Παράλληλη Επεξεργασία

Εξεταστική Περίοδος: Φεβρουαρίου 2024 (δεν έγινε η εξέταση)

Π. Χατζηδούκας - Ε. Γαλλόπουλος

Όνομα: _____

Επώνυμο: _____

ΑΜ: _____ Έτος:_____

Θέμα 1ο (3,5 μονάδες)) [απαντήστε εδώ ή στην κόλλα σας]

1.1 Μεταβλητές τύπου pthread_mutex_t χρησιμοποιούνται σε προγράμματα για: a. Υλοποίηση ατομικών εντολών b. Φράγματα c. Συγχρονισμό d. Αμοιβαίο αποκλεισμό	**1.2** Ποιο από τα παρακάτω δεν είναι τμήμα του προγραμματιστικού μοντέλου OpenMP; a. Μεταγλωττιστής b. Οδηγίες προς τον μεταγλωττιστή c. Βιβλιοθήκη χρόνου εκτέλεσης d. Μεταβλητές συστήματος
1.3 Μια υλοποίηση του προτύπου των POSIX Threads επιτρέπεται να υλοποιήσει τα νήματα: a. Μόνο ως νήματα επιπέδου πυρήνα b. Μόνο ως νήματα επιπέδου χρήστη c. Είτε ως νήματα επιπέδου χρήστη είτε ως νήματα επιπέδου πυρήνα d. Μόνο ως έργα (tasks)	**1.4** Το παράλληλο προγραμματιστικό μοντέλο MPI έχει σχεδιαστεί κυρίως για συστήματα: a. Κοινής μνήμης b. Κατανεμημένης μνήμης c. Κατανεμημένης κοινής μνήμης d. Κοινής κατανεμημένης μνήμης

1.5 Η συλλογική (collective) επικοινωνία στο MPI, σε σχέση με την υλοποίηση των αντίστοιχων λειτουργιών χρησιμοποιώντας μόνο συναρτήσεις επικοινωνίας «σημείο-προς-σημείο» (point-to-point):
a. Υποχρεώνει το πρόγραμμα MPI να μεταφέρει περισσότερα δεδομένα μεταξύ διεργασιών
b. Κάνει το πρόγραμμα MPI πιο ευανάγνωστο και πιο αποτελεσματικό στη μεταφορά δεδομένων
c. Κάνει το πρόγραμμα MPI πιο γρήγορο στο τμήμα εκείνο που πραγματοποιεί τους υπολογισμούς
d. Όλα τα υπόλοιπα
	1.6 Δίνεται ο παρακάτω κώδικας OpenMP. **(1 μονάδα)**
	`int test_var = 0;`
	`#pragma omp parallel num_threads(4) reduction(+:test_var)`
	`{`
	`  test_var += omp_get_thread_num();`
	`}`
	Ποια θα είναι η τιμή της μεταβλητής test_var μετά την εκτέλεση της παράλληλης περιοχής; Εξηγήστε αναλυτικά.
	a. 0 b.4 c. 6 d. Απροσδιόριστη

Θέμα 2ο (3 μονάδες)

Η παρακάτω συνάρτηση υπολογίζει και εκτυπώνει το μέγιστο στοιχείο ενός διανύσματος αλλά και τη θέση αυτού. Παραλληλοποιήστε τη συνάρτηση με χρήση του μοντέλου OpenMP. Περιγράψτε και αιτιολογήστε τη λύση σας.

```
void find_max_and_pos(int *A, int N) {
   int max_val = A[0];
   int max_pos = 0;

   for (int i=1; i<N; i++) {
      if (A[i] > max_val) {
         max_val = A[i];
         max_pos = i;
      }
   }
   print("max value = %d at position %d\n", max_val, max_pos);
}
```

Σχετικές οδηγίες του OpenMP:

```
#pragma omp parallel
#pragma omp for
#pragma omp critical
```

Θέμα 3ο (2 μονάδες)

Στον παρακάτω κώδικα, η εξωτερική thread-safe συνάρτηση do_work() εκτελείται για κάποιο χρονικό διάστημα (π.χ. αρκετά δευτερόλεπτα) και επιστρέφει έναν πραγματικό αριθμό που διαφέρει κάθε φορά. Κάθε νήμα καλεί τη συνάρτηση αυξάνοντας κάθε φορά τον μετρητή (counter) με την επιστρεφόμενη τιμή.

a. Αναφέρετε πιθανά προβλήματα στον κώδικα.
b. Τροποποιήστε τον κώδικα ώστε να τα λύσετε και αιτιολογήστε συνοπτικά τη λύση σας.

```
#include <stdio.h>
#include <pthread.h>
extern float do_work(); // implemented elsewhere

float counter = 0;

void *func (void *arg)
{
  for (int i = 0; i < 10; i++)
    counter += do_work();        // execute some 'work'

  return NULL;
}

int main (int argc, char * argv[]) {
  pthread_t id[4];

  for (int i = 0; i < 4; i++)
    pthread_create(&id[i], NULL, func, NULL);

  printf("counter=%f\n", counter);
  return 0;
}
```

Σχετικές συναρτήσεις και σταθερές:

```
PTHREAD_MUTEX_INITIALIZER
pthread_mutex_init (pthread_mutex_t *mutex, const pthread_mutexattr_t *attr);
pthread_mutex_lock (pthread_mutex_t *mutex);
pthread_mutex_unlock (pthread_mutex_t *mutex);
```

Θέμα 4ο (1,5 μονάδες)

Δίνεται ο παρακάτω κώδικας MPI:

```
#include <mpi.h>
#include <stdio.h>

int main(int argc, char** argv)
{
   MPI_Init(&argc, &argv);

   int rank, size;
   MPI_Comm_rank(MPI_COMM_WORLD, &rank);
   MPI_Comm_size(MPI_COMM_WORLD, &size);

   double total = 0.0;
   double local = rank+1.0;

   MPI_Reduce(&local, &total, 1, MPI_DOUBLE, MPI_PROD, 0, MPI_COMM_WORLD);

   if (rank == 0) printf("%lf\n", total);

   MPI_Finalize();
   return 0;
}
```

Τι τιμή εκτυπώνει ο παραπάνω κώδικας όταν εκτελεστεί με 5 διεργασίες (mpirun -n 5 <executable>); Εξηγήστε αναλυτικά.

Καλή επιτυχία!