PARALLEL AND DISTRIBUTED COMPUTING

2018/2019 2nd Semester

2nd Exam July 2nd, 2019 Duration: 2h00

- No extra material allowed. This includes notes, scratch paper, calculator, etc.
- Give your answers in the available space after each question. You can use either Portuguese or English.
- Be sure to write your name and number on all pages, **non-identified pages will not be graded!**
- **Justify all your answers.**
- Do not hurry, you should have plenty of time to finish this exam. Skip questions that you find less comfortable with and come back to them later on.

I. (1 + 1 + 1 + 1 + 1 = 5 val.)

1. The following OpenMP code counts the number of odd integers in a given array, with N=1000000:

   ```
   int data[N];
   int oddCount=0;
   #pragma omp parallel for
   for ( int i = 0; i < N ; i++ )
     if( data[i]%2 )
       #pragma omp atomic
       oddCount++;
   ```

   Optimize this code in order to avoid the need for a `critical` or `atomic` directives and to minimize the overhead introduced by the OpenMP management and syncronization mechanisms.

2. Explain the meaning of *false sharing*.

3. Consider the following OpenMP code, assuming that the program was executed in a system with 4 threads (`OMP_NUM_THREADS=4`):

```
#define iter 16;

#pragma omp parallel for private(j)
for ( i = 0; i < iter ; i++ ) {
  for ( j = iter - (i+1); j < iter ; j++ ) {
    // This function has a computing time of 2s
    compute_iteration(i, j, ...) ;
  }
}
```

a) Fill out the following table with a possible thread allocation of the first loop iterations (index i), assuming a static scheduling defined by the OpenMP directive `schedule(static)`. Indicate which thread performs each iteration and how long that iteration takes.
Determine the approximate execution time per thread and the total execution time.

#iter	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Thread																
Time/iter [s]																

	Thread 0	Thread 1	Thread 2	Thread 3
Individual thread execution time [s]				
Total execution time [s]				

b) Repeat the previous question assuming a dynamic scheduling defined by the OpenMP directive `schedule(dynamic,2)`.

#iter	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Thread																
Time/iter [s]																

	Thread 0	Thread 1	Thread 2	Thread 3
Individual thread execution time [s]				
Total execution time [s]				

c) Justify which of the previous schedules would be best for a generic case (variable number of iterations and threads).

II. (1,5 + 1,5 + 1 + 1 = 5 val.)

1. What is the difference between `MPI_Waitall`, `MPI_Waitany` and `MPI_Waitsome`?

2. Explain the functionality of `MPI_Allgather`. For 4 processes, draw an illustration and explain how information is transmitted between each of the processes, during the execution of this function, assuming an optimized implementation.

3. Consider the function `MPI_Gatherv` presented below.

```
int MPI_Gatherv(const void *sendbuf, int sendcount, MPI_Datatype sendtype,
                void *recvbuf, const int *recvcounts, const int *displs,
                MPI_Datatype recvtype, int root, MPI_Comm comm)
```

Note that:

- `recvcounts` is an integer array containing the number of elements that are received from each process;
- `displs` is an integer array, for which entry `i` specifies the displacement relative to `recvbuf` at which to place the incoming data from process `i`;
- `root` is the rank of the receiving process.

Provide an implementation for `MPI_Gatherv`, using other MPI functions. Even a simple, but correct, implementation is acceptable.

III. (0,75 + 0,75 + 0,75 + 0,5 + 2,25 = 5 val.)

1. A parallel program running on a machine with 11 processors spends 10% in purely serial computation and the rest is completely parallel.

   a) What was the speedup obtained?

   b) What is the maximum possible speedup for this program, no matter how many processors are used?

2. Consider the Karp-Flatt Metric.

   a) What is the value of the Experimentally Determined Serial Fraction for a system with an ideal speedup?

   b) In general, why does the value of the Experimentally Determined Serial Fraction tend to increase with the number of processors used?

3. Consider a problem with a sequential algorithm that runs in $\Theta(n\sqrt{n})$ and with a parallel implementation that runs in $\Theta(\frac{n\sqrt{n}}{p}\log p)$ with p processors and whose overhead (communication + redundant computation) per processor is given by $\Theta(n)$. If the required memory grows with n, compute the scalability function for this parallel algorithm. Discuss the result obtained.

IV. (1,25 + 0,5 + 0,75 + 1,25 + 1,25 = 5 val.)

1. Explain the difference between the Strong and Weak Scalability metrics.

2. The precision of Monte Carlo methods increases at a rate of $1/\sqrt{n}$ where n is the number of samples used.

   a) Why is the speedup in this methods close to p, where p is the number of parallel processors used?

   b) In what sense can we say that a Monte Carlo method in a computer system with p processors improves the accuracy by $\sqrt{p}$?

3. Consider a parallel system with p processors running an application that has been split in parallel tasks, and that the average runtime of a task is t. Discuss for which values of p and t a centralized work-pool model would be more effective.

4. Briefly explain the main difference between the plain Parallel Quicksort algorithm and the Hyper-quicksort algorithm. What is the main objective of this improvement?