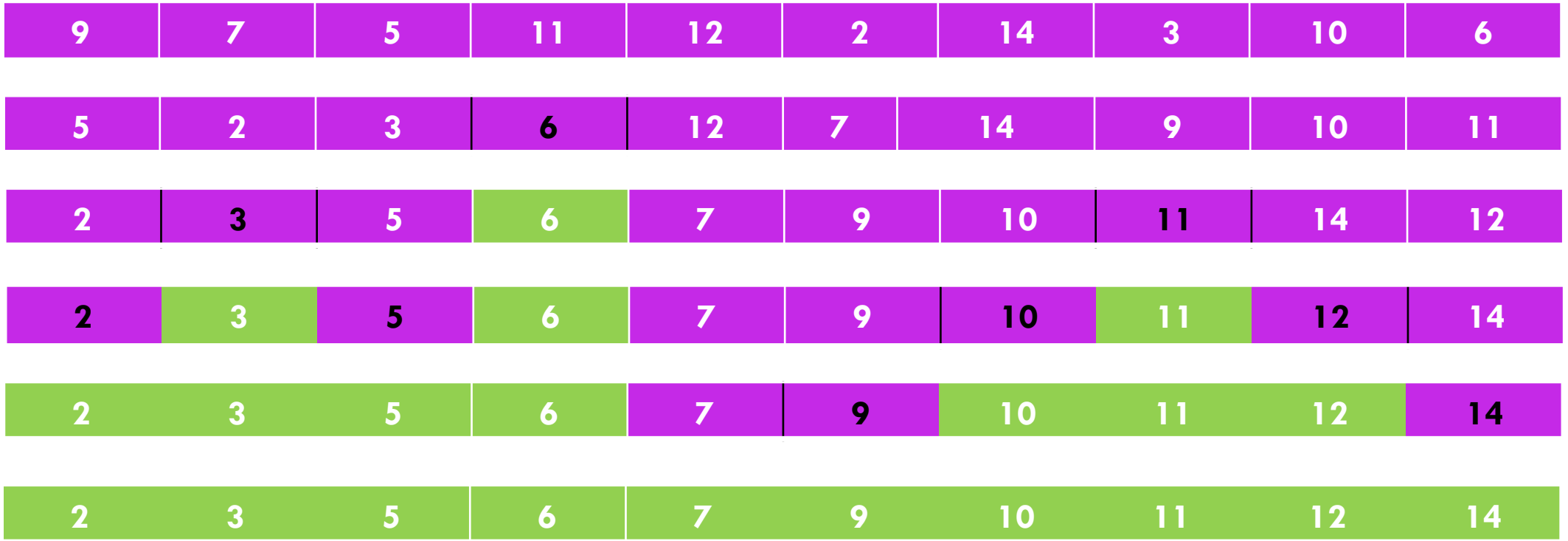
QUICKSORT

JORGE SIQUEIRA SERRÃO

Método do Quicksort

- O Quicksort se baseia na estratégia de dividir para conquistar.
- O algoritmo seleciona um pivô para separar o vetor em duas partições.
- Então o algoritmo reorganiza a lista de tal forma que todos os elementos anteriores ao pivô sejam menores que ele, e todos os elementos posteriores sejam maiores. No final o pivô estará na posição correta e haverá duas sub listas não ordenadas.
- E por fim o algoritmo irá ordenar as sub listas recursivamente até que o todo vetor esteja ordenado.

Funcionamento



Pseudo Código

```
QUICKSORT(A, lo, hi) {
    if(lo < hi) {
        p = PARTITION(A, lo, hi);
        QUICKSORT(A, lo, p-1);
        QUICKSORT(A, p+1, hi);
    }
}
```

```
PARTITION(A, lo, hi) {
    pivo = A[hi];
    i = lo – 1;
    for(k = lo to hi - 1) {
        if(A[k] < pivo) {
            i = i+1;
            swap(A[i], A[k]);
        }
    }
    swap(A[i+1], A[hi]);
    return i+1;
}
```

Cálculo da função do tempo

$T(n) = 2T(n/2) + O(n)$

$= 2T(n/2) + n$

$= 2[2T(n/2/2) + n/2]$

$= 4T(n/4) + n + n$

$= 4[2T(n/4/2) + n/4]$

$= 8T(n/8) + n + n + n$

Cálculo da função do tempo

$= 2^kT(n/2^k)$ com $n = 2^k$

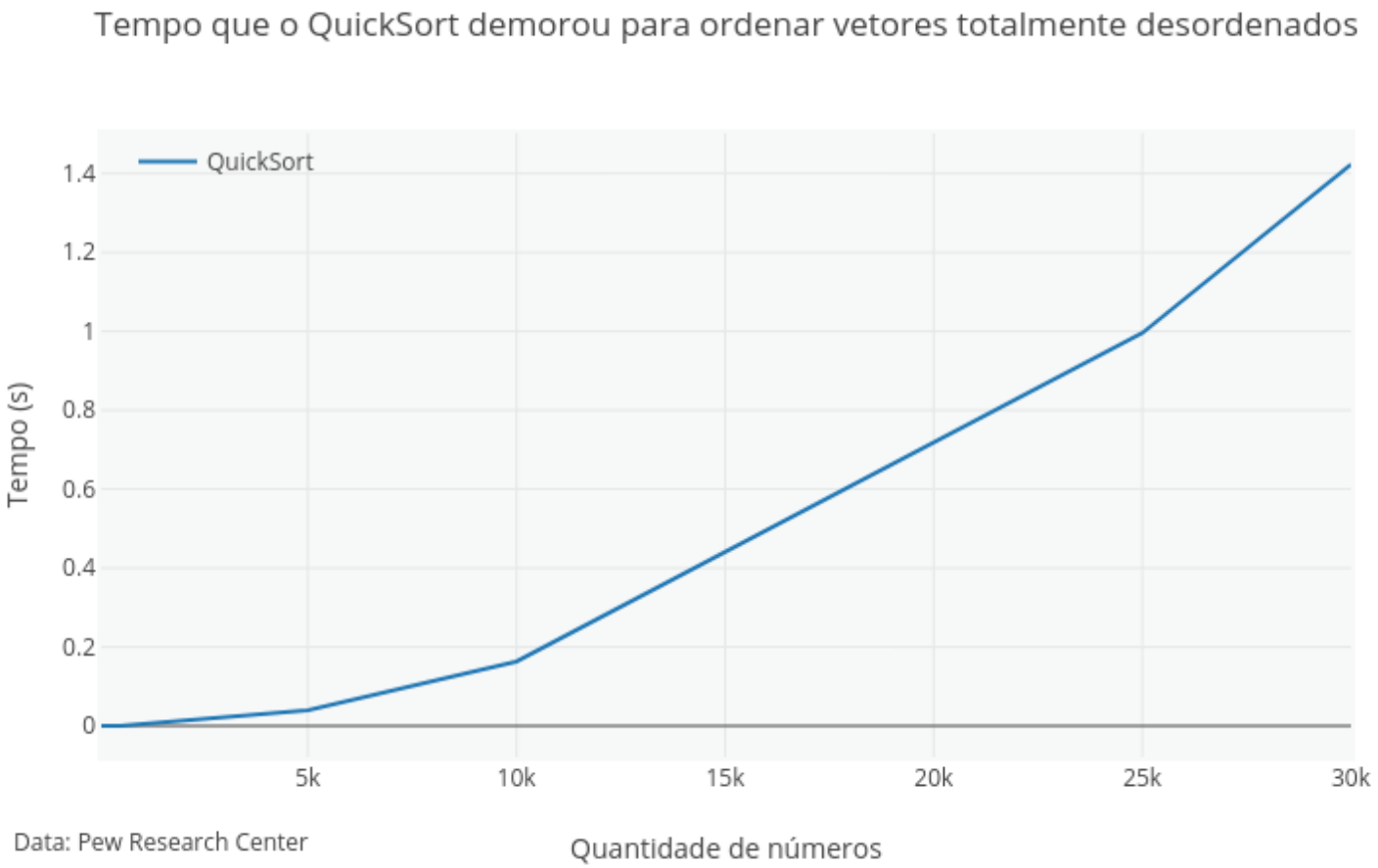
$= 2^kT(n/2^k) + kn$ $k = lg(n)$

$= nT(1) + nlgn$ $T(1) = 0$

$T(n) = nlgn$

Complexidade = O(nlgn)

Gráfico Tempo Quicksort



https://chart-studio.plotly.com/~jorgefilho/2

Analizando Mergesort

$T(n) = \{1 \text{ se } n = 1\}$ e $\{3T(n/3) + n + 1 \text{ se } n > 1\}$

$$T(n) = 3T(n/3) + n + 1$$

$$T(n/3) = 3T[3T(n/9) + (n/3) + 1] + n + 1$$

$$T(n/3) = 9T(n/9) + 2n + 3 + 1$$

$$T(n/9) = 9T[3T(n/27) + (n/9) + 1] + 2n + 3 + 1$$

$$T(n/9)*27T(n/27) + 3n + 9 + 3 + 1$$

Analizando Mergesort

$T(n) = \{1 \text{ se } n = 1\}$ e $\{3T(n/3) + n + 1 \text{ se } n > 1\}$

$$T(n/27) = 27T[3T(n/81) + (n/27) + 1] + 3n + 9 + 3 + 1$$

$$T(n/27) = 81T(n/81) + 4n + 27 + 9 + 3 + 1$$

$$T(n) = (3^4)T(n/(3^4)) + 4n + 3^3 + 3^2 + 3^1 + 3^0$$

$$T(n) = (3^k)T(n/(3^k)) + k*n \text{ ((somatorio(}\Sigma\text{) de i=0 até k – 1) } 3^i)$$

$$T(n) = (3^k)T(n/(3^k)) + k*n + ((1 - 3^{(k-1)})/(1-3))$$

$$T(n) = (3^k)T(n/(3^k)) + k*n (((3^{(k-1)}) - 1)/2)$$

Analizando Mergesort

$T(n) = \{1 \text{ se } n = 1\}$ e $\{3T(n/3) + n + 1 \text{ se } n > 1\}$

$$T(n) = (3^k)T(1) + k*n + ((3^{(k-1)}) - 1) / 2$$

$$T(n) = 3^k + k*n + ((3^{(k-1)}) - 1) / 2$$

$$T(n) = n + n*\log n \text{ base } 3 + (3^{\log n-1 \text{ base } 3} - 1 / 2$$

$$T(n) = n + n*\log n \text{ base } 3 + (n - 3) / 6 \Rightarrow O(n*\log n \text{ base } 3)$$

Para o melhor caso, pior caso e caso médio o custo será O(n*logn)

O mergesort neste caso possui uma complexidade similar ao quicksort

OBRIGADO