CZ4102 High Performance Computing

Tutorial 3

To be discussed on 27 Oct 2006.

Please solve the tutorial questions in advance.

1. One of the advantages of non-blocking communication operations is that they allow the transmission of the data to be done concurrently with computations. Discuss the type of restructuring that needs to be performed on a program to allow for the maximal overlap of computation with communication. Is the sending process in a better position to benefit from this overlap than the receiving process?

2. Suppose n pieces of work are allocated in cyclic fashion to p processes.

   (a) Which pieces of work are assigned to process k, where $0 \leq k \leq p\text{-}1$?
   (b) Which process is responsible for piece of work j, where $0 \leq j \leq n\text{-}1$?
   (c) What are the most pieces of work assigned to any process?
   (d) Identify all processes having the most pieces of work.
   (e) What are the fewest pieces of work assigned to any process?
   (f) Identify all processes having the fewest pieces of work.

3. Given a set of five unsigned, eight-bit integers with decimal values 13, 22, 43, 64, and 99, determine the decimal result of the following reductions:

   (a) Add
   (b) Multiply
   (c) Maximum
   (d) Minimum
   (e) Bitwise or
   (f) Bitwise and
   (g) Logical or
   (h) Logical and

4. A simple streaming media player consists of a thread monitoring a network port for arriving data, a decompressor thread for decompressing packets and generating frames in a video sequence, and a rendering thread that displays frames at programmed intervals. The three threads must communicate via shared buffers - an in-buffer between the network and decompressor, and an out-buffer between the decompressor and renderer. Implement this simple threaded framework. The network thread calls a dummy function listen_to_port to gather data from the network. For the sake of this program, this function generates a random string of bytes of desired length. The decompressor thread calls function decompress, which takes in data from the in-buffer and returns a frame of predetermined size. For this exercise, generate a frame with random bytes. Finally the render thread picks frames from the out buffer and calls the display function. This function takes a frame as an argument, and for this exercise, it does nothing. Draw the thread model, and simulate the operations of this threaded framework based on shared3.java.

Please solve the tutorial questions in advance to gain the maximum benefit from our tutorial session.