COSC 6352, Advanced Operating Systems

Fall 2018, Programing Assignment #2

Date Assigned: Monday, October 29, 2018

Due Date: Friday, November 09, 2018 at 11:59 p.m. (use blackboard)

DESCRIPTION

Write a C or C++ program named berkeley.c or berkeley.cpp (executable name should be berkeley) to implement the Berkeley time synchronization algorithm as discussed in the class using MPI for communication. The screenshot from jumpshot should be saved as berkeley.pdf. Read the values for the clocks at each processor and the coordinator processor from an input file specified as a command line argument.

INPUT TO THE PROGRAM

The input will be in the following format:

coordinator process rank
time for process with rank 0
time for process with rank 1
time for process with rank 2
...
time for process with rank n
integer value (an abs(value) which deviated by more than this value is ignored)

Example file named berkeley.txt. The content in brackets is not part of the input file.
3 (coordinator process)
16:20 (time at rank 0)
13:58 (time at rank 1)
14:12 (time at rank 2)
09:48 (time at rank 3)
02:12 (Ignore if the time difference between coordinator and another process is more than abs(2:12))

Sample Execution:
mpirun -np 4 berkeley berkeley.txt

PROGRAM OUTPUT

First print the rank of the process performing the coordinator using a printf() statement like this:

printf("I am process with rank %d acting as the coordinator process", rank);

The coordinator process should then broadcast it's time read from the file to all other processes. When the processes receives the time print a message. Use printf() statements like these:

printf("Coordinator process is sending time %d\n", rank);
printf("Process %d has received time %s\n", rank, time);

Once the processes receive the time they should calculate the differential of their clocks and send it back to the coordinator process. Use printf() statements like these:

printf("Process %d is sending time differential value of %d to process %d\n", rank, diff, rank);
printf("Process %d has received time differential value of %d\n", rank, diff);

If the coordinator ignores any values use printf() statements to output the information.

printf("Coordinator Process is ignoring time differential value of %d from process %d\n", diff, rank);

After the coordinator process calculates the average of the time differential's display it using a printf() statement

printf("Time differential average is %f", average);

Calculate the amount by which each clock's value should be adjusted, and send it to the appropriate processor. Once the processors receive the value adjust the local clocks and display their local time. Use printf() statements like these to clearly show the process.

printf("Coordinator process is sending the clock adjustment value of %f to process %d, value, rank);
printf(Process %d has received the clock adjustment value of %f, rank, value);
printf("Adjusted local time at process %d is %s", rank, newtime);

MISCELLANEOUS

1. Your MPI program will read the file name as command line argument and write the output to standard output.
2. The only MPI functions you are permitted to use are MPI_Init(), MPI_Finalize(), MPI_Comm_rank(), MPI_Comm_size(), MPI_Send(), MPI_Recv(), MPI_Bcast() and MPI_Barrier()