Due Date: Friday, May 6 2016, 11:59pm EST

In this project, you will implement different parallel versions of a one-dimensional redblack relaxation code. You will write your parallel codes in openMP and CUDA, targeting multi-core systems and GPUs, respectively. You will write one openMP version, and two CUDA versions. One CUDA version uses the shared memory (software controlled cache), while the other uses only the device global memory.

As part of the project, you will also work on a report that shows the performance and possible tradeoffs of your codes running on different machines using different numbers of cores, and different problem sizes. You may also use different language specific parameters, such as different scheduling heuristics as provided in openMP. Your report should discuss at least two different experiments, with experimental results illustrated by a graph. The report should be in PDF and no more than three pages long. To gather experimental results, you should run your experiments multiple times and preferably on machines at times when they are not overloaded. This will give you the best and most stable results. Show and explain all your results in the report.

Safety is a key concern in parallelizing the sequential redblack code. You have to make sure that your parallel program versions produce the same results.

Get Started
You can find two template files for your openMP and CUDA versions in the ilab directory ~uli/cs515/projects/proj3/students. This directory also contains a sequential version of the code which you can use to generate the correct output for a given problem size (time steps and array size combination). In addition, there is a Makefile to generate the different versions of your code.

All questions regarding this project should be posted on sakai!

What to submit
Your need to submit your two CUDA program versions and your openMP program version. In addition, you will submit your report as a PDF document. The details of the submission will be posted on sakai.
What machines to use?
All iLab machines currently have the CUDA 5.5 api installed (in "/usr/local/cuda-5.5"). OpenMP should also be available on all machines as part of the gcc compiler.

--1 GPU Machine: "atlas.cs.rutgers.edu"
1x Tesla K40: 12 GB memory, 2880 CUDA cores, CUDA capability 3.5
2x Quadro K4000: 3 GB memory, 768 CUDA cores, CUDA capability 3.0

--21 GPU Machines: List "A" below.
1x GT 630: 2 GB memory, 192 CUDA cores, CUDA capability 2.1
--8 GPU Machines: List "B" below.
1x GT 425M: 1 GB memory, 96 CUDA cores, CUDA capability 2.1

List "A":
adapter.cs.rutgers.edu
builder.cs.rutgers.edu
command.cs.rutgers.edu
composite.cs.rutgers.edu
decorator.cs.rutgers.edu
design.cs.rutgers.edu
facade.cs.rutgers.edu
factory.cs.rutgers.edu
flyweight.cs.rutgers.edu
interpreter.cs.rutgers.edu
mediator.cs.rutgers.edu
null.cs.rutgers.edu
patterns.cs.rutgers.edu
prototype.cs.rutgers.edu
singleton.cs.rutgers.edu
specification.cs.rutgers.edu
state.cs.rutgers.edu
strategy.cs.rutgers.edu
template.cs.rutgers.edu
utility.cs.rutgers.edu
visitor.cs.rutgers.edu

List "B":
cpp.cs.rutgers.edu
pascal.cs.rutgers.edu
java.cs.rutgers.edu
perl.cs.rutgers.edu
lisp.cs.rutgers.edu
basic.cs.rutgers.edu
CS515 Programming Languages and Compilers I
Spring 2016
Project 3: Parallelization

prolog.cs.rutgers.edu
assembly.cs.rutgers.edu