Worksheet For OpenMP Training

Hello World

Q1: How many times is Hello World Printed without the "#pragma omp parallel"? _____

Q2: How many times is Hello World printed with the "#pragma omp parallel" construct? _____

Q3: How many times is Hello World printed with the "#pragma omp critical"? _____

Parallelize Compute Pointes on a Line (OpenMP ←→ C++ Lambda)

Q1: (01NoLambda.cpp) How many global variables needed for initial implementation? _____

Q2: (02Lambda.cpp) How many global variables needed for C++ lambda Implementation? _____

Q3: (03Lambda_SeparateTasks.cpp) How many utility variables needed for breaking loop into 4 distinct sections? _____

Q4: (04Lambda_MultiProcessor.cpp) What happens if MAXTHREADS is set to 8 ? _____

Q5: (05Lambda_ManualPatition.cpp) What happens if "num_threads(MAXTHREADS)" is removed? _____

Q6: (06Lambda_ParallelFor.cpp) Change the scheduler to be "schedule(static,34)", what happens? _____

Chapter 6 Worksheet

Objective 1: Compile and run the serial Mandelbrot program

1. Go to Chapter6/obj61/
2. Run the command:
   make
3. Run the program with:
   ./61.exe
4. Record the printed times below:
   Time_to_calc: _____
   Time_to_print: _____
   Time_to_signature: _____
   Time_total: _____
   Wall_Time: _____

Objective 2: Use the compiler's loop profiler to find hotspots in the code

1. Go to Chapter6/obj62b
2. Run the command:
   make
3. Run the program with:
   ./62b.exe

4. This program is compiling using the exact same source code as 61.exe so why is it running slower? To get a better idea, open the Makefile in obj61/ and obj62a/ and compare the compiler options.
5. Start the loopprofileviewer to browse the XML file generated using:
   loopprofileviewer.sh <name_of_xml>.xml

This program was run with the compiler flag -inline-level=0 to stop the inlining of loops so that the loopprofileviewer could see them. To see which loops the compiler inlines follow these steps:

6. Go to Chapter6/obj62c
7. Run the command:
   make
8. Run the program with:
   ./62c.exe
9. You can the open ipo_out.optrpt to view where the compiler used inlining using:
   vi ipo_out.optrpt

Objective 3: Use the compiler's auto-parallizer to find parallizable loops

1. Go to Chapter6/obj63a
2. Run the command:
   make
3. Run the program with:
   ./63a.exe
4. Record the printed times below and compare them to the times in Objective 1:
   Time_to_calc: _____
   Time_to_print: _____
   Time_to_signature: _____
   Time_total: _____
   Wall_Time: _____
5. You can the open ipo_out.optrpt to view where the compiler auto parallelized using:
   vi ipo_out.optrpt

Objective 4: Use Amplifier XE to analyze hotspots

1. Go to Chapter6/obj64b
2. Run the command:
   make
3. Follow the instruction for Activity 6-4 on page 173 starting at step 3 and skipping step 6. Make sure for step 4b you point to 64b.exe.