ACM/CS 114
Parallel algorithms for scientific applications

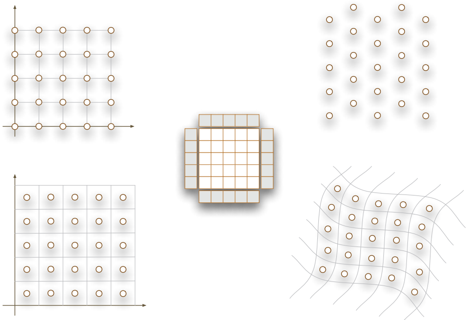
Michael A. G. Aïvázis

California Institute of Technology

Winter 2012

Overview

- lattices: logically Cartesian grids
  - flow, lattice dynamics, wave propagation, image processing
- suitable when it is possible to find an invertible map ϕ from the problem domain Ω to $\mathbb{Z}^d$
- data representation: multi-dimensional arrays
  - ϕ maps points in Ω to loop indices
  - with *guard cells* for enforcing boundary conditions

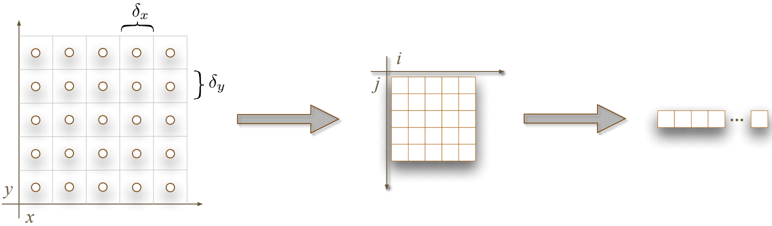


Advantages and disadvantages

- advantages: logically rectangular
  - indexing: easy traversal using loops
  - fixed stride: predictable memory layout
  - topology: finding neighbors is trivial
- disadvantages: many problems don't fit in simple boxes...
  - non-trivial geometries are hard to model
  - the representational simplicity disappears quickly as modeling complexity increases
    - domain feature resolution
    - complex initial and boundary conditions
    - adaptive refinement: allowing the properties of the solution to direct where computational resources are spent

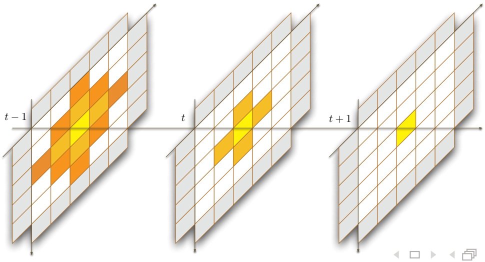
Data layout and performance

- in multi-tier memory architectures
  - good data locality enables efficient cache use
- multi-dimensional arrays: naïve implementations do not perform well
  - for large problem sizes
  - for complex physics updates that require keeping track of multiple fields
- representing scalar, vector and tensor fields
  - optimal layout is problem dependent
  - goal is to minimize cache misses while updating the fields
- the conventional mapping to arrays lays out the data in matrix form
  - not necessarily the most convenient convention
- take ownership of the indexing function



Updating the grid

- two broad categories of problems
  - steady state problems: iterates represent progress towards enforcing a spatial relationship dictated by the differential equation
  - time dependent problems: iterates represent the time evolution of the solution to the spatial problem
- two broad strategies:
  - *implicit* solvers cast the constraints among iterates as a large system of simultaneous equations
  - *explicit* solvers keep track of only a small number of the iterates and use them to advance the solution one step at a time
- the complexity of the update determines the *stencil*



Computing derivatives

- there are three different first order approximations
  - forward difference:

$$\partial\, \square\blacksquare\square = \frac{1}{\delta}\left(\square\square\blacksquare - \square\blacksquare\square \right) \tag{1}$$

  - backward difference:

$$\partial\, \square\blacksquare\square = \frac{1}{\delta}\left(\square\blacksquare\square - \blacksquare\square\square \right) \tag{2}$$

  - central difference:

$$\partial\, \square\blacksquare\square = \frac{1}{2\delta}\left(\square\square\blacksquare - \blacksquare\square\square \right) \tag{3}$$

  where δ is the uniform grid spacing
- forward and backward differences are most often used for *explicit* time integration
- central differences are used to compute spatial derivatives
- the second order central difference is given by

$$\partial\, \square\square\blacksquare\square\square = \frac{1}{12\delta}\left(- \square\square\square\square\blacksquare + 8\, \square\square\square\blacksquare\square - 8\, \square\blacksquare\square\square\square + \blacksquare\square\square\square\square \right) \tag{4}$$

Partial derivatives in two dimensions

- let δ_x and δ_y be the uniform grid spacing along each dimension
- then, the first order central difference approximations to the spatial derivatives are given by

$$\partial_x \begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} = \frac{1}{2\delta_x}\left(\begin{smallmatrix}\square&\square&\square\\\square&\square&\blacksquare\\\square&\square&\square\end{smallmatrix} - \begin{smallmatrix}\square&\square&\square\\\blacksquare&\square&\square\\\square&\square&\square\end{smallmatrix} \right) \tag{5}$$

$$\partial_y \begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} = \frac{1}{2\delta_y}\left(\begin{smallmatrix}\square&\square&\square\\\square&\square&\square\\\square&\blacksquare&\square\end{smallmatrix} - \begin{smallmatrix}\square&\blacksquare&\square\\\square&\square&\square\\\square&\square&\square\end{smallmatrix} \right) \tag{6}$$

- and the second order derivatives are given by

$$\partial_{xx} \begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} = \frac{1}{\delta_x^2}\left(\begin{smallmatrix}\square&\square&\square\\\square&\square&\blacksquare\\\square&\square&\square\end{smallmatrix} - 2\begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} + \begin{smallmatrix}\square&\square&\square\\\blacksquare&\square&\square\\\square&\square&\square\end{smallmatrix} \right) \tag{7}$$

$$\partial_{yy} \begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} = \frac{1}{\delta_y^2}\left(\begin{smallmatrix}\square&\square&\square\\\square&\square&\square\\\square&\blacksquare&\square\end{smallmatrix} - 2\begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} + \begin{smallmatrix}\square&\blacksquare&\square\\\square&\square&\square\\\square&\square&\square\end{smallmatrix} \right) \tag{8}$$

$$\partial_{xy} \begin{smallmatrix}\square&\square&\square\\\square&\blacksquare&\square\\\square&\square&\square\end{smallmatrix} = \frac{1}{4\delta_x\delta_y}\left(\begin{smallmatrix}\square&\square&\square\\\square&\square&\square\\\square&\square&\blacksquare\end{smallmatrix} - \begin{smallmatrix}\square&\square&\blacksquare\\\square&\square&\square\\\square&\square&\square\end{smallmatrix} - \begin{smallmatrix}\square&\square&\square\\\square&\square&\square\\\blacksquare&\square&\square\end{smallmatrix} + \begin{smallmatrix}\blacksquare&\square&\square\\\square&\square&\square\\\square&\square&\square\end{smallmatrix} \right) \tag{9}$$

Solving a simple PDE on a uniform structured grid

- Laplace equation over some domain $\Omega \in \mathbb{R}^d$, subject to Dirichlet boundary conditions

$$\nabla^2 \phi = 0 \quad \text{with} \quad \phi(\partial\Omega) = f \tag{10}$$

- let the grid be uniform: $\delta_x = \delta_y$
- in two dimensions, using first order central differences, Eq. 10 becomes

$(\partial_{xx} + \partial_{yy})$  $=$ 0 (11)

4  $=$  $+$  $+$  $+$  (12)

- and translates into the following constraint among grid elements

 $= \frac{1}{4}$  (13)

using a shorthand for the sum of the neighboring cells

An example

- specifically,
  - let Ω be the unit box in two dimensions
  - and let ϕ satisfy the following boundary conditions

$$\begin{array}{rcl} & & \phi(x,0) & = & \sin(\pi x) & 0 \le x \le 1 \\ & & \phi(x,1) & = & e^{-\pi}\sin(\pi x) & 0 \le x \le 1 \\ \phi(0,y) & = & \phi(1,y) & = & 0 & 0 \le y \le 1 \end{array} \tag{14}$$

- the exact solution is given by

$$\phi(x,y) = e^{-\pi y}\sin(\pi x) \tag{15}$$

- we will solve this equation using the Jacobi iterative scheme:
  - make an initial guess for ϕ over a discretization of Ω
  - apply the boundary conditions
  - interpret Eq. 13 as an update step to compute the next iteration

 $_t = \frac{1}{4}$  $_{t-1}$ (16)

  - stop when a convergence criterion is met