Παράλληλος Προγραμματισμός 2018
Προγραμματιστική Εργασία #1

Ονοματεπώνυμο: Μώκος Θεόδωρος

Α.Μ: Π2012028

Ο κώδικας μου περιέχει τα δύο αρχεία matrix1.c και matrix2.c , τα οποία περιέχουν τον κώδικα. Ξεκινώντας με το αρχείο matrix1.c , δηλώνουμε την getwalltime καθώς και τις μεταβλητές και δεσμεύουμε τον απαιτούμενο πίνακα γραμμικά με ένα loop. Στη συνέχεια, Συμπληρώνουμε τα στοιχεία του πίνακα με μια τιμή (το 1.0 για το συγκεκριμένο παράδειγμα) εξίσου γραμμικά. Έπειτα λαμβάνουμε στο start time και εκτελούμε το workload, όπου εκτελείται γραμμή προς γραμμή ,στου οποίου το εσωτερικό του διπλού loop χρησιμοποιούμε μια πράξη για να αποφύγουμε την απαλοιφή των loops. Μετά λαμβάνουμε το end time, ελέγχουμε αν το αποτέλεσμα του workload είναι σωστό. Τέλος , υπολογίζουμε το χρόνο εκτέλεσης και τα Maccesses , τα προβάλουμε και τα περνάμε σε ένα αρχείο ‘results.csv’ , όπου αποθηκεύει τα αποτελέσματα. Το αρχείο matrix2.c είναι σχεδόν ίδιο, με τη μόνη διαφορά ότι η εκτέλεση του διπλού loop γίνεται στήλη προς στήλη. Οι κώδικες και των δύο αρχείων εκτελέστηκαν με σταθερά NCOLS=100 και NROWS=(100,1000,10000,100000), δέκα φορές για την κάθε NROWS παράμετρο.
Τα αποτελέσματα του χρόνου εκτέλεσης του workload και των Maccesses παραθέτονται στο αρχείο `results.csv` .

Όπως παρατηρούμε από τα αποτελέσματα των εκτελέσεων των δύο αρχείων , ο χρόνος εκτέλεσης αυξάνεται όσο αυξάνουμε το NROWS. Παρόλα αυτά η εκτέλεση γραμμή προς γραμμή απαιτεί λιγότερο χρόνο και έχει περισσότερα Maccesses/sec σε σχέση με την εκτέλεση στήλη προς στήλη. Ο λόγος που συμβαίνει αυτό είναι επειδή, με βάση την αρχιτεκτονική του ο υπολογιστής δεσμεύει συνεχόμενες θέσεις μνήμης. Κατά την εκτέλεση του matrix1.c , ο υπολογιστής έχει δεσμεύσει

γραμμικά μνήμη και εφόσον η προσπέλαση είναι γραμμή προς γραμμή η εκτέλεση είναι γρήγορη. Κατά την εκτέλεση του matrix2.c όμως ο υπολογιστής αναγκάζεται να δεσμεύσει μέρος της κύριας μνήμης και αυτό απαιτεί χρόνο, γιαυτό και η εκτέλεση είναι πιο αργή.