10 et 12 oct. 2018 | # Les tris récursifs | I2 — Promo 122

A. Introduction

L'algorithme des tris récursifs est basé sur le principe suivant :

- **Diviser** : on divise la liste à trier en deux,
- **Régner** : on trie les deux listes obtenues,
- **Combiner** : on combine ces deux listes.

Il existe deux tris récursifs :

1. le tri rapide, « l'intelligence » du tri se trouve au niveau de la division (partitionnement) de la liste à trier.
2. le tri fusion, « l'intelligence » du tri se trouve au niveau la combinaison (interclassement) des deux listes obtenues après la division de la liste à trier.

B. Tri rapide

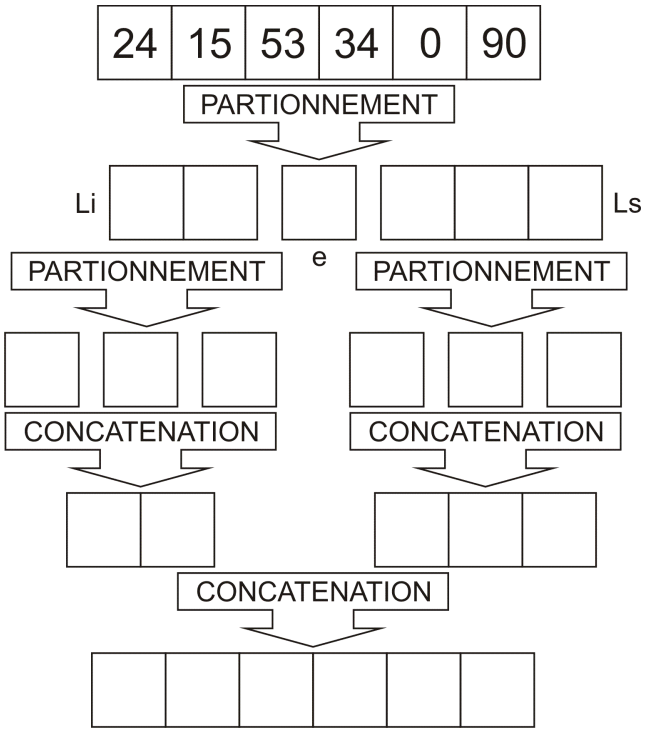
1. Présentation

Le principe du tri rapide d'une liste L est le suivant :

- si la liste L admet plus d'un seul élément alors
  - on choisit un élément quelconque $e \in L$,
  - on construit deux sous-listes Li formée des éléments inférieurs à e et Ls formée des éléments strictement supérieurs,
  - le résultat est la concaténation de la liste Li récursivement triée, de la liste $[e]$ et de la liste Ls elle aussi récursivement triée.

Remarque : Pour ce TP, nous choisissons comme élément quelconque e le premier élément de la liste L.

Exercice 1 Effectuer l'algorithme « à la main » sur la liste $[24, 15, 53, 34, 0, 90]$ en complétant la figure ci-dessous. Compter le nombre de comparaisons réalisées.



Exercice 2 Compter le nombre de comparaisons réalisées pour les listes $[0, 15, 24, 34, 53, 90]$ et $[90, 53, 34, 24, 15, 0]$.

Exercice 3 Pour quelle type de listes a-t-on le maximum de comparaisons ?
Pour une liste de n termes, combien y a-t-il de comparaisons au maximum ?
En déduire la complexité maximale du tri rapide.

Exercice 4 Écrire le pseudo-code de la fonction **partitionnement(L)** qui retourne les deux sous-listes Li et Ls.

Exercice 5 Écrire le pseudo-code de la fonction **tri_rapide(L)** qui utilise l'algorithme du tri rapide pour trier la liste L.

2. Implémentation en Python

Exercice 6 Coder la fonction **partitionnement(L)** qui retourne les deux sous-listes Li et Ls. Tester votre fonction avec $[24, 15, 53, 34, 0, 90]$.

Exercice 7 Coder la fonction **tri_rapide(L)** qui retourne la liste L triée en utilisant la technique du tri rapide.

A l'exercice 3, vous avez du déterminer une complexité maximale quadratique pour le tri rapide tout comme le tri à bulles. Dans ce cas, pourquoi cet algorithme de tri porte-t-il le nom de « tri rapide » ? Nous allons répondre à cette question en étudiant sa complexité moyenne.

Exercice 8 Compléter le tableau ci-dessous en générant 100 listes aléatoires de 100, 250, 500 et 1000 entiers compris entre 1 et 10 000 afin d'évaluer en moyenne le temps d'exécution du tri rapide.

Nombre d'éléments : n	100	250	500	1000
Temps d'exécution : t				
$\frac{t}{n}$				
$\frac{t}{n.ln(n)}$				
$\frac{t}{n^2}$				

En déduire la complexité moyenne du tri rapide. La comparer à celle du tri à bulles.

C. Tri fusion

1. Présentation

Le principe du tri fusion d'une liste L est le suivant :

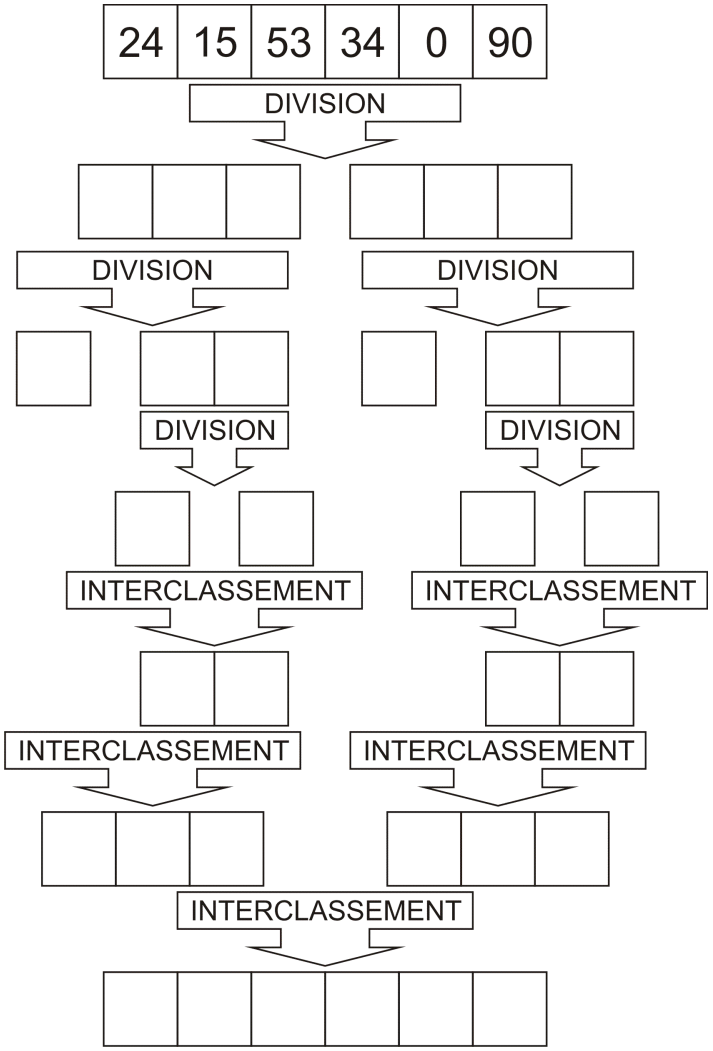
- si la liste L admet plus d'un seul élément alors
  - on divise la liste en deux sous-listes $L1$ et $L2$ de même longueur (à plus ou moins 1 près),
  - on trie récursivement les deux listes $L1$ et $L2$,
  - on fusionne les deux listes $L1$ et $L2$ en interclassant leurs éléments.

Pour rassembler deux listes $L1$ et $L2$ déjà triées, on peut les interclasser de la façon suivante :

- on compare les premiers éléments de chacune d'elles,
- on insère le plus petit des deux dans une nouvelle liste L,
- on supprime l'élément inséré de sa liste initiale,
- on poursuit en comparant successivement les deux plus petits éléments jusqu'à obtenir une des deux listes $L1$ et $L2$ vide,
- on insère dans la liste L les éléments de la liste non vide.

Remarque : Pour ce TP, nous choisissons d'obtenir la liste $L1$ avec un élément en moins par rapport à la liste $L2$ lorsque la longueur de la liste L est impaire.

Exercice 9 Effectuer l'algorithme « à la main » sur la liste $[24, 15, 53, 34, 0, 90]$ en complétant la figure ci-dessous. Compter le nombre de comparaisons réalisées.



Exercice 10 Compter le nombre de comparaisons réalisées pour les listes $[0, 15, 24, 34, 53, 90]$ et $[90, 53, 34, 24, 15, 0]$.

Vous pouvez remarquer que le maximum de comparaisons n'est pas atteint comme c'est le cas avec le tri rapide.

Exercice 11 Écrire le pseudo-code de la fonction **interclassement(L1,L2)** qui retourne la liste triée L contenant les éléments des listes triées $L1$ et $L2$.

Exercice 12 Écrire le pseudo-code de la fonction **tri_fusion(L)** qui utilise l'algorithme du tri fusion pour trier la liste L.

2. Implémentation en Python

Exercice 13 Coder la fonction **interclassement(L1,L2)** qui retourne la liste triée L contenant les éléments des listes triées $L1$ et $L2$. Tester votre fonction avec $[15, 24, 53]$ et $[0, 34, 90]$.

Exercice 14 Coder la fonction **tri_fusion(L)** qui retourne la liste **L** triée en utilisant la technique du tri fusion.

Exercice 15 Compléter le tableau ci-dessous en générant 100 listes aléatoires de 100, 250, 500 et 1000 entiers compris entre 1 et 10 000 afin d'évaluer en moyenne le temps d'exécution du tri fusion.

Nombre d'éléments : n	100	250	500	1000
Temps d'exécution : t				
$\frac{t}{n}$				
$\frac{t}{n.ln(n)}$				
$\frac{t}{n^2}$				

En déduire la complexité moyenne du tri fusion.