Évaluation de performances
Rapport du TP

Par Yutai Zhao
Github : https://github.com/yutaizhao

Table des matières

1 Introduction

1.1 Objectif

Consigne du TP

En utilisant fio, vous allez devoir répondre à des énoncés vous demandant d'effectuer des comparaisons entre plusieurs tâches I/O en faisant varier un (ou plusieurs) paramètre(s). Ces comparaisons prendront la forme de graphes de bande-passante générés par vos soins et d'un texte mettant en avant les différences observées et présentant une réflexion sur la raison de ces différences. Pour l'énoncé 3, vous présenterez également un graphe de déciles des latences observées.

Organisation du rapport

La présentation suivra l'ordre des énoncés.

Selon le consigne, pour les énoncés 1-4, ces éléments seront fournis :

— Des graphes de bande-passante
— Des textes descriptifs sur les différences observées
— Des textes analytiques sur la raison de ces différences

Le graphe de déciles des latences sera spécialement présentée pour l'énoncé 3, ainsi que 2 graphes de percentiles supplémentaires pour les analyses.

1.2 Environnement

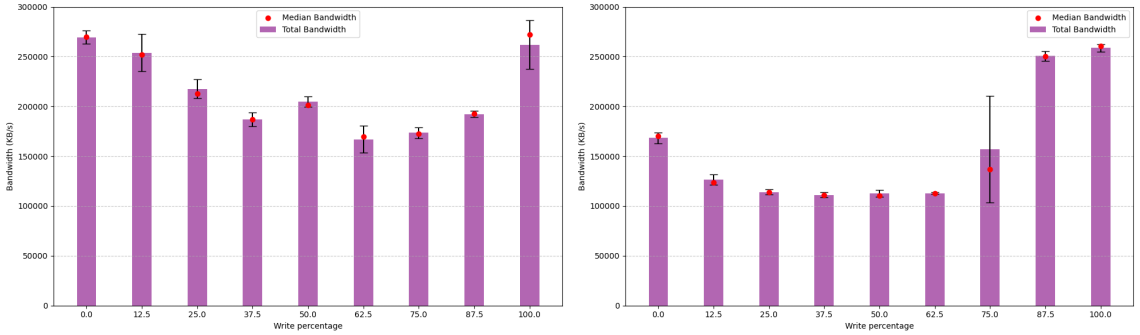
— OS : macOS
— CPU : Intel i5
— Disque/Taille : SSD/500GB
— fio version : fio-3.38

2 Énoncés

Par defaut, les parametres sont : -runtime=60 -numjobs=1 -rwmixwrite=30 -blocksize=16k -filesize=1G -rw=randrw -direct=0

2.1 Énoncé 1

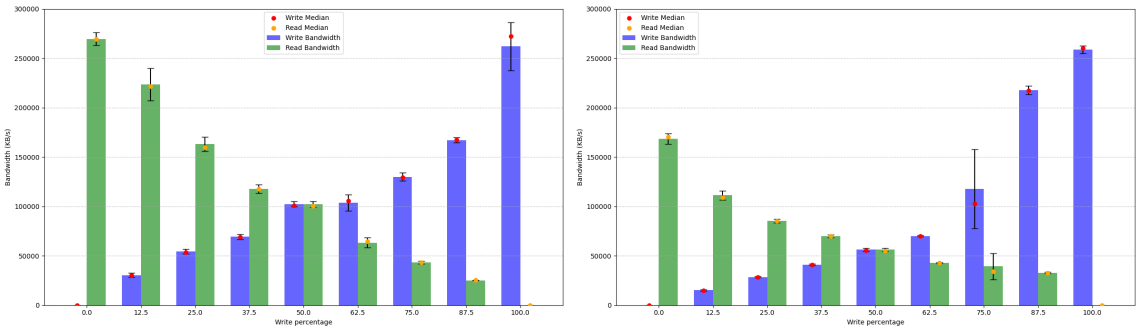
Tâches avec un pourcentage d'écriture variant de 0 à 100% par pas de 12.5 pour des accès aux données séquentiels, puis aléatoires



(a) Séquentiel : Bande-passante totale (b) Aléatoire : Bande-passante totale

FIGURE 1 – Comparaison des performances des tâches en lecture/écriture séquentielle et aléatoire

En général, on observe que les bandes passantes de la version séquentielle sont plus basses que celles de la version aléatoire, cela se voit directement à partir de Fig1.a et Fig1.b. Il est en particulier évident pour les premières bandes passantes, quand le pourcentage de l'écriture est faible, autrement-dit quand les requêtes s'agissent principalement de la lecture. On va ainsi séparer les bandes passantes pour les accès en lecture et en écriture afin de pouvoir analyser plus en détail les différences entre les versions séquetielle/aléatoire.



(a) Séquentiel : Lecture/Écriture (b) Aléatoire : Lecture/Écriture

FIGURE 2 – Comparaison des performances des tâches en lecture/écriture séquentielle et aléatoire

On observe que pour les accès séquentiels, la bande passante d'écriture augmente de manière linéaire avec le pourcentage de écriture(Fig2.a) et la bande passante de lecture diminue ainsi. En revanche, pour les accès aléatoires, la bande passante d'écriture augmente de manière brutalement quand le pourcentage de écriture atteint 75%, qui donne en même temps un écart-type élevé(Fig2.b).

On va analyser cette différence des bandes passantes en consultant d'abord 2 références. Dans cet article 'Blog : Understanding I/O: Random vs Sequential', on nous explique que dans le cas de HDD, cela peut s'expliquer par le fait que, lors des accès séquentiels, les données sont situées de manière contiguë sur le disque, ce qui permet de minimiser le temps de recherche. Cependant, on est dans le cas de SSD, où l'auteur déclare l'absence de ce concept de contiguïté des donnés. Or selon 'Superuser : there is no physical concept of blocks being adjacent or contiguous ?' on nous confirme que envoyer une requête d'I/O à un disque a des coûts supplémentaires que celui du transfert de données, le système d'exploitation prend du temps pour la préparer. Mais en réalisant des accès I/O séquentiels, les opérations peuvent être optimisées par le matériel, il peut combiner les requêtes à traiter. En effet ces références ne nous ont pas vraiment précisé les facteurs qui peuvent engendrer ces coûts supplémentaires, c'est ce que l'on va essayer de découvrir dans la suite du rapport avec nos observations.

Lecture : On nous a confirmé que des lectures séquentielles peut être réalisées en une seule opération d'I/O. Cela s'aligne parfaitement avec notre observation sur la lecture où la bande passante de lecture est très élevée dans la version séquentielle par rapport à la version aléatoire.

On remarque aussi que la bande passante de lecture est plus élevée que celle de l'écriture, car cette dernière déclenche la garbage collection, que l'on va en apporter dans la suite.

Ecriture : Discutons maintenant le cas de l'écriture où l'effet de 'Write amplification' pouvant se produit fréquement, c'est là où la garbage collection joue un rôle majeur.

Tout d'abord, cherchons à connaître comment un accès en écriture en SSD est effectué :

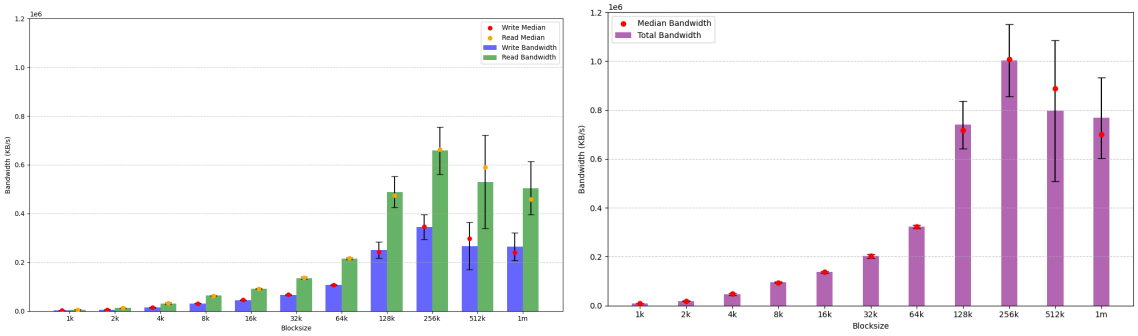
SSD est constitué principalement de mémoires flash, un mémoire flash est constitué de blocs, un bloc est constitué de pages(souvent de tailles 4kb/8kb). Cette mémoire flash ne peut que écrire mais pas réécrire une page, il faut d'abord effacer le contenu existant pour pouvoir écrire de nouveau. Cela doit se faire au niveau du bloc entier. Ainsi, lorsqu'il n'y a plus de pages libres, il est nécessaire de trouver un bloc qui contient des données invalides, d'effacer le bloc entier (y compris toutes les pages dedans), puis de choisir une page libre pour y écrire les nouvelles données.

Ainsi, les opérations d'écriture nécessitent souvent des effacements préalables de blocs de mémoire sur les SSD. Cela peut entraîner une variabilité accrue dans le temps de traitement des requêtes d'écriture. On peut estimer donc lors de l'écriture des données séquentielles, le bloc de mémoire flash entier est rempli de manière séquentielle avec des données liées au même fichier. Si le système d'exploitation décide de effacer un bloc, l'ensemble du bloc peut être marqué comme invalide, sans qu'il soit nécessaire de prendre le temps de lire certaines parties de plusieurs blocs, collecter les données et les réécrire dans un autre bloc.

Maintenant concentrons nous dans la version aléatoire, concernant des basses bandes passantes observées avant le pourcentage d'écriture augmente 75%, cela peut dues au fait que à faible pourcentage d'écriture, le SSD doit fréquemment effectuer des opérations d'effacement et d'écriture, augmentant ainsi l'effet d'amplification d'écriture. Mais lorsque la proportion d'écriture devient élevée, la plupart des accès sont en écriture, donc le système a une grande chance de bénéficier un maximum des écritures dans un requête, donc il peut faire beaucoup d'écriture (opérations) en un requête et cela réduit l'effet d'amplification. L'écart-type élevé à 75% peut donc être attribué au fait que parfois l'effet d'amplification survient et parfois non, comme on peut toujours tomber sur des accès fragmentés en écriture/lecture.

2.2 Énoncé 2

Tâches avec une taille de requête variant de 1ko à 1Mo par pas de puissance de 2



(a) Blocksize : Lecture/Écriture

(b) Blocksize : Bande-passante totale

FIGURE 3 – Comparaison des performances des tâches en variant la taille de requêtes

On observe que la bande passante augmente avec la taille des blocs jusqu'à atteindre environ 256 kb. Cependant, lorsque la taille des blocs devient plus grande, à savoir entre 512 Kb et 1 Mb, la bande passante se réduit.

Précédemment, nous avons déjà vu que la taille des requêtes pouvait influencer le nombre d'opérations par seconde (IOPS). Dans ce cas là, vu que les accès sont fixes (accès aléatoire avec 30% d'écriture), il est probable que d'autres facteurs influencent la bande passante.

En effet, on sait que la bande passante est déterminée par la formule suivante :

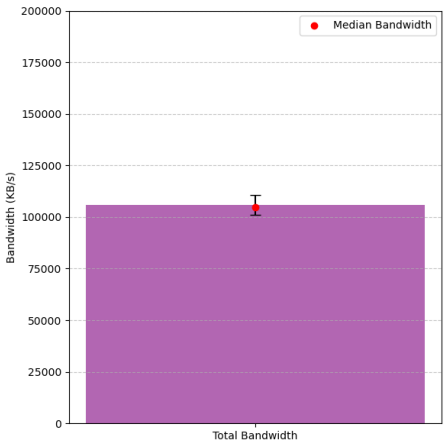
Bande passante = Taille de I/O × IOPS.

Sachant que la taille de IO représente le volume de données transféré par un requête, on peut en déduire que la taille maximale de IO que le système peut gérer de manière optimale est d'environ 256 kb, et que la quantité de données avec cette taille à transférer par requête est déjà très importante. Ainsi, lorsque la taille d'un requête dépasse cette limite, le système doit prendre du temps supplémentaire pour diviser cette requête en plusieurs requêtes plus petites pour les traiter, ce qui se traduit par une réduction de la bande passante. La taille de requête est donc un facteur qui peut influencer la bande-passante.

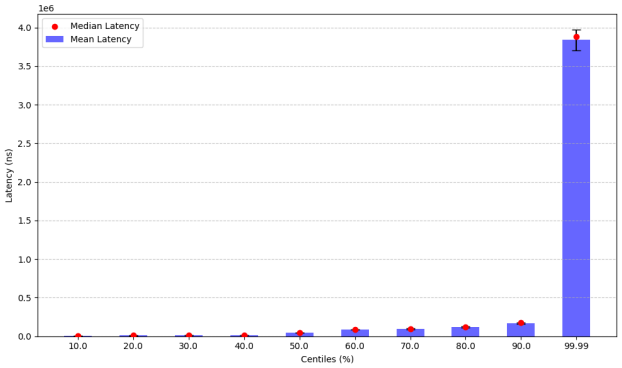
Par ailleurs, ce résultat contribue également à mieux comprendre le cas précédent des accès en lecture. Nous pouvons en déduire que c'était principalement IOPS qui influençait la bande passante, sachant que la taille de requête de 16 kb est inferieure à la taille maximale optimale 256 kb. Cela implique qu'une optimisation automatique sur les accès séquentielles en lecture a probablement été réalisée, permettant ainsi une augmentation de IOPS par rapport aux accès aléatoires, ce qui a contribué à maintenir une bande passante plus élevée.

2.3 Énoncé 3

Tâche de requêtes en écriture avec 30% de requêtes de 4ko, 60% de 16ko et 10% de 64ko (bssplit)



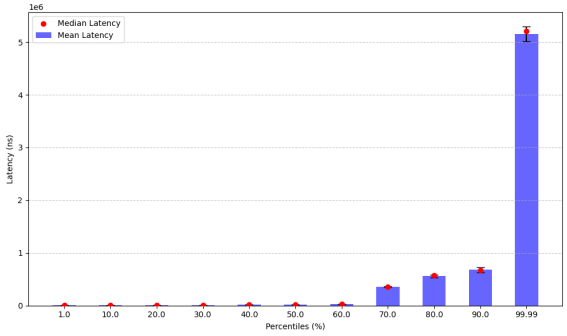
(a) bssplit : Bande-passante totale



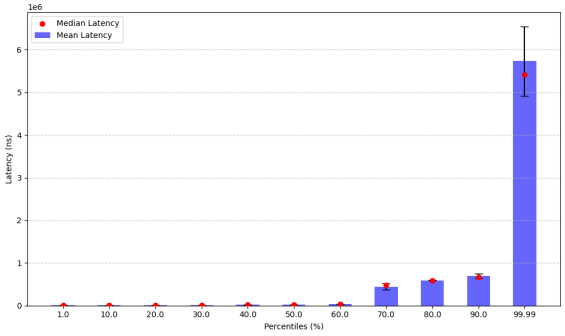
(b) bssplit : Latence

FIGURE 4 – Comparaison des performances des tâches avec bssplit = 4k :30,16k :60,64k :10

On observe que pour les centiles inférieurs (de 10%à 90%), les latences sont très faibles. Au centile 99,99% considéré comme 100%, la latence moyenne augmente considérablement, par rapport aux autres centiles. Remarquons d'abord que $64kb < 256kb$ la taille optimale trouvée dans la partie précédente, donc à priori cette augmentation n'est pas engendrée par le fait que la taille soit trop grande. On peut donc estimer que les raisons principaux de ce temps supplementaire se resident dans un autre problème de la gestion des requêtes ou/et sur l'amplification d'écriture. On a ainsi fait encore 2 tests :



(a) bssplit : 64k :30,256k :40,1m :30



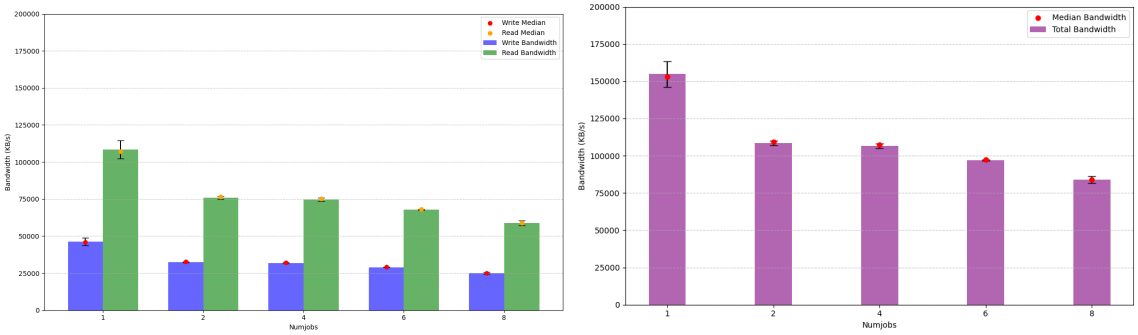
(b) bssplit : 64k :30,256k :40,512k :20,1m :10

FIGURE 5 – Comparaison des performances des tâches avec bssplit différents (1% inclus)

En effet, on peut observer que la taille des requêtes jouent un rôle comme déduit précédement, pour les tailles initiales 4kb-64kb on n'atteint même pas $4 * 10^6$, cependant pour 64kb-1mb on peut atteindre $5 * 10^6$ ns(Fig5.a) et $6 * 10^6$ ns (Fig5.b) , mais cette condition n'explique pas la différence entre les 2 graphes de Fig5 comme les gammes des tailles des requêtes sont identiques. En effet la différence c'est que dans le cas où la latence est plus élevée (Fig5.b) possède une variation plus importante des tailles de requêtes. On peut faire une dernière comparaison de la bande passante de Fig4.a avec les denières bandes passantes de Fig2, car dans ces cas toutes les conditions sont identiques : avec le pourcentage d'écriture à 100%, sauf que dans ce 3eme énoncé on a fait une variation des tailles de requêtes et cela a engendré un baisse d'environ $2.5 = \frac{250000}{100000}$ de bande passante. On peut conclure que même si les tailles ne dépasse pas la taille optimale, le système prend le temps de plus pour gérer cette variation de tailles.

2.4 Énoncé 4

Tâches avec 1, 2, 4, 6 et 8 processus tournant en parallèle



(a) numjobs : Lecture/Écriture

(b) numjobs : Bande-passante totale

FIGURE 6 – Comparaison des performances des tâches en versions parallèles

On peut observer que lorsque le nombre de tâches (numjobs) augmente, la bande passante de lecture et d'écriture diminue presque de manière homogène. En plus quand il y a 2 ou 4 processus parallèles, on peut constater que la bande passante de lecture et d'écriture reste relativement stable. D'abord cette diminution de bande passante peut relever un autre problème de la gestion du stockage. Quand plusieurs processus envoient simultanément des requêtes d'IO au SSD, le système doit gérer à la fois tous ces requêtes ce qui augmente le temps de traitement. Par exemple pour assurer l'atomicité de ces régions critiques, le système doit faire de nombreux vérouillages/déverouillages. Ensuite, le comportement similaire des bandes passantes pour les accès en lecture et en ériture peut dû au fait que ces accès partage certains materiels de gestion. Enfin, on peut dire que les ressources du système de gestion pour 2 processus sont suffisantes pour traiter les requêtes envoyés par 4 processus. Cependant, en général les problèmes de concurrence deviennent plus importants quand le nombre des processus augmente, entraînant une baisse des performances, cela indique un état de saturation des ressources comme dans le cas où la taille de requêtes depasse 256kb.

3 Conclusion

Pour conclure, les facteurs qui peuvent influencer la performance IO sont respectivement : Accès aléatoire/séquentiel, Write amplification, taille de requêtes, la variation des tailles de requêtes, les accès concurrents.

4 Référence

— Blog : Understanding I/O: Random vs Sequential
— Superuser : there is no physical concept of blocks being adjacent or contiguous ?
— Blog : SSD principle (Chinois)
— Wikipedia : IOPS
— Wikipdia : Write amplification
— CM2 d'EDP sur Workload