Algorithme de tri rapide

I. Fonctionnement

On considère un tableau $t = [t_0; t_1; \dots t_{n-1}]$ de taille n. Le but de l'algorithme est de trier le tableau t **en place**, en effectuant uniquement des permutations d'éléments, sans allouer de nouveau tableau.

I. 1. Fonctionnement général

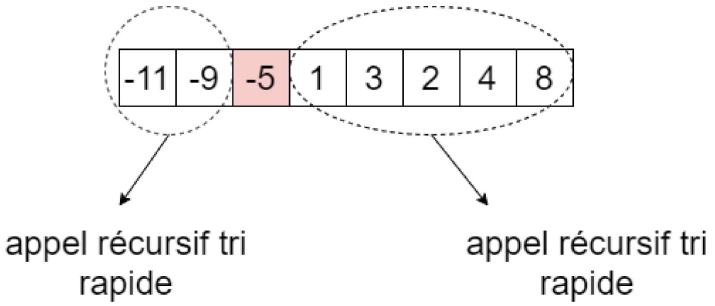
L'algorithme est globalement récursif mais fait appel à une sous-fonction itérative. L'algorithme travaille en place, directement dans le tableau, en effectuant uniquement des échanges de valeurs (transpositions, cf chapitre de mathématiques à venir sur le groupe symétrique).
Comme dans l'algorithme de recherche dichotomique, la récursivité s'opère sur la taille de la fenêtre de travail. Celle-ci est délimitée par la donnée de deux indices : `l` (indice gauche) et `r` (indice droit strict). Les éléments du tableau sur lesquels on travaille sont compris dans l'intervalle d'indices $[l, r-1]$. La taille de la fenêtre de travail est donc $r-l$ et on a toujours $0 \leq l \leq r \leq n$.

Tripartition : On suppose que l'on dispose d'une fonction de tripartition qui permet de répartir les éléments d'un tableau en trois segments par rapport à une valeur de référence, appelée **pivot** :

- **un segment gauche** qui ne contient que des valeurs strictement inférieures au pivot,
- **un segment central** qui contient toutes les valeurs égales au pivot,
- **un segment droit** qui ne contient que des valeurs strictement supérieures au pivot.

Initialisation. On choisit une valeur du tableau comme pivot et on active l'algorithme de tripartition sur l'ensemble du tableau

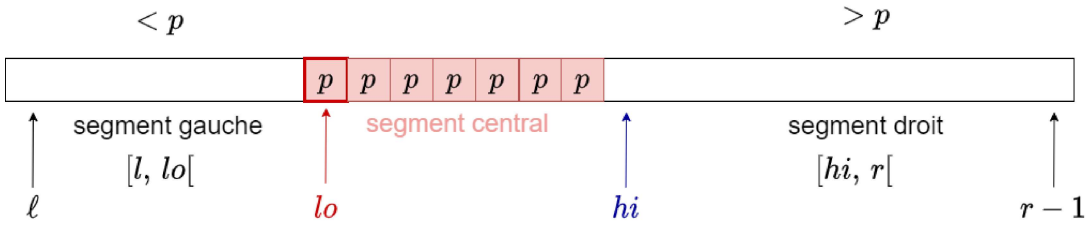
Récursivité. On appelle ensuite récursivement l'algorithme de tri rapide sur le segment gauche et sur le segment droit.



Cas de base. L'imbrication des appels récursifs cesse lorsque la taille de la fenêtre de travail est inférieure ou égale à 1 (aucun ou un seul élément dans le tableau).

I. 2. Détail de l'algorithme de tripartition

Le but de l'algorithme de tripartition est de séparer les éléments d'une fenêtre de travail $[l, r-1]$ en trois segments autour d'une valeur pivot p comme ceci :



De façon assez classique, on choisit la première valeur de la fenêtre comme pivot :

$$p = t[\ell]$$

mais n'importe quelle valeur de pivot peut être choisie et l'algorithme s'adapte facilement. Le pivot reste le même tout au long de l'algorithme de tripartition.
Pour effectuer la tripartition, on fait évoluer deux indices séparateurs lo et hi et un indice i correspondant à la valeur en cours de traitement, de sorte que, à chaque étape, on ait $\ell \leq lo < i \leq hi \leq r$:

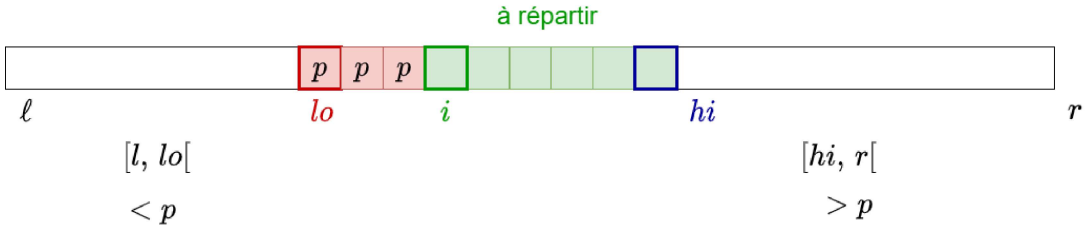
dans le segment gauche correspondant aux indices $[\ell, lo[$: tous les éléments strictement inférieurs à p

dans le segment central correspondant aux indices $[lo, i[$: tous les éléments égaux à p

dans le segment correspondant aux indices $[i, hi[$: tous les éléments restants à traiter

dans le segment droit correspondant aux indices $[hi, r[$: tous les éléments strictement supérieurs à p

L'algorithme de tripartition fonctionne de manière itérative : une boucle `while` sur l'indice i de la valeur à analyser nous permet à chaque itération de ranger la nouvelle valeur $t[i]$ dans le bon segment et de mettre en place les indices i, lo et hi pour la prochaine itération.



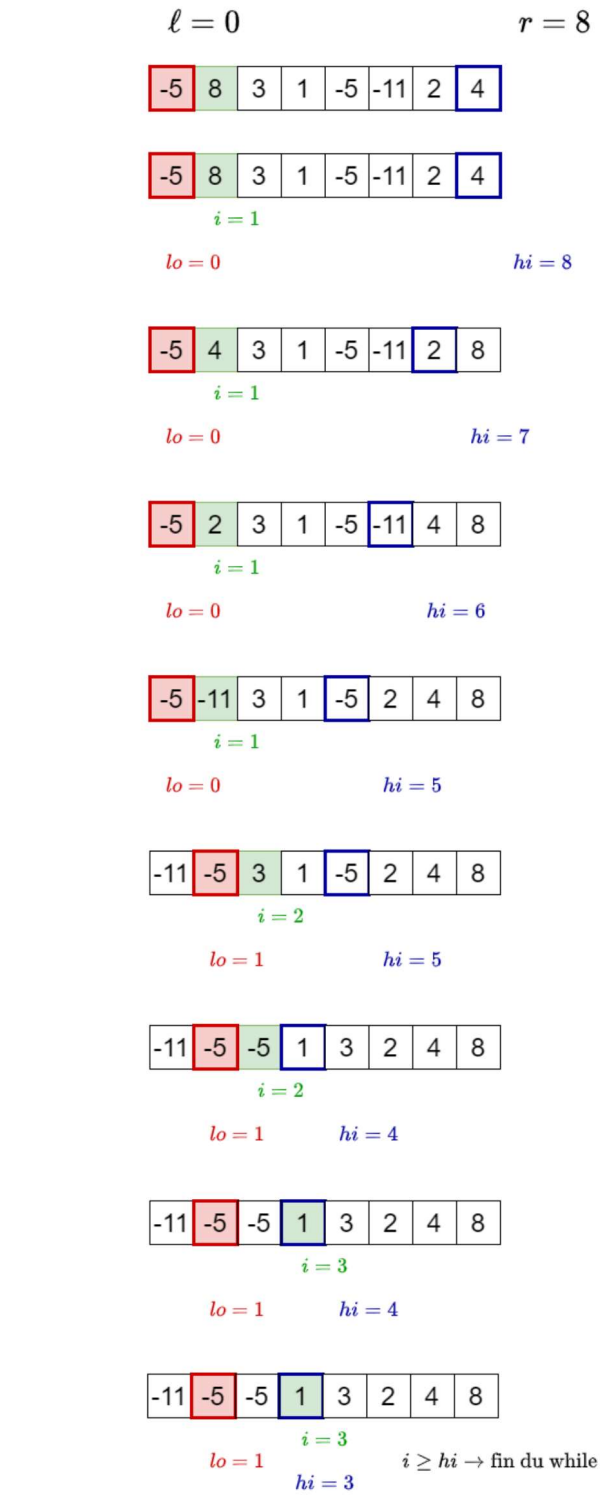
Plus précisément, à chaque itération, tant que i reste contenu dans l'intervalle des valeurs à répartir :

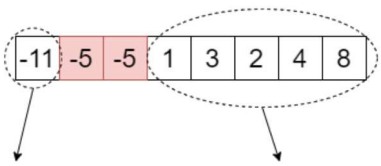
si $t[i] < p$, on échange $t[i]$ et $t[lo]$, on incrémente lo car le segment gauche a gagné une valeur, et on incrémente i pour le placer sur la prochaine valeur à répartir ;

si $t[i] = p$, $t[i]$ est donc déjà bien placée dans le segment central des valeurs égales au pivot. On incrémente simplement i pour le placer sur la prochaine valeur à répartir ;

si $t[i] > p$, on échange $t[i]$ et $t[hi - 1]$, et on décrémente hi de sorte que l'ancienne valeur $t[i]$ devient la valeur la plus à gauche du segment droit. Nul besoin de modifier i car la valeur maintenant située à l'indice i a été échangée avec une valeur à répartir, et sera traitée au prochain tour de boucle sans qu'il soit nécessaire de modifier i.

La figure ci-dessous montre un exemple de fonctionnement de l'algorithme de tripartition :





appel récursif tri rapide

$\ell = 0$

$r = lo = 1$

appel récursif tri rapide

$\ell = hi = 3$

$r = 8$

cas de base

$r - l \leq 1 \rightarrow$ fin des appels récursifs