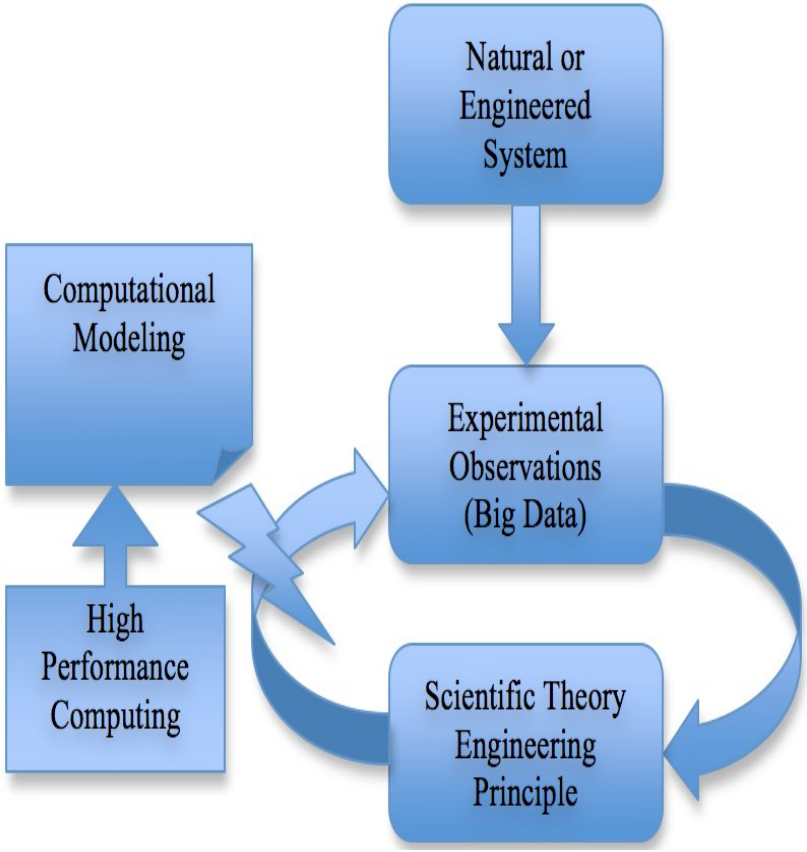
# High-Performance Computing for Big Data: Introducing parallel programming and big data in the core algorithms curriculum

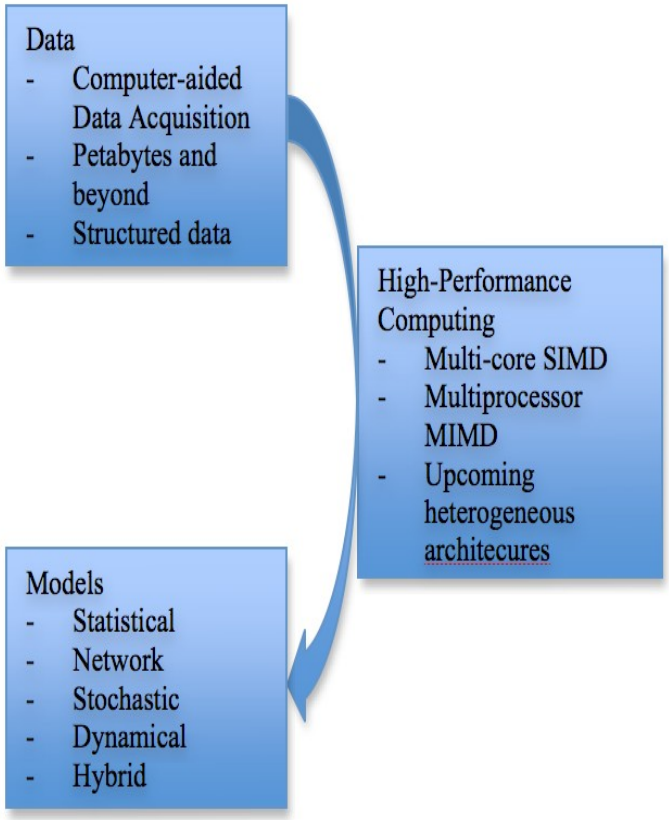

Faraz Hussain, Narsingh Deo, Sumit K. Jha
Electrical Engineering and Computer Science Department
University of Central Florida
Orlando FL 32816 USA

# The role of computational modeling and parallel algorithms in sciences and engineering




- Introducing parallel computational thinking based problem-solving techniques for real-world problems.
- Urgent need for scientists that are well trained in the art and science of parallel computer programming
- Ubiquitous demand for parallel programming

# Transforming data into knowledge using HPC




## 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. Leaning communities in complex evolving networks

## 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

- Statistical model checking
- Symbolic model checking

# Proposed modules and their pedagogical goals

| 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. |

# Conclusion

- Adequate tools to our future scientists and engineers so that they can leverage advances in high-performance computing (MPI, CUDA, GPGPU, hands-on experience with large datasets)
- Incorporating industry standard programming (example MapReduce algorithms)
- 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
  - Parallel complexity
  - Specification, simulation and verification of parallel programs