Παράλληλος προγραμματισμός 2018

Προγραμματιστική εργασία #1

Ονοματεπώνυμο: Ηλέκτρα Βλάχου

ΑΜ: Π2013073

Τα δύο δοκιμαστικά προγράμματα αναπτύχθηκαν στο IDE Netbeans 8.1 σε περιβάλλον Windows 10, φροντίζοντας να βάλουμε τα κατάλληλα flags στο μεταγλωττιστή και θέτοντας το πλήθος των γραμμών μέσω ορίσματος.

Το πλήθος των γραμμών για το οποίο επιλέξαμε να τρέξουμε τα δοκιμαστικά προγράμματα ήταν: 100, 1.000, 10.000, 100.000, ενώ το πλήθος των στηλών παρέμεινε σταθερό στις 100. Για να μην εξαλειφθούν τα loops από το μεταγλωττιστή αρχικοποιήσαμε δύο πίνακες (*random1*, *random2*) με τυχαίες τιμές και στη συνέχεια σε ένα νέο loop, στο οποίο κάναμε και τις μετρήσεις μας, κάναμε έναν υπολογισμό μεταξύ των τιμών αυτών και το αποτέλεσμα αποθηκεύτηκε στο πίνακα *table*. Ο υπολογισμός αυτός φαίνεται στην παρακάτω γραμμή χωρίς τα indexes να ανταποκρίνονται στον πραγματικό κώδικα.

```
table[i,j]=random1[i, j]*random2[i, j] + random2[i, j];
```

Στην πραγματικότητα ο δισδιάστατος πίνακας μεγέθους (DNROWSxNCOLS) απλώνεται στη μία διάσταση με το συνολικό πίνακα να καταλαμβάνει DNROWSxNCOLS θέσεις μνήμης. Στο αρχείο matrix1.c η προσπέλαση γίνεται γραμμή-προς γραμμή, ενώ στο αρχείο matrix2.c η προσπέλαση γίνεται στήλη-προς-στήλη. Ο τρόπος με τον οποίο επιτυγχάνεται η προσπέλαση σε κάθε περίπτωση παρουσιάζεται παρακάτω:

matrix1.c

```
for (i=0;i<DNROWS;i++) {
   for (j=0;j<NCOLS;j++){
      table[i*NCOLS+j]=random1[i*NCOLS+j]*random2[i*NCOLS+j] + random2[i*NCOLS+j];
   }
}
```

matrix2.c

```
for (i=0;i<NCOLS;i++){
   for (j=0;j<DNROWS;j++) {
      table[i+j*NCOLS]=random1[i+j*NCOLS]*random2[i+j*NCOLS] + random2[i+j*NCOLS];
   }
}
```

Το loop αυτό είναι το μόνο διαφορετικό κομμάτι στον κώδικα μεταξύ των δύο προγραμμάτων. Για να μετρήσουμε την απόδοση για τις δύο περιπτώσεις κάνουμε τους

εξής υπολογισμούς, όπου te είναι η χρονική στιγμή αμέσως μετά το τέλος του loop και ts η χρονική στιγμή (σε δευτερόλεπτα) πριν την αρχή του loop υπολογισμού:

```
mflops = (DNROWS*NCOLS*2.0)/((te-ts)*1e6);
maccesses = (DNROWS*NCOLS*4.0)/((te-ts)*1e6);
printf("Time elapsed: %f\n",te-ts);
printf("Mflops/sec: %f\n",mflops);
printf("Maccesses/sec: %f\n",maccesses);
```

Το πλήθος των επαναλήψεων του loop είναι όσο το πλήθος των στοιχείων του πίνακα (DNROWSxNCOLS). Σε κάθε επανάληψη πραγματοποιούνται 2 πράξεις (πρόσθεση και πολλαπλασιασμός), ενώ έχουμε 4 (3 για την ανάκτηση των δεδομένων των 3 πινάκων και 1 για την αποθήκευση του αποτελέσματος) προσπελάσεις στη μνήμη. Από αυτά τα δεδομένα προκύπτουν οι σχέσεις για τις μετρήσεις που παρατέθηκαν προηγουμένως, ενώ πολλαπλασιάζουμε το χρονικό διάστημα με την τιμή 10^6 για να παραθέσουμε τα αποτελέσματα στην κλίμακα Mflops (και Maccesses) που χρησιμοποιείται κατά κόρον.

Τα αποτελέσματα για κάθε περίπτωση παρουσιάζεται στους παρακάτω πίνακες:

matrix1.c

Γραμμές	Χρόνος (sec)	Mflops/sec	Maccesses/sec
100	0	inf	inf
1000	0.001001	199.823916	399.647832
10000	0.007507	266.414965	532.82993
100000	0.049572	403.453636	806.907272

matrix2.c

Γραμμές	Χρόνος (sec)	Mflops/sec	Maccesses/sec
100	0	inf	inf
1000	0.003004	66.576254	133.152508
10000	0.019156	104.406044	208.812088
100000	0.333536	59.963558	119.927117

Βλέπουμε πως και στα 2 προγράμματα όσο αυξάνεται το πλήθος των γραμμών, τόσο αυξάνεται και ο χρόνος εκτέλεσης, καθώς αυξάνεται ο όγκος δεδομένων προς επεξεργασία. Επιπλέον, όταν οι γραμμές είναι 100, ο υπολογισμός γίνεται σε πολύ μικρό χρόνο, που δεν είναι ικανός να καταγραφεί με την ακρίβεια του μηχανήματος στο οποίο έτρεξαν τα προγράμματα.

Στο *matrix1.c*, όσο αυξάνεται το πλήθος των γραμμών, αυξάνεται επίσης και το πλήθος των υπολογισμών ανά δευτερόλεπτο (Mflops/sec) και το πλήθος προσβάσεων στη μνήμη ανά δευτερόλεπτο (Maccesses/sec). Αντίθετα, στο *matrix2.c*, τα Mflops/sec και τα Maccesses/sec αυξάνονται μέχρι την τιμή 10.000, αλλά μειώνονται όταν το πλήθος των γραμμών αυξάνεται στην τιμή 100.000.

Στο matrix1.c η προσπέλαση γίνεται γραμμή-προς-γραμμή που σημαίνει ότι το πρόγραμμα απαιτεί προσπέλαση συνεχόμενων θέσεων μνήμης, των οποίων τα περιεχόμενα μπορούν να μεταφερθούν κατά ομάδες στη μνήμη cache του μηχανήματος. Αντίθετα, στο matrix2.c, που η προσπέλαση γίνεται στήλη-προς-στήλη, απαιτείται προσπέλαση θέσεων μνήμης που απέχουν μεταξύ τους κατά NCOLS=100 θέσεις, οπότε υπάρχουν περισσότερες αστοχίες στη μνήμη cache. Αυτός είναι ο λόγος που ο χρόνος είναι μεγαλύτερος στο matrix2 από ότι στο matrix1 σε όλες τις περιπτώσεις.

Το μηχάνημα στο οποίο τρέξανε τα προγράμματα διαθέτει τον επεξεργαστή i7-4510U, ο οποίος έχει τις ακόλουθες μνήμες cache.

Data width	**64 bit**
The number of CPU cores	**2**
The number of threads	**4**
Floating Point Unit	Integrated
Level 1 cache size	2 x 32 KB 8-way set associative instruction caches 2 x 32 KB 8-way set associative data caches
Level 2 cache size	**2 x 256 KB 8-way set associative caches**
Level 3 cache size	**4 MB 16-way set associative shared cache**

Εικόνα 1: Μνήμες cache i7-4510U (Πηγή: http://www.cpu-world.com/CPUs/Core_i7/Intel-Core%20i7-4510U%20Mobile%20processor.html)

Το μέγεθος του κάθε πίνακα (Rows*Cols*sizeof(double)) σε κάθε περίπτωση είναι το εξής:

Γραμμές	**Μέγεθος πίνακα (MB)**	**Συνολικό μέγεθος δεδομένων**
100	0.08	0.24
1000	0.800	2.4
10000	8	24
100000	80	240

Κάθε φορά που το πρόγραμμα ζητά κάτι από τη μνήμη και δεν το βρίσκει στη cache, τότε εκτός των δεδομένων που ζήτησε μεταφέρονται και ομάδες δεδομένων από συνεχόμενες θέσεις μνήμης που είναι πολύ πιθανό να ζητηθούν στη συνέχεια του προγράμματος. Όταν το μέγεθος του κάθε πίνακα φτάνει τα 80MB, είναι σαφές ότι στην περίπτωση του matrix2.c, υπάρχουν περισσότερες αστοχίες λόγω της απόστασης που έχουν τα δεδομένα μεταξύ τους. Αυτό έχει σαν συνέπεια την πτώση των δεικτών Mflops/sec και Maccesses/sec στην τελευταία μέτρηση στο matrix2.c, καθώς οι ίδιοι υπολογισμοί και οι προσπελάσεις της μνήμης γίνονται σε μεγαλύτερο χρόνο. Αντίθετα, στο matrix1.c αυξάνεται μεν ο χρόνος, αλλά όχι τόσο απότομα όσο στο matrix2.c, καθώς το γεγονός ότι το πρόγραμμα ζητά δεδομένα από συνεχόμενες θέσεις, ρίχνοντας σε μικρότερο βαθμό τα misses στη cache. Αυτό καταδεικνύεται από τη συνεχή αύξηση των δεικτών Mflops/sec και Maccesses/sec ακόμα και στην τελευταία μέτρηση.