Projet HPCA

GPU Merge Path - A GPU Merging Algorithm

M2 SFPN

Encadrant : Lokmane ABBAS-TURKI

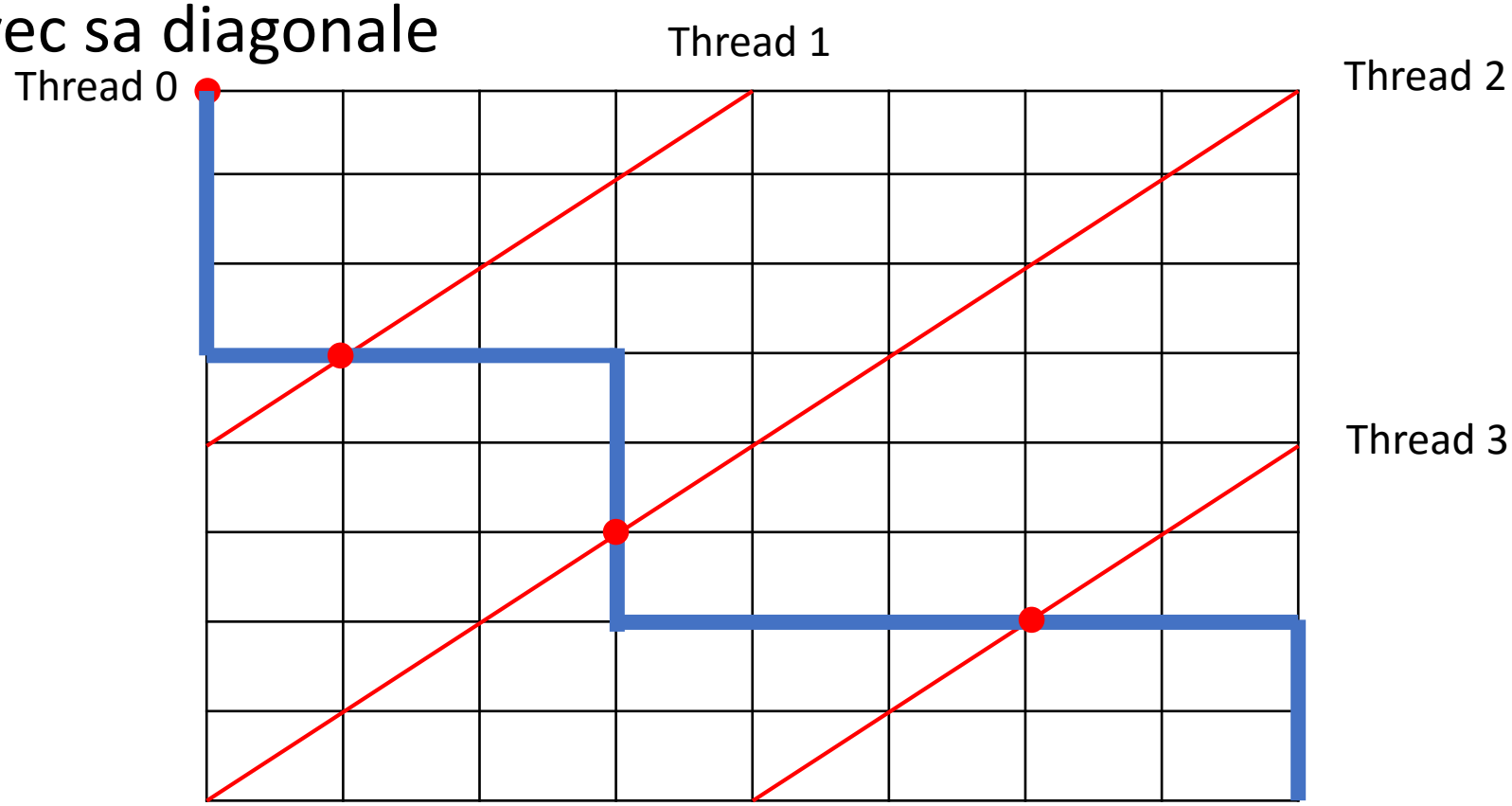
Moussa TIRERA
Phan Nguyet NGUYEN
Su ZHOU

Introduction

- Merge-path nous demande de combiner les deux tableaux déjà triés pour un nouveau tableau trié.
- Implémenter l’algorithme sur GPU en CUDA C.
- Augmenter la performance de notre calcul.

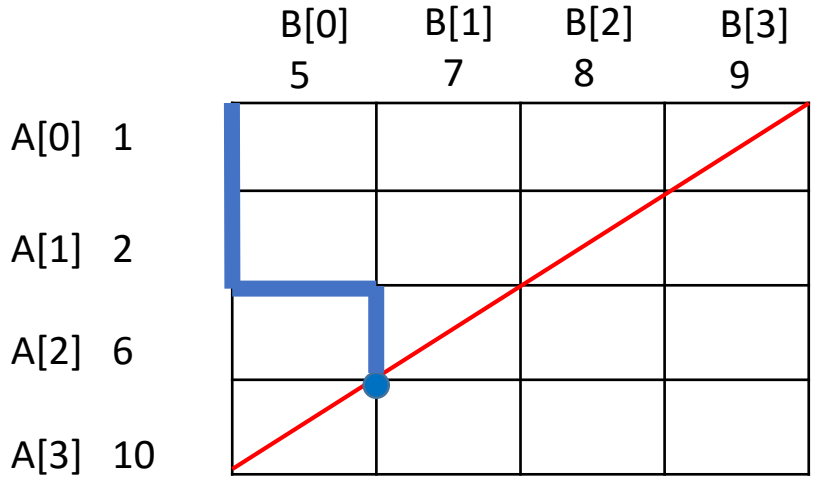
Merge

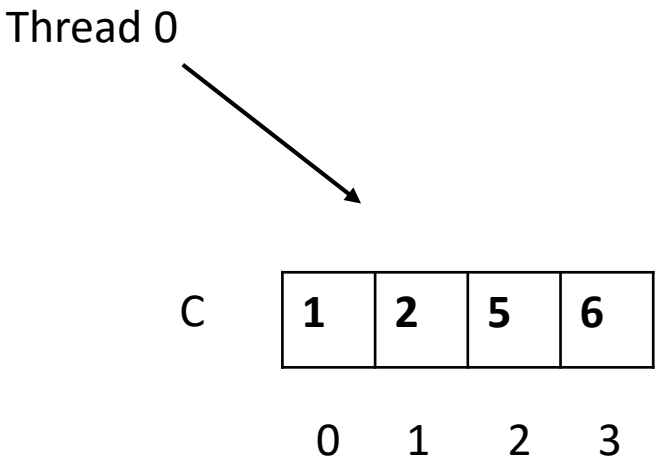
- Partie 1 : Chaque thread cherche le point d'intersection de Merge-path avec sa diagonale



Merge

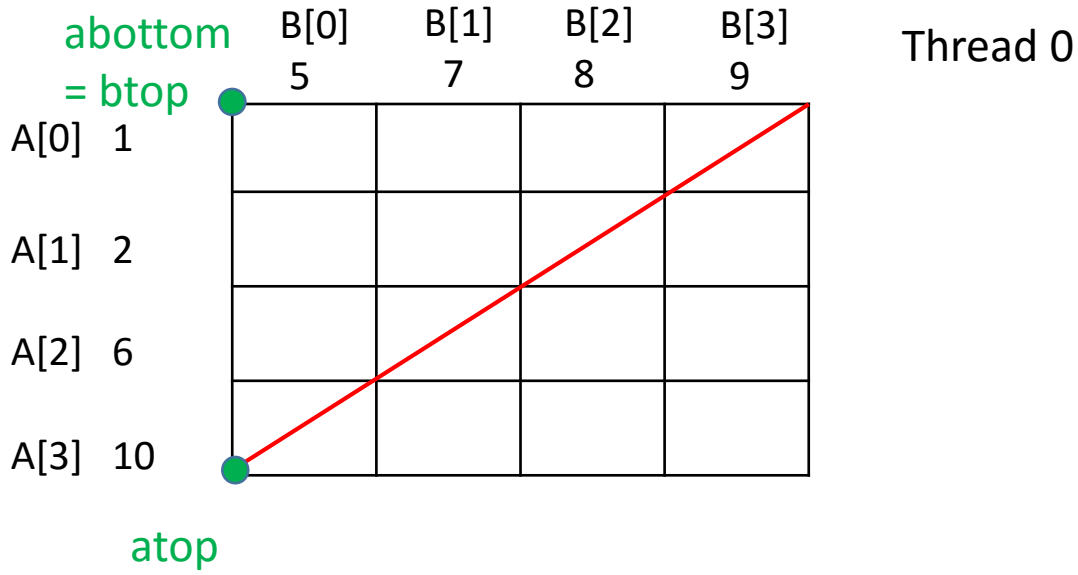
- Partie 2 : Chaque thread merge ses sous-tableaux





Partie 1 : Chercher le point d'intersection par la recherche binaire (dichotomique)

- Chaque thread détermine la matrice

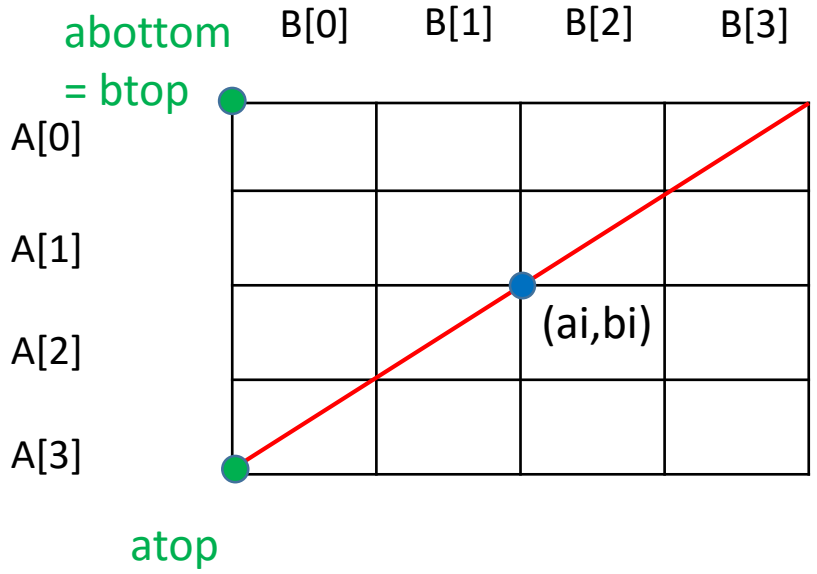


Thread 0

Offset = 2

Partie 1 : Chercher le point d'intersection par la recherche binaire (dichotomique)

- La recherche commence par le point au milieu

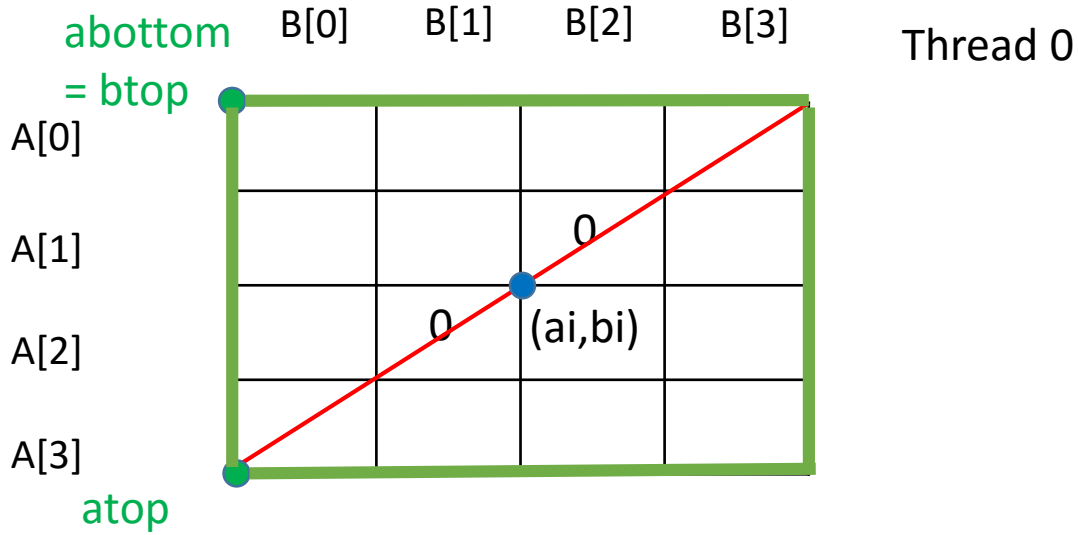


Thread 0

ai = atop – offset
bi = btop + offset

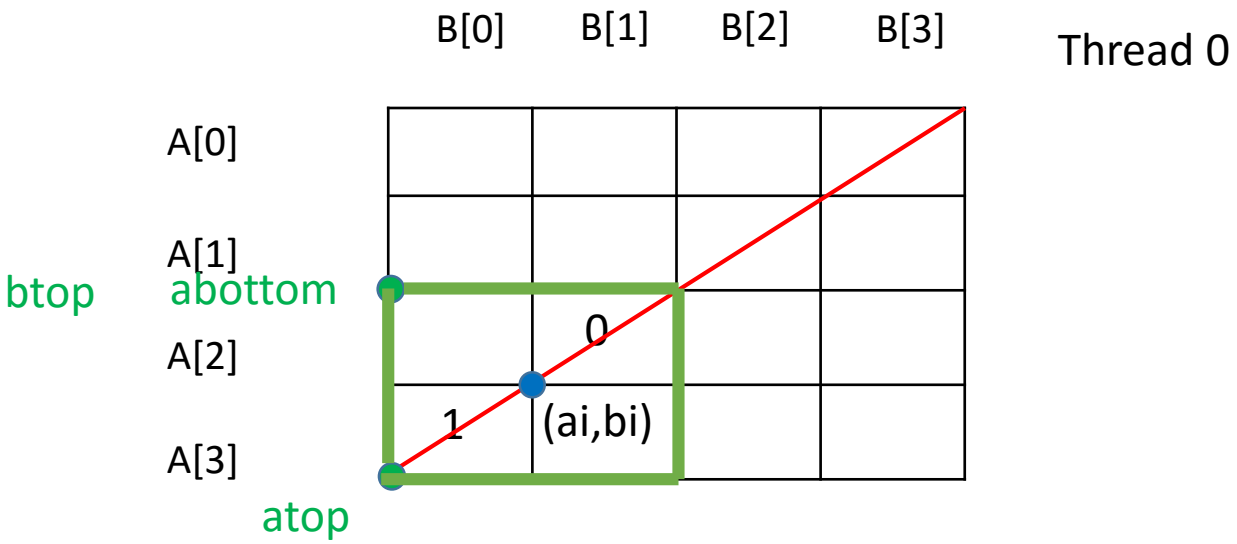
Partie 1 : Chercher le point d'intersection par la recherche binaire (dichotomique)

- Si (0,0) au point milieu sur la diagonale



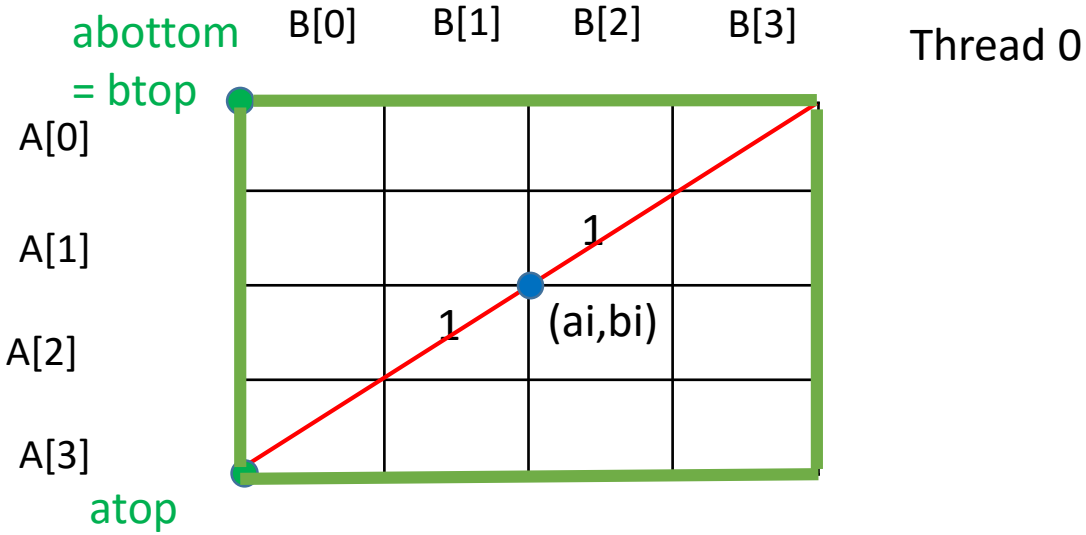
ai = atop – offset
bi = btop + offset

Partie 1 : Chercher le point d'intersection par la recherche binaire (dichotomique)



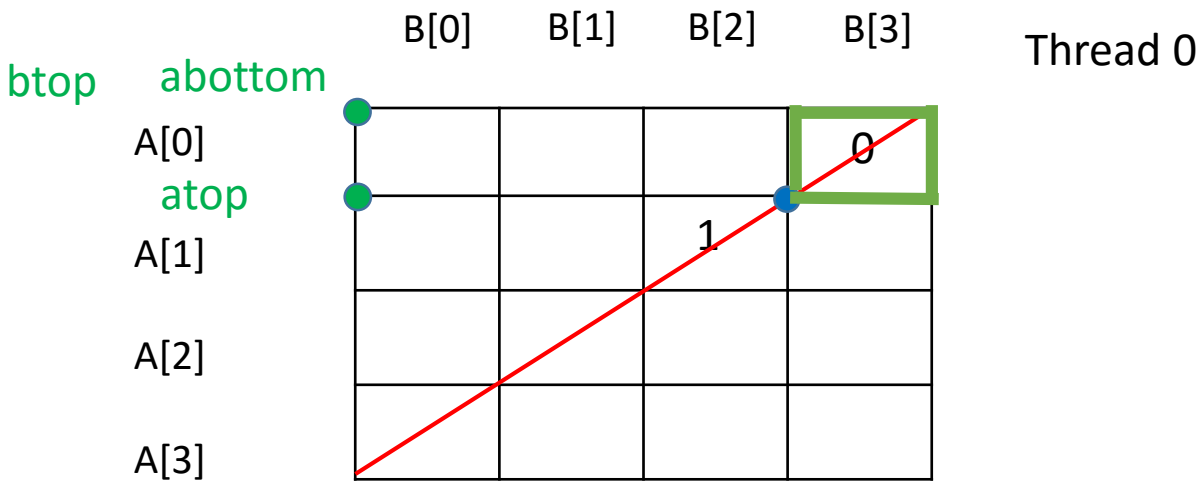
Partie 1 : Chercher le point d'intersection par la recherche binaire (dichotomique)

- Si (1,1) au point milieu sur la diagonale



ai = atop – offset
bi = btop + offset

Partie 1 : Chercher le point d'intersection par la recherche binaire (dichotomique)



Implémentation sur GPU

- Le nombre de threads <= (la taille de tableau C)/ 2

Chaque thread a au moins 2 éléments de C

NEPT = 2

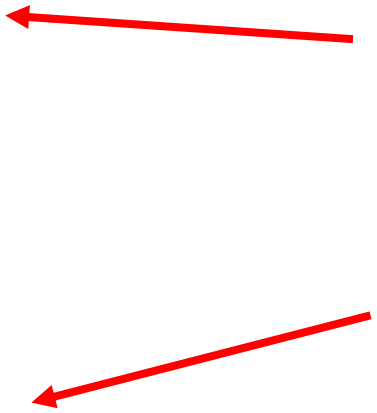
NTPB = SIZE_A / NEPT

NB = (SIZE_C + NEPT * NTPB – 1) / NEPT * NTPB

Implémentation sur GPU

- Lecture sur la mémoire globale

```
/*A la recherche de point d'intersection*/
If(A[ai] > B[bi-1]){
        if(A[ai-1] <= B[bi]){
        …
        }
}
/*Merge*/
While(il y a encore des éléments à comparer){
    If(A[ai] <= B[bi]){
        C[i] = A[ai];
        …
    }
}
```

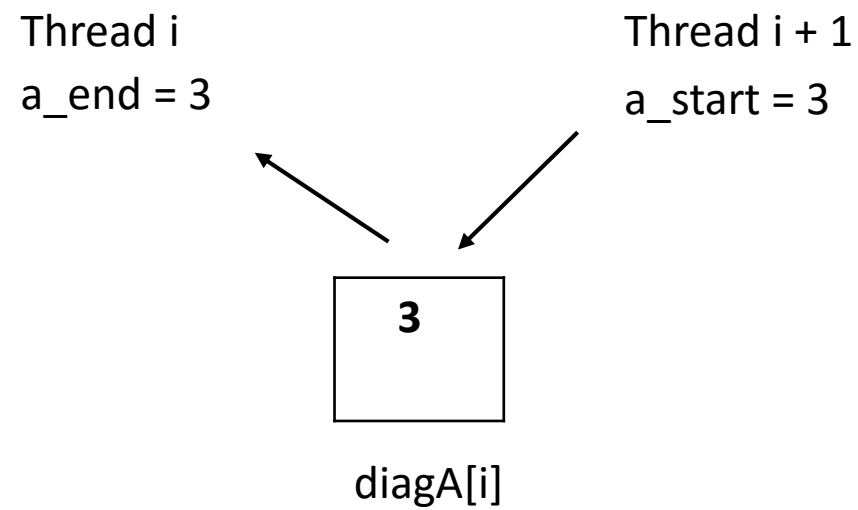


Problème :

lecture sur la mémoire globale

Implémentation sur GPU

```
/*Merge*/
__shared__ int diagA[NTPB];
__shared__ int diagB[NTPB];
```



Solution :

Utiliser deux mémoires partagées pour échanger les points d'intersections entre les threads

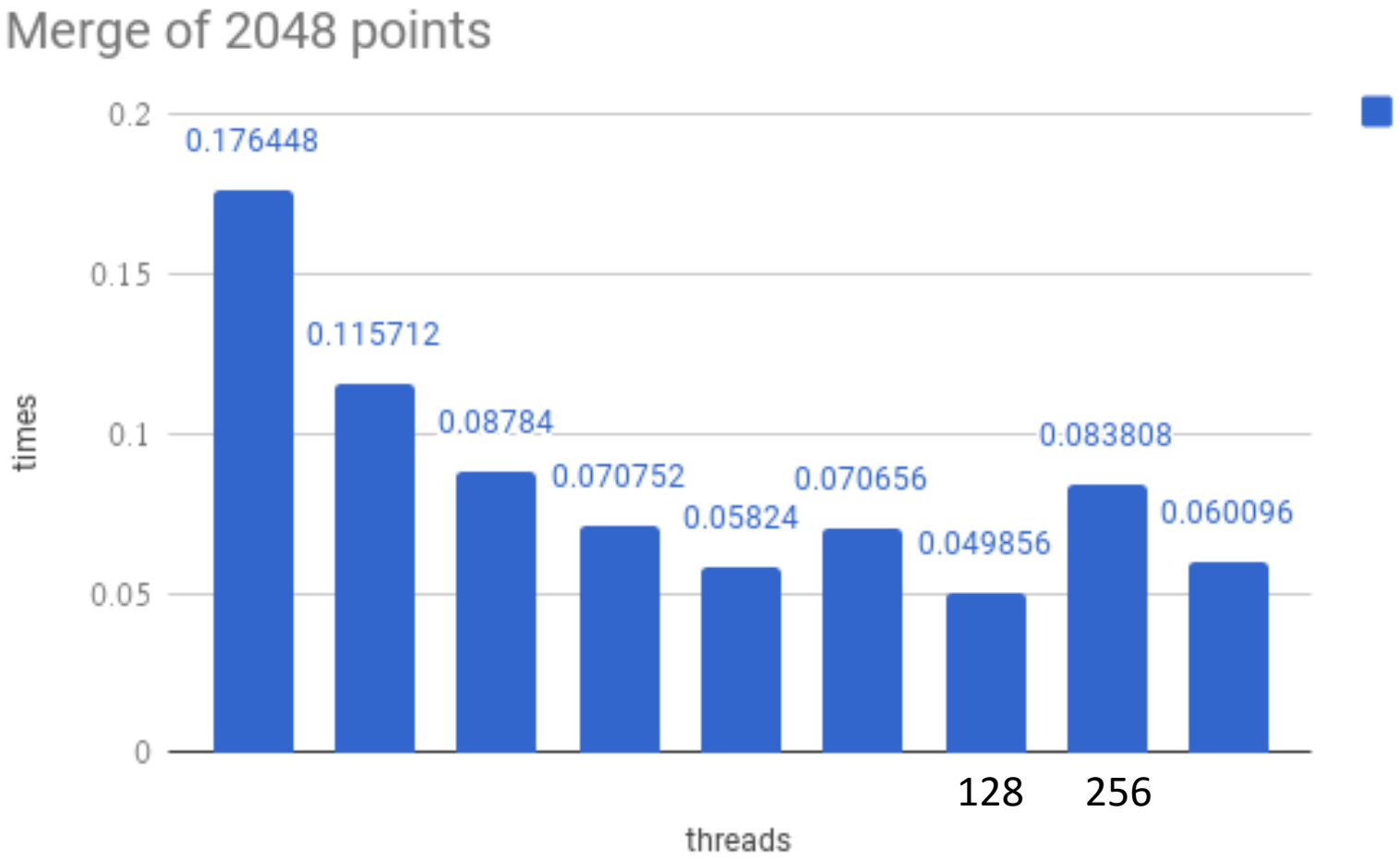
Implémentation sur GPU

```
While(a_start < a_end && b_start < b_end &&..){
        If(A[a_start] <= B[b_start]){
                C[i] = A[a_start];
                …
}
```

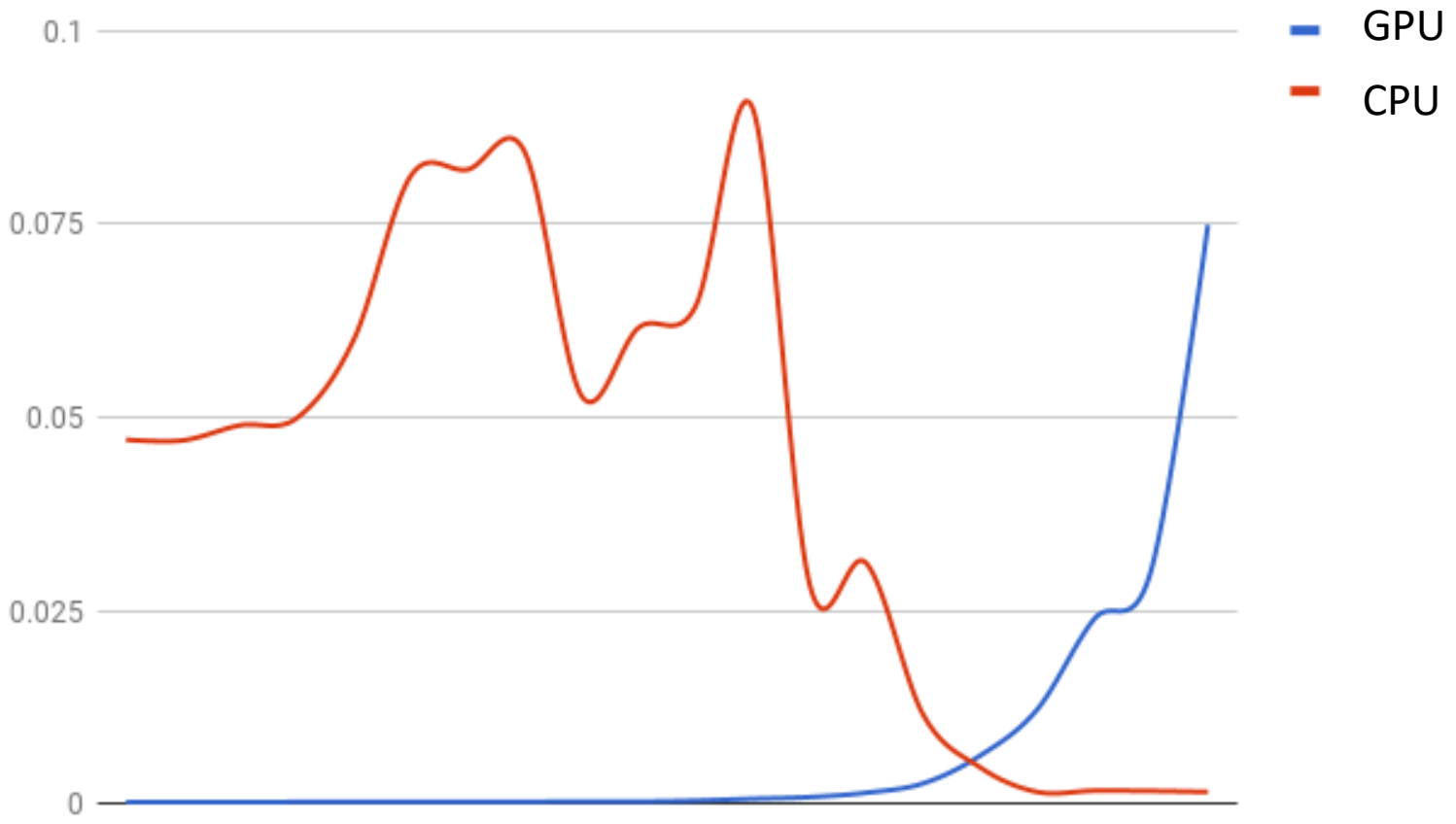
Solution :

Diminuer la lecture sur la mémoire globale

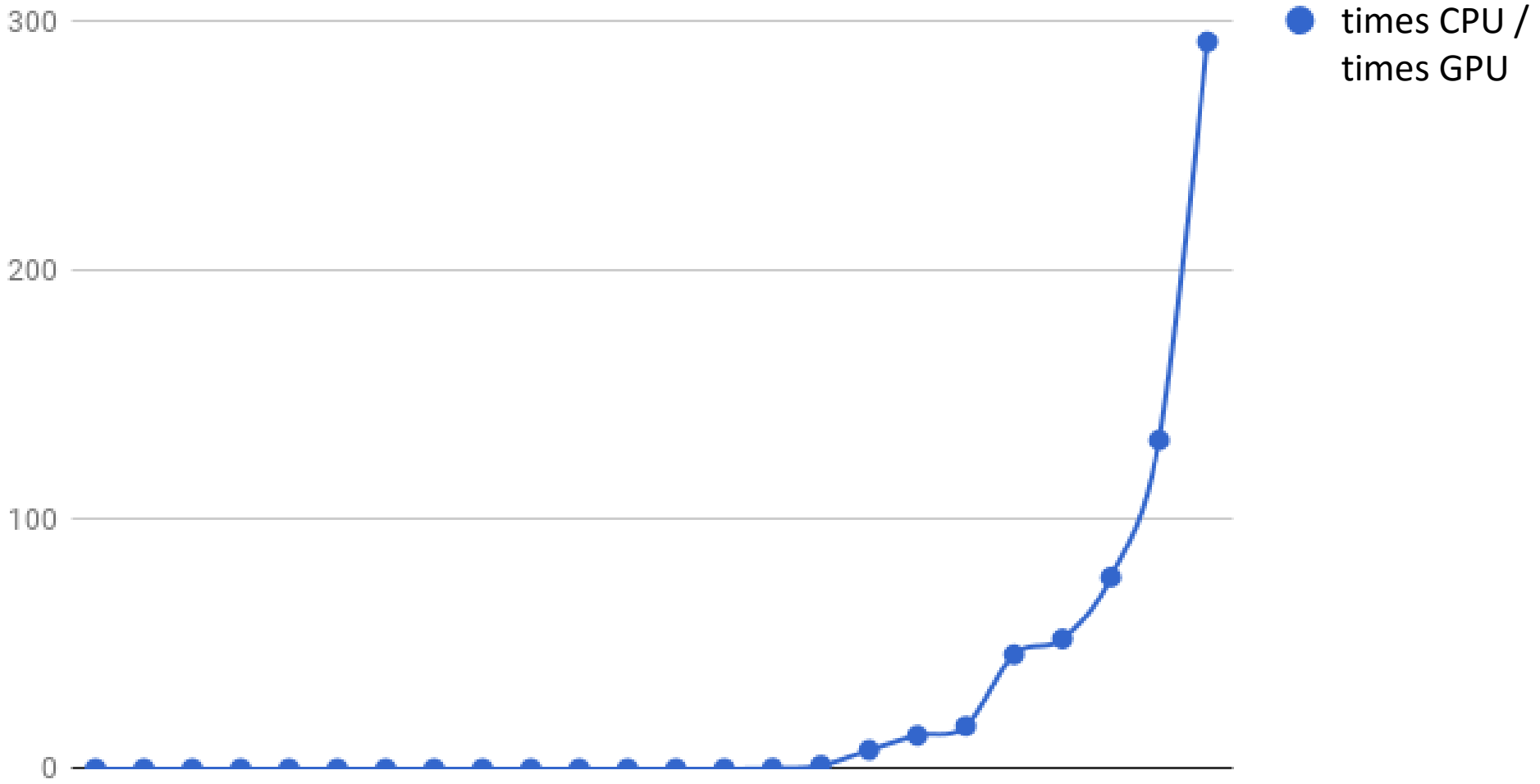
Performance



Pour le même nombre de threads et le même nombre de block

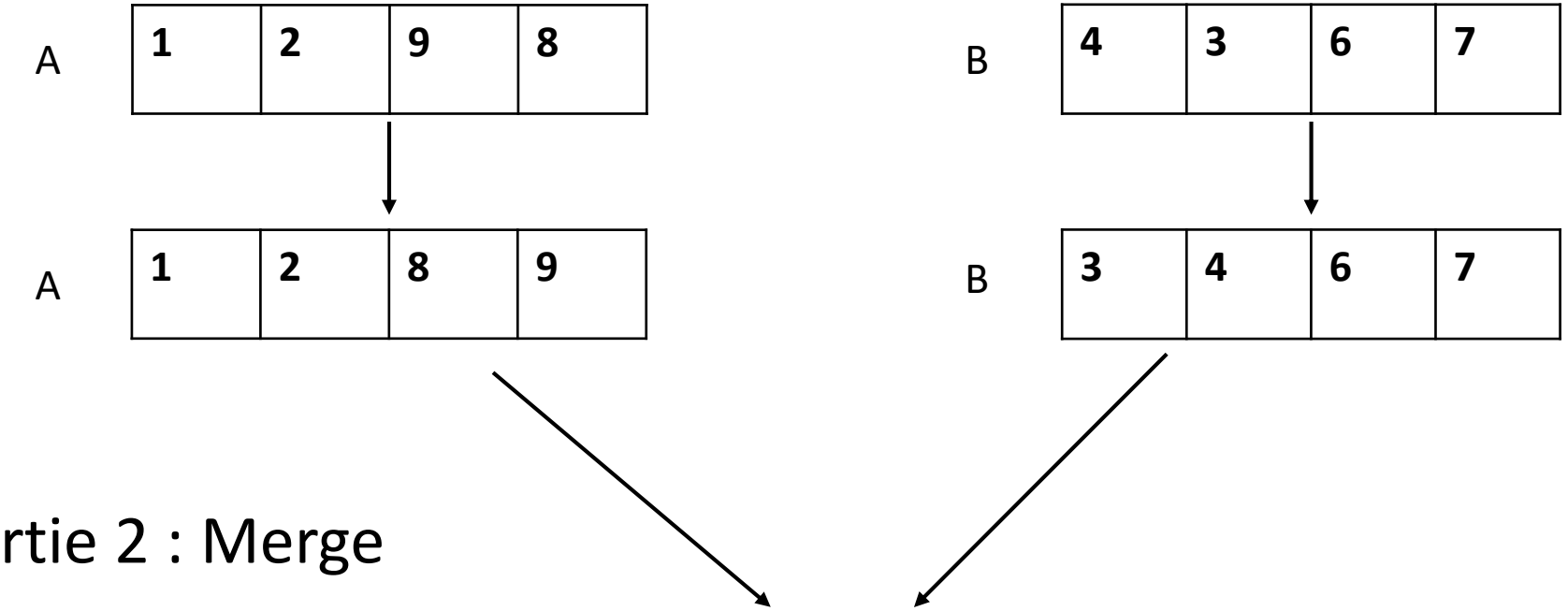


Gain de temps CPU / GPU



Application Merge-Sort

- Partie 1 : Sort



- Partie 2 : Merge

Conclusion

- Comprendre le parcours de la méthode Merge-path.
- Implémenter la programme parallèle avec CUDA pour augmenter la performance.
- Bien tester les codes et comparer la performance.

Merci !