Slide 1

Administrivia

- One purpose of the syllabus is to spell out policies (review today).
- Most information will be on the Web, on either my home page (office hours) or the course Web site.
- A request: If you spot something wrong with course material on the Web, please let me know!

Slide 2

A Few Words About Teaching and Learning This Semester

- The good news: We're back in the classroom! (We can see at least part of each others' faces.)
- The bad news: We're back in the classroom! (News about the pandemic is getting scary again.)
- I feel more at risk than the average person, but Trinity really wants in-person teaching. So: Classes in person, virtual office hours.

Slide 3

Course Web Site

- "Course Web site" is meant to point you to pretty much all information for the course — readings, assignments, etc.
- You can find it via TLearn, or via the link from my home page (should be findable from the page about me in TU's Web site, or by doing a Web search on my name).
- A request: If you spot something wrong with course material on the Web, please let me know!

Slide 4

Syllabus

- (Review syllabus.)
- (Review course Web site.)

Slide 5

About the Textbook

- Why are we using "my" book when there are books that are more textbook-like, and also more recent? because I think it comes closer than any other book I know to covering the material I think is perhaps best learned from a book.
- (And no, I'm not getting rich off the royalties! though extra income is not unwelcome.)
- I somehow neglected to turn in a bookstore order until the last minute, so who knows when the bookstore will get copies. I've put one copy of the book on 1-day reserve at the library.

Slide 6

What is Parallel/Distributed Computing?

- Some computational jobs are just too much for one processor — no way to get them done in reasonable time.
- For jobs done by people, what do you do when the job is too much for one person?

Slide 7

What is Parallel/Distributed Computing?

- For jobs done by people, if too much for one person you assign a team — but you have to figure out
  - How to divide up work among team members.
  - How to coordinate activities of team members.
- Same idea applies to computing — if too much for one processor, use multiple processors. Issues are similar — how to divide up work, how to coordinate.

Slide 8

Simple Examples

- "People job" examples:
  - Digging a hole.
  - Building a house.
  - Baby.

  What do you notice about the last one in particular?
- Computer examples:
  - Adding up a lot of numbers.
  - Computing Fibonacci numbers.

  But these don't seem too "big" . . .

Slide 9

How Much Calculating is "A Lot"?

- Dated examples from computational biology — how many operations per second are needed to get things done fast enough to be useful?:
  - Sequence the genome — 10^{12} ops/second (500 2-Gigahertz processors).
  - Protein/protein interactions — 10^{14} ops/second (25,000 4-Gigahertz processors).
  - Simulating whole-body response to a drug — 10^{16} operations/second (1,250,000 8-Gigahertz processors).
- (Source — Intel's former life sciences industry manager.)

Slide 10

What Are Some Other Hard Problems? Traditional HPC

- Crash simulation / structural analysis.
- Oil exploration.
- Explosion simulations (why Los Alamos is interested).
- Astrophysics simulations (e.g,, Dr. Lewis's work on Saturn's rings).
- Fluid dynamics.
- "Rendering" for computer-generated animation.
- And many others . . .

Slide 11

What Are Some Other Hard Problems? Recent

- "Big data"!
- AI, machine learning, what Dr. Hibbs and Dr. Lewis do . . .

Slide 12

About the Course

- Can think of this course as the equivalent of CS1 for parallel (and to some extent concurrent and distributed) programming. As with CS1, many things to learn all at once:
  - A new "box of tools" — or several boxes of tools (different languages/libraries/paradigms). Must learn syntax/functions, plus tools such as compilers and runtime systems.
  - How to use the stuff in the box of tools to solve interesting problems — from low-level "what is this syntax good for?" to algorithm design.
  - How to think about "does it work?"
  - How to think about "how fast is it?"
- Also as with CS1, the idea will be to teach a mix of technical skills and basic concepts, with emphasis on learning by doing.

Slide 13

A Little About Me

- Short version of biography: Undergrad degrees from UT Austin, math and Plan II. More than ten years in what we now call IT. Back to school for master's and PhD in computer science. Two years as a postdoc, then at Trinity since Fall 1999.
- I teach a variety of courses, but currently focusing more on courses "close to the machine". My research area (sadly neglected for some years) is parallel computing.
- (What do I do for fun? well . . .)

Slide 14

About Minute Essays

- Most lectures will end with a "minute essay" — as a quick check on your understanding, a way for me to get some information, etc., and also to track attendance.
- Send me your answer(s) by e-mail (no word-processor attachments please). And *please* put "minute essay" and the course in the Subject line. This makes it much easier for me to pick them out of my inbox and save them for my attendance-tracking scripts.

Slide 15

Minute Essay

- What are your goals for this course?
- Are you reasonably comfortable with C? How about Java? In the past I've had students do an assignment in Java, but this year maybe it should be Scala?
- I'm told that our CS2 course now includes some exposure to multithreaded programming? What do you remember from that, and do you have other experience with any kind of parallel or distributed programming?
- Anything else you want to tell me? about the course, about what you did over the summer . . .