Introducing parallel programming across the undergraduate curriculum through an interdisciplinary course on computational modeling

Narsingh Deo, Faraz Hussain, Sumit Kumar Jha, *Mahadevan Vasudevan

Electrical Engineering and Computer Science Department, University of Central Florida
Orlando FL 32816 USA

Abstract

We argue a case for including an interdisciplinary course of computational modeling with a focus on parallel programming across the undergraduate curriculum. The construction of computational models has become a fundamental process in the discovery process for all scientific disciplines, and there is little instructional support to enable the next generation of scientists and engineers to effectively employ massively parallel high-performance computing machines in their scientific process. We believe that a first course in computer programming must be followed by a second interdisciplinary course in computational modeling with a focus on parallel programming for students across the undergraduate curriculum.

Key-words

Parallel programming; Parallel algorithms; Numerical methods; Discrete algorithms; Computational modeling

Introduction

- An era of massively parallel computing where every programmer must be trained in the art and science of parallel programming
- Ubiquitous demand for parallel programming will require the training of undergraduate students
-
- Our focus is on introducing parallel computational thinking based problem-solving techniques for real-world problems across the entire spectrum of the undergraduate curriculum.

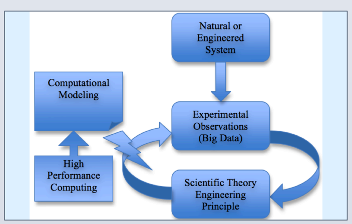

Figure 1: The role of computational modeling and parallel algorithms in sciences and engineering.

Building Big Systems for Biting into Big Data

- The nature of available data for building models has changed in two fundamental ways
  - Petabytes of computer aided data
  - Structured data from social media
- Transforming such data into knowledge is vital and knowing the techniques to model such data in parallel is invaluable

I LEARNING COMPUTATIONAL MODELS

A. Learning statistical models from big data
B. Learning dynamical graph models from structured data
C. Learning parameters of computational models
D. Learning communities in complex evolving networks

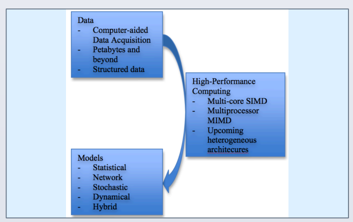

Figure 2: Transforming data into knowledge using HPC

II MODEL SIMULATION

a. Parallel simulation of ODE models
   - Ordinary Differential Equations
b. Parallel simulation of CTMCs
   - Continuous Time Markov Chains
c. Parallel simulation of ABMs
   - Agent-Based Models
d. Parallel simulation of SDEs
   - Stochastic Differential Equations

III ANALYSIS AND VALIDATION OF MODELS

a. Statistical model checking
   – use of statistical hypothesis testing in the determination of the correctness of a model with respect to a stated formal specification
   – students will be exposed to both trivial and non-trivial kinds of parallelization
b. Symbolic model checking
   – symbolic data structures like Binary Decision Diagrams, algorithms like the DPLL based satisfiability techniques, and theorem-proving methods
   – study parallel variants of popular symbolic data structures

Module	Pedagogical value in parallel programming	Additional teaching goals
Decision Trees	Distributed-memory Master-slave Algorithm	Building a recommendation system from big data via classical machine learning.
Dynamical graph models	Partitioning of structured data in distributed memory applications.	Building models of structured data. Theory of complex network models.
Parameter Synthesis	Trivial parallelism, parallelizing sequential code, thinking in a parallel manner.	Use of evolutionary algorithms, simulated annealing, and optimization methods
Learning communities	Provoking parallel solutions to Graph theoretic algorithms.	Modeling intensive biological data and observing correlation with real-world data
Parallel Simulation	SIMD and use of static analysis methods for discovering parallelism.	Simulation of ODEs, SDEs, ABMs and CTMCs.
Model Validation	Parallel complexity, discrete and numerical parallel algorithms, cache-coherency.	Model checking, symbolic data structures, statistical hypothesis testing.

Figure 3: Proposed modules and their pedagogical goals.

INCORPORATING INDUSTRY STANDARD PROGRAMMING

- Key challenges faced by students transitioning from academia to the industry lies in adapting to newer technologies used by modern day web-based, online-centric companies.
- The advent of cloud-based technologies, web-based solutions, mobile platforms and the demand for social networking applications has increased the usage of distributed computing models and frameworks for processing large data sets.
- Ubiquitous nature of big data across medical, social and marketing domains continuously creates the opportunity for the usage of distributed processing framework such as MapReduce.

CONCLUSION

We have proposed a new interdisciplinary course for teaching parallel computational thinking to senior undergraduate students across the undergraduate curriculum. Our proposed course will provide adequate tools to our future scientists and engineers so that they can leverage advances in high-performance computing to excel in the new and upcoming area of data-driven sciences (See Figure 1). We have also summarized the key modules involved in the design of such a course along with their pedagogical goals in Figure 3. It is our belief that such a broad course that focused on computational modeling will enable a large number of undergraduate students to learn the art and science of high-performance computing in a setting that they are likely to revisit in their professional life as scientists and engineers.

ACKNOWLEDGEMENTS

SC12 Educators Program for their participation in the LittleFe program

IEEE Early Adopter