ECE445 Παράλληλοι και Δικτυακοί Υπολογισμοί
Χειμερινό Εξάμηνο 2022-2023
Εργασία 3

Εισαγωγικά

(Ημερομηνία Παράδοσης: Παρασκευή 03.02.2023, 12:00μμ)
Ακολουθείστε το πρότυπο κείμενο (template.docx) για να γράψετε τις απαντήσεις σας στις παρακάτω ερωτήσεις και την αναφορά σας στα πειραματικά αποτελέσματα των προγραμμάτων σας.
Γράψτε καθαρά τα ονοματεπώνυμά σας και τα ΑΕΜ σας στην πρώτη σελίδα. Το κείμενο σας πρέπει να είναι καλογραμμένο και ευανάγνωστο ενώ θα πρέπει να δικαιολογείτε **ΠΛΗΡΩΣ** τα βήματα που ακολουθήσατε, και να σχολιάζετε τα αποτελέσματα από κάθε άσκηση (πίνακες, γραφικές παραστάσεις κλπ.
Προτείνεται να προγραμματίστε σε C/C++ και σε περιβάλλον Linux, χρησιμοποιείστε Makefile για compile/link/execution των προγραμμάτων σας.
Ετοιμάστε την αναφορά (report) σας, γράφοντας τις θεωρητικές λύσεις των προβλημάτων, περιγράφοντας την υλοποίηση και το μηχάνημα που τρέξατε τα προγράμματά σας και κάνατε τα πειράματά σας.
Καταγράψτε τις παρατηρήσεις σας, και παρουσιάστε τα αποτελέσματα σε γραφήματα και πίνακες ώστε να φαίνεται η κλιμάκωση του χρόνου σε σχέση με το μέγεθος των εισόδων. Παραδώστε μια φορά την εργασία σας μέσω eclass σε ένα zip αρχείο (ergasia1_AEM1_AEM2.zip), το οποίο θα περιέχει το κείμενο με τις λύσεις/αποτελέσματα και σχόλια σας, τον κώδικά σας (.c, .h αρχεία και το Makefile σας). Σημειώνεται ότι οι ομάδες δεν αλλάζουν στη διάρκεια του εξαμήνου.

Άσκηση 1 – Εισαγωγή στους αραιούς πίνακες (θεωρητική) (20 μονάδες)

Σε πολλά πραγματικά προβλήματα οι πίνακες που προκύπτουν είναι αραιοί, δηλαδή έχουν πολλά μηδενικά στοιχεία. Αν η διάσταση του πίνακα είναι N, οι πυκνοί πίνακες έχουν N^2 μη μηδενικά στοιχεία, ενώ οι αραιοί συνήθως έχουν O(N) μη μηδενικά στοιχεία NZ. Για παράδειγμα ο πίνακας A, όπου

$$A = \begin{bmatrix} 11 & 0 & 0 & 14 & 0 & 0 \\ 0 & 22 & 0 & 0 & 0 & 26 \\ 31 & 0 & 33 & 0 & 0 & 0 \\ 41 & 0 & 0 & 0 & 45 & 0 \\ 0 & 0 & 53 & 0 & 55 & 0 \\ 0 & 0 & 0 & 64 & 0 & 66 \end{bmatrix} \text{είναι αραιός.}$$

Οι πίνακες αυτοί αποθηκεύονται σε συμπιεσμένη μορφή (https://netlib.org/utk/people/JackDongarra/etemplates/node372.html) για εξοικονόμηση μνήμης και μείωση υπολογιστικού κόστους. Η πληροφορία που καταγράφεται είναι οι μη μηδενικές τιμές του πίνακα και οι θέσεις τους. Σε κάθε περίπτωση μας δίδεται το N και το NZ.

Αραιοί πίνακες σε Coordinate Format.
Ένας πρώτος τρόπος αποθήκευσης αραιών πινάκων είναι το **Coordinate Format (COO)** κατά γραμμές στο οποίο αποθηκεύονται σε τρία διανύσματα α_values, i_index, j_index οι τιμές των στοιχείων, η γραμμή και η στήλη στην οποία βρίσκονται ως εξής:
το στοιχείο α_values(k) βρίσκεται στην i_index(k) γραμμή και j_index(k) στήλη του πίνακα. Για τον παραπάνω πίνακα τα διανύσματα αυτά είναι τα:

$a\_\_values$ = [11 14 22 26 31 33 41 45 53 55 64 66]
$i\_\_index$ = [1 1 2 2 3 3 4 4 5 5 6 6]
$j\_\_index$ = [1 4 2 6 1 3 1 5 3 5 4 6]
Τα στοιχεία του A, αποθηκεύονται κατά γραμμές μέσα στα 3 διανύσματα. Τα στοιχεία της κάθε γραμμής αποθηκεύονται με αύξουσα σειρά κατά στήλη. Αντίστοιχα υπάρχει το **Coordinate Format** κατά στήλες**.**

Αραιοί πίνακες σε Compressed Sparse Row Format.
Ένας δεύτερος τρόπος αποθήκευσης αραιών πινάκων είναι το **Compressed Sparse Row Format (CSR)** στο οποίο αποθηκεύονται σε τρία διανύσματα α_values, i_ptr, j_index οι τιμές των στοιχείων, η γραμμή και η στήλη στην οποία βρίσκονται ως εξης: το στοιχείο α_values(k) βρίσκεται στην j_index(k) στήλη του πίνακα. Το διάνυσμα i_ptr περιέχει τις θέσεις στα διανύσματα α_values και j_index που ξεκινά η κάθε γραμμή του A. Έτσι τα στοιχεία της γραμμής i του πίνακα A βρίσκονται στις θέσεις i_ptr(i) εως i_ptr(i+1) -1
Για τον παραπάνω πίνακα τα διανύσματα αυτά είναι τα :
$a\_\_values$ = [11 14 22 26 31 33 41 45 53 55 64 66]
$i\_\_ptr$ = [1 3 5 7 9 11 13]
$j\_\_index$ = [1 4 2 6 1 3 1 5 3 5 4 6]
Τα στοιχεία του A, αποθηκεύονται κατά γραμμές μέσα στα 3 διανύσματα. Τα στοιχεία της κάθε γραμμής αποθηκεύονται με αύξουσα σειρά κατά στήλη. Αντίστοιχα υπάρχει το **Compressed Sparse Column Format.**

Υπάρχουν κι άλλα formats για αποθήκευση αραιών πινάκων, που προκύπτουν από τα προβλήματα που δημιουργούν τους πίνακες αλλά και τις πράξεις/συναρτήσεις που πρέπει να εφαρμόσουμε σε αυτούς (π.χ., banded formats, diagonal formats κλπ.)
Για τη αποθήκευση τέτοιων πινάκων στην δευτερεύουσα μνήμη του υπολογιστή (π.χ. HDD) υπάρχουν γνωστά file formats[1] [2] [3] όπως Matrix Market Exchange Format (που υποστηρίζει το COO format), Harwell-Boeing Exchange Format (που υποστηρίζει το CSR format) κλπ., και βιβλιοθήκες που βοηθούν στο I/O μέρος του προγραμματισμού των αλγορίθμων για αραιούς πίνακες.

<u>Σημείωση:</u> Η αρίθμηση στα παραπάνω διανύσματα, ακολουθεί η μαθηματική λογική και ξεκινά από το 1 (όχι από το 0). Στις παρακάτω πηγές θα βρείτε περισσότερες πληροφορίες για τα sparse formats, κώδικες και σχετικούς αλγορίθμους.

Ερώτημα 1
Γράψτε και μελετήστε τους σειριακό αλγόριθμο για πολλαπλασιασμό αραιού πίνακα (αποθηκευμένο σε **COO** και **CSR)** με διάνυσμα. Υπολογίστε το κόστος του σε συνάρτηση των N και NZ.
Ερώτημα 2
Γράψτε και μελετήστε τους αντίστοιχους παράλληλους αλγόριθμους για N στοιχεία σε p παράλληλες εργασίες. Επιχειρήστε διαχωρισμό του προβλήματος κατά block γραμμών (N/p) ανά επεξεργαστή. Υπολογίστε το κόστος του σε συνάρτηση των N, NZ και p. Υπολογίστε την χρονοβελτίωση και την απόδοση του αλγόριθμου.
Ερώτημα 3
Γράψτε και μελετήστε τον αντίστοιχο (για τον αρχικό σειριακό της άσκησης 1) παράλληλο αλγόριθμο για N στοιχεία σε p παράλληλες εργασίες. Επιχειρήστε διαχωρισμό του προβλήματος κατά block γραμμών έτσι ώστε να υπάρχουν NZ/p περίπου στοιχεία ανά επεξεργαστή. Υπολογίστε το κόστος του σε συνάρτηση των N, NZ και p.

[1] Matrix Market, http://math.nist.gov/MatrixMarket/
[2] https://math.nist.gov/MatrixMarket/formats.html
[3] File formats, http://math.nist.gov/MatrixMarket/formats.html

Άσκηση 2 – Σειριακή Υλοποίηση (20 μονάδες)

Υλοποιήστε το σειριακό αλγόριθμο (που μελετήσατε στην άσκηση 1 για πολλαπλασιασμό αραιού πίνακα (αποθηκευμένο σε **CSR**) με διάνυσμα.

Άσκηση 3 – Παράλληλη Υλοποίηση και Αξιολόγηση (30 μονάδες)

Υλοποιήστε τον αντίστοιχο παράλληλο αλγόριθμο (που μελετήσατε στην άσκηση 2), είτε σε openMP είτε σε MPI, για N στοιχεία σε p παράλληλες εργασίες. Επιχειρήστε διαχωρισμό του προβλήματος κατά block γραμμών (N/p) ανά επεξεργαστή.
Φτιάξτε πίνακες όπως τον παρακάτω και σημειώστε τους χρόνους για σειριακή και παράλληλη εκτέλεση και μέσα σε παρένθεση το speedup που είχατε σε κάθε περίπτωση. Κάντε γραφική παράσταση για το «πλήθος των threads vs καλύτερο speedup». Τι παρατηρείτε; Συμφωνούν τα αποτελέσματά σας με τη θεωρητική ανάλυση;

N vs p	0	1	2	3	4	…
	105	60 (1,75)				
…						

Άσκηση 4 – Επίλυση Αραιών Συστημάτων Εξισώσεων (30 μονάδες)

Γράψτε ένα παράλληλο πρόγραμμα με χρήση των παραπάνω τεχνικών (σε **CSR** και είτε σε openMP είτε σε MPI), που υλοποιεί την μέθοδο Jacobi για τη λύση του συστήματος

$$Ax = b \iff \begin{bmatrix} 2 & -1 & \\ -1 & \ddots & -1 \\ & -1 & 2 \end{bmatrix} x = \begin{bmatrix} 0 \\ \vdots \\ 0 \\ N+1 \end{bmatrix}$$

όπου N η διάσταση του συστήματος. Ανατρέξτε στις διαφάνειες 8-11 του lecture3_parallel_algo.pdf από το eclass του μαθήματος για περισσότερα στοιχεία σχετικά με την μέθοδο Jacobi.
Το πρόγραμμά σας θα πρέπει να :
α) τρέχει για m επαναλήψεις
β) σε κάθε επανάληψη να υπολογίζει την νέα προσέγγιση της λύσης, xnew, (και να εμφανίζει στην οθόνη) το υπόλοιπο της προσέγγισης, δηλ. $||b - A\, xnew||_2$ και τη διαφορά των 2 τελευταίων προσεγγίσεων, δηλαδή $||xold – xnew||_2$, με κατάλληλα σχόλια, π.χ.

iter = *, residual = *, difference = *

γ) εμφανίζεται ο χρόνος εκτέλεσης υπολογισμών, ο χρόνος επικοινωνίας μετά και την τελευταία επανάληψη.

computations time = *, communication time = *

δ) Εκτελέστε το πρόγραμμά σας για διαφορετικό πλήθος tasks και στην αναφορά σας προσθέστε ένα πίνακα, στον οποίον θα φαίνεται ο αριθμός των tasks και ο αντίστοιχος χρόνος εκτέλεσης στον υπολογιστή σας.
<u>Υπόδειξη:</u> Η λύση του συστήματος είναι το διάνυσμα [1 2 3 ... N]. Χρησιμοποιήστε $N=10^6$, m = 20 και nt = 1,2,4,8,16,32,...

Number of tasks	1	2	4	8	16	32	64
Computations Time (sec)							