Slide 1

Administrivia

- Homework 4 on Web. Due next Monday.
- Information about projects to be on Web soon.
- Guest speaker Friday.

Slide 2

Example Application: Mandelbrot Set, Continued

- Review:
- Code is a loop over points in a 2D space, where at each point we calculate until divergence / maximum iterations and then plot the result (to something implicitly or explicitly shared).
- Consider parallelizing for a distributed-memory environment (possibly for shared-memory environment too).

Slide 3

Parallelization — Strategy

- For distributed memory, assign one process (master) to manage the display and possibly assign/distribute work. Other processes (workers) do calculations and send back results.

  Some choices about how to distribute work — by blocks? dynamically? round-robin? perhaps try all of them.

- For shared memory, distribute work among threads and allow only one thread at a time to update the display. (Could try to emulate the strategy used for distributed memory, but it might be tricky to coordinate master with workers.)

  Similar choices about how to distribute work.

  Also consider whether to make "one thread at a time" section the update of a single point or something else.

Slide 4

Parallelization — Code

- (Look at code, multiple versions.)

Slide 5

Minute Essay

- For homework 4, your mission will be to parallelize quicksort. What algorithm structure seems to fit best — *Task Parallelism* (like the numerical integration example), *Divide and Conquer*, *Geometric Decomposition* (like the heat equation), *Recursive Data*, *Pipeline*, or *Event-Based Coordination*?

Slide 6

Minute Essay Answer

- *Divide and Conquer* seems like a good fit.