WELCOME
About the workshop

- **Intended audience**
  - Researchers in Comp. Sci. Eng. Medicine (CSEM) at Duke
    - Faculty, postdocs, senior grads with experience
  - With the need to shorten computation from a week to a day (from a day to an hour)

- **Lessons learned**
  - In parallel programming (old and new, options)
  - How to make a successful transition

- **Lectures**
  - From the perspective of a new bicycle rider or vehicle driver
  - Common problems, explanations in terms of principle concepts & solution tips
  - Examples, templates, tools
Objectives

- Help immediate start ( $0.0 cost, save some time and trouble)
- Advance CSEM research at Duke
- Re-enforce collaboration among CSEM community @Duke
- S.t. the constraints in resource, time, experience, attention span (coffee available)

This workshop is inspired by

- NSF-CISE Workshop on Broad Impact on Research & Discovery (BIRD), 06/28/2010
- America Competes Reauthorization Act (ACRC) of 2010, 05/28/2010
- CSEM researchers at Duke in email survey in 09/2010
• **Introduction** (20 min.s, Xiaobai)
  - Why multicore programming, among others
  - Who make the transition: collaboration vs subcontract

• **Basic multicore programming** (25+5, 25+5 min.s, Nikos)
  - Threading a sequential program (Pthread, OpenMP, examples)
  - Critical sections in thread coordination (arch. and alg. means)

• **Threading subroutines** (25+5 min.s, Bo)
  - Across subroutines & within a subroutine
  - Tips for evolutional parallelization in application SW

• **Intermediate multicore programming** (25+5 min.s, Nikos)
  - Data-dependent parallelization
  - Performance tuning and evaluation

• **Summary & Feedback**
Introduction

- Why multicore programming
  - A brief glance at a short history and current landscape of parallel computing
  - The central position of multicore programming

- How to make a successful transition
  - Options
  - Measures of success
  - Approaches we took
Parallel architectures (system HW and SW)

- Memory-processor (MP) configuration
  - Distributed memory and processors
  - Shared memory and multiple processors
  - Distributed-shared memory

- Within a CPU
  - Vector registers
  - Multicores (<64), many cores (≥ 64, such as GPUs)
  - APU = CPU + GPU (to come)
  - From desktop, laptop, to hand-held devices
  - Darwin death to single-core CPUs

- Memory
  - Larger on our laps than a cluster of computers with distributed memories a decade ago
Programming/Threading on new CPUs

- Parallel programming models & environments
  - Distributed-memory programming
    - MPI, most notable among others
  - Shared-memory programming (multi-threading)
    - CUDA, CTM on GPUs
    - Pthread, OpenMP (among others) on multicores
  - Distributed-memory programming + shared-memory programming

- Threading mechanism and methodologies (AAAAs)
  - Automation : dream
  - Annotation with OpenMP : easy to start
  - Argument passing to Pthread lib. : flexible in parallelization fashion
  - Algorithm renovation : perhaps the most important to application SW
• Reshape certain core concepts in programming
  
  – Subroutines: each cloned in multiple threads, but need coordination
  – Speed up $\geq$ #cores, $\#$threads/$\#$cores $\geq$ 1
  – Task parallelism, data parallelism & load balance (?)
    
    • Thread a ‘task’ at multiple spatial scales, time scales, in different fashions
    • Data and work in the same space, within same duration
    • Thread life span $\leq$ Execution duration

• Not just programming: Re-think, restore dependency & concurrency
Make a successful transition

- **Contract**
  - Outsource to a CS expert or a programmer
  - Assign to a postdoc or a senior grad

- **Collaboration**
  - Find capable postdocs and/or senior grads
    - with domain knowledge and programming experience
    - To take the *pilot* position (cf. Brooks’ book)
  - Find a CS expert (*pilot instructor*), if necessary
  - All collaborators (*pilot, co-pilot, instructor, controller*):
    - Identify dominant factors (which may change over the development course)
    - Specify short-term targets in research & development
      - Toward long-term goals
    - Evaluate progress accordingly
    - Locate problems and find solutions
    - Distill and disseminate new findings, understandings, or tools

- **Measures of success**
  Impact on advance in research and researcher career
• Speaking from experience

  – I have taken all but instructor positions at different times

  – We have completed the first version of *parallel FMM (>60,000 lines)* at expected speedup on desktop/laptop
    • with Bo as the pilot, Nikos as the instructor
    • Started with a sequential code by, and in collaboration with, Prof. Huang @amath.unc

  – We have presented some *research finding* at certain conferences and workshops, with the support of the development techniques and strategies presented in part at this workshop