Uppsala University
Department of Information Technology
Scientific Computing

Programming of Parallel Computers
2009-06-03

Programming of Parallel Computers, 2009-06-03

Time: $14^{00} - 19^{00}$
Help: None
Each of the six problems below can give up to five points. For maximum points, you must give detailed answers and motivate your assumptions.

1. a) Describe the different scheduling classes of the for-directive in OpenMP (`static, dynamic, guided, auto, runtime`).
   b) Explain how one can do explicit user controlled partitioning of a loop in OpenMP.
   c) What is the purpose of the nowait clause in OpenMP and what is the effect of this clause (why should you be careful when using this)? In what situations is it useful?

2. In OpenMP 3.0 a new directive has been introduced, the *task*-directive. Explain how this directive works and show how it can be used, preferably by constructing an example code.

3. The enumeration-sort algorithm can be written as:

```
for ( j=0; j<len; j++ )
   {
   rank=0;
   for ( i=0; i<len; i++)
           if (indata[i]<indata[j]) rank++;
   outdata[rank]=indata[j];
   }
```

   The algorithm can be parallelized either on the j-loop or on the i-loop or on both loops. Make a nested two-level parallelization of both loops in OpenMP of the enumeration-sort algorithm. Assume that you have totally 8 threads available.

4. When communicating data with MPI, a significant part of the overhead is due to communication start-up delays. What can we do to avoid/minimize these delays? Under what circumstances will the performance impact of start-up delays be particularly noticable?

5. Assume that we have 11 independent tasks (can be done in any order) with relative run-times $\{2, 10, 2, 3, 1, 5, 8, 2, 10, 12, 15\}$ and three processors. Describe an algorithm to load balance the tasks to the processors and apply your algorithm to the problem.

6. In assignment 2 you were required to parallelize the Gram-Schmidt algorithm in two ways with OpenMP. Now, sketch a distributed memory parallelization of the algorithm using MPI. The syntax does not have to be exact and the program not complete but the relevant code segments, details and arguments must be included. Assume that m is divisible with P (the number of processors) and that all vectors v are initiated on processor P_0. In the end all vectors q should be collected to processor P_0.

```
**program** Modified-Gram-Schmidt
**declare**
   real $v(1:n, 1:m)$, $q(1:n, 1:m)$
**begin**
   $q(:,1) = v(:,1)/\|v(:,1)\|_2$;

   for $i = 2$ to $m$ do
      for $j = i$ to $m$ do
         $\sigma = q(:,i-1)^T v(:,j)$
         $v(:,j) = v(:,j) - \sigma q(:,i-1)$;
      $q(:,i) = v(:,i)/\|v(:,i)\|_2$;
**end**
```

Good Luck!