Slide 1

Administrivia

- Reminder: Homework 2 due Monday. (Unless . . . I'm having trouble making leapfrogging work with OpenCL. I'll send e-mail probably tomorrow, either giving you some hints about how to make it work or — something.)

Slide 2

Numerical Integration Revisited

- The OpenMP version of this example is simple and compact. However, it might not be the best model for the OpenMP version of Homework 2 — seems like you want to be able to control which random sequence each iteration of the main loop uses, and no obvious way to do that with OpenMP parallel `for`.
- So instead we could try an SPMD-style approach . . .
- Also note that while declaring variable `x` outside the parallel block allows it to be an example of how to deal with such data, really it could be local to the thread or even to the loop, no?
- (Example revised.)

Slide 3

Task-Based Versus Data-Based Decomposition – Review

- Two basic approaches to decomposing a problem — task-based and data-based. Usually one will seem more logical than the other, but may need to think through both.
- Either way, you'll look at both tasks and data; difference is in which you look at first, and then the other follows.

Slide 4

Examples — Recap

- Simplified generic molecular dynamics application — timestepped simulation, where in each step we compute forces on all atoms ($O(N^2)$ unless we ignore distant particles) and then use forces to recompute positions and velocities.
- Heat-diffusion equation — repeatedly compute new values for each point based on old value of it and neighbors, continuing until we get convergence.

Slide 5

Decomposition — Examples

- Next slides will show working through our two examples. For purposes of illustration, we'll do one starting with a *Task Decomposition* and inferring a *Data Decomposition*, the other one the other way around.

Slide 6

Molecular Dynamics Example — Task Decomposition

- Tasks that find the vibrational forces on an atom.
- Tasks that find the rotational forces on an atom.

  (Together, these are tasks to compute "bonded forces" — those due to chemical bonds.)
- Tasks that find the non-bonded forces on an atom (the ones due to electrical charges).
- Tasks that update the position and velocity of an atom.
- Tasks that update the neighbor list for an atom. (Or we could consider updating all the neighbor lists as one task, as in the book, if we think it won't be done very often and therefore is not worthwhile to parallelize.)

Slide 7

Molecular Dynamics Example — Data Decomposition

- Key data structures:
  - An array of atom coordinates, one element per atom.
  - An array of atom velocities, one element per atom.
  - An array of lists, one per atom, each defining the neighborhood of atoms considered to be "close".
  - An array of forces on atoms, one element per atom.
- Decompose each of these to match task decomposition — into elements corresponding to individual atoms.

Slide 8

Heat Diffusion Example — Data Decomposition

- Key data structures:
  - Array for "old values" (time step k).
  - Array for "new values" (time step $k + 1$).
- Most computation involves updating or otherwise operating on these two arrays. Think of partitioning into "chunks".

Slide 9

Heat Diffusion Example — Task Decomposition

- Tasks to compute new values from old values, one per chunk.
- Tasks to compute maximum difference between new and old values, one per chunk.
- Task to swap pointers (to fake copying new values to old values).

Slide 10

Group Tasks

- Once you've broken down problem into tasks / data chunks, need to put it back together as design for parallel algorithm.
- First step — look for "groups of tasks" — logically related, or interdependent, or all with same constraints, etc. Often just one group.

Slide 11

Order Tasks

- Next step — identify constraints on groups of tasks. Possibilities:
  - "First this, then that."
  - "All of these together."

Slide 12

Molecular Dynamics Example — Group Tasks, Order Groups

- Task groups based on list of a few slides back — each type of task (e.g., compute rotational forces) defines a task group.
- Ordering constraints, for each timestep:
  - Task group to compute neighbor list must run before task group to compute non-bonded forces.
  - Task groups to compute bonded and non-bonded forces must run before task group to update positions and velocities.
  - Task group to update positions and velocities must run before next timestep.
- (Also see Figure 3.4 in the book.)

Slide 13

Heat Diffusion Example — Group Tasks, Order Groups

- Task groups based on list of a few slides back — each type of task (e.g., compute new values from old values) defines a task group.
- Ordering constraints, for each timestep:
  - Three tasks groups must run in sequence.
  - All task groups must run before next timestep.

Slide 14

Minute Essay

- None really — just sign in.