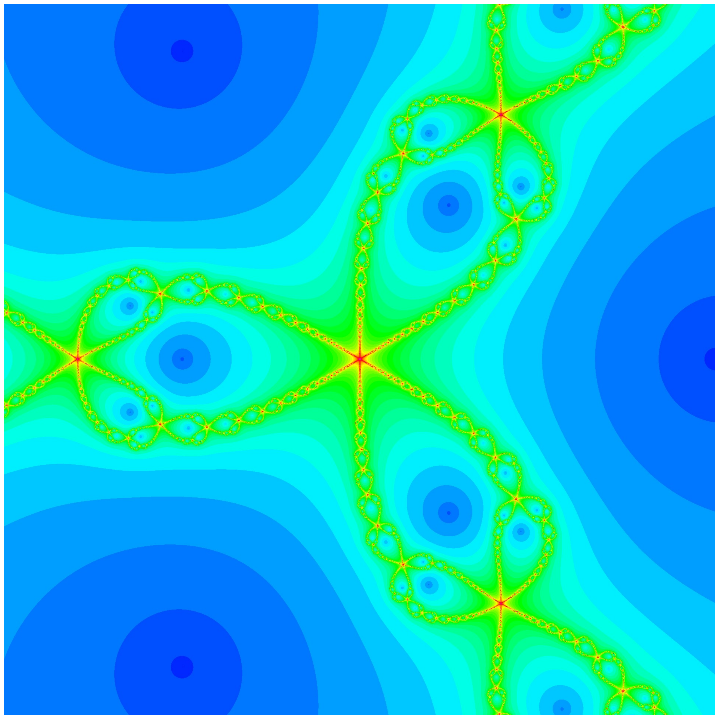
# THREAD LEVEL PARALLELISM



05/02/17

TLP Project (DUE: Fri. 5/12/2017, 10AM)

Now that you understand the basics of pthreads and MPI, you have been assigned the following task:

OwensCorp Management needs a recommendation regarding the next purchase of computers. *Should management purchase fewer more expensive quad-core systems or more single core systems that will be clustered? How does processing power on multi-core compare with clusters?*

# Thread Level Parallelism

## BACKGROUND

Suppose your company wants to purchase some new computers. There are two types of computers that you can choose: a large cluster of single-core systems, or fewer expensive quad-core systems. Suppose your budget is limited. Please write a report to a technical manager who needs to make purchase decisions with your suggestions about the best type of computers to purchase.

## INTRODUCTION

In the TLP project, you must parallelize a program that generates a fractal image.  The code lends itself to parallelizing.  You must write 2 parallel versions of this code: one using pthreads and the other using MPI.  You must then run the parallelized programs on various pthreads and MPI configurations in order to understand how performance scales when you add threads or cluster members.  Understanding this is important, as the report you will write will recommend either higher performance (and higher cost) quad-core machines, or cheaper single-core machines that will be clustered.

The application used in this project is a simple program that computes the Newton fractal for a polynomial function. Starting at a point in the complex plane, the program uses Newton's Method to converge to one of the roots of the polynomial. The resulting image is color-coded based on how many iterations it took for each point to converge. For a more detailed explanation, please visit Wikipedia (http://en.wikipedia.org/wiki/Newton_Fractal).

## TOOLCHAINS

- Pthreads how-to (http://www.yolinux.com/TUTORIALS/LinuxTutorialPosixThreads.html)
- MPI Documentation (https://www.open-mpi.org/doc/current/)
- MPI Hello World (http://mpitutorial.com/tutorials/mpi-hello-world/)

## SAMPLE CODE

(Download from Canvas: Resources / Projects / TLP / Code)

- newton.c - Sample code that serially computes a Newton fractal and outputs a BMP file.
- vecsum.c - Sample code showing MPI scatter and gather in action.
- dotprod_pthreads.c - finds the dot product of two vectors using pthreads
- dotprod_mpi.c - finds the dot product between two vectors using MPI
- compare.c - compares your parallelized image to the image from the sample code.

## OTHER REFERENCES

- Gather Web page - Web page illustrating MPI gather (http://mpi.deino.net/mpi_functions/MPI_Gather.html)

## BEFORE YOU START

### 1. Make sure you have followed the MPI setup instructions from the warmup

### 2. View the processor speed information

```
cat /proc/cpuinfo
```

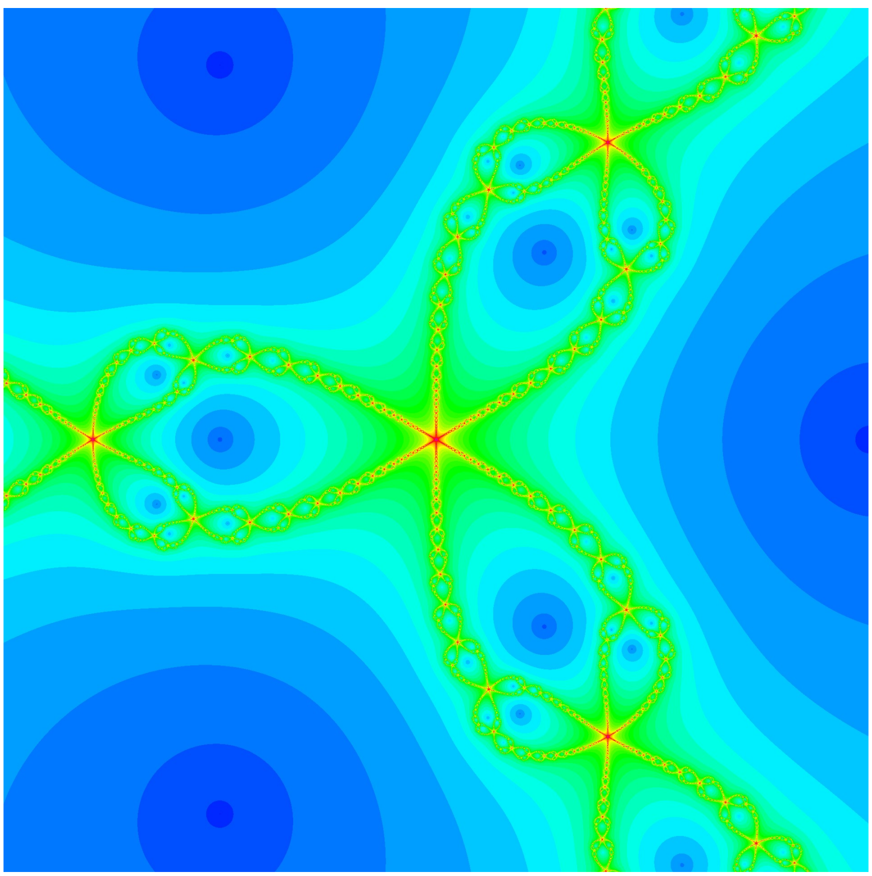
### 4. Compile serial version of newton.c

newton.c requires the math library (-lm) to compile, so you can compile it like this:

```
gcc -o newton newton.c -lm
```

Run the sample program.

Output (newton.bmp) for the default settings in newton.c:



### 5. Parallelize the Newton fractal program

The goal here is to review the serial code and determine a way to parallelize the algorithm. I suggest you go for parallelizing along the rows, as the algorithm is already set up that way. You need to create 2 parallel Newton fractal programs: the first using pthreads and the other using MPI. In your report you will briefly describe how you did this step, both for pthreads and MPI.

Tips:

- Review the documentation and the programs from the warmup to learn more about using pthreads and MPI.
- Write your programs so that you can easily change the number of threads/hosts.
- Verify that the BMP file produced in parallel matches the file you get running the sample code. You can use the command `md5sum newton.bmp` to generate a checksum, or use the compare.c program on SmartSite.

## THREADS PERFORMANCE

You need to determine how to characterize the performance of the algorithm as you spread the work across a number of threads.

1. Login to one of the ECE Linux systems.  Record the processor speed information for later use in your report and recommendation.
2. Run the pthreads version of Newton using various numbers of threads, recording the performance for each run (program runtime in msecs). You should test over a wide range of possibilities to make sure you find the best results.

## USING MPI

You need to determine how to characterize the performance of the algorithm as you spread the work among cluster members using MPI.  Please debug your code on 2 ECE systems at most before using more nodes.

1. Login to one of the ECE Linux systems and record the processor speed information for later use in your report and recommendation.
2. Create the cluster config file to point LAM at other ECE systems for cluster use.  (Available machine names can be found on SmartSite: **Resources / ECE_Hostnames.txt** )
3. Run the MPI version of Newton to use various numbers of cluster members, recording performance for each cluster size (program runtime in msecs).

NOTE: Here is the procedure to access command line arguments with MPI.

1. Write your MPI-capable code to process command line arguments after the call to MPI_Init().

   ```
   MPI_Init (&argc, &argv);
   /* do not access command-line arguments until after MPI_Init()
    *       is called */
   numnodes = atoi(argv[2]);  /* first arg should be number 2 */
   ```

2. Run the MPI version of the code, passing in the argument (5 in this case):

   ```
   mpirun -n 3 newton -- 5
   ```

# SUBMISSION

Turn in with Canvas: **1)** the source code for the pthreads version of Newton, **2)** the MPI version of Newton, and **3)** your report in PDF format (see requirements below).

**REPORT GUIDELINES:**

- Your report should target a technical manager who needs to make design decisions. You can assume that your manager understands parallel architectures (threads, clusters, etc.) but doesn't know the code you're evaluating, or your methodology. Make a design recommendation, then support your recommendation with data and analysis.
- Do a little research to compare prices of cheap dual-core machines and more complex quad-core or 8-core machines such as the ECE systems. Use the information you find to present performance per cost figures that support your recommendation. In your report, you should also briefly mention where you found your cost information.
- Make sure your report discusses the following points on multithreading:
  - How many threads can profitably be put on a threads-capable machine? How does performance per thread scale with more threads? How does overall performance scale with more threads?
- Make sure your report discusses the following points on MPI:
  - How does performance per MPI node scale with more nodes? How does overall performance scale with more nodes; at what point does it not make sense to add more nodes?
- The report, including graphs and tables, should be no more than 3 pages in PDF format with 1 inch margins and no smaller than 10 point type.
- You are encouraged to include graphs to better illustrate how your experimental results support your conclusions. However, don't feel like you have to make graphs of everything. A single graph or a few graphs that emphasize important points are best.
- Make sure to include a section describing how you parallelized the Newton program using both pthreads and MPI.
- Be sure you have thoroughly read the assignment and include all information it asks for.

**DUE DATE: Friday 5/12 at 10AM.**