Why Parallel Programming?

- Faster (sometimes speed is important)
- Methods for Parallel Programs can use less memory per machine (because problem is subdivided)
- Sometimes problem data is scattered, so it makes sense to move the "solver" near the data.

How to Parallel Program?

- Use a special-purpose language and compiler (generally we don't have these)
- Use ordinary languages + ParProg API & Library

Where to Parallel Program

- High Performance Supercomputers
- Clusters of workstations on a LAN
- Some graphics cards have parallel instructions

Thinking About / Designing Parallel Programs

Textbook suggests three ways:

- Result-oriented
- Agenda-oriented
- Specialist-oriented

Best explained by examples.

"Compute $-\sqrt{\pi}$ and $\log_{\frac{1}{2}}(2.5 \times 10^{14})$"

unrelated tasks can be done in parallel

Result-oriented parallelism

"Tailor 50 suits"

→ refine this into more concrete, smaller jobs

↓

measure 50 customers

↓

cut, stretch cloth, etc

↓

sew

↓

test against customer

↓

done

Not all steps have equal size ... can be done "anonymously" by a team of "worker bees"