Εθνικο & Καποδιστριακο Πανεπιστημιο Αθηνων
Τμημα Πληροφορικης & Τηλεπικοινωνιων
Μαθημα: **Παραλληλα Συστηματα**

Εργασία 2 (*υποχρεωτική*) –Προγραμματισμός με OpenMP

Ακαδημαϊκο Ετος 2022 – 2023
(εκφωνηση) Παρασκευη 16 Δεκεμβριου 2022
(παραδοση στο eclass μεχρι) **Δευτερα 9 Ιανουαριου 2023**

Πληροφορίες για τις Υποχρεωτικές Εργασίες του μαθήματος

- Κάθε ομάδα μπορεί να αποτελείται **από 1 ή 2 φοιτητές**. Όλα τα μέλη της ομάδας πρέπει να έχουν ισότιμη συμμετοχή και να γνωρίζουν τις λεπτομέρειες της υλοποίησης της ομάδας.
- Για την εξεταστική Σεπτεμβρίου δε θα δοθούν άλλες εργασίες. Το Σεπτέμβριο εξετάζεται μόνο το γραπτό.
- Επίσημη υπολογιστική πλατφόρμα του μαθήματος είναι το δίκτυο των υπολογιστών του Τμήματος με λειτουργικό σύστημα Linux Ubuntu (linux01.di.uoa.gr έως linux30.di.uoa.gr).
- Μαζί με τον κώδικά σας καλείστε να υποβάλετε και τα σχετικά αρχεία Makefile.
- Στην αναφορά σας καλείστε να δώσετε πληροφορίες σχετικά με το όνομα του υπολογιστικού συστήματος που χρησιμοποιείτε, καθώς επίσης και το μοντέλο επεξεργαστή, τον αριθμό των πυρήνων, την έκδοση του λειτουργικού συστήματος, και την έκδοση του μεταγλωττιστή. Για τα πειραματικά δεδομένα που παρουσίαζετε, καλείστε να αναφέρετε ρητά στα εκάστοτε σημεία της αναφοράς τις εισόδους που χρησιμοποιήσατε. Καλείστε να εκτελέσετε κάθε πείραμα (το οποίο ορίζεται ως η εκτέλεση ενός προγράμματος για συγκεκριμένο αριθμό νημάτων και παραμέτρους εισόδου) πολλές φορές (για παράδειγμα, 4 φορές) και να παρουσιάσετε τον μέσο όρο των αποτελεσμάτων (σύσταση: δημιουργήστε scripts για την εκτέλεση των πειραμάτων, ακόμα και για την επεξεργασία των αποτελεσμάτων και την δημιουργία γραφημάτων).
- Προαιρετικά, μπορείτε να υποβάλετε και τα scripts που χρησιμοποιήσατε για να τρέξετε τα πειράματα και να δημιουργήσετε τα σχετικά γραφήματα.
- Καλείστε να προσεγγίσετε την κάθε άσκηση στην αναφορά σας ως εξής: περιγραφή προβλήματος, σύντομη περιγραφή της λύσης σας, παράθεση πειραματικών αποτελεσμάτων (χρήση πινάκων ή γραφημάτων), και σχολιασμός αποτελεσμάτων.
- Σε περίπτωση αντιγραφής θα μηδενίζονται όλες οι ομάδες που μετέχουν σε αυτή.
- Η παράδοση της **Εργασίας** πρέπει να γίνει μέχρι τα **μεσάνυχτα της προθεσμίας ηλεκτρονικά** και μόνο στο eclass (να ανεβάσετε ένα μόνο αρχείο zip ή rar με την αναφορά σας σε PDF και τον κώδικά σας). **Μην περιμένετε μέχρι την τελευταία στιγμή.**

Άσκηση 2.1

Γράψτε ένα πρόγραμμα OpenMP που να χρησιμοποιεί τη μέθοδο Μόντε Κάρλο για την εκτίμηση της τιμής του π (για την περιγραφή του αλγορίθμου, δείτε την εκφώνηση της Άσκησης 1.1 από τη 1η Εργασία). Το πρόγραμμα θα πρέπει να παίρνει το πλήθος των ρίψεων από τον χρήστη πριν την εκκίνηση νημάτων. Χρησιμοποιήστε έναν όρο reduction για τον υπολογισμό του πλήθους των βελών που πετυχαίνουν το εσωτερικό του κύκλου και φροντίστε ώστε η εκτύπωση του αποτελέσματος να γίνεται μετά την ένωση όλων των νημάτων. Προτιμήστε τον τύπο long long int για το πλήθος των βελών εντός του κύκλου και για το πλήθος των ρίψεων, αφού και οι δύο αυτοί αριθμοί θα πρέπει να είναι πολύ μεγάλοι (για παράδειγμα, το πλήθος ρίψεων να είναι 10^8 ή 10^9) για να πάρετε μια καλή εκτίμηση για το π.

Συγκρίνετε την απόδοση του παράλληλου προγράμματος σας που χρησιμοποιεί OpenMP για διαφορετικό αριθμό νημάτων και διαφορετικό αριθμό πλήθος ρίψεων σε σχέση με τον σειριακό αλγόριθμο. Παρατηρείτε επιτάχυνση και γιατί; Συγκρίνετε επίσης την απόδοση του παράλληλου προγράμματος σας που χρησιμοποιεί OpenMP με το αντίστοιχο παράλληλο πρόγραμμα που γράψατε για τη 1η Εργασία με Pthreads. Παρατηρείτε επιτάχυνση και γιατί; Παρατηρείτε διαφορές στην επίδοση;

Άσκηση 2.2

Σε αυτή την άσκηση καλείστε να τροποποιήσετε το πρόγραμμα πολλαπλασιασμού μήτρας-διανύσματος του βιβλίου «Εισαγωγή στον Παράλληλο Προγραμματισμό» που σας δίνεται (omp_mat_vect_rand_split.c), ώστε να υπολογίζει αποδοτικά (δηλαδή να αποφεύγει αχρείαστους υπολογισμούς) τον πολλαπλασιασμό άνω τριγωνικού πίνακα με διάνυσμα, με χρήση OpenMP. Υπενθυμίζεται ότι άνω (κάτω) τριγωνικός λέγεται ένας πίνακας στον οποίο είναι μη μηδενικά μόνο τα στοιχεία πάνω (κάτω) από την κύρια διαγώνιο, καθώς και αυτά της κύριας διαγωνίου. Παρατηρείτε διαφορές στην επίδοση καθώς αυξάνεται ο αριθμός των νημάτων; Μπορείτε να βελτιώσετε την επίδοση μέσω της επιλογής ανάθεσης επαναλήψεων σε νήματα; Ποια είναι η επιλογή που μειώνει περισσότερο το χρόνο εκτέλεσης; Ίσως σας φανεί χρήσιμος ο όρος schedule(runtime) και η μεταβλητή περιβάλλοντος OMP_SCHEDULE.

Άσκηση 2.3 (Προαιρετική)

Για την επίλυση μεγάλων γραμμικών συστημάτων, συχνά χρησιμοποιούμε την απαλοιφή Gauss ακολουθούμενη από μια αντικατάσταση προς τα πίσω. Η απαλοιφή Gauss μετατρέπει ένα γραμμικό σύστημα n × n σε άνω τριγωνικό χρησιμοποιώντας τις παρακάτω πράξεις στις γραμμές:

- Πρόσθεση ενός πολλαπλάσιου μίας γραμμής σε άλλη γραμμή
- Αντιμετάθεση δύο γραμμών
- Πολλαπλασιασμό μίας γραμμής με μια μη μηδενική σταθερά

Ένα άνω τριγωνικό σύστημα έχει μόνο μηδενικούς συντελεστές κάτω από τη διαγώνιο που εκτείνεται από την άνω αριστερή γωνία τού συστήματος στην κάτω δεξιά.

Για παράδειγμα, το γραμμικό σύστημα:

$$2x_0 - 3x_1 = 3$$
$$4x_0 - 5x_1 + x_2 = 7$$
$$2x_0 - x_1 - 3x_2 = 5$$

μπορεί να απλοποιηθεί στην άνω τριγωνική μορφή:

$$2x_0 - 3x_1 = 3$$
$$x_1 + x_2 = 1$$
$$-5x_2 = 0$$

Αυτό το σύστημα μπορεί να λυθεί εύκολα: βρίσκουμε το x_2 χρησιμοποιώντας την τελευταία εξίσωση, μετά βρίσκουμε το x_1 από τη δεύτερη εξίσωση και, τέλος, βρίσκουμε το x_0 χρησιμοποιώντας την πρώτη εξίσωση.

Μπορούμε να χρησιμοποιηθούν διάφοροι σειριακοί αλγόριθμοι για την αντικατάσταση προς τα πίσω. Ένας τέτοιος αλγόριθμος που διατρέχει το σύστημα πρώτα κατά γραμμή έχει τη μορφή

```
for (row = n-1; row >= 0; row--) {
      x[row] = b[row];
      for (col = row+1; col < n; col++)
            x[row] -= A[row][col]*x[col];
      x[row] /= A[row][row];
}
```

Εδώ η «δεξιά πλευρά» του συστήματος αποθηκεύεται στη συστοιχία b, οι συντελεστές των αγνώστων αποθηκεύονται στη διδιάστατη συστοιχία A, και οι λύσεις αποθηκεύονται στη συστοιχία x. Ένας εναλλακτικός αλγόριθμος, που διατρέχει το σύστημα πρώτα κατά στήλη, είναι ο παρακάτω:

```
for (row = 0; row < n; row++)
      x[row] = b[row];

for (col = n-1; col >= 0; col--) {
      x[col] /= A[col][col];
      for (row = 0; row < col; row++)
            x[row] -= A[row][col]*x[col];
}
```

1. Εξετάστε αν ο εξωτερικός βρόχος του «κατά γραμμή» αλγορίθμου μπορεί να παραλληλοποιηθεί.
2. Εξετάστε αν ο εσωτερικός βρόχος του «κατά γραμμή» αλγορίθμου μπορεί να παραλληλοποιηθεί.
3. Εξετάστε αν ο (δεύτερος) εξωτερικός βρόχος του «κατά στήλη» αλγορίθμου μπορεί να παραλληλοποιηθεί.
4. Εξετάστε αν ο εσωτερικός βρόχος του «κατά στήλη» αλγορίθμου μπορεί να παραλληλοποιηθεί.
5. Γράψτε ένα πρόγραμμα OpenMP για καθέναν από τους βρόχους που θεωρήσατε ότι μπορούν να παραλληλοποιηθούν. Ίσως σας φανεί χρήσιμη η οδηγία single –όταν ένα μπλοκ κώδικα εκτελείται παράλληλα και ένα υπομπλοκ στο εσωτερικό του πρέπει να εκτελεστεί από ένα μόνο νήμα, το υπομπλοκ μπορεί να προσδιοριστεί με μια οδηγία #pragma omp single. Τα νήματα της ομάδας που εκτελεί τον κώδικα θα μείνουν μπλοκαρισμένα στο τέλος του υπο-μπλοκ κώδικα μέχρι να φτάσουν σε αυτό το σημείο όλα τα νήματα.
6. Τροποποιήστε τον παράλληλο βρόχο σας χρησιμοποιώντας τον όρο schedule(runtime) και δοκιμάστε το πρόγραμμα με διάφορα χρονοδιαγράμμα-τα. Αν το άνω τριγωνικό σύστημα έχει 10.000 μεταβλητές, ποιο χρονοδιάγραμμα δίνει τις καλύτερες επιδόσεις;