Sur la complexité moyenne du tri rapide avec une tentative de justifier l'équirépartition du pivot.

Rémi Lajugie

11 août 2018

Dans cette note, nous nous intéressons à l'analyse de la complexité moyenne du tri rapide. Il est bien connu que celle-ci est en $\Theta(n\log(n))$ alors que la complexité pire des cas (liste triée dans l'ordre croissant ou décroissant) est en $\Theta(n^2)$ et la complexité meilleur des cas est en $\Theta(n\log(n))$.

1 L'implémentation que l'on va étudier

Nous allons nous borner à analyser l'implémentation PYTHON suivante du tri rapide dans le cas où le pivot est choisi comme étant l'élément le plus à gauche de la liste PYTHON à segmenter.

```
1  def partition(t, i, j):
2      #On segmente entre les indices i et j-1 inclus
3      pivot = t[i]
4      a     = j-1 # a est l'indice de coupure :
5      #  IB : "Tous les elements a droite de a sont plus grands que le pivot"
6
7      for b in range(j-1, i, -1): # indices variant de j-1 a i+1
8          if t[b] > pivot :
9              t[a], t[b] = t[b], t[a]
10             a = a - 1
11
12     t[i], t[a] = t[a], t[i]
13     return a # a est l'indice de coupure, on notera dans la preuve g la partie t[i:a]
14              # d la partie t[a+1:j]
15
16 def triRapide(t, i, j):
17
18     if i + 1 < j:
19         a = partition(t, i, j)
20         triRapide(t, i, a)
21         triRapide(t, a + 1, j)
```

Nous supposerons, sans trop perdre en généralité, que la liste à trier est obtenue en permutant les n premiers entiers, i.e. l'ensemble $\{1,\dots,n\}$.

Proposition 1 *Supposons que toutes les permutations de S_n sont initialement équiprobables. Alors la fonction de partition du tri rapide coupe la liste initiale en une paire de listes (g,d) d'éléments permutés de manière équiprobables.*

Preuve 1 *Supposons que chaque permutation est initialement équiprobable. Nous identifions une permutation $\sigma \in S_n$ à la liste $[\sigma(1),\dots,\sigma(n)]$ des images de $[1,\dots,n]$.*

Analysons le fonctionnement du tri rapide. Avec la fonction partition, on sépare l'ensemble $\{1,\dots,n\}$ en deux sous ensembles g et d sur lesquels on effectue un appel récursif. g contient les éléments strictement

plus petits que le pivot et d ceux qui sont strictement plus grands. Le pivot se retrouve «à sa place finale» (puisqu'il est placé entre ces deux sous zones). Il s'agit désormais de s'assurer que g et d sont des ensembles de nombres permutés de manière équiprobable.

Notons k et $n-k-1$ les cardinaux respectifs de g et d.

***Fait :** La fonction de partition ne change pas l'ordre relatif des éléments de g.*

Ce fait est évident à la lecture de la fonction de partition. En effet, les éléments sont parcourus de la gauche vers la droite.

Soit l la liste à trier, avec les notations PYTHON, *$l = [a_1, \ldots, a_n]$. Par commodité on va noter $k+1$ la valeur du pivot, ici a_1. L'ordre relatif des éléments $[a_2, \ldots, a_n]$ induit une permutation de S_{n-1}. Notons qu'il est immédiat de vérifier que la distribution de cette permutation est également uniforme sur l'ensemble S_{n-1}.*

Considérons l'application $\phi : \sigma \in S_{n-1} \mapsto (\tau, \mu) \in S_k \times S_{n-k-1}$ définie comme suit : partant de $[a_2, \ldots, a_n]$, on construit deux permutations :

1. *la permutation $[a_{\alpha_1}, \ldots, a_{\alpha_k}]$ obtenue en extrayant les a_i correspondant aux k éléments plus petit que le pivot,*
2. *la permutation correspondant à la permutation des éléments plus grands que le pivot (après renumérotation, en gros $k+2, k+3, \ldots, n$ sont envoyés sur $1, 2, \ldots, n$).*

Soit $(\tau, \mu) \in S_k \times S_{n-k-1}$. Intéressons nous au cardinal de $\phi^{-1}(\tau, \mu)$, c'est à dire au nombre de listes représentant une permutation dont l'image est (τ, μ). Une liste de $\phi^{-1}(\tau, \mu)$ peut se construire de manière univoque de la manière suivante :

1. *on choisit k emplacements parmi les $n-1$ derniers éléments du tableau pour placer les éléments $\tau(1), \ldots, \tau(k)$ (dans cet ordre).*
2. *Ensuite on place les éléments restant dans l'ordre imposé par μ (en décalant les indices, car 1 doit être envoyé sur $k+1$, 2 sur $k+2$ etc.). Il n'y a qu'une seule possibilité pour faire ce placement*

Ainsi, la la probabilité $P(\phi(\sigma) = (\tau, \mu)) = \frac{1}{(n-1)!} \times \binom{n-1}{k} = \frac{1}{k!(n-k-1)!}$. Autrement dit, les variables aléatoires donnant τ et μ sont uniformes[1] *et indépendantes.*

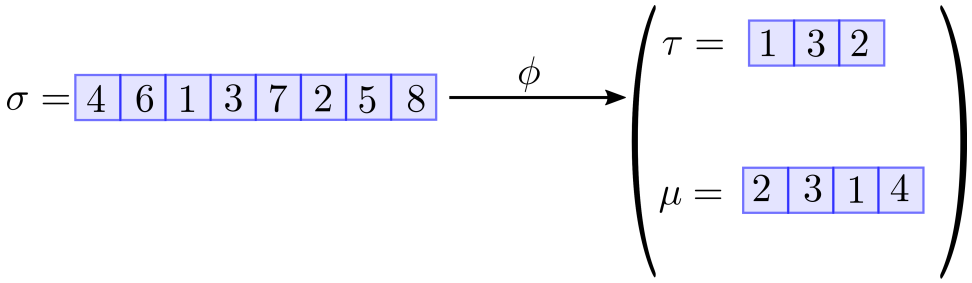

FIGURE 1 – Illustration de l'application ϕ utilisée dans la preuve de la complexité du tri rapide. Dans cet exemple, la valeur du pivot vaut 4.

Ainsi les deux listes séparées par le pivot induisent chacune des permutations uniformes de leurs valeurs et de plus ces permutations sont des variables aléatoires indépendantes.[2].

De cette proposition, on déduit la propriété suivante :

Proposition 2 *A chaque appel récursif du tri rapide, les pivots sont uniformément distribués parmi les éléments des sous-listes à trier.*

1. Pour cela il suffit de calculer la probabilité que la première composante de $\phi(\sigma)$ vaille τ (resp la deuxième composante vaille μ.
2. En fait, conditionnellement à la valeur du pivot les variables aléatoires donnant τ et μ sont indépendantes.

Preuve 2 *Par la proposition précédente, à chaque appel appel à la fonction de partition, on va séparer la liste en un couple de sous-listes indépendamment permutées de manière uniforme. On en déduit alors que le pivot de chacun de ces deux sous-tableaux(que l'on prend dans notre implémentation comme étant le premier élément du tableau) peut prendre n'importe quelle valeur du sous-tableau avec même probabilité. Les pivots sont donc répartis de manière uniforme.*

Théorème 1 *La complexité moyenne du tri rapide est en* $\Theta(n\log(n))$.

Preuve 3 *On rappelle que l'on suppose que la liste à trier est composée des n entiers de l'ensemble* $E = \{1, 2, 3, \ldots, n\}$.

Nous allons démontrer la complexité . Comme il est usuel pour les méthodes de tri opérant par comparaisons et échanges, nous allons uniquement nous contenter d'estimer le nombre de comparaisons faites au cours du tri[3].

Soit T_{π_n} *la complexité pour le tri rapide d'un tableau de taille n obtenu en permutant les éléments de E avec la permutation* π. *A la première étape du tri rapide, on choisit un pivot qui a une certaine valeur k. La liste est alors séparée en deux : d'un côté les k éléments strictement plus petits que le pivot et de l'autre les* $n-k-1$ *éléments strictement plus grands que le pivot. La complexité satisfait alors à une relation de récurrence de la forme* $T_{\pi_n} = T_{\pi_k} + T_{\pi_{n-k-1}} + n - 1$, *où* π_k *et* π_{n-k-1} *sont les permutations correspondant aux listes obtenues après partition autour du pivot (qui s'appelaient* τ *et* μ *dans la preuve précédente). Le troisième terme de la somme est le nombre de comparaisons impliqué dans la procédure de segmentation.*

On note c_n *la complexité en moyenne.* $c_n = \mathbb{E}[T_{\pi_n}] = \frac{1}{n!}\sum_{\pi_n \in S_n} T_{\pi_n}$, *a priori, mais on peut réarranger la somme suivant la valeur du premier pivot :*

$$c_n = \frac{1}{n!}\sum_{k=1}^{n}\sum_{\pi_k, \pi_{n-k-1}} T(\pi_k) + T(\pi_{n-k-1}),$$

où, comme précédemment, π_k *est la permutation obtenue en restreignant* π *aux k premiers éléments (éléments placés à gauche du pivot après l'étape de partition) et* π_{n-k-1} *est la permutation correspondant à la partie située à droite du pivot. Par la proposition qui précède, on a uniforme répartition des pivots à chaque appel. Donc* $\sum_{\pi_k, \pi_{n-k-1}} T(\pi_k) + T(\pi_{n-k-1}) = (n-1)!c_k + c_{n-k-1}$, *on en déduit alors :*

$$c_n = \frac{1}{n}\sum_{k=0}^{n-1}(c_k + c_{n-k-1}) + n - 1 = \frac{2}{n}\sum_{k=0}^{n-1} c_k + n - 1,$$

Cette récurrence peut se résoudre de la manière suivante.
On a :
— $nc_n = n(n-1) + 2c_{n-1} + 2\sum_{k=0}^{n-2} c_k$,
— $(n-1)c_{n-1} = (n-1)(n-2) + 2\sum_{k=0}^{n-2} c_k$.
On fait la différence et il vient que :

$$nc_n - (n+1)c_{n-1} = 2(n-1),$$

On divise par $n(n+1)$ *et en remarquant que* $\frac{4}{n+1} - \frac{2}{n} = \frac{2(n-1)}{n(n+1)}$, *on obtient que :* $\frac{c_n}{n+1} - \frac{c_{n-1}}{n} = \frac{4}{n+1} - \frac{2}{n}$.
On somme cette suite télescopique et on obtient que :

$$\frac{c_n}{n+1} - c_0 = 2\sum_{k=0}^{n-1}\frac{1}{k} + \frac{4}{n+1} - 4$$

Or $c_0 = 0$, *donc* $c_n = 2(n+1)(\sum_{k=0}^{n-1}\frac{1}{k}) - 4n = \Theta(n\log(n))$ *en utilisant l'équivalent asymptotique de la série harmonique.*

3. Dans ce genre de tri le nombre d'échanges est au plus égal au nombre de comparaisons puisqu'avant d'échanger deux éléments, il faut les avoir comparé.