MP3: Heat Transfer

Submission Instructions. For this assignment, you will parallelize the same code using **both** CUDA **and** OpenACC. This assignment is worth 40 points total. Submission instructions are at the end of the assignment.

WebCode – Online Editor – http://gpu.cse.eng.auburn.edu:8000/

As described in class, we have developed an online code editor, called WebCode, that we would like to use in future offerings of this course. Hopefully, it will be easier and faster than compiling and running on DMC. Usernames and passwords for WebCode were distributed in class on March 30.

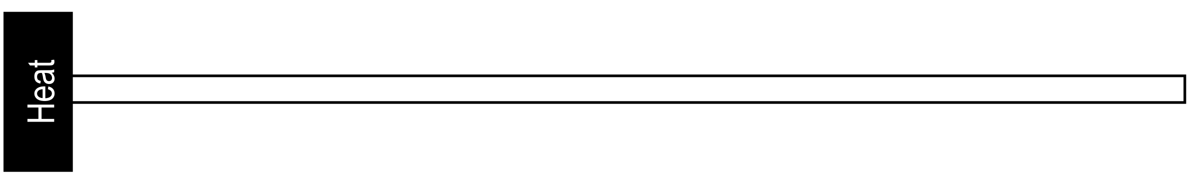
Please use WebCode to write (and run) your code for this assignment. If WebCode does not work correctly, or if you have any problems/complaints/suggestions, please e-mail both William Hester <weh0008@auburn.edu> and Jeff Overbey <joverbey@auburn.edu> so we can address them.

You must be on Auburn's network, or connected to the VPN, to access WebCode. We have only provided logins to students in the course. We're trusting you; **your code is not sandboxed.** It would be very easy to write malicious code that would destroy our GPU server. Please don't.

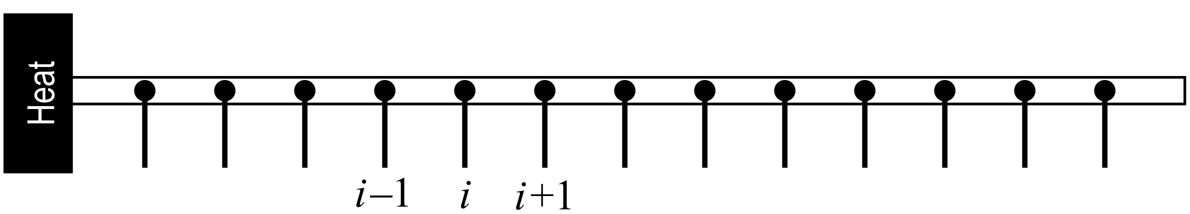
Background

In this assignment, you will be parallelizing a program that simulates heat transfer.[1]

Suppose you have a long, thin rod at temperature 0, and a heat source is applied to the left-hand side. As time progresses, the heat will transfer across the rod.



We can "sample" the temperature at equally spaced points along the rod. Moreover, we can repeat this "sampling" to understand how the temperature changes over time.



[1] For a better description of the mathematics (and physics) behind this simulation, see this assignment from MIT: http://courses.csail.mit.edu/6.884/spring10/labs/lab3.pdf

Initially, the temperature at every point will be 0.

Let's use $t_{old}[i]$ to denote the temperature of the *i*-th point at a particular unit of time. Then the temperature of that point at the next unit of time is given by

$$t_{new}[i] = t_{old}[i] + \alpha(t_{old}[i-1] + t_{old}[i+1] - 2\,t_{old}[i])$$

where α is a constant (the "thermal diffusivity").

Assignment

In WebCode, there is a template called **MP3** that simulates heat transfer (as described above) for 10,000 time steps. The simulation is done entirely on the CPU.

For this assignment, you will parallelize this simulation using both CUDA and OpenACC.

- Log into WebCode. Starting from the **MP3** template, create **two** new projects—one OpenACC, one CUDA.
- If your last name (family name) starts with A–L, please parallelize the simulation using **CUDA first**. Then, parallelize it using **OpenACC afterward**, i.e., after you have completed the CUDA code.

  CUDA First: Alkofahi, Almohaishi, Bordelon, Brooks, Calvert, Caufield, Farmer, Feist, Gong, Hancock, Hester, Hoover, Jennings, Kilgore, Li

- If your last name (family name) starts with M–Y, please parallelize the simulation using **OpenACC first**. Then, parallelize it using **CUDA** after you have completed the OpenACC code.

  OpenACC First: Mukhopadhyay, O'Rourke, Perreault, Pierce, Price, Ravipati, Rowley, Sawyer, Scott, Sprunger, Stewart, Tang, W. Wang, X. Wang, Y. Wang, Wei, Yan

(In the next MP, you'll get to work with the other language first.)

Submission Instructions

After you have finished writing your code, you will need to copy it out of the WebCode editor and paste it into a text editor, so you can create a file to upload to Canvas.

- **Submit three files in Canvas:**
  - Your **CUDA** code (mp3.cu)
  - Your **OpenACC** code (mp3.c)
  - A **text file** (mp3.txt) with *brief* answers to the following questions:
    1. Which version did you write first: CUDA or OpenACC?
    2. Which version was *easier* to write, OpenACC or CUDA? Why?
    3. If you needed to parallelize code like this in the future, would you prefer to use CUDA or OpenACC?