Indução Matemática (Prof. Flávio L. C. de Moura)

Considere o seguinte problema:

Qual é o resultado da soma $1 + 2 + 3 + \ldots + 98 + 99 + 100$?

Conta-se que uma professora de Matemática, na tentativa de obter silêncio na classe e ocupar seus alunos, propôs que os mesmos calculassem o resultado da soma dos números naturais de 1 a 100. Entre seus alunos estava o alemão Karl Friedrich Gauss (1777-1855) com então 8 anos de idade, que resolveu o problema em poucos segundos! Como Gauss fez isto?

Observe o seguinte padrão:

$$\begin{array}{ccccccccccccc} 1 & + & 2 & + & 3 & + & \ldots & + & 98 & + & 99 & + & 100 \\ 100 & + & 99 & + & 98 & + & \ldots & + & 3 & + & 2 & + & 1 \\ \hline 101 & + & 101 & + & 101 & + & \ldots & + & 101 & + & 101 & + & 101 \end{array}$$

Assim temos 100 parcelas iguais a 101, ou seja, o resultado da soma é igual a $\frac{100 * 101}{2} = 5050$, já que a soma desejada está sendo computada duas vezes.

Será possível repetir este padrão para computar outras somas semelhantes? É fácil ver que sim. Podemos então generalizar este raciocínio? Isto é, para um dado número natural n, qual é o valor da soma $1 + 2 + 3 + \ldots + n$? Como veremos, a resposta é sim, e esta generalização pode ser garantida pelo **princípio de indução**. O princípio de indução nos permite provar propriedades sobre os números naturais $\mathbb{N} = \{1, 2, 3, \ldots\}$, e a partir de alguns exemplos, semelhantes ao dado acima, poderíamos conjecturar que

$$1 + 2 + \ldots + n = \frac{n(n+1)}{2} \tag{1}$$

Mas será que esta equação é verdadeira? Ou seja, como garantir que a equação (1) é correta para qualquer natural n? Para isto utilizaremos o chamado Princípio da Indução Matemática (PIM).

Para provar que uma propriedade M sobre os naturais é válida precisamos mostrar duas coisas:

1. **Base da Indução (BI)**: O número 1 satisfaz a propriedade, isto é, precisamos de uma prova de $M(1)$;

2. **Passo Indutivo (PI)**: Assumindo que o natural n satisfaça a propriedade M, precisamos provar que $n + 1$ também satisfaz a propriedade, isto é, precisamos construir uma prova de $M(n) \to M(n+1)$.

Este princípio é também conhecido com o *princípio da indução finita*.

O Princípio da Indução Finita: Primeira Forma

Formalmente o princípio da indução finita pode ser apresentado da seguinte forma:

Definição 1. *Seja $P(n)$ uma propriedade associada aos números naturais. Em outras palavras, P é um predicado unário cujo argumento é um número natural, e suponhamos que:*

(a) **Base da Indução (BI)***: $P(1)$ é verdadeira;*

(b) **Passo Indutivo (PI)***: para todo número natural n, se $P(n)$ é verdadeira então $P(n+1)$ é verdadeira.*

Nestas condições, a propriedade $P(n)$ é verdadeira para todo número natural n.

A suposição $P(n)$ no passo indutivo é chamada de **hipótese de indução**.
Esquematicamente o princípio de indução pode ser representado da seguinte forma:

$$[\text{BI} + \text{PI}] \Rightarrow \forall n, P(n).$$

O Princípio da Indução Finita Generalizado

Pode ocorrer também que a propriedade a ser provada sobre os naturais só seja verdadeira a partir de um certo valor n_0 que não precisa necessariamente ser igual a 1. Podemos generalizar o princípio apresentado acima da seguinte forma:

Definição 2 (Princípio da Indução Finita Generalizado)**.** *Seja $P(n)$ uma propriedade sobre os números naturais que satisfaz as seguintes condições:*

1. **(BI)***: O número natural m_0 satisfaz a propriedade P;*

2. **(PI)***: Se um número natural n satisfaz a propriedade P então seu sucessor também satisfaz a propriedade P.*

Então todos os números naturais maiores ou iguais a m_0 satisfazem a propriedade P.

Vamos então retornar ao nosso problema inicial: provar que a equação (1) é verdadeira para qualquer número natural. Seja $P(n)$ a propriedade que expressa a seguinte ideia:

$$P(n): \quad 1+2+\ldots+n = \frac{n(n+1)}{2}$$

A base de indução corresponde a asserção: (BI) $P(1): \quad 1 = \dfrac{1(1+1)}{2}$, que é trivialmente verdadeira. Para provarmos o (PI), assumimos que $P(n)$ é uma proposição verdadeira, e precisamos mostrar que $P(n+1)$ também é verdadeira. Em outras palavras, se a igualdade

$$1+2+\ldots+n = \frac{n(n+1)}{2}$$

é verdadeira, então precisamo mostrar que $1+2+\ldots+n+(n+1) = \dfrac{(n+1)(n+2)}{2}$. De fato,
$1+2+\ldots+n+(n+1) =$

$\dfrac{n(n+1)}{2} + (n+1) =$
$\dfrac{n(n+1) + 2(n+1)}{2} =$
$\dfrac{(n+1)(n+2)}{2}$, que por sua vez corresponde a $P(n+1)$.

Logo, $P(n+1)$ é verdadeiro, e portanto $P(n)$ é verdadeira para todo $n \geq 1$.

Como outro exemplo, considere a seguinte afirmação: $2n+1 < 2^n$, para $n \geq 3$. Para $n = 3$ (BI), a desigualdade é verdadeira pois $7 < 8$. Para mostrarmos o (PI) assumimos que $2n+1 < 2^n$ (hipótese de indução), e vamos tentar concluir que $2(n+1)+1 < 2^{n+1}$. De fato, $2(n+1)+1 = (2n+1)+2 < 2^n + 2$, onde a última desigualdade é obtida aplicando-se a hipótese de indução. Para $n \geq 3$, temos que $2^n + 2 < 2^n + 2^n = 2^{n+1}$, e portanto $2(n+1)+1 < 2^{n+1}$ como queríamos concluir.

Princípio da Indução Forte

Em diversos problemas, pode não ser claro como provar $P(n+1)$ a partir de $P(n)$. Por exemplo, suponha que queiramos mostrar que todo número natural $n \geq 2$ pode ser escrito como um produto de números primos, isto é,

$$n = \prod_{i=1}^{k} p_i$$

onde $k \geq 1$ e p_i $(1 \leq i \leq k)$ é primo.

Para resolvermos este problema utilizaremos o chamado Princípio da Indução Forte (PIF):

Teorema 1. *Seja P uma propriedade referente aos números naturais. Dado $n \in \mathbb{N}$, se a validade de P para todo número natural menor do que n implicar que P é verdadeira para n, então P é verdadeira para todos os números naturais. Ou ainda, se $\forall n(\forall m, m < n \to P(m)) \to P(n)$ então $\forall n, P(n)$.*

Exercícios

1. Prove:
   (a) $n^2 < 2^n$, para $n \geq 5$.
   (b) $n! > 2^n$, para $n \geq 4$.
   (c) Para todo número natural n, 3 divide $n^3 - n$.
   (d) Demonstre que a soma dos n primeiros números ímpares é igual a n^2.
   (e) $1^2 + 2^2 + \ldots + n^2 = \dfrac{n(n+1)(2n+1)}{6}$
   (f) O conjunto $\{a_n | n \in \mathbb{N}\}$ é definido por:
   $a_0 := 0$
   $a_1 := 1$
   $a_{i+2} := 2a_{i+1} - a_i$ $(i \in \mathbb{N})$.

   Encontre uma fórmula geral para a_n e prove a correção de sua afirmação.

(g) O conjunto $\{a_n | n \in \mathbb{N}\}$ é definido por:
$a_0 := 0$
$a_1 := 4$
$a_{i+2} := 4(a_{i+1} - a_i)$ $(i \in \mathbb{N})$.

Prove que, para todo n, $a_n = n.2^{n+1}$.

2. Prove que todo número natural $n \geq 2$ pode ser escrito como um produto de números primos, isto é,

$$n = \prod_{i=1}^{k} p_i$$

onde $k \geq 1$ e p_i $(1 \leq i \leq k)$ é primo.

3. Prove a equivalência entre os princípios da indução forte (PIF) e da indução matemática (PIM).

Indução Estrutural

O conjunto dos números naturais pode ser definido indutivamente:

$$n ::= 0 \mid S\ n \tag{2}$$

A gramática (2) nos diz que um natural n é igual a 0, ou então é igual $S\ m$, se m é um natural já construído. Assim, a partir do 0 podemos construir o $S\ 0$, que normalmente chamamos de 1. A partir de $S\ 0$ podemos construir o $S(S\ 0)$, que normalmente denotamos por 2, e assim sucessivamente. Mostrar que uma propriedade P vale para os números naturais corresponde a mostrar que P vale para cada elementos do conjunto $\{0, S\ 0, S(S\ 0), \ldots\}$, o que corresponde ao PIM visto anteriormente.

As listas também constituem uma estrutura indutiva importante em Computação:

$$l ::= [] \mid h :: l \tag{3}$$

Similarmente, a gramática (3) nos diz que a lista vazia [] é uma lista, e que se l é uma lista já construída, e h um elemento, então $h :: l$ denota a lista cujo primeiro elemento é h, e cuja cauda é igual a l.

Exercícios

Considere a estrutura de listas definida como a seguir:

$$l ::= [] \mid a :: l$$

onde [] representa a lista vazia, e $a :: l$ representa a lista com primeiro elemento a e cauda l. O comprimento de uma lista é definido recursivamente por:

$$|l| = \begin{cases} 0, & \text{se } l = [] \\ 1 + |l'|, & \text{se } l = a :: l' \end{cases}$$

A concatenação de listas também pode ser definida por uma função recursiva:

$$l_1 \circ l_2 = \begin{cases} l_2, & \text{se } l_1 = [] \\ a :: (l' \circ l_2), & \text{se } l_1 = a :: l' \end{cases}$$

O reverso de listas é definido por:

$$rev(l) = \begin{cases} l, & \text{se } l = [] \\ (rev(l')) \circ (a :: []), & \text{se } l = a :: l' \end{cases}$$

1. Prove que $|l_1 \circ l_2| = |l_1| + |l_2|$, quaisquer que sejam as listas l_1, l_2.
2. Prove que $l \circ [] = l$, qualquer que seja a lista l.
3. Prove que a concatenação de listas é associativa, isto é, $(l_1 \circ l_2) \circ l_3) = l_1 \circ (l_2 \circ l_3)$ quaisquer que sejam as listas l_1, l_2 e l_3.
4. Prove que $|rev(l)| = |l|$, qualquer que seja a lista l.
5. Prove que $rev(l_1 \circ l_2) = (rev(l_2)) \circ (rev(l_1))$, quaisquer que sejam as listas l_1, l_2.
6. Prove que $rev(rev(l)) = l$, qualquer que seja a lista l.

Outra estrutura indutiva importante em Computação é a de árvores. Árvores pode ser definidas simplesmente como grafos sem ciclos, mas este conjunto também admite uma definição indutiva:

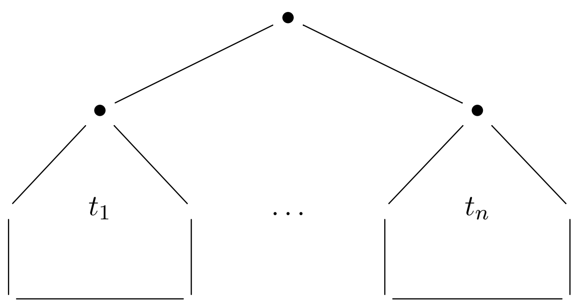
Definição 3. *Árvores são definidas indutivamente da seguinte forma:*

- *Um nó é uma árvore;*
- *Se $t_1, t_2, \ldots, t_n$ são árvores então podemos construir uma nova árvore adicionando um novo nó que será a raiz da nova árvore, que por sua vez será ligado por uma aresta a cada raiz das árvores $t_1, t_2, \ldots, t_n$.*

Graficamente, a definição acima nos diz que um nó é uma árvore:

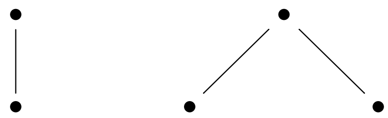
$$\bullet$$

e adicionalmente, se $t_1, t_2, \ldots, t_n$ são árvores então podemos construir uma nova árvore como esquematizado abaixo:

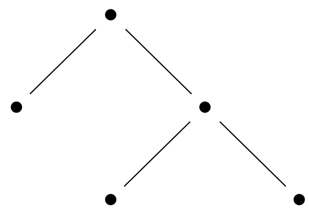


A altura de uma árvore é a maior distância entre uma folha e a raiz da árvore. Assim, a árvore

tem altura 0, enquanto que as árvores

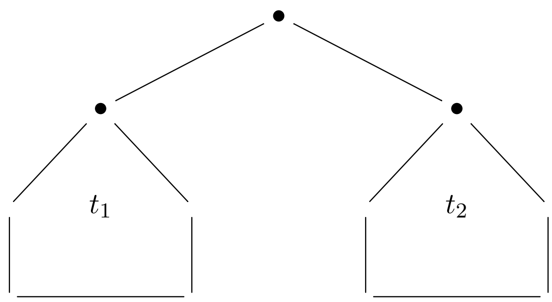


têm altura 1. Já a árvore



tem altura 2.

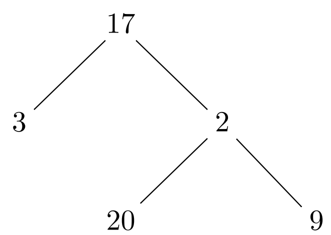
Esta noção de altura pode ser definida por uma função recursiva. Utilizaremos uma notação plana para representar árvores de forma que escreveremos $t_1 \cdot t_2$ ao invés de



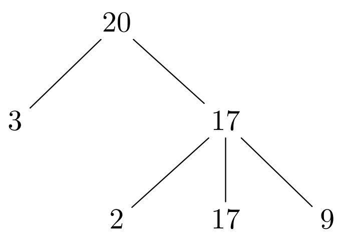
Assim, se t é uma árvore binária, podemos calcular a sua altura por meio da seguinte função:

$$h(t) = \begin{cases} 0, & \text{se } t = \bullet \\ 1 + \max\{h(t_1), h(t_2)\}, & \text{se } t = t_1 \cdot t_2 \end{cases}$$

Podemos utilizar os nós de uma árvore para armazenar dados:



Dizemos que uma árvore satisfaz a propriedade de *heap*, se o valor armazenado em um nó é maior ou igual ao valor armazenado nos nós filhos. A árvore acima não tem a propriedade de *heap*. Já a árvore



tem a propriedade de *heap*.

Exercícios

1. Defina uma notação plana adequada para árvores arbitrárias, e em seguida escreva a função h' que calcula a altura $h'(t)$ de uma árvore t qualquer.

2. Mostre que em uma árvore binária com altura h, e contendo n nós, vale a seguinte desigualdade $n \leq 2^{h+1} - 1$.

3. Mostre que se uma árvore tem a propriedade de *heap* então o valor armazenado na raiz da árvore é maior ou igual que qualquer outro valor armazenado na árvore.

Problema: Agora utilizaremos o conhecimento adquirido sobre indução para provar a correção de um algoritmo de ordenação de listas conhecido como *insertion sort*, ou ordenação por inserção. O pseudocódigo deste algoritmo é dado a seguir:

$$InsertionSort(l) = \begin{cases} l, & \text{se } l = [] \\ Insert(h, InsertionSort(l')), & \text{se } l = h :: l' \end{cases}$$

onde

$$Insert(x, l) = \begin{cases} x :: [], & \text{se } l = [] \\ x :: l, & \text{se } l = h :: l' \text{ e } x \leq h \end{cases}$$

Nosso objetivo é provar que o algoritmo acima é correto, mas o que isto significa? Como expressar este fato por meio de um teorema? Quais passos intermediários você julga que serão necessários para provar este teorema? Explique e justifique sua solução, e os passos que utilizou para obtê-la, de forma clara e completa por meio de um relatório detalhado.