i,j} = \begin{cases} 1 & \text{se i e j são adjacentes} \\ 0 & \text{caso contrário} \end{cases} \tag{7}$$

Para calcular a quantidade de caminhos de tamanho 2 do vértice u até o vértice v, basta multiplicar a matriz M por ela mesma. Temos a seguinte relação:

$$M[u][v] = \sum_{w \in V} M[u][w] + M[k][w]$$

Essa recorrência nos diz que para obter os números de caminhos de tamanho $k+1$, basta sabermos a matriz que contem quantos caminhos de tamanho k e multiplicarmos com ela mesma. Considerando a matriz que contem quantos caminhos de tamanho 1 como caso base, basta usar o algoritmo de exponenciação matricial mostrado neste artigo e consultar o valor $M[u][v]$.

6. Conclusão

Geralmente, para se resolver uma função de recorrência linear é usado técnicas como memorização e programação dinâmica e essas técnicas normalmente tem uma complexidade maior ou igual a $O(n)$. Neste artigo, vimos como modelar função de recorrência linear com matrizes e vimos também que se a matriz possui tamanho constante, a resolução do problema se torna $O(logn)$, ficando assim melhor que todas essas as outras técnicas.

Referências

Fuadi, A. (2011). Solving linear recurrence for programming contest.

[2]Para mais informações sobre o problema, acesse `https://www.urionlinejudge.com.br/judge/pt/problems/view/1713`