COM150 – Fundamentos Matemáticos para Computação

Prof. Dr. Marcos Tadeu de Oliveira Pimenta

Texto de revisão

Neste breve texto, responderemos às dúvidas levantadas pelos estudantes da disciplina Fundamentos Matemáticos para a Computação, visando a sua devida preparação para a avaliação final. Entre as dúvidas, algumas foram demasiadamente genéricas, de forma que fica difícil levantar pontos relevantes aqui, sendo mais aconselhável, nesses casos, que se faça uma revisão de todo o material da semana correspondente. Entretanto, responderemos abaixo dúvidas versando sobre diversos assuntos, apresentando exemplos ilustrativos que ajudem na sua compreensão.

Dúvida: Escrever algoritmos recursivos e o uso da indução forte.

Resposta: A indução forte, ou indução completa, difere da indução fraca pela presença de uma hipótese de indução mais forte. De fato, no Princípio de Indução Forte, para provar a validade de uma sentença aberta P(n), para todo $n \geq 1$, temos como hipóteses que:

i. P(1) é verdadeira;
ii. P(r) é verdadeira, para todo $1 \leq r \leq k \Longrightarrow$ P(k+1) é verdadeira.

Note que a hipótese ii. é mais forte do que aquela correspondente ao Princípio de Indução Fraca. Logo, em geral, é mais fácil verificar a validade da hipótese ii. do Princípio da Indução Forte do que aquela correspondente ao Princípio da Indução Fraca. Uma importante observação é que o número 1 acima poderia ser substituído por qualquer $n_0 \in \mathbb{N}$ fixo.

Exemplo: Provemos que $\forall n \in \mathbb{N}, n \geq 2$, n é primo ou então é o produto de números primos.

De fato, utilizemos o Princípio da Indução Forte e consideremos a sentença aberta.

$P(n)$: n é primo ou então é o produto de números primos.

Note que:

- P(2) é verdadeira, pois 2 é um número primo.
- Fixado $k \in \mathbb{N}$, suponhamos que P(r) seja verdadeira para todo $2 \leq r \leq k$, ou seja, suponhamos que todo número $2 \leq r \leq k$ seja primo, ou o produto de primos. Consideremos agora o número natural k+1. Se k+1 for primo, então P(k+1) será verdadeira. Caso contrário, por definição, podemos escrever $k + 1 = a.b$, em que $2 \leq a, b \leq k$ e $a, b \in \mathbb{N}$. Assim, como $2 \leq a, b \leq k$, pela hipótese de indução, cada um deles é ou primo, ou o produto de números primos. Em qualquer das situações, podemos afirmar que $k + 1 = a.b$ será o produto de números primos e, portanto, P(k+1) se verifica.

Dessa forma, pelo Princípio da Indução Forte, podemos afirmar que $\forall n \in \mathbb{N}, n \geq 2$, n é primo ou então é o produto de números primos.

Os algoritmos recursivos são alternativas para resolver problemas que, quando solucionados com uma técnica não recursiva, podem ser bastante complexos. Em geral, envolvem estruturas do tipo **se, então, senão**. Vejamos um exemplo de algoritmo recursivo escrito em pseudocódigo para calcular a n-ésima entrada da seguinte sequência definida recursivamente:

1. T(1) = 1
2. T(n) = T(n-1) + 3, para todo $n \geq 2$.

Função T(n), n inteiro

Início

Se n = 1, então

T(n):=1

Senão

T(n) := T(n-1)+3

Fim

Dúvida: Como a fórmula para solução de recorrência linear de primeira ordem é resolvida?

Resposta: A fórmula de recorrência linear visa apresentar uma fórmula fechada, ou seja, que expresse uma sequência S(n) explicitamente em termos de n, não dependendo dos valores S(k) para $k \leq n$. Conforme visto no texto-base, se S(n) for uma relação de recorrência linear de primeira ordem com coeficientes constantes, então ela se escreve como:

S(n) = c.S(n-1) + g(n),

em que c é uma constante e g é uma função envolvendo n. Ainda, pelo texto-base, usando a técnica de expandir, supor e verificar, pode-se provar que a fórmula fechada para S(n) é dada por:

$$S(n) = c^{n-1}S(1) + \sum_{i=2}^{n} c^{n-i}g(i).$$

Possivelmente a notação de somatório possa tornar um pouco difícil de compreender tal fórmula, entretanto, é apenas uma notação utilizada para fazer a expressão “ficar menor”. De fato, sem utilizar a notação de somatório, tal expressão se escreve como:

$$S(n) = c^{n-1}S(1) + \ c^{n-2}g(2) + c^{n-3}g(3) + \cdots + c^{0}g(n)$$

Vejamos a utilização dessa fórmula através de um exemplo.

Exemplo: Encontre uma solução de forma fechada para a relação de recorrência T, em que:

1. T(1) = 1.
2. T(n) = T(n-1) + 3, para $n \geq 2$.

Note primeiramente que T é uma relação de recorrência linear e de primeira ordem. Ainda, para esse exemplo particular, note que c = 1 e g(n) = 3, para todo $n \geq 2$. Dessa forma, para encontrarmos uma fórmula fechada para T(n), basta aplicarmos a fórmula acima. Assim,

$$T(n) = c^{n-1}T(1) + \sum_{i=2}^{n} c^{n-i}g(i)$$

$$= \ 1^{n-1}.T(1) + \ 1^{n-2}.3 + 1^{n-3}.3 + \cdots + 1^{0}.3$$

$$= T(1) + 3 + 3 + \cdots + 3$$

$$= 1 + (n-1).3$$

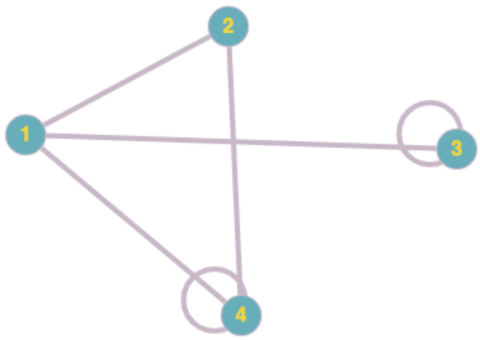
$$= 3n - 2.$$

Logo, na resolução, lembre-se de que o somatório é apenas uma notação para simplificar a escrita de uma expressão que envolve a soma de muitas parcelas.

Dúvida: Minha dúvida é entender operações matriciais: elementos de uma matriz, matrizes quadradas, matriz simétrica, operações matriciais.

Resposta: As matrizes são ferramentas básicas na matemática e suas aplicações, consistindo em uma tabela formada, em geral, por números reais. Como vimos ao longo dos estudos dos grafos não direcionados, existe uma relação intrínseca entre esses e matrizes simétricas. Matrizes simétricas são matrizes $[a_{i,j}]_{1\leq i,j\leq n}$, para as quais o elemento $a_{i,j}$ é igual ao elemento $a_{j,i}$, ou seja, o elemento que ocupa a i-ésima linha e j-ésima coluna é igual ao que ocupa a j-ésima linha e i-ésima coluna. De fato, a representação de grafos não orientados com n nós, através de matrizes n x n, dá-se inserindo no elemento $a_{i,j}$, o número de arcos entre os nós i e j do grafo. Isso forma o que chamamos de **matriz de adjacências associada ao grafo.**

Exemplo: Considere o seguinte grafo não orientado:



Note que a matriz de adjacências será 4 x 4, haja vista que temos quatro nós. Logo, a matriz será:

$$\begin{bmatrix} 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 \end{bmatrix}.$$

As operações matriciais usuais são a soma e produto de matrizes, o que é amplamente coberto por textos básicos do Ensino Médio. Porém, quando se trata de matrizes booleanas, ou seja, de matrizes cujos elementos sejam apenas 0’s ou 1’s, definem-se novas operações, como a conjunção “∧” ou a disjunção “∨” de matrizes.

Dadas duas matrizes booleanas $A = \left[a_{i,j}\right]_{1\leq i,j\leq n}$ e $B = \left[b_{i,j}\right]_{1\leq i,j\leq n}$, definimos

$$A \wedge B = \left[a_{i,j} \wedge b_{i,j}\right]_{1\leq i,j\leq n},$$

ou seja, a matriz cujas entradas são as conjunções de $a_{i,j}$ e $b_{i,j}$, lembrando que:

p	q	p ∧ q
1	1	1
1	0	0
0	1	0
0	0	0

De maneira análoga, define-se:

$$A \vee B = \left[a_{i,j} \vee b_{i,j}\right]_{1\leq i,j\leq n},$$

ou seja, a matriz cujas entradas são as disjunções de $a_{i,j}$ e $b_{i,j}$, lembrando que:

p	q	p ∨ q
1	1	1
1	0	1
0	1	1
0	0	0

Exemplo: Considere as matrizes booleanas:

$$A = \begin{bmatrix} 1 & 1 & 0 \\ 0 & 0 & 0 \\ 1 & 1 & 0 \end{bmatrix} \; e \; B = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}.$$

Calcule $A \wedge B$ e $A \vee B$.

Note que, pela definição das operações de conjunção e disjunção de matrizes,

$$A \wedge B = \begin{bmatrix} 1 \wedge 0 & 1 \wedge 0 & 0 \wedge 0 \\ 0 \wedge 1 & 0 \wedge 0 & 0 \wedge 1 \\ 1 \wedge 1 & 1 \wedge 0 & 0 \wedge 0 \end{bmatrix} = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 0 & 0 \\ 1 & 0 & 0 \end{bmatrix}.$$

Por outro lado,

$$A \vee B = \begin{bmatrix} 1 \vee 0 & 1 \vee 0 & 0 \vee 0 \\ 0 \vee 1 & 0 \vee 0 & 0 \vee 1 \\ 1 \vee 1 & 1 \vee 0 & 0 \vee 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}.$$