ECE 574 – Cluster Computing
Lecture 1

Vince Weaver

https://web.eece.maine.edu/~vweaver

vincent.weaver@maine.edu

11:00am, Barrows 133

31 August 2026

Welcome to ECE574

- Distribute and go over syllabus
- Course website, all assignments and class notes will be posted here:

  `https://web.eece.maine.edu/~vweaver/classes/ece574/ece574_2026f.pdf`

ECE574 Syllabus – Instructor Info

- I'm Professor Weaver
- Office is 203 Barrows
- Office hours 1:30pm-2:30pm Tuesday/Thursday
- Feel free to e-mail for appointment too, or just stop by if my door is open

ECE574 Syllabus – Textbook

- None

ECE574 Syllabus – Computer Accounts

- Will be handing out computer accounts, please use them responsibly.

ECE574 Syllabus – Programming

- Will involve a lot of coding, mostly C or C-like languages.

ECE574 Grading – Homeworks

- Homeworks, 50% – roughly 10, lowest dropped.
- Generally will be due on Thursday (or Friday?)
- Will usually have at least a week to do them.
- Submission by e-mail, grades sent in response to that e-mail
- Will send out e-mail when assignment posted on website.
- Note: we will not be using Brightspace

ECE574 Grading – Exams

- Two Midterms, totaling 25%, after Fall break and near end
- No final exam

ECE574 Grading – Final Project

- Project, 20%
- HPC related project, open-ended
- Can program in any language
- Can work in groups
- Presentation during last week of classes
- Final writeup
- More details as we get closer.

ECE574 Grading – Participation

- Class participation: 5%

ECE574 Grading – Late Work

- Work will lose points for being late (up to 10% per day)
- Might be best to submit what you have at deadline, and if you make further progress can submit as late work
- Note that a large part of your grade are homeworks so try not to miss too many

ECE574 Grading – Class Notes

- Class notes will be posted on the website
- Generally within a few days of the lecture
- Note: lecture notes are now Title II compliant!

ECE574 — Pandemic Policy

- We will follow whatever the current UMaine recommendation is
- If really sick, please don't come to class
- if slightly sick, wear a mask if possible

ECE574 — Coding Help

- If you have questions often the most efficient way is to send me your code via e-mail
- Sending actual code is best and will get better results than sending screenshots

ECE574 — Academic Honesty

- I hate to dwell on this but...
- **For coding assignments please only submit code you wrote yourself**
- Do not turn in as yours other people's code, either from this class, copied off the internet, or via AI
- Do not share your code with others in the class, even after the submission deadline. If you share your code and the person you shared with submits it too, **you both will face consequences**

ECE574 — Academic Honesty Consequences

- The minimal consequence for this is a zero on the assignment.
- Note: a zero for cheating will not be dropped as part of the "lowest homework grade is dropped"

ECE574 — Academic Honesty Part 2

- Note that this doesn't mean you can't get help. The following are allowed
  - You can always ask me (the professor) for help
  - You can always discuss and ask for high-level help from others about the assignment (just don't copy code)
  - You can even ask someone to look at your code to spot errors you might be missing (ideally do this without sending it to them)

ECE574 — AI Policy

- I don't like AI and I ask you not to use it for this class
- Especially do not use it for the coding assignments

ECE574 — Syllabus Boilerplate

- Required info from UMaine

Cluster Computing

What is cluster computing?

Basically it's just a large number of computers working together.

So why is that interesting?

Road to Parallelism – Serial

- Originally computers were serial, did one thing at a time (though could hide this via multi-tasking)
- Moore's Law meant performance would increase
- At some point limit hit for serial execution, so much effort went into trying to find parallelism in code automatically in hardware (pipelining, out-of-order execution, superscalar)
- We spend a whole semester on this in ECE571

Road to Parallelism – SMP

- Moore's Law continued, but serial speedup trailed off (Memory Wall a big issue)
- Instead transistors used for multiple-cores in system.
- Suddenly parallelism was everyone's problem
- Even in embedded systems and phones

Multicore Systems

- Multicore systems need special programming to run parallel code
- It's possible even then you'll want more processing power than a single multi-core system can provide
- In that case you'll need to spread the processing across multiple machines: a cluster

Supercomputers/HPC

InsideHPC defines HPC: "High Performance Computing most generally refers to the practice of aggregating computing power in a way that delivers much higher performance than one could get out of a typical desktop computer or workstation in order to solve large problems in science, engineering, or business."

And a supercomputer is similarly vague, just a larger computer than the kind you'd normally use.

Related terms

- **Supercomputer** – a large computer. No real definition, "I know one if I see one".
- **High Performance Computing** – using large computers
- **Cluster** – one way to make a large computer out of many small computers
- **Distributed systems** – a system that is made out of many subnodes that communicate by passing messages to work on large problems. Sort of the old CS term for cluster computing.

More Related terms

- **Grid computing** – idea to make computing like the power grid, a utility you access to run your jobs. A large loosely coupled cluster.
- **Cloud computing** – more or less a newer and more buzzword-friendly term for grid computing. Often uses virtualization, not often used for high performance mostly because usually the network is not optimized in a cluster fashion.

Even More Related terms

- **Datacenter** – a large building with lots of network connected computers. Not a cluster unless they are working together. This gets complicated when things like google are involved.
- **Mainframe** – a type of large computer. Usually differentiated from a supercomputer because a mainframe concentrates on reliability and I/O performance rather than CPU performance.

What about AI?

- Systems used for AI overlap a lot with those for HPC
- Traditional HPC uses a lot of 64-bit floating point math
- AI instead operates on much smaller numbers, often FP16 or smaller
- While we won't cover much on AI in this class, a lot of what we will cover is relevant to low-level AI implementations

What about Quantum Computing?

- Quantum has been in development for years and it's unclear if it will ever be fully practical
- There have been some breakthroughs recently
- In theory some problems currently done with HPC might be done faster with quantum computing
- I don't think all HPC workloads can necessarily be replace by quantum
- I did attend some talks at SC'24 on this topic
- ECE class on it?

Cluster Programming

- The easiest way is to just have single-thread programs but run multiple of them
- If instead you want to run one program but have it run faster across multiple cores you'll need to re-write it to be parallel
- You will need to do this to see a speed benefit on multi-core systems (cluster or otherwise)

Parallel Programming

- It's hard
- It's really hard
- It's possibly orders of magnitude harder

Parallel Programming

- Most systems, even embedded systems and cellphones are parallel these days
- Languages don't support it well
- Human brain has trouble grasping it
- It should probably be taught earlier than a 500 level grad course

Can it be someone else's problem?

- Can code be automatically made parallel?
- Why not have compiler auto-parallelize code?
- Oh they try, how they try.
- Compilers and CPU hardware designers (OoO) wring out as much as they can
- Maybe AI will save the day?
- For now it has to all be done manually.

Parallel Programming Languages

- C?
- Fortran?
- C++?
- What about things like Javascript, Python? Python widely used but actual parallel stuff in C libraries (global lock means until recently Python couldn't scale)
- What about things like Go or Rust or Zig?

HPC Workloads

- Linear Algebra
- Modeling (weather, chemistry, biology, nuclear, aerospace)
- Business (high-speed trading)
- Simulation
- 3d rendering (movie studios, games)
- Data crunching (geology exploring)
- AI? Bitcoin Mining?

A note on Models

- "It's tough to make predictions, especially about the future." — Yogi Berra
- "All models are wrong, but some are useful" — George E. P. Box
- Wasted lots of time on computer architecture simulations
- GIGO