Overview

- Introduction and Motivation
- Structured Grids
  - Adaptive structured grids
- Unstructured Grids
  - Adaptive unstructured grids
- Particles and Spatial Search
  - Regular grids
  - Trees
Introduction / Motivation

There are only a few ways to represent the problem domain:
- Structured Grids
- Unstructured Grids
- Particles

Knowing the basic terms helps you talk to application folks, and understand their code.
Grids in General
So you’re trying to represent some physical situation, like heat flow

You decide to divide up space into a bunch of little pieces:

Grids: Introduction

- Element, or Cell, or Volume
- Node, or Vertex, or Point
Grids: Location of Data

- **Element Centered Data**
  - **Fluid Dynamics, most PDEs**
  - Data values constant (or simple) in a cell

- **Node Centered Data**
  - **Structural dynamics/FEM**
  - “Shape function” interpolates between nodes

Hybrids too, like Arakawa C-grid
Grids: Motion of Grid and Data

- **Eulerian**: non-moving grid
  - E.g., pressure waves move through the grid in CFD

- **Lagrangian**: moving grid
  - E.g., grid deformation follows the structure deformation in FEM

Or hybrid, e.g. “ALE”
Structured Grids
Structured Grids: Introduction

- AKA “Regular Grid”, since grid cells lie in regular rows and columns
- Cells are stored in a 3D array
- Cells can lie along axes (“rectilinear grid”); or curve through space
Structured Grids: Terminology

- “Stencil” of source cells to compute a destination cell
  - Classic GPU algorithm
  - Common in fluid dynamics
  - Also found in PDE solvers

- Read-only “Ghost” or “Dummy” cells around boundary
Structured Grids: Applications

- Fluid Dynamics
  - Classical fluid dynamics grid
- Jacobi and other PDE solvers
  - “Finite Difference” formulation
- Level set methods
  - E.g., fluid solidification phase field
- Image processing
  - Just a 2D pixel array!
Adaptive Structured Grids
Adaptive Structured Grids: Intro

- “Adaptive Mesh Refinement”/AMR
- Cells are stored in small 3D arrays, linked together with pointers
- For regular refinement, use quadtree (2D) or octree (3D); can be irregular “block structured AMR”
Adaptive Structured Grids: Terms

- "Refinement" and "Coarsening" criteria control evolution of mesh
  - Basically simulation error estimates

- "Hanging Node Constraint"
  - Neighbors must have similar (±1) refinement level

![Diagram of Adaptive Structured Grids]

bad!
Adaptive Structured Grids: Apps

- Adaptive physics solvers
- LLNL SAMRAI C++ Framework
- NASA GSFC PARAMESH
- AMRITA (James Quirk)
- INRIA GPU Gems 3:5
Unstructured Grids
Unstructured Grids: Introduction

- AKA “Mesh”
- Cells are stored in 1D array
- Vertices (“nodes”) of each cell (“element”) are listed explicitly
- Mesh consists of triangles and/or quadrilaterals (2D); tetrahedra, cubes/hexahedra, prisms, pyramids (3D)
Unstructured Grids: Terms

- “Ghost regions”, like structured grids
- “Shared nodes” along partition boundaries:
  - Run computation on separate pieces
  - Add up node forces along boundaries
Unstructured Grids: Terms

- “Conformality”
  - Nodes never land in middle of element
  - Enforced during mesh generation/modification
Unstructured Grids: Applications

- Structural Mechanics
  - This is the classic finite element mesh

- Fluid Dynamics
  - In strange domains, where structured grids are tough to automatically generate

- Can be extended to Adaptive Meshes!
Adaptive Unstructured Grids
Adaptive Unstructured Grids: Intro

- AKA “Mesh Refinement”, shades into from-scratch “Mesh Generation”
- Cells still stored in 1D arrays, but the cells can now change
- Must respect conformality
- Must ensure element “quality”
- Must work in parallel
Adaptive Meshes: Terminology

- "Delaunay" mesh and "flip"

- "Edge bisection": cut edge in middle
Almost every unstructured mesh program wants to be adaptive.

- Charm++ Triangle Mesh Refinement (Wilmarth)
- Charm++ PMAF3D (Wilmarth)
- Charm++ Tet Data Transfer Library (Lawlor)
Particle Methods and Spatial Search
Particles and Spatial Search

- To work on a particle, you need nearby particles
  - E.g., all particles within cutoff $r$
    - Used for molecular dynamics (NAMD)
  - or, all $k$ nearest particles
    - Used by Smoothed Particle Hydrodynamics (SPH) methods

- Search for neighboring particles is spatial, so need a "spatial search structure"
  - Can use: structured grid, adaptive search tree, unstructured grid, ...
... using Structured Grids

- E.g., NAMD molecular dynamics
  - Particles are Atoms
  - Search structure is based on “Patches” of space in regular, rectilinear grid

- E.g., Charm++ Collision Library
  - Search structure is based on regular rectilinear voxel grid

... never talk to atoms over here...
... using Search Trees

- E.g., Cosmology simulations
  - Particles are stars, galaxies
  - Search structure is a spatial octree

- SPH: “Smoothed particle hydrodynamics”

- Barnes-Hut gravity
  - “Tree walk”
Conclusions
Conclusions

- There are only a few ways to represent the problem domain:
  - Structured Grids
  - Unstructured Grids
  - Particles

- There are a lot of specialized terms, but very few concepts