Aum Sri Sai Ram

Programming for Performance:

Focus: The clear focus of the course is to experiment with performance optimizations applicable to numerically intensive scientific codes. However, the techniques learnt in this course will be relevant even for generic codes. We will work on the state-of-the-art architectures such as Intel MIC, Intel SandyBridge, AMD Opetron and NVIDIA Kepler.

(18.01)
1. Motivation and Organization of the course
   MMM multiple versions [Lecture 1]

(19.01)
[LS1] Tools for writing correct code [Demonstration 1]
Valgrind and gdb. What is a memory leak ? Types of errors detected by Valgrind.
Parallel debuggers … are there any good ones like Totalview on TACC systems).

[Demonstration 2] (21.01 ?)
[Different XSEDE systems and their usage, modules etc.] Submitting batch jobs.
[Have to pick up during the class: how to use gnuplot, how to write python code, latex etc.]
[HW] Find out the peak performance theoretically of each core of a socket on a node, each socket of a node, and each node of all the XSEDE systems that we have access to.

Compute Core Optimizations
(22.01 - 27.01) In-core optimizations (ILP - pipelining, superscalar etc. branch predictions etc.)
[Lecture 2A] What Compilers can and cannot do.
[Lecture 2B] Assembly level optimizations.
[Demonstration 3] Profiling tools: perfexpert, perf. Tips to write branchless code.
[LS2] Which sort is better Quick or Merge ? Which can be better optimized for performance ?
[PA1] Multiple code snippets for core level optimizations.

Benchmarking
(28.01 - 29.01) Benchmarking (Issues with accurately timing a code) [Lecture 3]
Nano level benchmarking : Discuss X-ray paper. (only Lecture)
[LS3] Timer granularity. compare c-time, gettimeofday and rdtsc etc.

Vectorization
(30.01 - 4.02) SIMD vectorization [Lecture 4]
Programming Assignment: Run this assignment on Gordon and Stampede MIC
[LS4] SIMD programming (scalar dot product)
[PA 2] How to write code so as to make compiler generate effective SIMD code.

Memory Locality Optimizations
(5.02 - 10.02) Locality (Memory specific optimizations) [Lecture 5]
Prefetching, caching, cache blocking, register blocking etc.
Loop specific optimizations.
Discuss membench (and memory mountain) [Only lecture]
[LS5a] 6 versions of MMM using perfexpert, perf and papi.
[LS5b] Optimize Matrix Transpose

(**Application Performance Bounds**)
(11.02 - 17.02) [Lecture 6]
When do you say the tuning of an app reached an upper bound ? How to find the upper bound on the performance of an application once its characteristics are known. (Describe the roofline model).

[PA 3] Data Access Optimizations: Based on STREAM benchmark, Have to read the 3rd chapter from Hager's book.

(Code for Speed) Programming Contest (3 hrs) 18.02

Autotuning
(19.02) [Lecture 7] ATLAS for dense (brief lecture) ATLAS vs OSKI; Sparse Matrix Multiplication (Sparse Kernels, Sparsity. (only Lecture)

MIC Specific Optimizations
(20.02 - 28.02) [Lecture 8] MIC specific programming

Optimizing MMM: A Kernel where all the optimization discussed till now can be applied
[PA 4] Optimize MMM on Stampede using a core on MIC as well for Sandy Bridge core.

(Till here the concentration will only be on optimizing for a single core).

Multi-core specific optimizations
(1.03 - 3.03) [LS5] memory affinity, false sharing
[PA 4B] Multi-core MMM on MIC and Sandy Bridge

GPU specific optimizations
(4.03 - 10.03) [Lecture 9] GPU specific optimizations.
[LS6] CUDA programming basics.
[Meetup] GPU tools (each student group will present a tool related to GPUs and also they briefly discuss one of the HiPC student papers. (http://hipc.org/hipc2012/studentsymposium.php)

[PA 5] Optimizing Reduction on Stampede

How to fool the masses (How not to fool yourself) while quoting performance numbers
(10.03 - 11.03) [Demonstration/Discussion 3]
Papers on why GPUs are not as rosy as they projected them to be. (Recent microsoft paper) Debunking.

Mini-Project:
(12.03 till the last working day of the semester)
Choose a scientific computing kernel or application and improve performance on a machine.
On the last working day, presentations of the project will be scheduled.

Project (Optional):
(12.03 - …)
Choose a topic on which you can apply the optimizations used in this class. Could work on this through the summer of 2013 and aim for a decent paper once you succeeded in getting good results.

Instructions:
All LS (Lab Sessions) should ideally be done individually. It is strictly not allowed to show or explain your code to others. However, you are welcome to discuss with your peers while working on PAs. May be from PA 3, groups of 2 can be formed. Please do abide by the instructions. No issue at all if you do not submit anything, however, kindly do not submit the work done by someone else as your work.

The evaluation will not be liberal and it will be done the way it has to be done. This will be done at whatever may be the cost it takes, the idea is to have an intense and enthusiastic participation from everyone involved. The schedule is too tight and cannot afford lose even a single minute.

If one gets through this course successfully, I am sure he will be very confident and can face any technical interview. This course involves the fundamentals of Computer Science and will be very handy for whichever subfield you may work on later.

Marks:
LSs : 20%
PAs: 40%
Class participation: 10% (includes piazza)
Contest: 10%
Project: 20%