Christophe Peretti - https://github.com/oneil78/Res-Labo2

Réseaux – Laboratoire 2

But

Le but de ce laboratoire est de mettre en place une série de classes permettant de traiter des données d'expérimentation, de travailler avec les entrées-sorties en java, et de mettre en place une méthode de sérialisation de ces données (format csv). Le format csv va nous permettre de pouvoir facilement trier les données de performance et de pouvoir en tirer des graphiques par exemple. Les performances testées sont les temps mis pour lire et écrire des fichiers Byte par Byte ou Bloc par Bloc dans avec des flux bufferisé ou non.

Méthode

J'ai choisi de mettre en place la structure proposée dans le webcast, c'est-à-dire de définir trois interfaces (IData, IRecorder, et ISerializer) ainsi que trois classes (MyExperimentData, CsvSerializer, FileRecorder), implémentant ces trois interfaces.

MyExperimentData :

Cette classe contient une HashMap<String, Object> nous permettant de stocker les données, avec différentes clés. Elle contient aussi deux méthodes, une pour obtenir toutes les clés et l'autre pour avoir l'objet correpondant à une clé spécifique.

CsvSerializer :

Cette classe va nous permettre de placer une chaine de texte dans un flux reçu en paramètre. La chaine est formée à partir des data aussi reçues en paramètre. Dans notre cas, le flux va correspondre à un fichier csv.

FileRecorder :

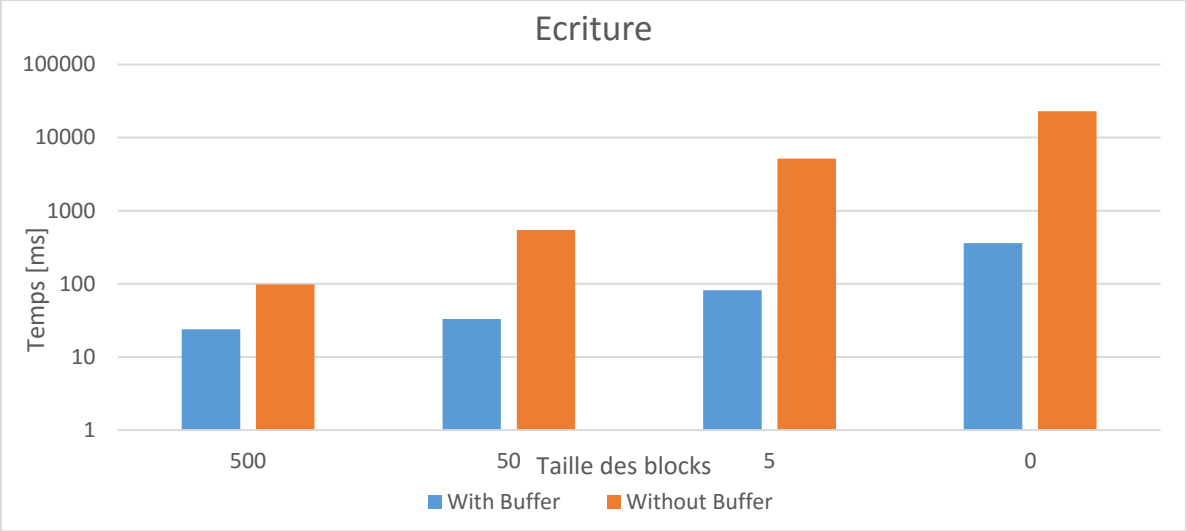
C'est cette classe qui va, à partir d'un nom de fichier, lui associer un flux de sortie, et lancer la sérialisation des données afin de les écrire dans le fichier.

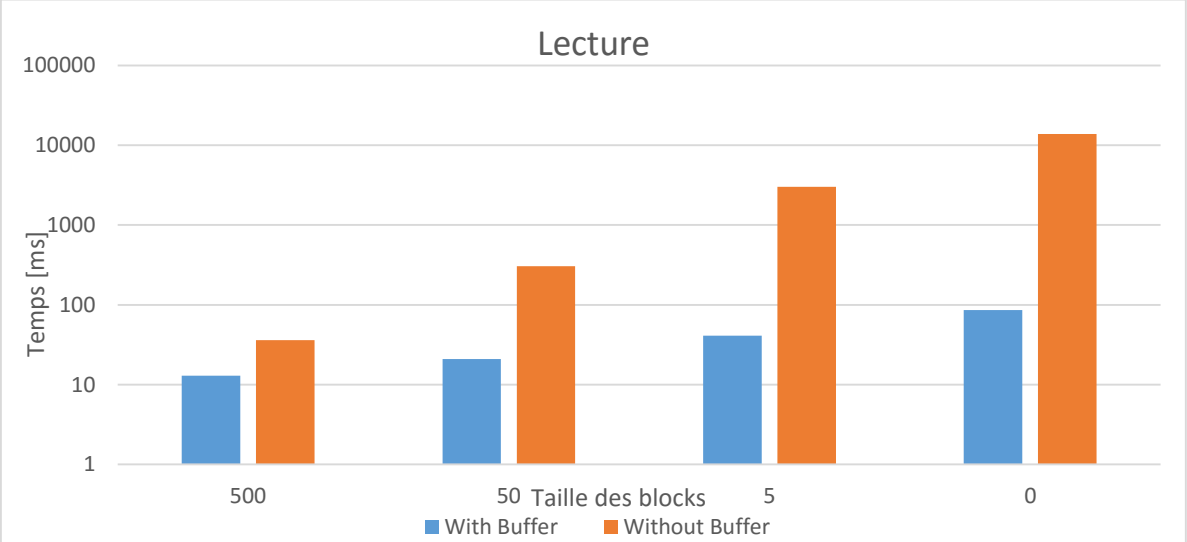
Résultats :

WRITE	BlockByBlockWithBufferedStream	500	10485760	24
WRITE	BlockByBlockWithBufferedStream	50	10485760	33
WRITE	BlockByBlockWithBufferedStream	5	10485760	82
WRITE	ByteByByteWithBufferedStream	0	10485760	360
WRITE	BlockByBlockWithoutBufferedStream	500	10485760	98
WRITE	BlockByBlockWithoutBufferedStream	50	10485760	548
WRITE	BlockByBlockWithoutBufferedStream	5	10485760	5182
WRITE	ByteByByteWithoutBufferedStream	0	10485760	22874
READ	BlockByBlockWithBufferedStream	500	10485760	13
READ	BlockByBlockWithBufferedStream	50	10485760	21
READ	BlockByBlockWithBufferedStream	5	10485760	41
READ	BlockByBlockWithBufferedStream	0	10485760	86
READ	BlockByBlockWithoutBufferedStream	500	10485760	36
READ	BlockByBlockWithoutBufferedStream	50	10485760	305
READ	BlockByBlockWithoutBufferedStream	5	10485760	3020
READ	ByteByByteWithoutBufferedStream	0	10485760	13783

Christophe Peretti - https://github.com/oneil78/Res-Labo2

Ce tableau est le résultat du fichier csv produit avec les données, en ayant choisi le délimiteur « ; ». La lecture du fichier avec Excel organise automatiquement les colonnes en supprimant le délimiteur. Cela nous permet d'en tirer des graphiques très facilement, sans avoir à copier-coller les données une par une.





Nous pouvous voir sur ces graphiques que le temps de lecture/écriture de flux sans buffer prend beaucoup plus de temps qu'avec un flux bufferisé. L'échelle étant logarithmique, la différence entre les deux est bien plus que du simple au double. Avec des blocs de petite taille, la différence est encore plus marquée. Nous pouvons voir que globalement, les résultats sont assez similaires lors de la lecture et de l'écriture, même si c'est toujours plus rapide de lire que d'écrire.

Les flux bufferisés permettent d'écrire des blocs de bytes sans nécessairement faire appel au système à chaque byte à écrire. Cela raccourci donc grandement la procédure.