**Set 3 - OPENMP and GPUs**

**Φοιτητής 1**

Ονοματεπώνυμο: Αγγουρά Ρουμπίνη Μαρία

ΑΜ: 1084634

Έτος Σπουδών: 5ο

**Φοιτητής 2**

Ονοματεπώνυμο: Παυλόπουλος Ιάσονας

ΑΜ: 1084565

Έτος Σπουδών: 5ο

## **Επεξήγηση Υλοποίησης**

**Ερώτημα 1**

Για την υλοποίηση του complex matrix multiplication, ακολουθούμε τα εξής βήματα:

* Αρχικοποιούμε τις θέσεις μνήμης των μεταβλητών που θα χρειαστούμε στον host (CPU).
* Εισάγουμε τυχαίες τιμές (με την rand()) σε κάθε μία από τις μεταβλητές.
* Αρχικοποιούμε τις θέσεις μνήμης των μεταβλητών που θα χρειαστούμε και στο device (GPU).
* Με την **#pragma omp target,** κάνουμε execute τον κώδικα υπολογισμού του πολλαπλασιασμού στην GPU.
* Με την **teams distribute parallel for**, χωρίζουμε τα loops σε διαφορετικές ομάδες threads στην GPU και τα εκτελούμε παράλληλα
* Με την **collapse(2)**, ενώνουμε τα for loops για I και j για καλύτερο παραλληλισμό.
* Με την **map(to: A, B, C, D),** αντιγράφουμε τα μητρώα A,B,C,D από το CPU στο GPU
* Με την **map(from: E, F),** αντιγράφουμε τα αποτελέσματα πίσω στο CPU μετά τον υπολογισμό.

Επίσης, υλοποιήσαμε το complex matrix multiplication και στο CPU, με triple nested loop, το οποίο όπως θα δούμε παρακάτω προφανώς είναι πολύ αργό, αλλά χρησιμεύει για την σύγκριση και επαλήθευση των αποτελεσμάτων σε σχέση με το offloading στο GPU (και σε σχέση με τον χρόνο αλλά και με την ορθότητα των αποτελεσμάτων).

Για την χρονομέτρηση του complex matrix multiplication στο host χρησιμοποιήσαμε το timeval.

Για την επαλήθευση των αποτελεσμάτων και των 2 υλοποιήσεων, χρησιμοποιούμε την συνάρτηση **verify\_results()** η οποία παίρνει ως ορίσματα τους 4 πίνακες (2 για κάθε υλοποίηση), e και f, καθώς επίσης και ένα tolerance και συγκρίνει τα αποτελέσματα ένα προς ένα και αν υπάρχει κάποιο mismatch μας το εκτυπώνει.

Έχοντας λοιπόν τρέξει το complex\_matrix\_multiplication\_omp.cu, για διαφορετικά μεγέθη Matrix συλλέξαμε τα εξής δεδομένα σε μορφή διαγράμματος:

Παρατηρούμε, πως η επιτάχυνση του complex matrix multiplication, είναι εκθετική (ειδικά στην αρχή) όσο αυξάνεται το matrix size, χρησιμοποιώντας την υλοποίηση στην GPU, ωστόσο σε σύγκριση με την προηγούμενη άσκηση το Speedup είναι πολύ μικρότερο (περίπου 500 φορές πιο αργό για 2048 matrix size)

Συγκεκριμένα, μπορείτε να δείτε τις μετρήσεις (οι χρόνοι σε ms) παρακάτω:

|  |  |  |  |
| --- | --- | --- | --- |
| Matrix Size | CPU Time | GPU Time | Speedup |
| 64 | 8.00 | 7.00 | 1.14 |
| 128 | 34.00 | 18.00 | 1.89 |
| 256 | 259.00 | 31.00 | 8.35 |
| 512 | 3505.00 | 244.00 | 14.36 |
| 1024 | 36170.00 | 1835.00 | 19.71 |
| 2056 | 566203.00 | 19961.00 | 28.37 |

**Παρατηρήσεις**

1. Όπου κρίθηκε αναγκαίο, υπάρχουν τα απαραίτητα Comments στον κώδικα για την ευκολότερη ανάγνωση και κατανόησή του.
2. Τα παραπάνω αποτελέσματα των χρόνων εκτέλεσης και του Speedup, συλλέχθηκαν με τον υπολογισμό του μέσου όρου 5 εκτελέσεων της κάθε μεθόδου.
3. Μπορείτε να αλλάξετε το Matrix Size από το Makefile που δημιουργήσαμε για την εύκολη μεταγλώττιση και εκτέλεση του κώδικα.