Introduction

Avoir une idée de quel outil utiliser afin d’optimiser le temps d’exécution de son programme est confortable voire nécessaire suivant le domaine d’application.

Dans ce laboratoire, nous allons nous concentrer sur les temps d’écriture et de lecture de données dans des fichiers, en fonction de différentes stratégies (Stream buffered ou non, écriture byte par byte ou bloc par bloc, en variant les tailles). Nous verrons, au moyen de graphiques, comment évoluent ces temps.

Améliorations apportées au programme fourni

De base, le programme fourni nous permet de mesurer le temps qu’il nous faut pour écrire ou lire des données en fonction de la stratégie utilisée et de la taille des blocs. Les résultats sont visibles dans la console.

La première amélioration consiste à exporter les résultats dans un fichier. En effet, il est fastidieux de devoir relever toutes les valeurs depuis la console, particulièrement si les mesures sont nombreuses. Avoir un fichier externe résout ce problème. Ce dernier est au format CSV, ce qui facilite son traitement (si l’on veut le parser ou l’importer comme feuille de calcul, par exemple).

Une seconde amélioration est l’ajout de tests pour des tailles de blocs différentes. En effet, si l’on veut avoir une idée plus précise du comportement du temps d’exécution en fonction de la taille des blocs, il est préférable de tester d’autres valeurs. Ce n’est donc plus des tailles de bloc de 500 à 5 que l’on testera mais de 50 000 000 à 5 (toujours en divisant la taille par 10 à chaque fois).

Implémentation des améliorations

Voyons à présent comment ont été implémentées les améliorations proposées précédemment.

* Pour la création du fichier CSV, un FileWriter est utilisé et les valeurs qui sont affichées dans la console sont aussi écrites dans le fichier. Pour ce faire, l’approche choisie ici consiste simplement à écrire dans le fichier juste après chaque log dans la console.
* L’ajout de tests consiste simplement en une boucle qui commence par tester une taille de 50 000 000 (plutôt que de 500) et qui finit par 5.

Analyse des résultats

Pour comparer les résultats obtenus, représentons-les sur des graphiques. Ces derniers représentent la taille des blocs (Block size) sur l’axe des abscisses et le temps d’exécution (Duration, en millisecondes) sur l’axe des ordonnées. Des échelles logarithmiques sont utilisées ici pour les deux axes afin de mieux présenter les données.

Intéressons-nous tout d’abord aux opérations d’écriture (WRITE, figure 1).

On peut voir dans un premier temps

Conclusion