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

# Εργασία 2 (*υποχρεωτική*) – Προγραμματισμός με OpenMP

**Ακαδημαϊκο Ετος 2025 – 2026**
(εκφωνηση) Δευτερα 8 Δεκεμβριου 2025
(παραδοση στο eclass μεχρι) **Παρασκευη 16 Ιανουαριου 2026**

**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. Τον 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).
- Για την ανάπτυξη των προγραμμάτων σας καλείστε να χρησιμοποιήσετε τη γλώσσα προγραμματισμού C.
- 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 παρουσιάζετε, 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 (20%)**

Σε αυτή την άσκηση καλείστε να γράψετε ένα πρόγραμμα OpenMP το οποίο υπολογίζει τον πολλαπλασιασμό πολυωνύμων. Πιο συγκεκριμένα, καλείστε να γράψετε ένα πρόγραμμα το οποίο: (i) δημιουργεί δύο τυχαία πλήρη πολυώνυμα n βαθμού οι συντελεστές των οποίων είναι ακέραιοι μη-μηδενικοί αριθμοί, (ii) τα πολλαπλασιάζει χρησιμοποιώντας τον σειριακό αλγόριθμο, (iii) τα πολλαπλασιάζει χρησιμοποιώντας τον παράλληλο αλγόριθμο, και (iv) επιβεβαιώνει ότι ο σειριακός και ο παράλληλος αλγόριθμος παράγουν τα ίδια αποτελέσματα. Το πρόγραμμά σας θα λαμβάνει σαν ορίσματα τον βαθμό n του κάθε πολυωνύμου και τον αριθμό των νημάτων, ενώ σαν έξοδο θα τυπώνει τον χρόνο δημιουργίας και αρχικοποίησης των πολυωνύμων (καθώς και όποιων άλλων δομών δεδομένων χρειάζεστε), τον χρόνο εκτέλεσης του σειριακού αλγόριθμου, τον χρόνο εκτέλεσης του παράλληλου αλγόριθμου, και την επιβεβαίωση της σωστής εκτέλεσης του παράλληλου αλγόριθμου σε σχέση με τον σειριακό. Συγκρίνετε την απόδοση του παράλληλου προγράμματος σας για διάφορους βαθμούς πολυωνύμων n (για παράδειγμα, $10^5$ ή $10^6$) και για διαφορετικό αριθμό νημάτων σε σχέση με τον σειριακό αλγόριθμο. Παρατηρείτε επιτάχυνση και γιατί; Χρειάστηκε να χρησιμοποιήσετε συγχρονισμό και γιατί; Στην άσκηση αποφύγετε την επίλυση μέσω Fast Fourier Transform. Τέλος, συγκρίνετε την απόδοση του παράλληλου προγράμματος σας που χρησιμοποιεί OpenMP με το αντίστοιχο παράλληλο πρόγραμμα που γράψατε για την Άσκηση 1.1 με Pthreads. Παρατηρείτε διαφορές στην επίδοση;

**Άσκηση 2.2 (60%)**

Σε αυτή την άσκηση καλείστε να γράψετε ένα πρόγραμμα OpenMP που να πραγματοποιεί αποδοτικά την κατασκευή αραιού πίνακα και τον πολλαπλασιασμό αραιού πίνακα με διάνυσμα (sparse matrix-vector multiplication). Αραιός (sparse) πίνακας ορίζεται ως ο πίνακας του οποίου τα περισσότερα στοιχεία είναι μηδενικά. Η βασική ιδέα εκμετάλλευσης των αραιών πινάκων είναι η χρήση εναλλακτικού σχήματος αναπαράστασης του πίνακα που κρατά πληροφορίες μόνο για τα μη-μηδενικά στοιχεία (non-zero elements). Τα βασικά οφέλη της χρήσης εναλλακτικής αναπαράστασης αραιού πίνακα είναι: (i) μείωση των απαιτήσεων μνήμης και αποθηκευτικού χώρου, και (ii) βελτίωση του χρόνου εκτέλεσης λόγω της αποφυγής πράξεων των στοιχείων του πίνακα με μηδενικά στοιχεία. Ωστόσο, αν ο πίνακας είναι αποθηκευμένος στην κλασσική, πυκνή (dense) μορφή, τότε απαιτείται και χρόνος κατασκευής των δομών δεδομένων που χρησιμοποιούνται για την αναπαράσταση του αραιού πίνακα.

Έχουν προταθεί και χρησιμοποιηθεί διάφορα σχήματα αναπαράστασης αραιού πίνακα. Περισσότερες πληροφορίες μπορείτε να βρείτε στον εξής σύνδεσμο: https://en.wikipedia.org/wiki/Sparse_matrix. Στην άσκηση θα χρησιμοποιήσετε το Compressed Sparse Row (CSR) σχήμα αναπαράστασης, το οποίο χρησιμοποιεί τρεις μονοδιάστατους πίνακες για την αναπαράσταση του αρχικού πίνακα. Ο πρώτος πίνακας περιέχει τις τιμές των μη-μηδενικών στοιχείων του αρχικού πίνακα. Ο δεύτερος πίνακας περιέχει τις τιμές των στηλών στις οποίες βρίσκονται τα μη-μηδενικά στοιχεία. Ο τρίτος πίνακας περιέχει πληροφορίες σχετικά με το πλήθος των μη-μηδενικών στοιχείων σε κάθε γραμμή.

Πιο συγκεκριμένα, το πρόγραμμά σας αρχικά θα κατασκευάζει με τυχαίο τρόπο τον πίνακα και το διάνυσμα με ακέραιους αριθμούς (αυτό το βήμα μπορεί να γίνει είτε σειριακά είτε παράλληλα, και δεν το λαμβάνετε υπόψη στις μετρήσεις) φροντίζοντας να υπάρχει κατάλληλο ποσοστό μηδενικών στοιχείων στον πίνακα. Στη συνέχεια, θα κατασκευάζει παράλληλα την αναπαράσταση CSR του πίνακα και θα τυπώνει τον χρόνο εκτέλεσης της κατασκευής. Έπειτα, θα πολλαπλασιάζει παράλληλα τον πίνακα με το διάνυσμα για έναν συγκεκριμένο αριθμό επαναλήψεων (το διάνυσμα του αποτελέσματος της κάθε επανάληψης είναι το διάνυσμα εισόδου της επόμενης επανάληψης), και θα τυπώνει αντίστοιχα τον συνολικό χρόνο του πολλαπλασιασμού. Τέλος, θα πραγματοποιεί παράλληλα τον πολλαπλασιασμό με τη χρήση της πυκνής μορφής για τον ίδιο αριθμό επαναλήψεων και θα τυπώνει τον αντίστοιχο χρόνο.

Για την παραλληλοποίηση των παραπάνω βημάτων, θα χρειαστεί αρχικά να γράψετε τις αντίστοιχες σειριακές εκδόσεις που θα χρησιμοποιήσετε σαν βάση τόσο για την ορθότητα της παραλληλοποίησης, όσο και για τις μετρήσεις σύγκρισης της επίδοσης της παράλληλης εκδοχής.

Το πρόγραμμά σας θα λαμβάνει σαν ορίσματα: (i) τον αριθμό των στοιχείων γραμμής/στήλης του πίνακα (θεωρείστε ότι ο πίνακας είναι τετραγωνικός), (ii) το ποσοστό μηδενικών στοιχείων του πίνακα (sparsity), (iii) τον αριθμό επαναλήψεων του πολλαπλασιασμού, και (iv) τον αριθμό των νημάτων.

Πειραματιστείτε με διαφορετικές διαστάσεις του πίνακα/διανύσματος (για παράδειγμα, $10^3$ έως $10^4$ στοιχεία ανά γραμμή), διαφορετικά ποσοστά μηδενικών στοιχειών (για παράδειγμα, 0% έως 99%), διαφορετικό αριθμό επαναλήψεων (για παράδειγμα, 1 έως 20) και διαφορετικό αριθμό νημάτων. Αρχικά επικεντρωθείτε στο πώς κλιμακώνει η παράλληλη υλοποίηση σας για την κατασκευή της αναπαράστασης CSR και του πολλαπλασιασμού καθώς αλλάζει ο αριθμός των νημάτων σε σχέση με την αντίστοιχη σειριακή εκτέλεση. Αναφέρετε αν βελτιώνεται η επίδοση μέσω της παραλληλίας και κατά πόσο. Στη συνέχεια, επικεντρωθείτε στη σύγκριση της επίδοσης μεταξύ της αναπαράστασης CSR και της πυκνής αναπαράστασης. Για ποιες τιμές των παραμέτρων παρατηρείτε καλύτερο χρόνο επίδοσης με τη χρήση της αναπαράστασης CSR, λαμβάνοντας υπόψη και τον χρόνο δημιουργίας της αναπαράστασης CSR; Στην αναφορά σας καλείστε να παραθέσετε γραφήματα (ή πίνακες) που να δείχνουν τον αντίκτυπο των παραπάνω παραμέτρων και να σχολιάσετε τα αποτελέσματα.

**Άσκηση 2.3 (20%)**
Σε αυτή την άσκηση καλείστε να παραλληλοποιήσετε τον αλγόριθμο της ταξινόμησης με συγχώνευση (mergesort), και πιο συγκεκριμένα την από πάνω προς τα κάτω υλοποίηση του αλγορίθμου κάνοντας χρήση της OpenMP και της οδηγίας task. Το πρόγραμμά σας θα πρέπει να δέχεται σαν ορίσματα: (i) το μέγεθος του πίνακα ακεραίων που θα ταξινομήσει, (ii) το αν θα εκτελεστεί ο σειριακός ή ο παράλληλος αλγόριθμος της ταξινόμησης, και (iii) τον αριθμό των νημάτων που θα χρησιμοποιηθούν στην παράλληλη ταξινόμηση. Στη συνέχεια, το πρόγραμμά σας θα πρέπει να παράγει με τυχαίο (ντετερμινιστικό) τρόπο τον προς ταξινόμηση πίνακα ακεραίων και έπειτα να τον ταξινομεί. Τέλος, το πρόγραμμά σας επιβεβαιώνει ότι ο πίνακας είναι ταξινομημένος και τυπώνει το χρόνο εκτέλεσης της ταξινόμησης. Δοκιμάστε να εκτελέσετε το πρόγραμμά σας για διαφορετικά μεγέθη πινάκων (για παράδειγμα, $10^7$ και $10^8$). Παρατηρείτε επιτάχυνση και γιατί; Ίσως σας φανεί χρήσιμος ο όρος if() της οδηγίας task.