Heterogeneous Computing for Undergraduates: A Module-Driven Approach

Apan Qasem
Texas State University
apan@txstate.edu

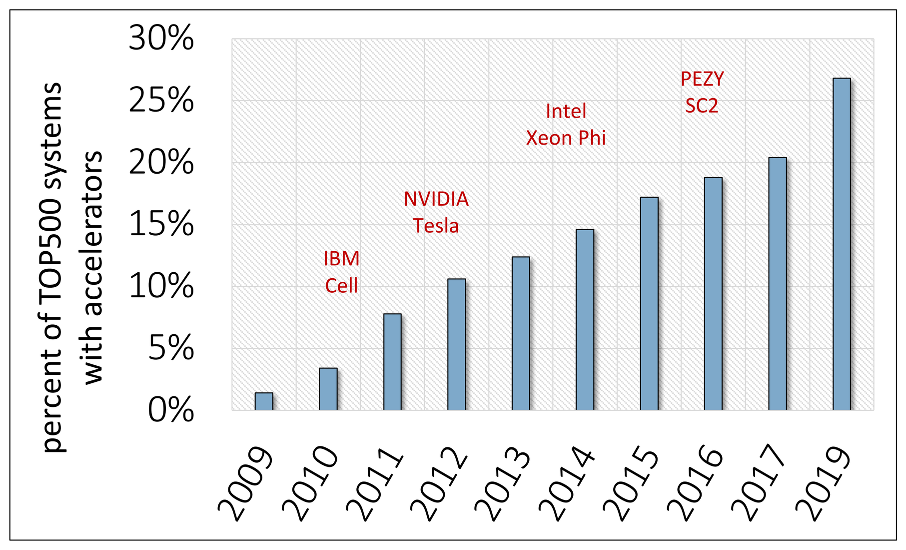
David P. Bunde
Knox College
dbunde@knox.edu

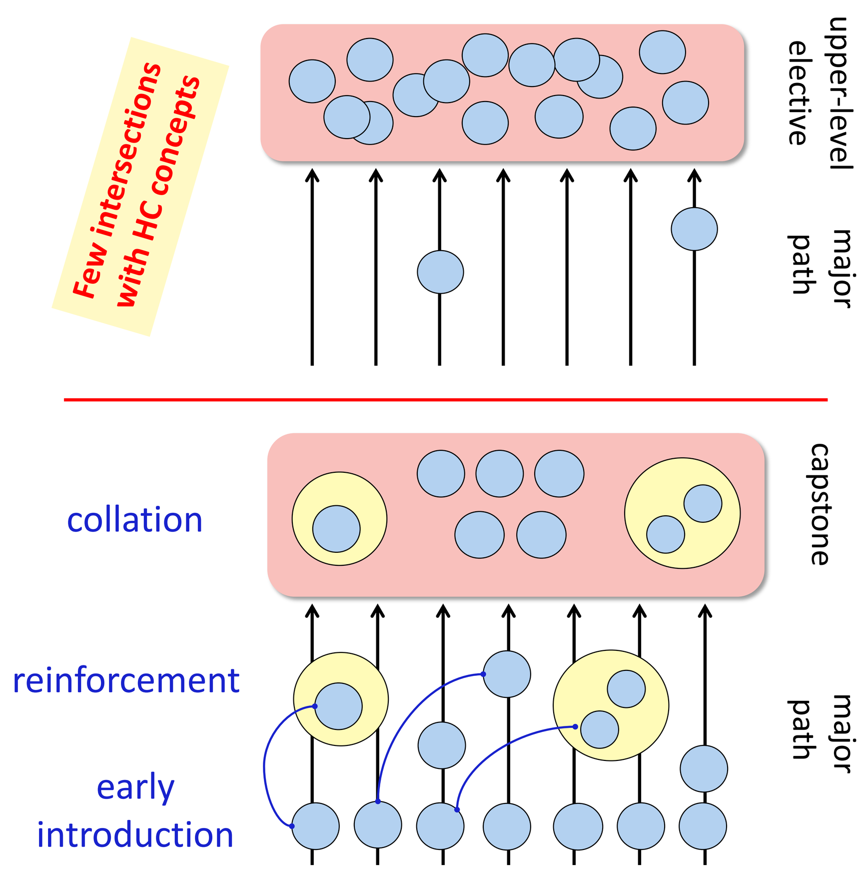
Philip Schielke
Concordia University Texas
Philip.Schielke@concordia.edu

Pervasive Heterogeneity: The need for increased performance per watt, coupled with the demands of processing diverse workloads, has led to a shift towards heterogeneous computing systems. Integration of high-performance CPUs with energy-efficient GPUs is now common in all classes of HPC systems. Architectural heterogeneity has also permeated mobile processing, cloud computing, the Internet of Things (IoTs), and machine learning. It is imperative that computer science graduates learn the skills to program these complex systems of the future.

Lack of coverage in current curricula: Current undergraduate computer science (CS) curricula lack sufficient coverage of heterogeneous computing (HC) concepts. Heterogeneity is covered only as an upper-level elective and is primarily limited to R1 institutions. Adding a required course in HC is challenging because the CS/CE curriculum is already crowded. Furthermore, HC spans different areas such as programming, algorithms, and architecture so it is better covered in multiple courses rather than a single course.

Our approach: This initiative attempts to integrate HC content into the curriculum using the early-and-often approach. We have developed modules that can be injected into several courses across the curriculum in a way that creates significant coverage of HC concepts along every path through the major. The enhanced curriculum is complemented with synergistic activities that include industry partnership, faculty training, and out-of-class training camps.





Design Principles

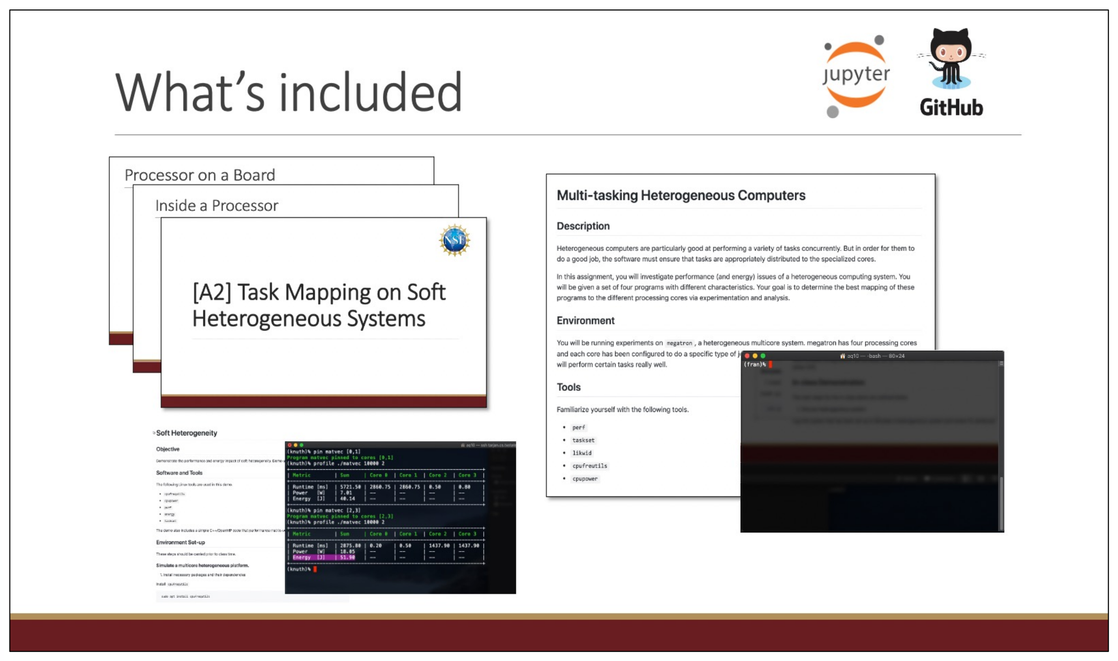
Abstraction: Topics selected based on level of abstraction at which they can be introduced.

Topics that can be explained at a higher level of abstraction are introduced early in the curriculum. Example: Amdahl's Law in CS1

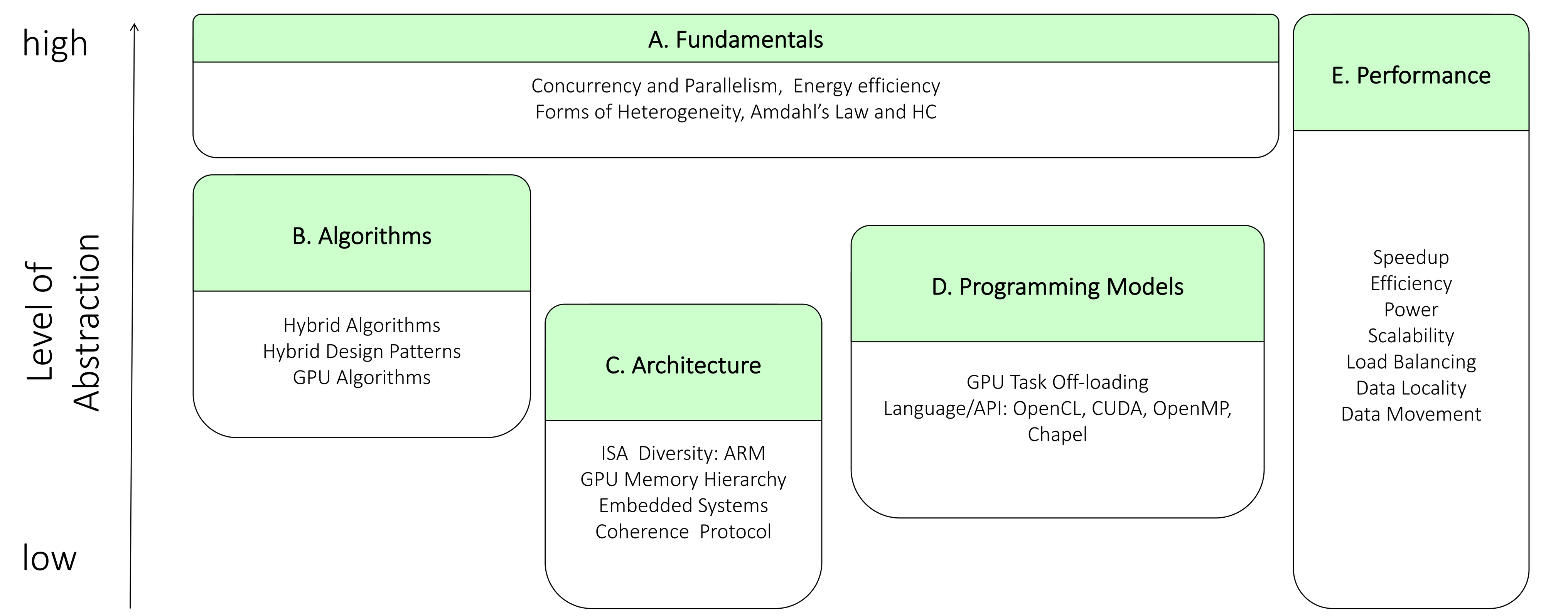
Context: Modules provide heterogeneous context to existing content, reducing the extra time needed

Active Learning: Modules emphasize active learning and include self-paced hands-on tutorials, demos, and assignments

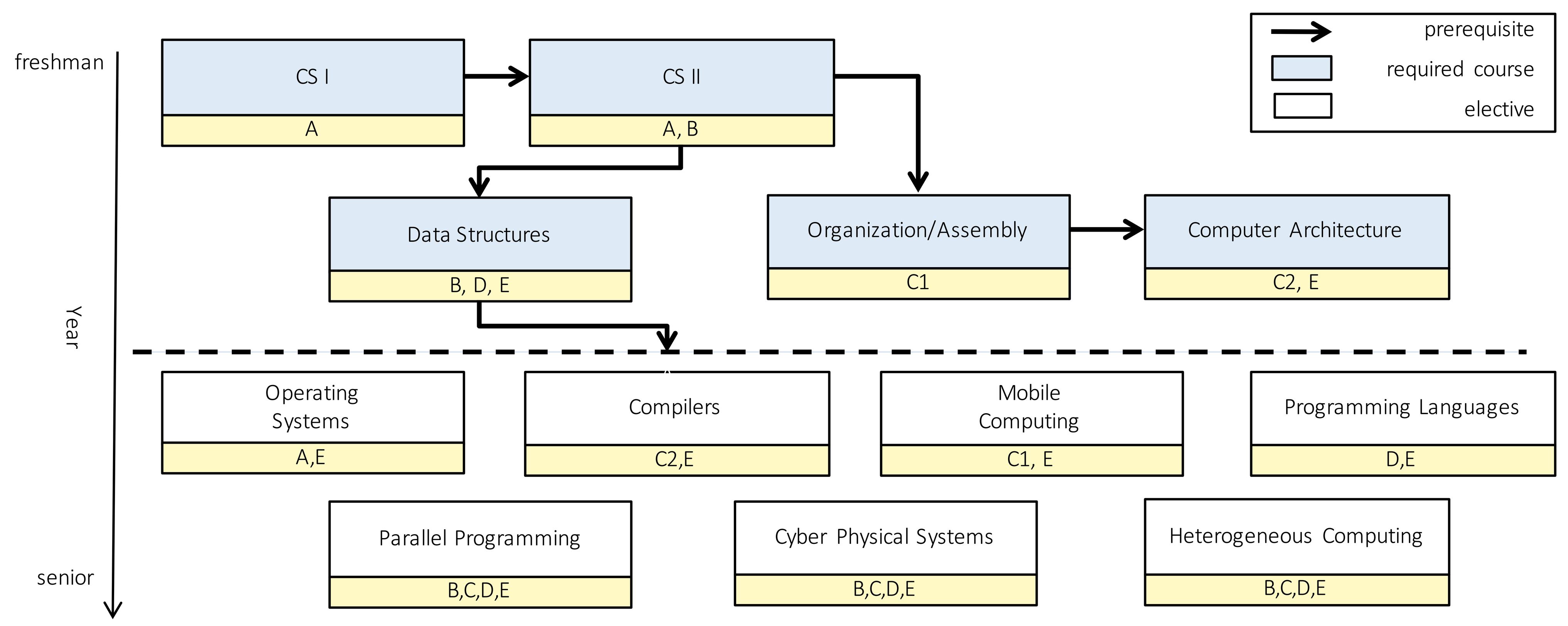
Adoption: Modules are short and self-contained for easy adoption across institutions. They are also free



Modules



Curricular Integration



Pollack's Rule

Topics: Pollack's rule (estimating perf. from core area), Amdahl's law, dynamic parallelism in multithreaded progs

Suitable for: OS/Systems, Arch/Org, maybe CS 2 if multithreading is used

Use: 50-minute lecture, plus HW (possible problems provided)

ARM ISA

Topics: Assembly programming w/ ARM and Thumb instruction sets on ARM processors, ISA diversity

Suitable for: Aimed at Arch/Org, but suitable for any course using assembly.

Use: 1-2 lectures, plus written and prog. HW (problems provided). Uses Raspberry Pi. Assumes prior assembly prog.

Synergistic Activities

Industry Partnerships: Advisory board with experts from Intel, AMD, NVIDIA, and Qualcomm provide suggestions on module design, review created modules, and recommend enhancements

Faculty training Half-day tutorials on specific modules at SIGCSE, CCSC

Multi-day workshops at host institutions (virtual due to COVID)

- Session 1: Overview of modules and pedagogical strategy
- Session 2: One-on-one deep dive into modules
- Session 3: Participant presentations outlining implementation plan
- Session 4: De-brief sessions, post-implementation

Boot camps: Short immersive camps for students

- Reinforce HC knowledge
- Industry speakers
- Networking opportunities

Acknowledgement: This project is supported by the National Science Foundation under award number OAC-1829644 and OAC-1829554

The project has been funded under a broader initiative of enhancing the Advanced Cyberinfrastructure workforce.