**ECE445 Παράλληλοι και Δικτυακοί Υπολογισμοί**

Χειμερινό Εξάμηνο 2022-2023

**Εργασία 1**

Ομάδα φοιτητών:

Επώνυμο Όνομα – ΑΕΜ : Αποστολοπούλου Ιωάννα 03121

Επώνυμο Όνομα – ΑΕΜ : Τολούδης Παναγιώτης 02995

## Άσκηση 1

Για την πράξη Ab = c θεωρούμε στον υπάρχον πίνακα πως το b είναι αποθηκευμένο στην κύρια διαγώνιο και το c αφού γίνουν οι υπολογισμοί θα αποθηκευτεί επίσης στην κύρια διαδώνιο.

Α) Χωρίζοντας τον πίνακα σε δύο επεξεργαστές με εναν προαφνή τρόπο όπως φαίνεται παρακάτω προκύπτουν τα παρακάτω δεδομένα.

Εικόνα που περιέχει σότζι, κτίριο, πύργος

Περιγραφή που δημιουργήθηκε αυτόματα

Χρόνος επικοινωνίας στο δίκτυο :

tcomm  = ts + th \* l + m \* tw

Όπου :

* ts : είναι ο χρόνος εκκίνησης που διαφέρει από μηχάνημα σε μηχάνημα.
* th \* l : είναι ο χρόνος που χρειάζεται να μεταβιβαστεί η πληροφορία από hop σε hop στην περίπτωσή μας από πυρήνα σε πηρήνα του επεξεργαστή.
* m \* tw : είναι ο χρόνος μεταφοράς του περιεχομένου της πληροφορίας. Στην συγκεκριμένη περίπτωση χωρισμού του πίνακα το m = 16 \* χ, με χ = το συνολικό μήκος της πληροφορίας σε words.

Επομένως προκύπτει :

tcomm  = ts + th \* l + (16\*χ) \* tw

Το m = 16 \* χ είναι το πλήθος των μηνυμάτων που θα σταλούν ανάμεσα στους δύο επεξεργαστές λόγω του πλήθους των στοιχείων του πίνακα που υπολογίζουμε.

Χωρίζοντας τον πίνακα σε τέσσερις επεξεργαστές με εναν προαφνή τρόπο όπως φαίνεται παρακάτω προκύπτουν τα παρακάτω δεδομένα.

Εικόνα που περιέχει σότζι, κτίριο, καλλυντικά

Περιγραφή που δημιουργήθηκε αυτόματα

Χρόνος επικοινωνίας στο δίκτυο :

tcomm  = ts + th \* l + m \* tw

Όπου :

* m \* tw : είναι ο χρόνος μεταφοράς του περιεχομένου της πληροφορίας. Στην συγκεκριμένη περίπτωση χωρισμού του πίνακα το m = 32 \* χ, με χ = το συνολικό μήκος της πληροφορίας σε words.

Επομένως προκύπτει :

tcomm  = ts + th \* l + (32\*χ) \* tw

Το m = 32 \* χ είναι το πλήθος των μηνυμάτων που θα σταλούν ανάμεσα στους δύο επεξεργαστές λόγω του πλήθους των στοιχείων του πίνακα που υπολογίζουμε. Τα πρώτα 16 μηνύματα είναι για την αποστολή των στοιχείων του b και τα υπόλοιπα 16 είναι για την αποστολή των στοιχείων που υπολογίστηκάν και θα αποθηκευτούν στον c.

Β) Σε έναν μη προφανή χωρισμό του πίνακα που φαίνεται παρακάτω προκύπτει ότι:

Εικόνα που περιέχει σότζι, κτίριο

Περιγραφή που δημιουργήθηκε αυτόματα

Χρόνος επικοινωνίας στο δίκτυο :

tcomm  = ts + th \* l + m \* tw

Όπου :

* m \* tw : είναι ο χρόνος μεταφοράς του περιεχομένου της πληροφορίας. Στην συγκεκριμένη περίπτωση χωρισμού του πίνακα το m = 21 \* χ, με χ = το συνολικό μήκος της πληροφορίας σε words.

Επομένως προκύπτει :

tcomm  = ts + th \* l + (21\*χ) \* tw

## Άσκηση 2

1. Γραφική Παράσταση

**Εικόνα 1.** Χρόνος από την συνάρτηση internal dot product με των αριθμό βηματισμού της επανάληψης.

b) Ο αριθμός των πράξεων παρατηρούμε πως είναι ίσος με τον αριθμό των θέσεων του διανύσματος.

Δηλαδή οι προσθέσεις ισούνται με 220 και οι πολλαπλασιαμοί επίσης με 220.

Οπότε:

Πλήθος = Ν \* Προσθέσεις + Ν \* Πολλαπλασιαμοί

Πλήθος = 220 \* 1 + 220 \* 1

Πλήθος = 440 πράξεις

c)

|  |  |  |
| --- | --- | --- |
| **N** | **cpu time** | **FLOPS** |
| 1 | 0,00000315 | 141166668 |
| 2 | 0,00000475 | 100466668 |
| 4 | 0,0000044 | 102300001 |
| 8 | 0,00000505 | 94416668 |
| 16 | 0,0000047 | 94600001 |

**Εικόνα 2. Πίνακας αποτελεσμάτων.**

d) Βάση των μετρήσεων που πραγματοποιήσαμε, παρατηρούμε πως για 8 threads η απόδοση του συστήματος είναι χειρότερη καθώς έχει τον μεγαλύτερο χρόνο χρήσης της cpu για τον υπολογισμό των αποτελεσμάτων.

Επομένως το πιο κρίσιμο k για τους υπολογισμούς είναι το k = 8.

e) Όσο σπάμε τον πίνακα σε blocks περιμένουμε πως η αποδοτικότητα του προγράμματος θα αυξηθεί, διότι μοιράζουμε σε περισσότερους επεξεργαστές λιγότερους υπολογισμούς.

Αντιθέτως από τις μετρήσεις παρατηρούμε πως το πρόγραμμά μας είναι πιο αποδοτικό όταν τρέχει σειριακά σε 1 επεξεργαστή.

## Άσκηση 3

1. Γραφική Παράσταση

**Εικόνα 3.** Χρόνος από την συνάρτηση product of table με των αριθμό NxN block της επανάληψης.

b)

|  |  |  |
| --- | --- | --- |
| **n** | **cpu time** | **FLOPS** |
| 1 | 0,00003885 | 1366157 |
| 2 | 0,0000386 | 1288728 |
| 4 | 0,00005115 | 945038,6 |
| 8 | 0,0000547 | 896055,6 |
| 16 | 0,00006115 | 881834,7 |

**Εικόνα 3. Πίνακας αποτελεσμάτων.**

c) Όσο σπάμε τον πίνακα σε blocks περιμένουμε πως η αποδοτικότητα του προγράμματος θα αυξηθεί, διότι μοιράζουμε σε περισσότερους επεξεργαστές λιγότερους υπολογισμούς.

Σύμφωνα με τις παραπάνω μετρήσεις βλέπουμε πως η απόδοση του προγράμματος επιδεινώνεται όσο σπάμε τον πίνακα. Επομένως η βέλτιστη λύση ως προς την απόδοση είναι να τρέξουμε σειριακά το πρόγραμμα σε 1 επεξεργαστή.

## Άσκηση 4

1. Stencil\_1D
   1. Περιγραφή Αλγορίθμου

Ο αλγόριθμος θέτει της θέσεις 0,1,Ν-1,Ν-2 σε 5000. Γίνεται επανάληψη από την δεύτερη μέχρι τη προ προ τελευταία θέση του πίνακα και παίρνει τον μέσο όρο από της παλιές τιμές σε εύρος -2 με +2 από την τιμή του i και υπολογίσουμε το νέο i, οπού το i είναι ο αριθμός της επανάληψης.

* 1. Περιγραφή Προσέγγισης Παραλληλισμού

Ξεκινήσαμε να πειραματιζόμαστε που θα είναι πιο βέλτιστο. Πρώτα βάλαμε παραλληλοποίηση στην for loop. Μετά προσπαθήσαμε να βάλουμε και στην while loop και τελικά δεν τόσο αποδοτικό. Και προσθέσαμε και στην for loop του initialization.

* 1. Πειραματικά Αποτελέσματα

1. Περιγραφή μηχανήματος: Βλέπε παράτημα περιγραφή μηχανήματος

ii)

* 1. Συμπέρασμα

Παρατηρούμε ότι ανάλογα με το μέγεθος του πίνακα είναι πιο αποδοτικό να το διαμοιράσουμε σε διεργασίες για να επιτύχουμε πιο γρήγορο χρόνο εκτέλεσης.

1. Stencil\_2D
   1. Περιγραφή Αλγορίθμου

Σε επανάληψη παίρνουμε τον μέσο ορό γύρο από την θέση σε σχήμα σταυρού με εύρος -2 με +2 και υπολογίζει την νέα τιμή της θέσης του πίνακα.

* 1. Περιγραφή Προσέγγισης Παραλληλισμού

Ξεκινήσαμε να παραλληλοποιούμε την αρχικοποίηση του τελικού και του αρχικού πίνακα. Στην συνέχεια παραλληλοποίησαμε την πρώτη επανάληψη και πήραμε την χρονική τιμή του χρόνου. Έπειτα κάναμε το ίδιο και στην δεύτερη επανάληψη. Παρατηρήσαμε αντί να μειώνεται ο χρόνος ότι αυξανόταν. Μετά από δοκιμές αποφασίσαμε να βγάλουμε την τελευταία παραλληλοποίηση.

* 1. Πειραματικά Αποτελέσματα

1. Περιγραφή μηχανήματος: Βλέπε παράτημα περιγραφή μηχανήματος

ii)

* 1. Συμπέρασμα

Παρατηρούμε ότι όσο αυξάνουμε τον αριθμό των νημάτων χρειάζεται λιγότερος χρόνος για να ολοκληρωθεί ο υπολογισμός. Υπάρχουν κάποιοι αριθμοί νημάτων για τους οποίους η επικοινωνία είναι μεγαλύτερη σε σχέση με άλλα.

1. Mat\_Vec
   1. Περιγραφή Αλγορίθμου

Ο αλγόριθμος παίρνει μια θέση από μια στήλη του πίνακα και προσθέτει τα αθροίσματα από τον πολλαπλασιασμό της θέσης του πίνακα Α( i, j) με την τιμή Β(j), το j είναι η θέση από τον γραμμή.

* 1. Περιγραφή Προσέγγισης Παραλληλισμού

Ξεκινήσαμε να παραλληλοποιούμε την αρχικοποίηση του πίνακα Α και του διανύσματος Β. Στην συνέχεια παραλληλοποίησαμε την πρώτη επανάληψη και πήραμε την χρονική τιμή του χρόνου. Έπειτα κάναμε το ίδιο και στην δεύτερη επανάληψη. Παρατηρήσαμε αντί να μειώνεται ο χρόνος αυξανόταν. Μετά από δοκιμές αποφασίσαμε να βγάλουμε την τελευταία παραλληλοποίηση.

* 1. Πειραματικά Αποτελέσματα

1. Περιγραφή μηχανήματος: Βλέπε παράτημα περιγραφή μηχανήματος

ii)

* 1. Συμπέρασμα

Παρατηρούμε ότι για 4 thread έχουμε την καλύτερη απόδοση. Εάν αυξήσουμε το μέγεθος του πίνακα Β αυξάνεται ραγδαία η απόδοση για μικρότερο μέγεθος του πίνακα Α. Σε τετραγωνικό πίνακα έχουμε τις καλύτερες επιδόσεις.

## Άσκηση 5

Στην άσκηση ο πίνακας είναι σταθερού μεγέθους στο πρόγραμμα και εκτελέστηκε με διαφορετικά processes.

## 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