Comparaison des Algorithmes de Tri

Analyse comparative entre le **Tri par Sélection** (utilisé dans la classe `File`) et le **Tri Fusion** (utilisé dans la classe `Pile`).

1. Complexité Temporelle (Time Complexity)

Tri par Sélection (Selection Sort)

Le tri par sélection parcourt la liste non triée à chaque itération pour trouver l'élément minimum (ou maximum) et le mettre à sa position finale.

- **Complexité**: $\boldsymbol{O(n^2)}$.
- **Explication**: Pour une liste de taille $\boldsymbol{n}$, il effectue $\boldsymbol{n}$ itérations. À chaque itération $\boldsymbol{i}$, il fait $\boldsymbol{n-1-i}$ comparaisons. le nombre des iteration devient exprime par la somme:

$$\sum_{i=0}^{n-1}(n-1-i)$$

$$=\sum_{i=0}^{n-1}(n-1)-\sum_{i=0}^{n-1}i$$

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

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

$$=\left(\frac{1}{2}\right)n^2-\left(\frac{1}{2}\right)n$$

Ce qui donne une complexité de l'ordre de $\boldsymbol{O(n^2)}$

Tri Fusion (Merge Sort)

Le tri fusion est un algorithme « Diviser pour Régner » qui divise la liste en deux moitiés récursivement jusqu'à ce que chaque sous-liste ne contienne qu'un seul élément, puis fusionne les moitiés triées.

- **Complexité**: $\boldsymbol{O(n \log n)}$
- **Explication**: La division de la liste prend un temps logarithmique ($\boldsymbol{\log n}$ niveaux de récursion), et la fusion des sous-listes prend un temps linéaire ($\boldsymbol{n}$) à chaque niveau car en chaque niveau, tous les éléments doivent être comparés et fusionnés.

2. Complexité Spatiale (Space Complexity)

Tri par Sélection

- **Complexité**: $O(1)$
- **Explication**: Cet algorithme n'utilise qu'un nombre constant d'espaces supplémentaires pour les variables temporaires, indépendamment de la taille de la liste à trier.

Tri Fusion

- **Complexité**: $O(n)$
- **Explication**: L'étape de fusion nécessite de créer des petites tableaux temporaires pour stocker les éléments triés avant de les recopier dans le tableau original.

3. Tableau de Comparaison

	Tri par Sélection	**Tri Fusion**
Complexité Temporelle	$O(n^2)$	$O(n \log n)$
Complexité Spatiale	$O(1)$	$O(n)$
Efficacité sur grandes listes	Faible	Élevée

Tableau 1. – Comparaison Selection Sort vs Merge Sort

4. Comparaison Pratique

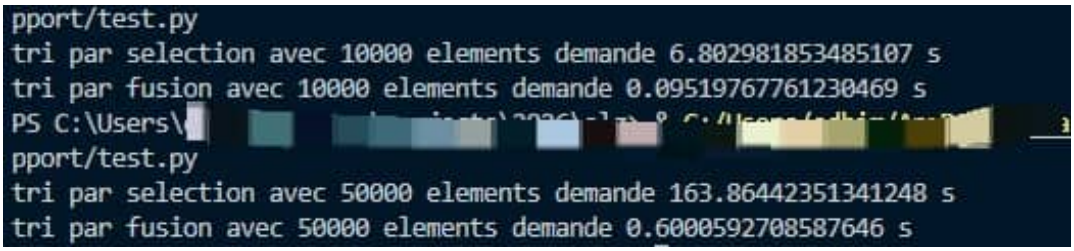

Fig. 1. – Performance des Algorithmes de Tri en Fonction de la Taille des Données