ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ

Συστήματα Παράλληλης και Κατανεμημένης Επεξεργασίας

Ενότητα: ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ No:05

Δρ. Μηνάς Δασυγένης

mdasyg@ieee.org

Τμήμα Μηχανικών Πληροφορικής και Τηλεπικοινωνιών

Εργαστήριο Ψηφιακών Συστημάτων και Αρχιτεκτονικής Υπολογιστών

http://arch.icte.uowm.gr/mdasyg

Άδειες Χρήσης

- Το παρόν εκπαιδευτικό υλικό υπόκειται σε άδειες χρήσης Creative Commons.
- Για εκπαιδευτικό υλικό, όπως εικόνες, που υπόκειται σε άλλου τύπου άδειας χρήσης, η άδεια χρήσης αναφέρεται ρητώς.

Χρηματοδότηση

- Το παρόν εκπαιδευτικό υλικό έχει αναπτυχθεί στα πλαίσια του εκπαιδευτικού έργου του διδάσκοντα.
- Το έργο «**Ανοικτά Ψηφιακά Μαθήματα του Πανεπιστημίου Δυτικής Μακεδονίας**» έχει χρηματοδοτήσει μόνο τη αναδιαμόρφωση του εκπαιδευτικού υλικού.
- Το έργο υλοποιείται στο πλαίσιο του Επιχειρησιακού Προγράμματος «Εκπαίδευση και Δια Βίου Μάθηση» και συγχρηματοδοτείται από την Ευρωπαϊκή Ένωση (Ευρωπαϊκό Κοινωνικό Ταμείο) και από εθνικούς πόρους.

Με τη συγχρηματοδότηση της Ελλάδας και της Ευρωπαϊκής Ένωσης

Περιεχόμενα

1. Σκοπός της άσκησης

- Παράλληλος υπολογισμός του **π** χρησιμοποιώντας αριθμητική ολοκλήρωση.
- Υπολογισμός αθροίσματος στοιχείων πίνακα.

2. Παραδοτέα

(A) 3 ερωτήσεις
(C) 4 ασκήσεις

3. Υπολογισμός π

Α) Γράψτε ένα πρόγραμμα που να υπολογίζει το **π** σύμφωνα με την εξίσωση:

$$\pi = \int_0^1 \frac{4}{1+x^2} dx$$

4. Χρόνος εκτέλεσης του προγράμματος

Βρείτε το χρόνο εκτέλεσης του προγράμματος.

Οδηγίες:

1. Για να υπολογίσουμε τον παραπάνω τύπο θα χρησιμοποιήσουμε την αριθμητική ολοκλήρωση από 0 έως 1, η οποία καταλήγει στην παρακάτω σχέση:

$$\pi = \int_0^1 \frac{4}{1+x^2} = \sum_{i=0}^{N-1} \frac{4W}{1+[(i+0.5)W]^2}$$

όπου:

- **`N`** ο αριθμός των διαστημάτων στον οποίον χωρίζουμε την περιοχή ολοκλήρωσης από 0 έως 1 *(η ακρίβεια αυξάνεται με τον αριθμό των διαστημάτων).*
- **`W`** το πλάτος του κάθε διαστήματος, το οποίο ισούται με **`1/N`**.

2. Για να υπολογίσουμε τον παραπάνω τύπο θα χρησιμοποιήσουμε την αριθμητική ολοκλήρωση από 0 έως 1, η οποία καταλήγει στην παρακάτω σχέση:

```
int  i, N=512;
float pi, W=1.0/N;
pi=0.0;
for (i=0;i<N;i++)
     {pi+=4*W/(1+(i+0.5)*(i+0.5)*W*W);}
```

Παρατηρήστε ότι ο παραπάνω αλγόριθμος υπολογίζει σε **N** βήματα το παραπάνω άθροισμα. Θα μπορούσαμε λοιπόν να υπολογίζουμε το άθροισμα με το να χρησιμοποιήσουμε έναν αριθμό διεργασιών όπου θα χωρίσουμε τα **N** βήματα στις διεργασίες και κάθε διεργασία θα υπολογίζει το άθροισμα στο διάστημα που της αντιστοιχεί.

3. **(C1)** Κατασκευάστε το πρόγραμμα **c1.c** το οποίο υπολογίζει το π σε μια διεργασία *(χωρίς παράλληλη επεξεργασία και συνεπώς δε χρειάζεται ανταλλαγή μηνυμάτων)* χρησιμοποιώντας τον αλγόριθμο που έχει δοθεί παραπάνω. Εκτελέστε τον και επιβεβαιώστε ότι το π έχει υπολογιστεί σωστά.
4. **(C2)** Αντιγράψτε το αρχείο **c1.c** στο αρχείο **c2.c** το οποίο θα δέχεται μια παράμετρο από τη γραμμή εντολών *(command line)* και η οποία θα είναι ο αριθμός N. Αν δε δίνεται παράμετρος στη γραμμή εντολών το πρόγραμμα θα τερματίζει. Η εκτέλεση δηλαδή του προγράμματος θα γίνεται ως εξής:

```
./c2 512
```

5. Αφού επιβεβαιώσετε την ορθή λειτουργία του **p_single2.c** αντιγράψτε το στο αρχείο **c3.c**.
6. **(C3)** Επεξεργαστείτε το αρχείο **c3.c** και προβείτε στις εξής αλλαγές:

    1. Θα πρέπει να γίνεται έλεγχος αρχικοποίησης **`MPI_Init()`**
    2. Θα πρέπει να εκτελείται με τουλάχιστον 2 διεργασίες. Τοποθετείστε την κατάλληλη συνθήκη ελέγχου για το μέγεθος του Communicator.
    3. Η διεργασία 0 θα διαβάζει μια παράμετρο από τη γραμμή εντολών η οποία θα είναι η **N** και θα κάνει broadcast την ακέραια τιμή **N** σε όλες τις υπόλοιπες διεργασίες.
    4. Κάθε διεργασία *(ακόμη και η 0)* θα υπολογίζει το άθροισμα που αντιστοιχεί στη συγκεκριμένη διεργασία. Θα πρέπει να βρείτε το μαθηματικό τύπο πως ακριβώς υπολογίζεται το διάστημα. Μη ξεχνάτε ότι δεν είναι γνωστός ο αριθμός των διεργασιών από πριν οπότε δε μπορείτε να προ-ρυθμίσετε το διάστημα που θα εργάζεται η κάθε διεργασία. Τα μόνα σχετικά στοιχεία που γνωρίζετε είναι ο αριθμός των συνολικών διεργασιών και ο αριθμός rank της κάθε διεργασίας. Για παράδειγμα αν έχουμε 3 διεργασίες, σημαίνει ότι θα χωρίσουμε τα βήματα **N** σε 3 διεργασίες και άρα η διεργασία 0 θα υπολογίσει το άθροισμα από 0 έως 1N/3 *(=N/3)*, η δεύτερη διεργασία N/3 έως 2N/3 και η τρίτη διεργασία 2N/3 έως 3N/3 *(συνολικά N)*. *(***(A1)** *σκεφτείτε αν μπορείτε να χρησιμοποιήσετε τη* **mpi_scatter();** *και δικαιολογήστε τη θετική ή αρνητική σας άποψη.)*

5. Όλα τα αποτελέσματα θα πρέπει να σταλούν σε μια διεργασία για να βγάλει το τελικό άθροισμα. Αν θέλουμε να σταλούν όλα τα αποτελέσματα στη διεργασία 0 τότε:
    1. Η διεργασία 0 θα έχει μέσα σε ένα βρόχο επαναλήψεων τόσες εντολές ασύγχρονης λήψης (`MPI_Irecv()`) όσες οι υπόλοιπες διεργασίες.
    2. Σε κάθε λήψη θα εκτυπώνει αυτό που έχει λάβει από κάθε διεργασία *(π.χ.* `Process 2 send value: 3.01`*)*
    3. Η διεργασία 0 θα έχει τόσες εντολές `MPI_Wait()` όσες είναι οι εντολές ασύγχρονης λήψης *(μια για τη κάθε λήψη)*. Αυτό απαιτείται για να μη συνεχίσει η εκτέλεση χωρίς να έχουμε έγκυρα δεδομένα.
    4. Κάθε διεργασία μόλις υπολογίζει το κομμάτι της εργασίας που της έχει ανατεθεί θα στέλνει τα δεδομένα σύγχρονα στη διεργασία 0.
    5. Μόλις η διεργασία 0 λάβει από όλες τις διαδικασίες τα αποτελέσματα τα προσθέτει και εκτυπώνει το τελικό αποτέλεσμα μαζί με τα βήματα που χρησιμοποιήθηκαν:
    (`After 512 steps p is: 3,1415`)

5. Υπολογισμός χρόνου εκτέλεσης του προγράμματος

(C4) Αντιγράψτε το αρχείο **c3.c** σε **c4.c**. Κάντε τις εξής αλλαγές:

1. Τοποθετήστε κατάλληλα μετά το σημείο `mpi_init()` τη συνάρτηση μέτρησης του χρόνου.
2. Τοποθετήστε στο τέλος, πριν το τελευταίο `printf()` τη συνάρτηση μέτρησης του χρόνου.
3. Κάντε την πράξη αφαίρεσης των 2 χρόνων και τροποποιήστε το τελευταίο `printf()`, ώστε να εκτυπώνει και το χρόνο και τον αριθμό των διεργασιών:

```
After 512 steps and 4.05 seconds using 10 processes p is: 3.1415
```

6. Υπολογισμός αθροίσματος στοιχείων πίνακα

Το παρακάτω πρόγραμμα υπολογίζει το άθροισμα των στοιχείων ενός πίνακα:

```
#include <stdio.h>
#define SIZE 50000
main (int argc, char *argv[])
{
int i, sum, Tsum,  data[SIZE];
/* αρχικές τιμές στον data[] */
for (i=0; i<SIZE; i++) { data[i] =  i+1; }
Tsum = 0;

/* υπολογισμός αθροίσματος */
for (i=0; i < SIZE; i++) {  Tsum = Tsum + data[i]; }
printf ("Total Sum  %d \n", Tsum);
return (0); }
```

Το παρακάτω πρόγραμμα υπολογίζει το άθροισμα των στοιχείων ενός πίνακα παράλληλα:

```
#include <stdio.h>
#include <mpi.h>
#define SIZE 50000

MPI_Status status;
main (int argc, char *argv[])
{
int size, proc, rank, dest, index, i, source, chunksize, sum, Tsum;
int data1[SIZE], data2[SIZE];
MPI_Init (&argc, &argv);
MPI_Comm_rank (MPI_COMM_WORLD, &rank);
MPI_Comm_size (MPI_COMM_WORLD, &size);
proc = size; /* Συνολικός Αριθμών proc */
/* πόσο κομμάτι του πίνακα θα στείλουμε σε κάθε rank */
chunksize = (SIZE / proc);

/* Η διεργασία 0 δίνει αρχικές τιμές, στέλνει σε κάθε υπόλοιπη διεργασία το τμήμα του
πίνακα που αναλογεί και λαμβάνει τα αποτελέσματα */
if (rank == 0) {
               /* αρχικές τιμές στον data1[] */
               for (i=0; i < SIZE; i++) {  data1[i] =  i+1; }
               index = 0;
               for (dest=1; dest < proc; dest++) {
  MPI_Send (&index, 1, MPI_INT, dest, 0, MPI_COMM_WORLD);
  MPI_Send (&data1[index], chunksize, MPI_INT, dest, 0,  MPI_COMM_WORLD);
               index = index + chunksize;  }

Tsum = 0;
for (i=1; i < proc; i++) {
       source = i;
       /* λήψη αποτελεσμάτων από κάθε διεργασία */
       MPI_Recv (&sum, 1, MPI_INT, source, 0,MPI_COMM_WORLD, &status);
Tsum = Tsum + sum;
printf("-----------------------------------------------------------------\n");
printf("Received from process: %d, Sum: %d \n", source, sum); }

printf("Total Sum Received: %d \n", Tsum);
printf("All Done! \n");  }

/* Κάθε διεργασία με rank>0 θα υπολογίζει το άθροισμα */
if (rank > 0) {
       MPI_Recv (&index, 1, MPI_INT, 0, 0, MPI_COMM_WORLD, &status);
```

```
        MPI_Recv (&data2[index], chunksize, MPI_INT, 0, 0,MPI_COMM_WORLD,&status);
sum = 0; sum = 0;
for (i = index; i < index + chunksize; i++) { sum = sum + data2[i]; }
/* κάθε διεργασία με rank>0 θα στέλνει το μερικό άθροισμα */
MPI_Send (&sum, 1, MPI_INT, 0, 0, MPI_COMM_WORLD);
}


MPI_Finalize();
return(0);
}
```

1. **(A2)** Να γράψετε το πρόγραμμα υπολογισμού σειριακού αθροίσματος στο αρχείο **sum_serial.c** και να το εκτελέσετε. Σημειώστε την τιμή που υπολογίζετε.

2. **(A3)** Να γράψετε το πρόγραμμα υπολογισμού παράλληλου αθροίσματος **sum_parallel.c** *(που αθροίζει τα ίδια στοιχεία)* και να σημειώστε την τιμή του αθροίσματος. Είναι η ίδια με την παραπάνω; Αν όχι γιατί; Αν είναι διαφορετική να προβείτε στις ανάλογες ενέργειες για να διορθωθεί το πρόβλημα που εντοπίσατε.

(συνεχίζεται στο επόμενο εργαστήριο…)