Algo

- Numero: 5
- Prof: Alexandre Duret-Lutz
- Date: 25 Octobre 2017

Tips

Essayer de calculer : $\sum_{i=0}^{\infty} ix^i$, avec $|x| < 1$

$$\begin{aligned}
S_n(s) &= \sum_{i=0}^{n} x^i \\
S_n(x) &= x^0 + x^1 + x^2 + \cdots + x^n \\
xS_n(x) &= x^1 + x^2 + \cdots + x^n + x^{n+1} \\
(1-x)S_n(x) &= x^1 + x^2 + \cdots + x^n + x^{n+1} \\
x \neq 1 \sum_{i=0}^{n} x^i &= \frac{1 - x^{n+1}}{1 - x} \\
\text{Donc } \sum_{i=0}^{\infty} x^i &= \frac{1}{1-x}
\end{aligned} \tag{1}$$

> La notation $\mathcal{O}(n)$ signifie que l'on est sur une fonction au pire lineaire. Mais elle peut etre plus rapide.

> La notation $\Theta(n)$ signifie que l'on est sur une fonction d'ordre lineaire.

Revenons a Heap

```
def BuildHeap(A, m):
  for i in range(n // 2 - 1, -1, -1):
    Heapify(A, m, i)
```

Nous somme sur une complexite: $\mathcal{O}(m \log m)$

QuickSort

```
def QuickSort(A, b, e):
    if (e - b > 1):
        m = Partition(A, b, e)
        QuickSort(A, b, m)
        QuickSort(A, m, e)
```

```
def Partition(A, b, e):
      p = A[b]; i = b; j = e - 1
    while True:
        while A[i] >= p:
            i += 1
        while A[j] <= p:
            j -= 1
        if (j > i):
            swap(A, i, j)
        else:
            return i + (b == i)
```

Complexite

`Partition` est de complexite $\Theta(n)$

`QuickSort` est de complexite $T_{QS}(n)$

Supposons que `Partition` nous donne la medianne. Du coup:

$$\begin{aligned} T(n) &= 2T\left(\frac{n}{2} + \Theta(n)\right) \text{ Comme MergeSort.} \\ \implies T(n) &= \Theta(n \log n) \end{aligned} \tag{2}$$

Supposons que `Partition` nous donne l'indice a 10% du debut.

$$\begin{aligned} T(n) &= T\left(\frac{n}{10}\right) + T\left(\frac{9}{10}n\right) + \Theta(n) \text{ Comme MergeSort.} \\ \implies T(n) &= \Theta(n \log n) \end{aligned} \tag{3}$$

On peut donc encadrer $T(n)$

$$
\begin{aligned}
& cn\log_{10} n \le T(n) \le cn\log_{10/9} n \\
& \Theta(n\log n) \le T(n) \le \Theta(n\log n) \\
\Longrightarrow\ & T(n) = \Theta(n\log n)
\end{aligned} \tag{4}
$$

Cas moyen:

> Partition peut retourner toutes les valeurs de m entre `b + 1` (inclus) et e (exclu). On suppose toutes les valeurs de m equiprobable.

$\widetilde{T}(n) = \frac{1}{n-1}\sum_{l=1}^{n-1}\left(\widetilde{T}(l) + \widetilde{T}(n-l) + \Theta(n)\right)$

$\widetilde{T}(n) = \Theta(n) + \frac{1}{n-1}\sum_{l=1}^{n-1}\widetilde{T}(l) + \widetilde{T}(n-l)$

$\widetilde{T}(n) = \Theta(n) + \frac{2}{n-1}\sum_{l=1}^{n-1}\widetilde{T}(l)$

$F(n) = cn + \frac{2}{n-1}\sum_{l=1}^{n-1}F(l)$

$(n-1)F(n) = cn(n-1) + 2\sum_{l=1}^{n-1}F(l)$

$(n-2)F(n) = cn(n-1)(n-2) + 2\sum_{l=1}^{n-2}F(l)$

$\frac{F(n)}{n} = \frac{2c}{n} + \frac{F(n-1)}{n-1}$

Donc `QuickSort` $\mathcal{O}(n^2)$
Mieux: $\Theta(n\log n)$ Moy: $\Theta(n\log n)$ Pire: $\Theta(n^2)$

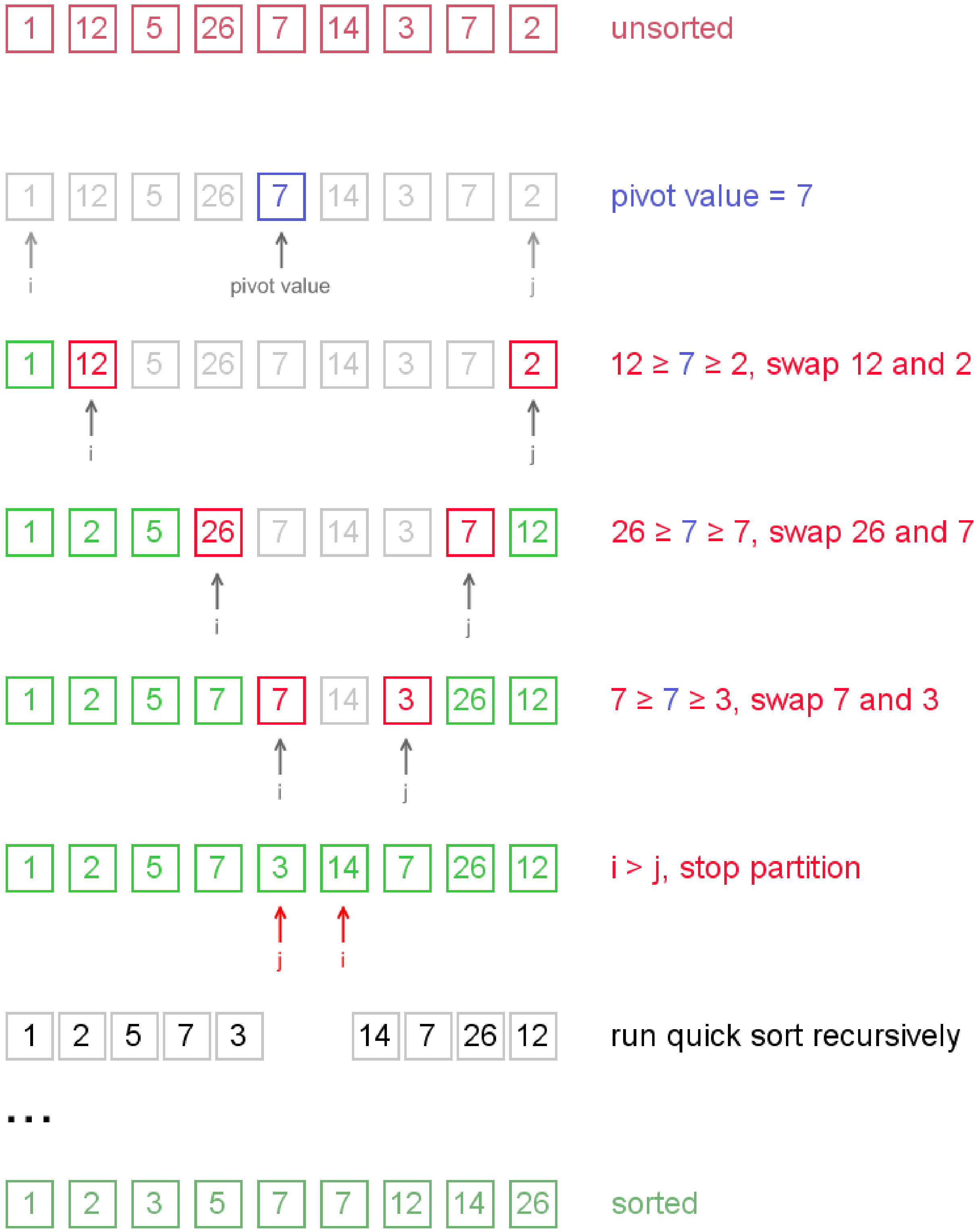

Figure 1: Quick Sort exemple