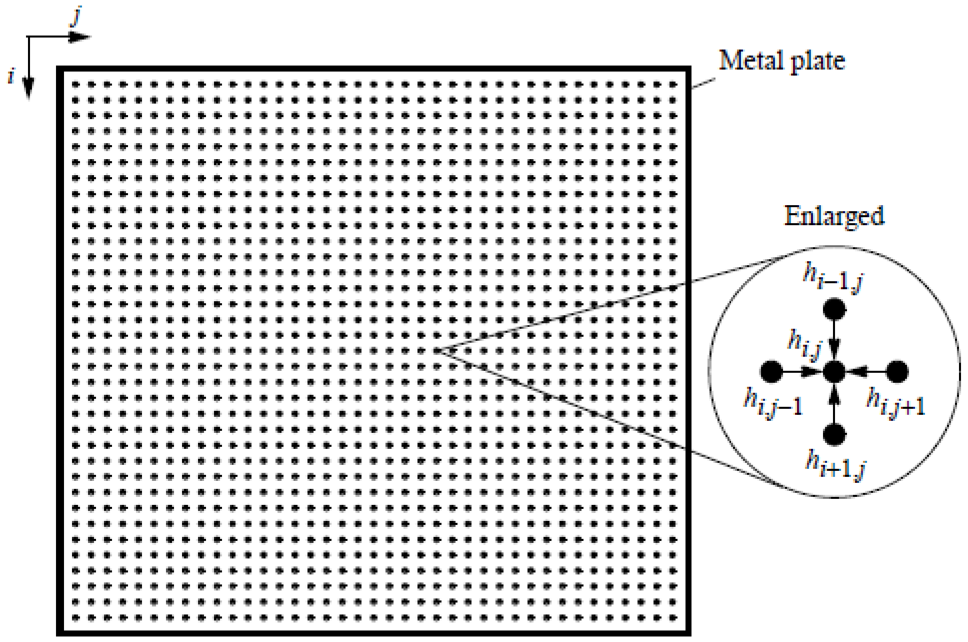
Graphics Processing Units (GPUs): Architecture and Programming
Programming Assignment 2

In this lab, you will write CUDA programs to determine the heat distribution in a space using synchronous iteration on a GPU. We have 2-dimensional square space and simple boundary conditions (edge points have fixed temperatures). The objective is to find the temperature distribution within. The temperature of the interior depends upon the temperatures around it. We find the temperature distribution by dividing the area into a fine mesh of points, *hi,j*. The temperature at an inside point is calculated as the average of the temperatures of the four neighboring points, as shown in the following figure.



The edge points are when i = 0, i = n-1, j = 0, or j = n-1, and have fixed values corresponding to the fixed temperatures of the edges. The temperature at each interior point is calculated as:

$$h_{i,j} = \frac{h_{i-1,j} + h_{i+1,j} + h_{i,j-1} + h_{i,j+1}}{4}$$

(**0 < i < n** ; **0 < j < n**, remember that edge points have fixed values)

Stopping condition: You can have a fixed number of iterations or when the difference between two consecutive iterations (calculated as average among all points) is less than or equal to some number *e*. For this lab, we will use a fixed number of iterations.

Assume we have NxN points (including edge points). The edge points are initialized as follows:

- Points (0,0) to (0,N-1) are initialized to 100F.
- Points (N-1,0) to (N-, N-1) are initialized to 150F.

- Points (1,0) to (N-2, 0) are initialized to 80F.
- Points (1, N-1) to (N-2, N-1) are initialized to 80F.

Assume also that all internal points are initialized to zero.

What you have to do:

1. As a way to help, we have provided you with source code heatdist.cu that contains the code needed to get input from command line, implement the sequential version, measure the time for both sequential and GPU versions. You need to fill the blanks and implement the GPU version. In order to do this you need two things:
   - Understand the source code.
   - Put all CUDA coda (including dynamic allocation, communication between host and device, etc) in function gpu_heat_dist()
2. Execute the programs (seq and GPU) with n = 100; 500; 1,000; and 10,000. Assume 50 iterations. For the GPU version, experiment with different numbers of blocks and threads per block. Then submit. i.e., set it in the code, the ones with the best average performance. You need to use the same configuration for all experiments.
3. At the top of the report you submit, write the best configuration you found.
4. Draw a bar-graph showing n (x-axis) and the time (y-axis) for the 2 versions of the program (2 bars per n on the same graph).
5. What are your conclusions regarding:
   a) When is GPU usage more beneficial (at which n and why)?
   b) When is the speedup (i.e. time of CPU version / time of CUDA version) at its lowest? And why?
   c) When is the speedup at its highest? And why?
6. Repeat #4, #5, and #6 with 100 iterations. That is, include another bar graph and re-answer the questions in #6.
7. What is the effect of increasing the number of iterations? And Why?

This means your report must have two bar-graphs (one for num iterations = 50 and another for number of iterations =100), answers to questions in #6 twice (once for num iterations = 50 and another for number of iterations =100), and an answer to question in #8.

What to submit:

- The source code you used for each program above. Name the source code files as heatdist.cu
- A report that contains the best config, the graphs, and answers to the questions mentioned.

Put all the above in a zip file named by your lastname-firstname.zip and submit as usual through Brightspace.

All the best!