Programmation sur Processeur Graphique – GPGPU
TD 9 : algorithme de réduction

Centrale Nantes
P.-E. Hladik, pehladik@ec-nantes.fr

—

Version bêta (8 décembre 2022)

1 Produit vectoriel

Objectif 1.1

— observer l'influence des warps
— mettre en œuvre un algorithme d'arbre de réduction

(1.1) Travail à faire : Dot

Implémentez un kernel qui effectue le produit vectoriel (dot en anglais) pour deux vecteurs a et b.

Votre noyau devrait être capable de gérer des vecteurs d'entrée de longueur arbitraire. Cependant, pour des raisons de simplicité, vous pouvez supposer que les vecteurs d'entrée comptera au maximum 2048 x 65535 éléments afin qu'ils puissent être traitée par un seul lancement de kernel.

La condition limite peut être traitée en remplissant par 0 la mémoire partagée du dernier bloc lorsque la longueur n'est pas un multiple de la taille du bloc. Supposons en outre que les réductions produites par les blocs seront additionnées par le CPU. On pourrait aussi effectuer la réduction du vecteur de sortie récursivement et prendre en charge toute taille d'entrée. Pour des raisons de simplicité, nous ne le mettrons pas en place ici (sauf ceux qui sont en avance).

Pour vous aider, vous pouvez utilisez le squelette fournit dans le fichier `dot.cu` et proposez deux versions, l'une sans prendre en compte les warp et l'autre avec. Comparez les performances.