Exemplo inducao

Definição

A ***altura*** *de uma árvore binária enraizada é definida de forma recursiva:*

- *A árvore de com um só no tem altura 1*
- *Se a árvore tem mais de um nó, então sua altura é igual a*

$$\max\{\textit{altura subárvore esquerda}, \textit{altura subárvore direita}\} + 1.$$

Exemplo inducao

Proposição

Para toda árvore binária enraizada temos

$$\textit{número de nós} \leq 2^{\textit{altura}} - 1.$$

Exemplo inducao

Proposição

Para toda árvore binária enraizada temos

$$\textit{número de nós} \leq 2^{\textit{altura}} - 1.$$

Prova: Por indução forte no número de nós n
Caso base: Árvore com $n = 1$ nó. Como temos 1 nó e a altura é 1, verificamos

$$1 \leq 2^1 - 1.$$

Exemplo inducao

Proposição

Para toda árvore binária enraizada temos

$$\textit{número de nós} \leq 2^{\textit{altura}} - 1.$$

Prova: Por indução forte no número de nós n
Caso base: Árvore com $n = 1$ nó. Como temos 1 nó e a altura é 1, verificamos

$$1 \leq 2^1 - 1.$$

Passo indutivo. Suponha que seja verdade pra toda árvore com $\leq n$ nós

Exemplo inducao

Proposição

Para toda árvore binária enraizada temos

$$\textit{número de nós} \leq 2^{\textit{altura}} - 1.$$

Prova: Por indução forte no número de nós n
Caso base: Árvore com $n = 1$ nó. Como temos 1 nó e a altura é 1, verificamos

$$1 \leq 2^1 - 1.$$

Passo indutivo. Suponha que seja verdade pra toda árvore com $\leq n$ nós

Precisamos provar pra árvore com $n + 1$ nós

Exemplo inducao

Exemplo inducao

Considere uma árvore com $n + 1$ nós.

Exemplo inducao

Considere uma árvore com $n + 1$ nós.

$$\#\text{ nós} = \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Exemplo inducao

Considere uma árvore com $n + 1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Exemplo inducao

Considere uma árvore com $n + 1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Portanto podemos aplicar a **hipotese indutiva** às sub-arvores:

Exemplo inducao

Considere uma árvore com $n+1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Portanto podemos aplicar a **hipotese indutiva** às sub-arvores:

$$\#\text{nós} = \underbrace{\#\text{ nós sub-árvore esquerda}}_{\leq 2^{\text{altura sub-árvore esquerda}} - 1} + \underbrace{\#\text{ nós sub-árvore direita}}_{\leq 2^{\text{altura sub-árvore direita}} - 1} + 1$$

Exemplo inducao

Considere uma árvore com $n+1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Portanto podemos aplicar a **hipotese indutiva** às sub-arvores:

$$\begin{aligned}\#\text{nós} &= \underbrace{\#\text{ nós sub-árvore esquerda}}_{\leq\, 2^{\text{altura sub-árvore esquerda}} - 1} + \underbrace{\#\text{ nós sub-árvore direita}}_{\leq\, 2^{\text{altura sub-árvore direita}} - 1} + 1 \\ &\leq 2^{\text{altura sub-árvore esquerda}} + 2^{\text{altura sub-árvore direita}} - 1\end{aligned}$$

Exemplo inducao

Considere uma árvore com $n+1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Portanto podemos aplicar a **hipotese indutiva** às sub-arvores:

$$\begin{aligned}
\#\text{nós} &= \underbrace{\#\text{ nós sub-árvore esquerda}}_{\leq\, 2^{\text{altura sub-árvore esquerda}} - 1} + \underbrace{\#\text{ nós sub-árvore direita}}_{\leq\, 2^{\text{altura sub-árvore direita}} - 1} + 1 \\
&\leq 2^{\text{altura sub-árvore esquerda}} + 2^{\text{altura sub-árvore direita}} - 1 \\
&\leq 2 \cdot 2^{\max\{\text{alt esq, alt dir}\}} - 1
\end{aligned}$$

Exemplo inducao

Considere uma árvore com $n+1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Portanto podemos aplicar a **hipotese indutiva** às sub-arvores:

$$\begin{aligned}
\#\text{nós} &= \underbrace{\#\text{ nós sub-árvore esquerda}}_{\leq\, 2^{\text{altura sub-árvore esquerda}} - 1} + \underbrace{\#\text{ nós sub-árvore direita}}_{\leq\, 2^{\text{altura sub-árvore direita}} - 1} + 1 \\
&\leq 2^{\text{altura sub-árvore esquerda}} + 2^{\text{altura sub-árvore direita}} - 1 \\
&\leq 2 \cdot 2^{\max\{\text{alt esq, alt dir}\}} - 1 \\
&= 2^{\max\{\text{alt esq, alt dir}\}+1} - 1
\end{aligned}$$

Exemplo inducao

Considere uma árvore com $n+1$ nós.

$$\#\text{ nós} \;=\; \#\text{ nós sub-árvore esquerda} + \#\text{ nós sub-árvore direita} + 1$$

Sub-árvores esquerda e direita tem estritamente menos nós que a árvore toda (então $\leq n$ nós)

Portanto podemos aplicar a **hipotese indutiva** às sub-arvores:

$$\begin{aligned}
\#\text{nós} &= \underbrace{\#\text{ nós sub-árvore esquerda}}_{\leq\, 2^{\text{altura sub-árvore esquerda}} - 1} + \underbrace{\#\text{ nós sub-árvore direita}}_{\leq\, 2^{\text{altura sub-árvore direita}} - 1} + 1 \\
&\leq 2^{\text{altura sub-árvore esquerda}} + 2^{\text{altura sub-árvore direita}} - 1 \\
&\leq 2 \cdot 2^{\max\{\text{alt esq, alt dir}\}} - 1 \\
&= 2^{\max\{\text{alt esq, alt dir}\}+1} - 1 \\
&= 2^{\text{altura árvore toda}} - 1.
\end{aligned}$$

Quicksort (2a tentativa)

Vamos novamente provar que o algoritmo **Quicksort** ordena um vetor de números corretamente

(De novo assumimos o vetor não tem repetição de números)

Quicksort (2a tentativa)

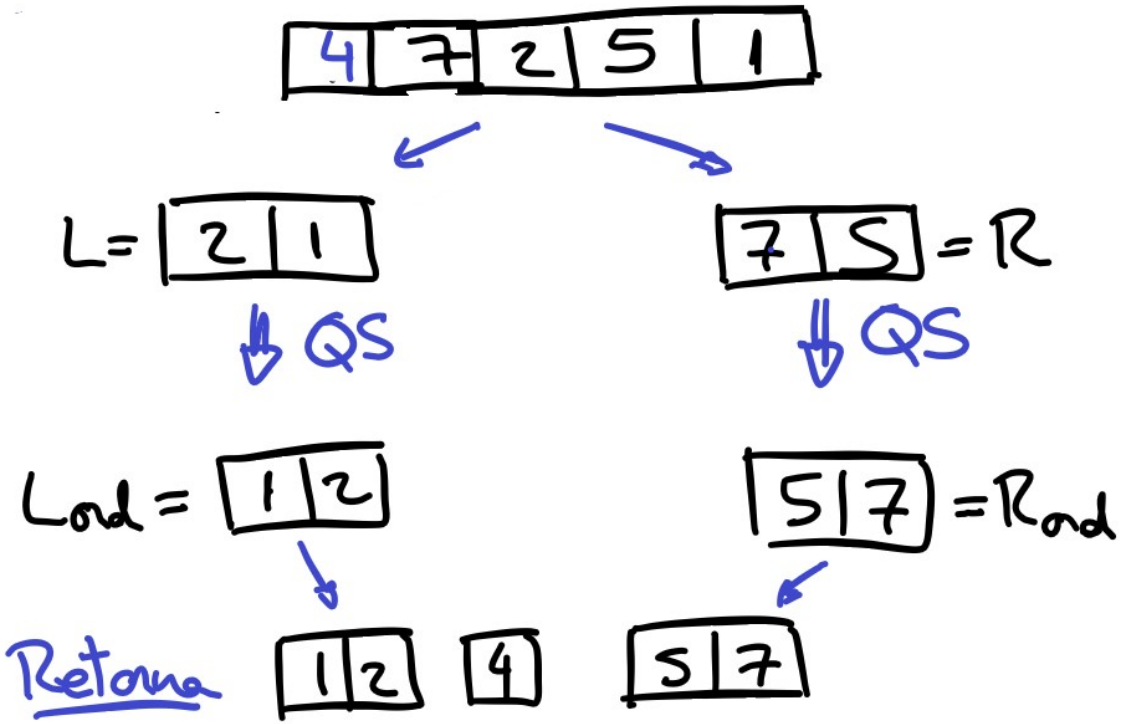
Vamos novamente provar que o algoritmo **Quicksort** ordena um vetor de números corretamente

(De novo assumimos o vetor não tem repetição de números)

Quicksort(A)

If $A.len = 0$ ou $A.len = 1$
    Return A
Else
    Construa vetor L com os números menores que $A[1]$
    Construa vetor R com os números maiores que $A[1]$
    $L_{ord} \leftarrow$ Quicksort(L)
    $R_{ord} \leftarrow$ Quicksort(R)
    Return concatenação de L_{ord}, $A[1]$ e R_{ord}
End if

Quicksort (2a tentativa)



Quicksort (2a tentativa)

Proposição

Quicksort(A) retorna o vetor A ordenado

Quicksort (2a tentativa)

Proposição

Quicksort(A) retorna o vetor A ordenado

Prova: Por indução forte no tamanho n do vetor A

Quicksort (2a tentativa)

Proposição

Quicksort(A) retorna o vetor A ordenado

Prova: Por indução forte no tamanho n do vetor A **Caso base: tamanho** $n = 0$ **ou** $n = 1$**.** Quicksort(A) retorna A, OK

Quicksort (2a tentativa)

Proposição

Quicksort(A) retorna o vetor A ordenado

Prova: Por indução forte no tamanho n do vetor A **Caso base: tamanho** $n = 0$ **ou** $n = 1$**.** Quicksort(A) retorna A, OK

Passo indutivo. Considere $n \geq 1$ e suponha que Quicksort funciona pra todo vetor de tamanho $\leq n$

Quicksort (2a tentativa)

Proposição

Quicksort(A) retorna o vetor A ordenado

Prova: Por indução forte no tamanho n do vetor A **Caso base: tamanho** $n = 0$ **ou** $n = 1$**.** Quicksort(A) retorna A, OK

Passo indutivo. Considere $n \geq 1$ e suponha que Quicksort funciona pra todo vetor de tamanho $\leq n$

Precisamos provar que funciona para vetores de tamanho $n + 1$

Quicksort (2a tentativa)

Considere um vetor A de tamanho $n + 1$

Quicksort (2a tentativa)

Considere um vetor A de tamanho $n + 1$

Note que os vetores L e R construídos têm tamanho estritamente menor que A (ou seja $\leq n$)

Quicksort (2a tentativa)

Considere um vetor A de tamanho $n + 1$

Note que os vetores L e R construídos têm tamanho estritamente menor que A (ou seja $\leq n$)

Então podemos aplicar a **hipótese indutiva** e obter que L_{ord} e R_{ord} estao ordenados

Quicksort (2a tentativa)

Considere um vetor A de tamanho $n+1$

Note que os vetores L e R construídos têm tamanho estritamente menor que A (ou seja $\leq n$)

Então podemos aplicar a **hipótese indutiva** e obter que L_{ord} e R_{ord} estao ordenados

Portanto, Quicksort(A) retorna

$$\underbrace{L_{ord}}_{\text{menores que } A[1] \text{ ordenados}} + A[1] + \underbrace{R_{ord}}_{\text{maiores que } A[1] \text{ ordenados}}$$

Quicksort (2a tentativa)

Considere um vetor A de tamanho $n+1$

Note que os vetores L e R construídos têm tamanho estritamente menor que A (ou seja $\leq n$)

Então podemos aplicar a **hipótese indutiva** e obter que L_{ord} e R_{ord} estao ordenados

Portanto, Quicksort(A) retorna

$$\underbrace{L_{ord}}_{\text{menores que } A[1] \text{ ordenados}} + A[1] + \underbrace{R_{ord}}_{\text{maiores que } A[1] \text{ ordenados}}$$

$\Rightarrow$ retorna vetor ordenado

Quicksort (2a tentativa)

Considere um vetor A de tamanho $n+1$

Note que os vetores L e R construídos têm tamanho estritamente menor que A (ou seja $\leq n$)

Então podemos aplicar a **hipótese indutiva** e obter que L_{ord} e R_{ord} estao ordenados

Portanto, Quicksort(A) retorna

$$\underbrace{L_{ord}}_{\text{menores que } A[1] \text{ ordenados}} + A[1] + \underbrace{R_{ord}}_{\text{maiores que } A[1] \text{ ordenados}}$$

$\Rightarrow$ retorna vetor ordenado

Conversor binário

Converter número em sua representação binária

Conversor binário

Converter número em sua representação binária

Saída: vetor $v = v_1, v_2, \ldots v_k$ onde v_1 é o digito binário mais significativo, v_2 o segundo mais significativo, etc.

Conversor binário

Converter número em sua representação binária

Saída: vetor $v = v_1, v_2, \ldots v_k$ onde v_1 é o digito binário mais significativo, v_2 o segundo mais significativo, etc.

Ex:

	Saída
$n = 6$	$(v_1, v_2, v_3) = (1, 1, 0)$
$n = 8$	$(v_1, v_2, v_3, v_4) = (1, 0, 0, 0)$
$n = 0$	vetor vazio

Conversor binário

Converter número em sua representação binária

Saída: vetor $v = v_1, v_2, \ldots v_k$ onde v_1 é o digito binário mais significativo, v_2 o segundo mais significativo, etc.

Ex:

	Saída
$n = 6$	$(v_1, v_2, v_3) = (1, 1, 0)$
$n = 8$	$(v_1, v_2, v_3, v_4) = (1, 0, 0, 0)$
$n = 0$	vetor vazio

Q: Qual é a relação entre n e o vetor v?

Conversor binário

Converter número em sua representação binária

Saída: vetor $v = v_1, v_2, \ldots v_k$ onde v_1 é o digito binário mais significativo, v_2 o segundo mais significativo, etc.

Ex:

	Saída
$n = 6$	$(v_1, v_2, v_3) = (1, 1, 0)$
$n = 8$	$(v_1, v_2, v_3, v_4) = (1, 0, 0, 0)$
$n = 0$	vetor vazio

Q: Qual é a relação entre n e o vetor v?

$$n = 2^{k-1}v_1 + 2^{k-2}v_2 + \ldots + 2^0 v_k$$

Conversor binário

Converte(n)

If $n = 0$

Return vetor vazio

Else if n **é par**

Vetor $u \leftarrow$ Converte$(\frac{n}{2})$

Return vetor $(u_1, u_2, \ldots, u_{u.len}, 0)$ [ou seja, u concatenado com 0 no final]

Else if n **é impar**

Vetor $u \leftarrow$ Converte$(\frac{n-1}{2})$

Return vetor $(u_1, u_2, \ldots, u_{u.len}, 1)$ [ou seja, u concatenado com 1 no final]

End if

Conversor binário

Converte(n)

If $n = 0$

    Return vetor vazio

Else if n **é par**

    Vetor $u \leftarrow$ Converte$(\frac{n}{2})$

    Return vetor $(u_1, u_2, \ldots, u_{u.len}, 0)$ [ou seja, u concatenado com 0 no final]

Else if n **é impar**

    Vetor $u \leftarrow$ Converte$(\frac{n-1}{2})$

    Return vetor $(u_1, u_2, \ldots, u_{u.len}, 1)$ [ou seja, u concatenado com 1 no final]

End if

Proposição

Para todo n, o vetor v retornado por Converte(n) é a representação binária de n: (k é o tamanho de v)

$$n = 2^{k-1}v_1 + 2^{k-2}v_2 + \ldots + 2^0 v_k$$

Conversor binário

Prova: Por indução forte em n **Caso base:** $n = 0$**.** Converte(0) retorna lista vazia, correto por definição

Conversor binário

Prova: Por indução forte em n **Caso base:** $n = 0$**.** Converte(0) retorna lista vazia, correto por definição

Passo indutivo. Considere $n \geq 1$. Suponha que o algoritmo funcione pra todo número $\leq n$

Conversor binário

Prova: Por indução forte em n **Caso base:** $n = 0$**.** Converte(0) retorna lista vazia, correto por definição

Passo indutivo. Considere $n \geq 1$. Suponha que o algoritmo funcione pra todo número $\leq n$

Precisamos provar que funciona pro número $n + 1$

Conversor binário

Caso 1: $n+1$ **é par.** Note que $\frac{n+1}{2}$ é estritamente menor que $n+1$, portanto podemos usar a **hipótese indutiva**

Conversor binário

Caso 1: $n+1$ **é par.** Note que $\frac{n+1}{2}$ é estritamente menor que $n+1$, portanto podemos usar a **hipótese indutiva**

Lembre $u = \mathit{Converte}(\frac{n+1}{2})$. Seja k o tamanho de u

Conversor binário

Caso 1: $n+1$ **é par.** Note que $\frac{n+1}{2}$ é estritamente menor que $n+1$, portanto podemos usar a **hipótese indutiva**

Lembre $u = \mathit{Converte}(\frac{n+1}{2})$. Seja k o tamanho de u

Converte$(n+1)$ retorna vetor $(u_1, \ldots, u_k, 0)$ com tamanho $k+1$ que tem valor

$$
\begin{aligned}
& 2^k u_1 + 2^{k-1} u_2 + \ldots + 2^1 u_k + \underbrace{2^0 \cdot 0}_{0} \\
= \; & 2 \cdot \left(2^{k-1} u_1 + 2^{k-2} u_2 + \ldots + 2^0 u_k \right) \\
= \; & 2 \cdot \frac{n+1}{2} && \text{(pela hipótese indutiva)} \\
= \; & n+1
\end{aligned}
$$

Conversor binário

Caso 2: $n+1$ **é impar.** Note que $\frac{(n+1)-1}{2}$ é estritamente menor que $n+1$, portanto podemos usar a **hipótese indutiva**

Conversor binário

Caso 2: $n+1$ **é impar.** Note que $\frac{(n+1)-1}{2}$ é estritamente menor que $n+1$, portanto podemos usar a **hipótese indutiva**

Lembre $u = Converte(\frac{(n+1)-1}{2})$. Seja k o tamanho de u

Conversor binário

Caso 2: $n+1$ **é impar.** Note que $\frac{(n+1)-1}{2}$ é estritamente menor que $n+1$, portanto podemos usar a **hipótese indutiva**

Lembre $u = \mathit{Converte}(\frac{(n+1)-1}{2})$. Seja k o tamanho de u

Converte$(n+1)$ retorna vetor $(u_1, \ldots, u_k, 1)$ com tamanho $k+1$ que tem valor

$$
\begin{aligned}
& 2^k u_1 + 2^{k-1} u_2 + \ldots + 2^1 u_k + 2^0 \cdot 1 \\
= \; & 2 \cdot \left(2^{k-1} u_1 + 2^{k-2} u_2 + \ldots + 2^0 u_k \right) + 1 \\
= \; & 2 \cdot \frac{(n+1)-1}{2} + 1 && \text{(pela hipótese indutiva)} \\
= \; & n+1
\end{aligned}
$$

Exercicio

Exercicio: Considere uma festa com n pessoas onde cada uma conheça no máximo 3 outras

Prove que é possível separar as pessoas em 4 grupos de modo que quaisquer duas pessoas do mesmo grupo não se conheçam