Faculté sciences & sciences de l'ingénieur
Université Bretagne Sud

Master d'Informatique

UE INF2245

Calcul haute performance pour le «Big Data»

Sujet d'examen terminal

(première session)

Durée : 2 heures
aucun document autorisé

F. Raimbault, N. Courty

mai 2022

Avertissement : Il sera principalement tenu compte dans la notation de la qualité de rédaction, de votre capacité à abstraire une solution et de la rigueur avec laquelle vous la décrivez.

Exercice 1 : Jeu d'instructions SIMD (4 points)

On vous demande de réaliser une fonction de recherche de la valeur max des valeurs de tous les pixels d'une image monochrome.

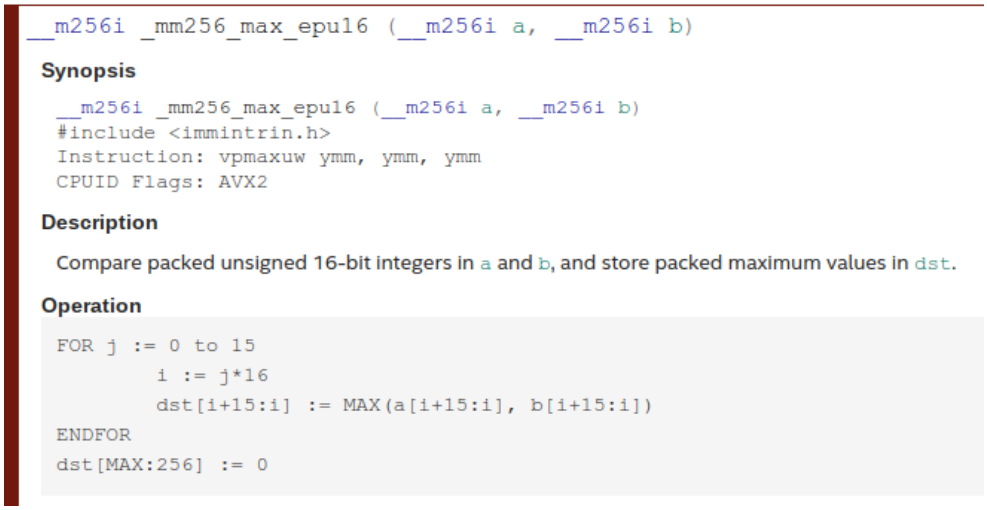
Les valeurs des pixels sont codées sur 16 bits. Les images sont mémorisées dans des tableaux à une dimension de N entiers courts non signés (**unsigned short** de 16 bits en C).

Pour accélérer la fonction de recherche vous choisissez de la programmer à l'aide de fonctions intrinsèques en C pour un jeu d'instructions Intel SIMD.

Vous disposez d'un processeur doté du jeu d'instruction AVX2 sur 256 bits. Dans la librairie de fonctions intrinsèques Intel vous trouvez la fonction `_mm256_max_epu16` dont la documentation est décrite dans la figure 1.

Répondez de manière précise et exhaustive aux questions suivantes (on ne demande pas de programme en C, par contre des schémas peuvent être utiles).

1. Quel va être le principe de la parallélisation de votre fonction de recherche avec des



__m256i _mm256_max_epu16 (__m256i a, __m256i b)

Synopsis

```
__m256i _mm256_max_epu16 (__m256i a, __m256i b)
#include <immintrin.h>
Instruction: vpmaxuw ymm, ymm, ymm
CPUID Flags: AVX2
```

Description

Compare packed unsigned 16-bit integers in a and b, and store packed maximum values in dst.

Operation

```
FOR j := 0 to 15
        i := j*16
        dst[i+15:i] := MAX(a[i+15:i], b[i+15:i])
ENDFOR
dst[MAX:256] := 0
```

FIGURE 1 – Extrait de la documentation en ligne Intel Intrinsics Guide

registres SIMD ? Comment terminer le calcul pour obtenir le résultat final dans un **short int** ?

2. On suppose que N (le nombre de pixels de l'image) est un multiple de 16. Combien d'itérations sont nécessaires (y compris celles pour terminer le calcul) ?
3. Comment procéder si N n'est pas un multiple de 16 ? Combien d'itérations sont alors nécessaires ?

Exercice 2 : Spark (8 points)

L'objectif de cet exercice est d'exprimer une solution en Spark pour rechercher le mot ayant le plus fort TF-IDF dans chaque texte d'une collection de documents. Le travail demandé est décrit dans le dernier paragraphe.

Définition Extrait de Wikipédia : « Le TF-IDF (de l'anglais term frequency-inverse document frequency) est une méthode de pondération souvent utilisée en recherche d'information et en particulier dans la fouille de textes. Cette mesure statistique permet d'évaluer l'importance d'un terme contenu dans un document, relativement à une collection ou un corpus. Le poids augmente proportionnellement au nombre d'occurrences du mot dans le document. Il varie également en fonction de la fréquence du mot dans le corpus.»

Formulation Le TF-IDF de chaque terme w d'un document d est défini formellement par :

$$TFIDF(w,d) = TF(w,d) \times IDF(w) = \frac{N(w,d)}{N(d)} \times \log_{10} \frac{D}{D(w)+1}$$

où :
— $TF(w,d)$ est la fréquence du terme w dans le document d
— $IDF(w)$ est la fréquence inverse de document du terme w
— $N(w,d)$ est le nombre d'occurrences du terme w dans le document d

— $N(d)$ est la somme de toutes les occurrences de tous les termes dans le document d, c-à-d. le nombre total de termes dans le document d.
— D est le nombre total de documents,
— $D(w)$ est le nombre de documents dans lesquels apparaît le terme w ; on ajoute 1 pour éviter la division par 0.

Algorithme On procèdera en cinq étapes qui produiront 4 RDD successifs et la table finale :

1. `RDD_W` : contenant l'ensemble des termes de tous les documents, obtenu par la lecture de l'ensemble des documents d'une collection.
2. `RDD_TF` : contenant la fréquence de tous les termes de tous les documents.
3. `RDD_IDF` : contenant la fréquence inverse de tous les termes
4. `RDD_TFIDF` : contenant le TF-IDF de tous les termes de tous les documents
5. `TABLE_IDF` : contenant pour chaque document son terme ayant le plus fort TF-IDF.

Programmation en Spark Donnez (en langage algorithmique) les actions et transformations successives qui permettent d'aboutir à la table recherchée en prenant soin de détailler les types et fonctions utilisées.

Exercice 3 : CUDA et complexité (8 points)

Tri à bulle. L'algorithme de tri à bulle consiste à regarder les différentes valeurs adjacentes d'un tableau et à les permuter si le premier des deux éléments est supérieur au second.

L'algorithme se déroule ainsi :
— Les deux premiers éléments du tableau sont comparés, et si le premier est supérieur au second on les permute.
— On compare ensuite les éléments 2 et 3 et on les permute si l'élément 2 est supérieur à l'élément 3
— Ainsi de suite jusqu'à la comparaison des éléments (n-1) et n
— On repart ensuite du début du tableau et on réitère le procédé n fois.

Questions :

1. Donnez une version de cet algorithme en Python. Donnez sa complexité dans le pire cas et au meilleur cas
2. **Amélioration** : après un parcours du tableau la case n du tableau contient le max, pour le deuxième parcours on ne va donc que jusqu'à la comparaison (n-2) et (n-1) et ainsi de suite. Cette amélioration change t'elle la complexité au pire cas (justifiez votre réponse) ?
3. **CUDA**.
   (a) Quelle difficulté allez vous rencontrer pour implémenter cet algorithme avec les principes de programmation en Cuda ?

(b) Proposez une implémentation en PyCuda d'un noyau effectuant la permutation ou non de deux éléments successifs dans le tableau (on supposera que la taille du tableau est un multiple de 2 pour simplifier un peu les choses).

(c) Donnez finalement le code python réalisant les appels successifs de ce noyau.

(d) (**Bonus**) en utilisant la fonction `_syncthreads()` qui permet de synchroniser tous les threads lors de l'exécution d'un noyau, proposez éventuellement une fonction réalisant le tri en un seul noyau.