**ECE445 Παράλληλοι και Δικτυακοί Υπολογισμοί**

Χειμερινό Εξάμηνο 2022-2023

**Εργασία 3**

Ομάδα φοιτητών:

Επώνυμο Όνομα – ΑΕΜ : Αποστολοπούλου Ιωάννα 03121

Επώνυμο Όνομα – ΑΕΜ : Τολούδης Παναγιώτης 02995

## Άσκηση 1

*Ερώτημα 1:*

Ψευδοκώδικας COO:

for i in range [0, num\_of\_rows]

res[i] = 0

for (k = 0; k < nz\_id; k++) {

i = i\_index [k]

j = j\_index [k]

res [i] += ( a\_values [k] \* vec [i] )

}

To Κόστος Πολυπλοκότητας του Αλγορίθμου προκύπτει από το πλήθος επαναλήψεων της for και τις 2 πράξεις που πραγματοποιούνται μέσα σε αυτήν, την πρόσθεση και τον πολλαπλασιασμό των στοιχείων. Η πολυπλοκότητα παραμένει ίδια ανεξάρτητα από το πλήθος των μη μηδενικών στοιχείων του αραιού πίνακα. Ωστόσο η μνήμη επιρεάζεται σηματικά από το πλήθος αυτό.

Οπότε:

T(n) = 2 \* nz

Ψευδοκώδικας CSR:

for i in range [0, num\_of\_rows]

res[i] = 0

start = i\_ptr [i]

end = i\_ptr [i + 1]

for id in range [start, end] {

j = j\_index [id]

res [i] += ( a\_values [id] \* vec [i] )

}

To Κόστος Πολυπλοκότητας του Αλγορίθμου προκύπτει από το πλήθος επαναλήψεων τως 2 for και τις 2 πράξεις που πραγματοποιούνται μέσα σε αυτήν, την πρόσθεση και τον πολλαπλασιασμό των στοιχείων.

Οπότε:

T(n) = 2 \* nz\_rows \* nz\_columns \* 1

Το \*1 προκύπτει γιατί το vector με το οποίο πολλαπλασιάζουμε τον πίνακα είναι dense.

*Ερώτημα 2:*

Ψευδοκώδικας COO:

for i in range [0, num\_of\_rows]

res[i] = 0

#pragma omp parallel for

for m in range [0, (n/processors)] {

for (k = (m \* processors); k < ((m + 1) \* processors); k++) {

i = i\_index [k]

j = j\_index [k]

res [i] += ( a\_values [k] \* vec [i] )

*}*

*}*

To Κόστος Πολυπλοκότητας του Αλγορίθμου προκύπτει από το πλήθος επαναλήψεων της for και τις 2 πράξεις που πραγματοποιούνται μέσα σε αυτήν, την πρόσθεση και τον πολλαπλασιασμό των στοιχείων διά τον αριθμό των επεξεργαστών.

Οπότε:

*T(n) = (2 \* n) / processors*

Ψευδοκώδικας CSR:

#pragma omp parallel for

for k in range [0, (n/processors)] {

for m in range [0, ((n/processors) )] {

i = (k \* (n/processors)) + m

res[i] = 0

start = i\_ptr [i]

end = i\_ptr [i + 1]

for id in range [start, end] {

j = j\_index [id]

res [i] += ( a\_values [id] \* vec [i] )

*}*

*}*

*}*

To Κόστος Πολυπλοκότητας του Αλγορίθμου προκύπτει από το πλήθος επαναλήψεων τως 2 for και τις 2 πράξεις που πραγματοποιούνται μέσα σε αυτήν, την πρόσθεση και τον πολλαπλασιασμό των στοιχείων.

Οπότε:

T(n) = 2 \* (n/processors)2

*Ερώτημα 3:*

Ψευδοκώδικας COO:

for i in range [0, num\_of\_rows]

res[i] = 0

#pragma omp parallel for

for m in range [0, (nz/processors)] {

for (k = (m \* processors); k < ((m + 1) \* processors); k++) {

i = i\_index [k]

j = j\_index [k]

res [i] += ( a\_values [k] \* vec [i] )

}

}

To Κόστος Πολυπλοκότητας του Αλγορίθμου προκύπτει από το πλήθος επαναλήψεων της for και τις 2 πράξεις που πραγματοποιούνται μέσα σε αυτήν, την πρόσθεση και τον πολλαπλασιασμό των στοιχείων διά τον αριθμό των επεξεργαστών.

Οπότε:

T(n) = (2 \* nz) / processors

Ψευδοκώδικας CSR:

#pragma omp parallel for

for i in range [0, num\_of\_rows]

res[i] = 0

start = i\_ptr [i]

end = i\_ptr [i + 1]

for id in range [start, end] {

j = j\_index [id]

res [i] += ( a\_values [id] \* vec [i] )

}

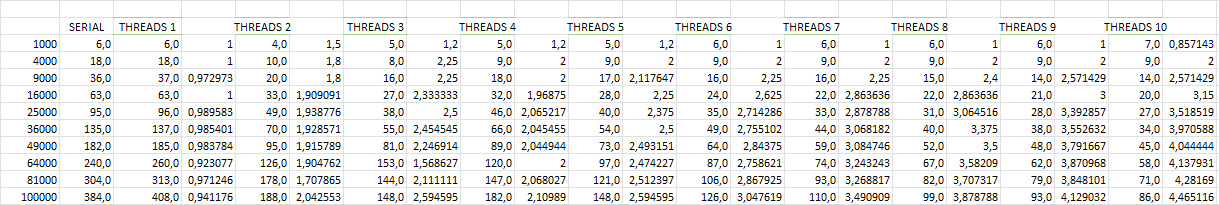
To Κόστος Πολυπλοκότητας του Αλγορίθμου προκύπτει από το πλήθος επαναλήψεων τως 2 for και τις 2 πράξεις που πραγματοποιούνται μέσα σε αυτήν, την πρόσθεση και τον πολλαπλασιασμό των στοιχείων, διά τον αριθμό των επεξεργαστών.

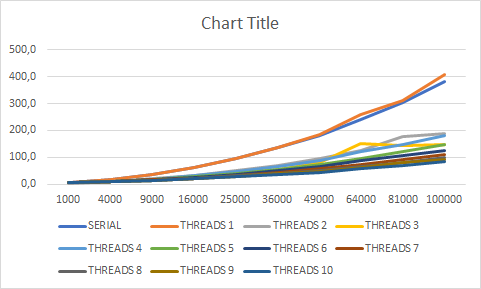
Οπότε:

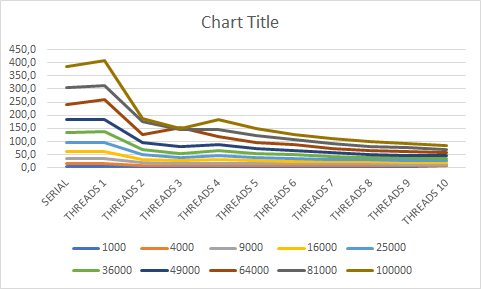
T(n) = (2 \* nz\_rows \* nz\_columns \* 1) / p

*Το \*1 προκύπτει γιατί το vector με το οποίο πολλαπλασιάζουμε τον πίνακα είναι dense.*

## Άσκηση 3







## Άσκηση 4

**ΣΗΜΕΙΩΣΗ**

Όλοι οι χρόνοι είναι μετρημένη σε second.

# ***ΠΑΡΑΡΤΗΜΑ***

## ***Περιγραφή του μηχανήματος*:**

Operating System: Windows Subsystem for Linux Ubuntu 20.04.5 LTS

Architecture: x86\_64

CPU op-mode(s): 32-bit, 64-bit

Byte Order: Little Endian

Address sizes: 48 bits physical, 48 bits virtual

CPU(s): 8

On-line CPU(s) list: 0-7

Thread(s) per core: 2

Core(s) per socket: 4

Socket(s): 1

Vendor ID: AuthenticAMD

CPU family: 23

Model: 17

Model name: AMD Ryzen 7 2700U with Radeon Vega Mobile Gfx

Stepping: 0

CPU MHz: 2195.886

BogoMIPS: 4391.77

Hypervisor vendor: Microsoft

Virtualization type: full

L1d cache: 128 KiB

L1i cache: 256 KiB

L2 cache: 2 MiB

L3 cache: 4 MiB