Parallel Programming
in C with MPI and OpenMP

Michael J. Quinn

Mc Graw Hill

Chapter 13

Finite Difference Methods

Outline

- Ordinary and partial differential equations
- Finite difference methods
- Vibrating string problem
- Steady state heat distribution problem

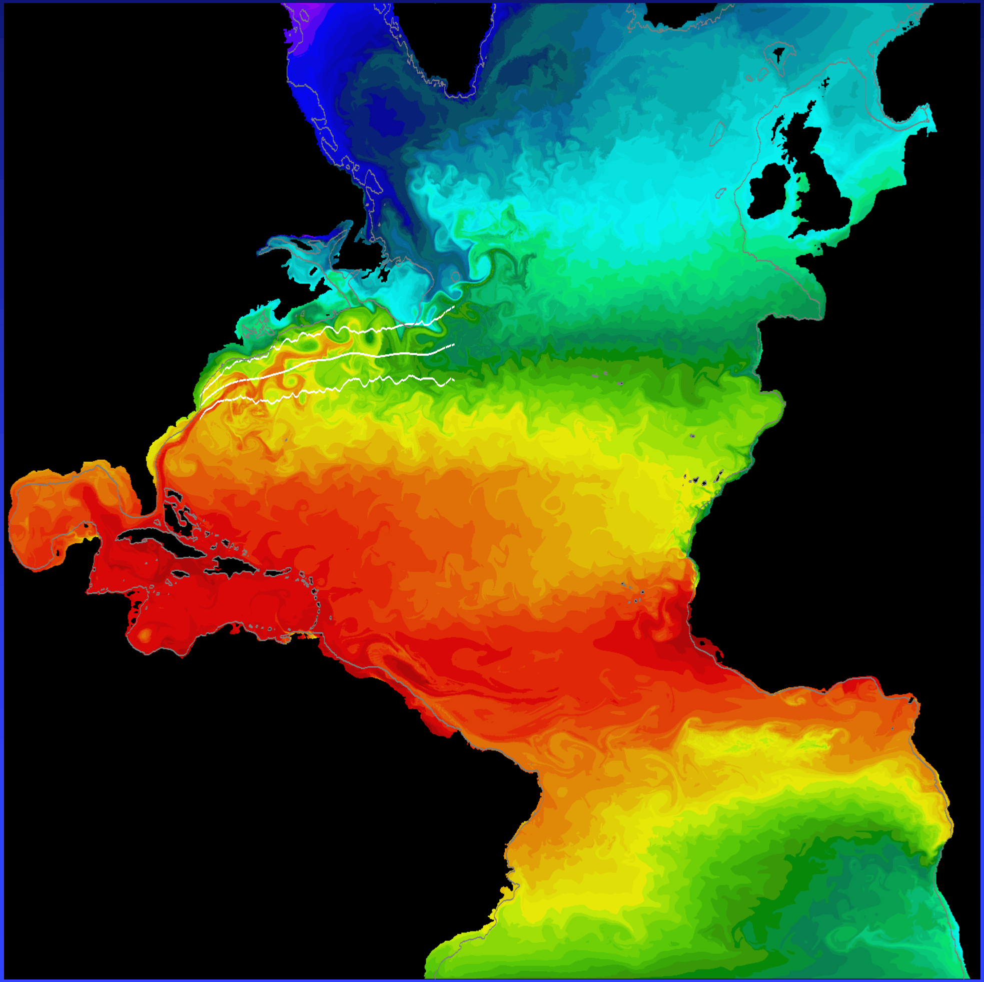
Ordinary and Partial Differential Equations

- Ordinary differential equation: equation containing derivatives of a function of one variable in one dimension
- Partial differential equation: equation containing derivatives of a function in higher Dimensions where the differentiation occurs in some dimension(s).

Examples of Phenomena Modeled by PDEs

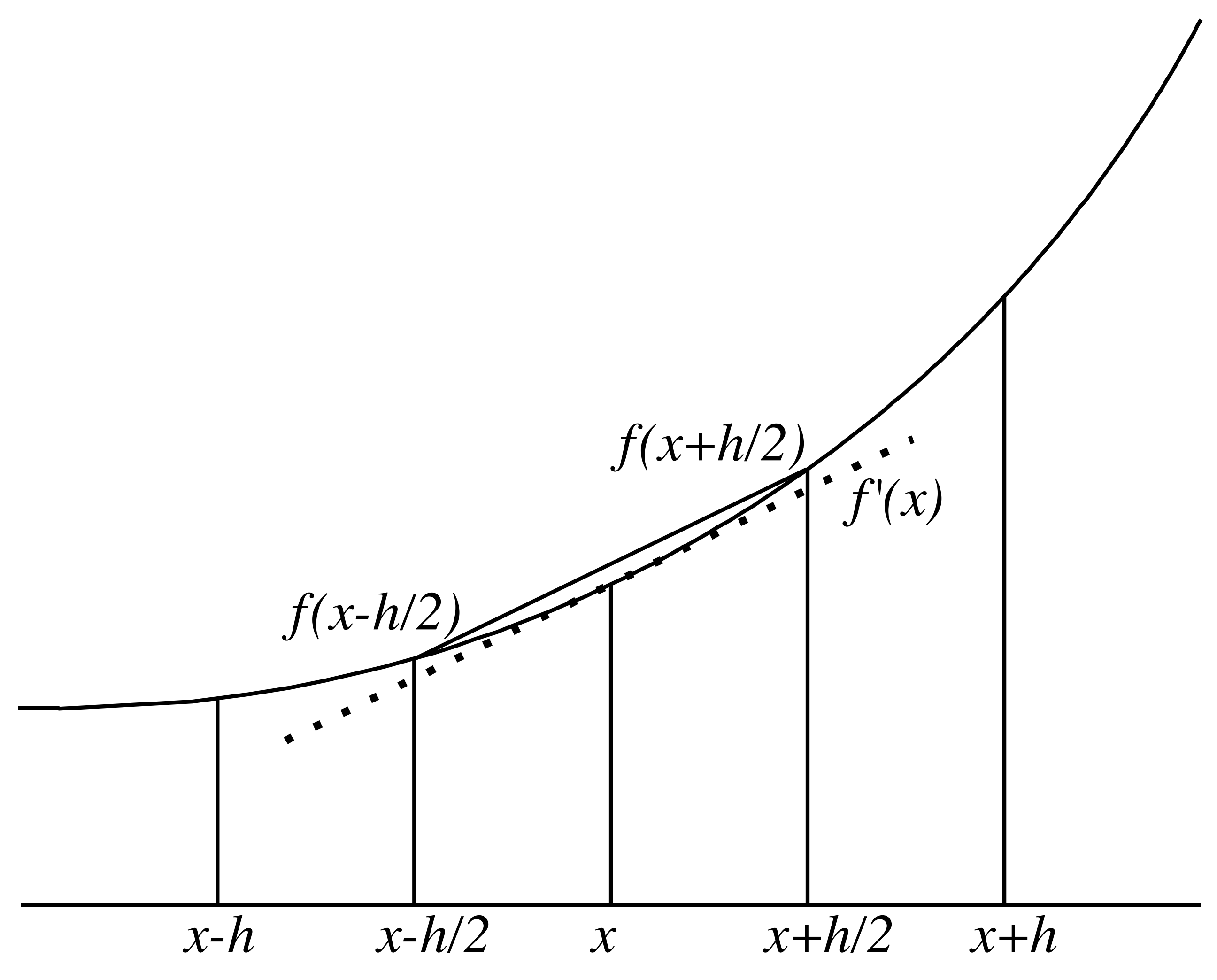
- Air flow over an aircraft wing
- Blood circulation in human body
- Water circulation in an ocean
- Bridge deformations as its carries traffic
- Evolution of a thunderstorm
- Oscillations of a skyscraper hit by earthquake
- Heat transfer

Model of Sea Surface Temperature in Atlantic Ocean



Courtesy MICOM group
at the Rosenstiel School
of Marine and Atmospheric
Science, University of Miami

Difference Quotients



Formulas for 1st, 2d Derivatives

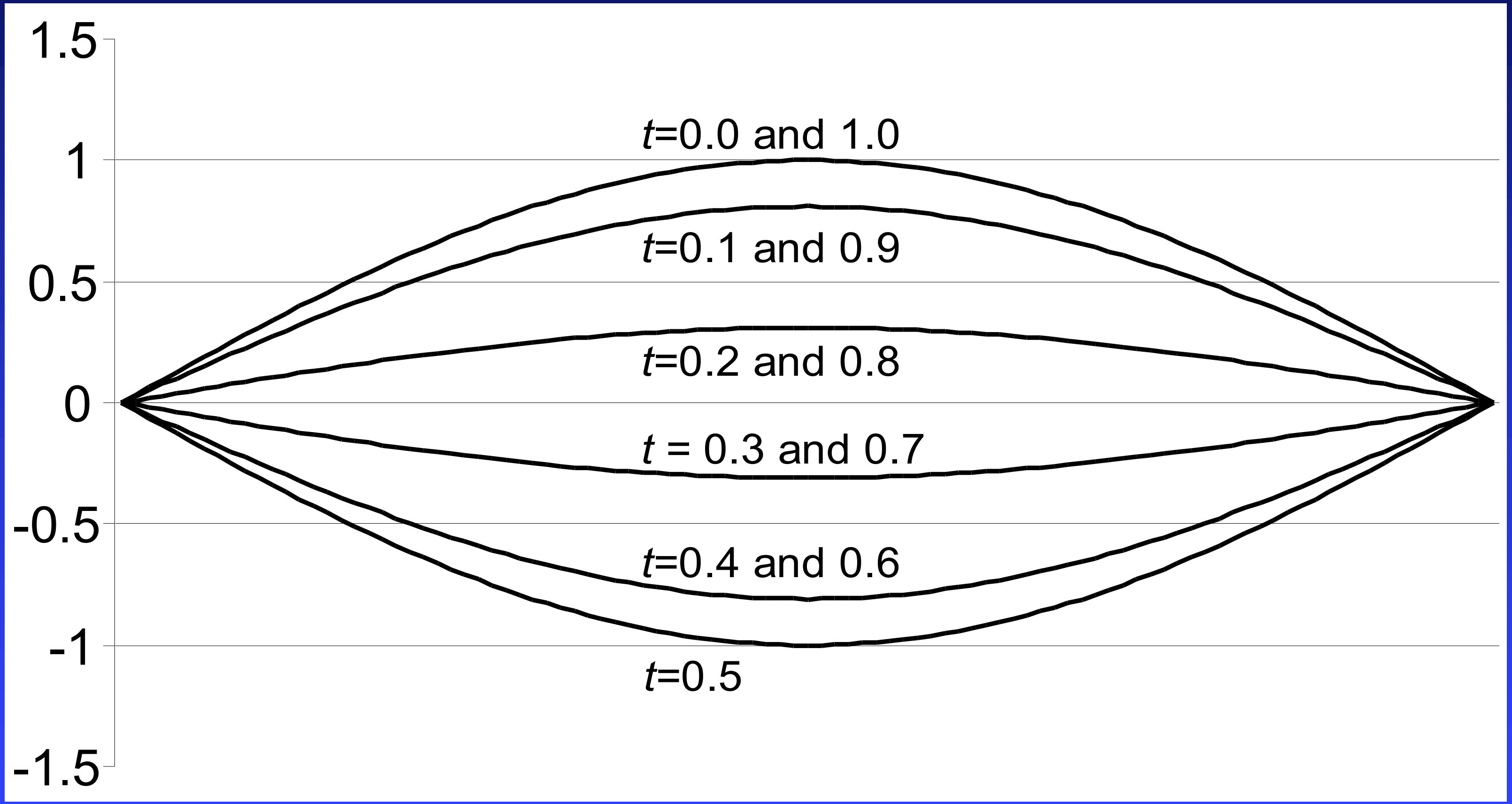
$$f'(x) \approx \frac{f(x+h/2) - f(x-h/2)}{h}$$

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

f'(x) and f''(x)

- f'(x) often called df/dx
  - first derivative, the rate of change of f in the x direction. If f is location then f' is speed.
- f''(x) often called d^2f/dx^2
  - second derivative, the rate of change of f' in x direction. If f' is speed, then f'' is accelleration.

Vibrating String Problem

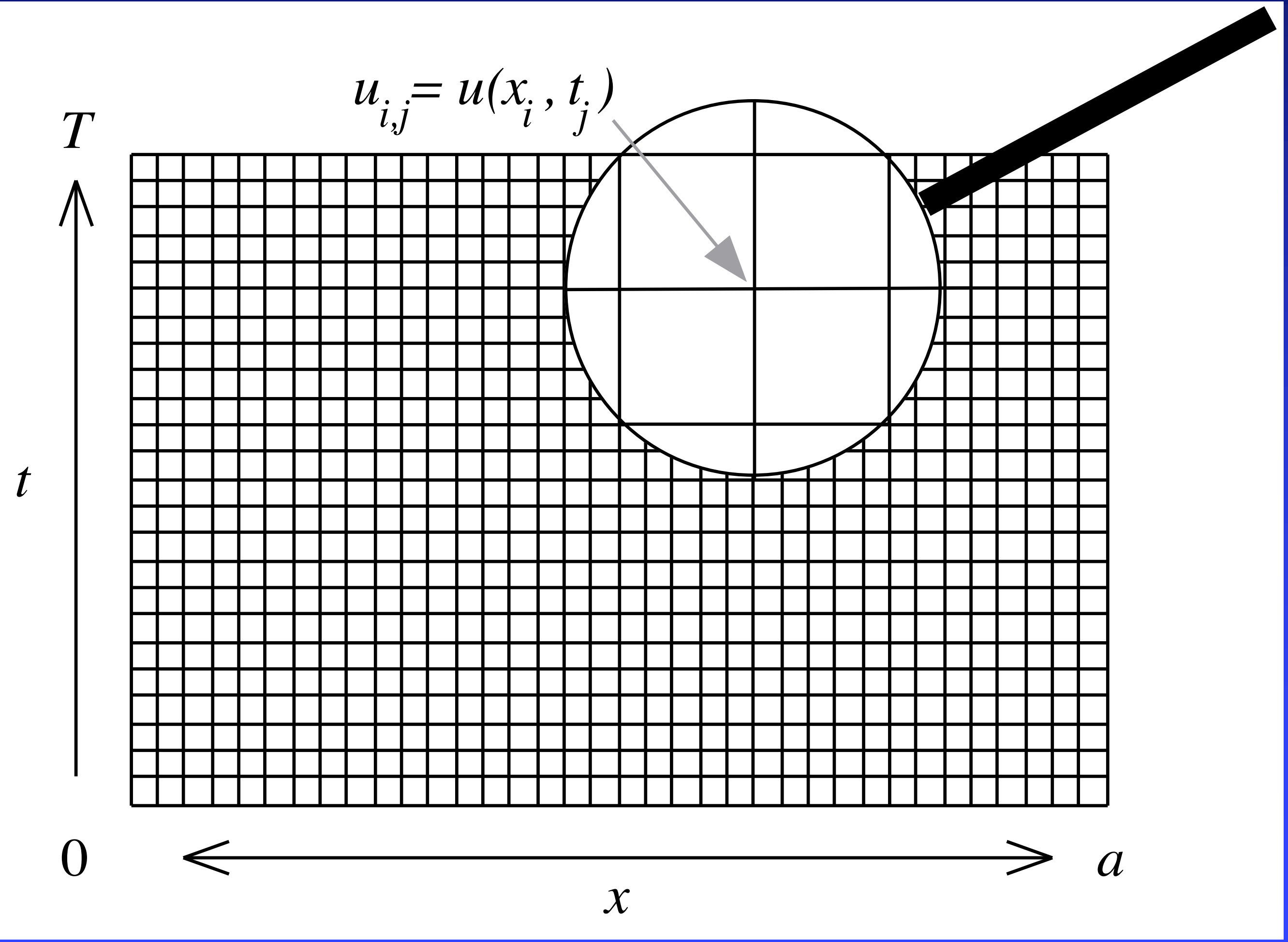


Vibrating string modeled by a hyperbolic PDE

Solution Stored in 2-D Matrix

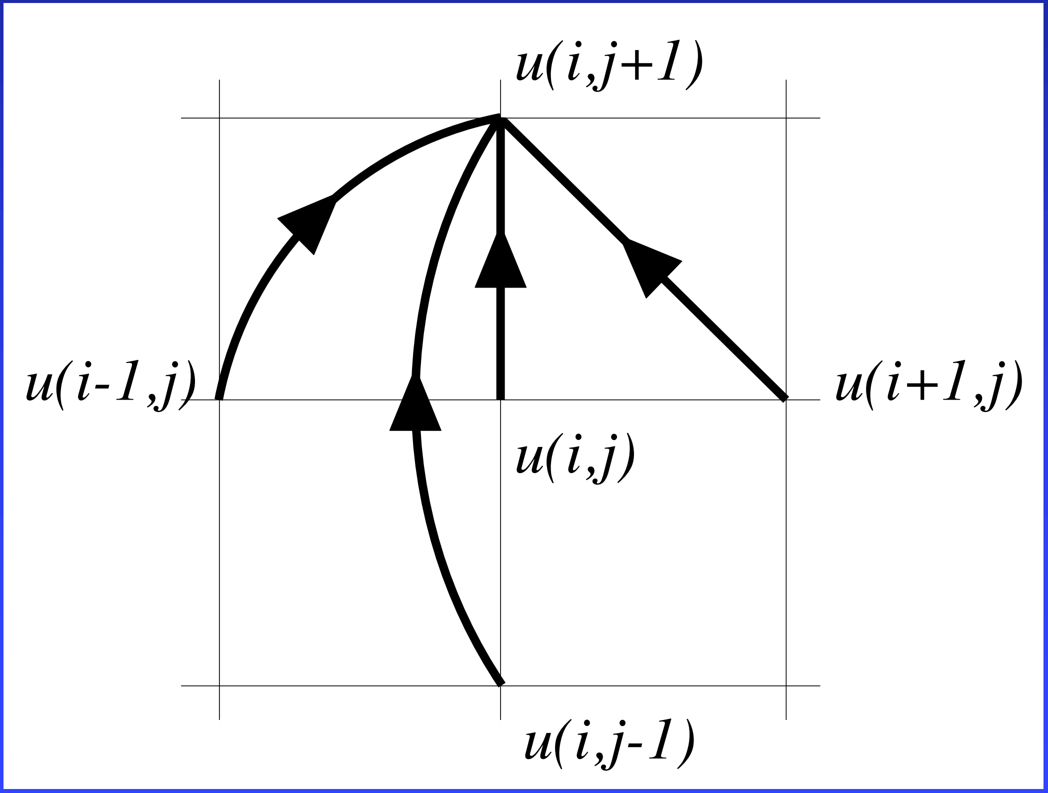
- Each row represents state of string at some point in time
- Each column shows how position of string at a particular point changes with time
- In some cases (jacobi 1D in PA4) we are just interested in the end state (heat transfer) and we keep only 2 vectors (prev and curr)

Discrete Space, Time Intervals Lead to 2-D Matrix



Heart of Sequential C Program

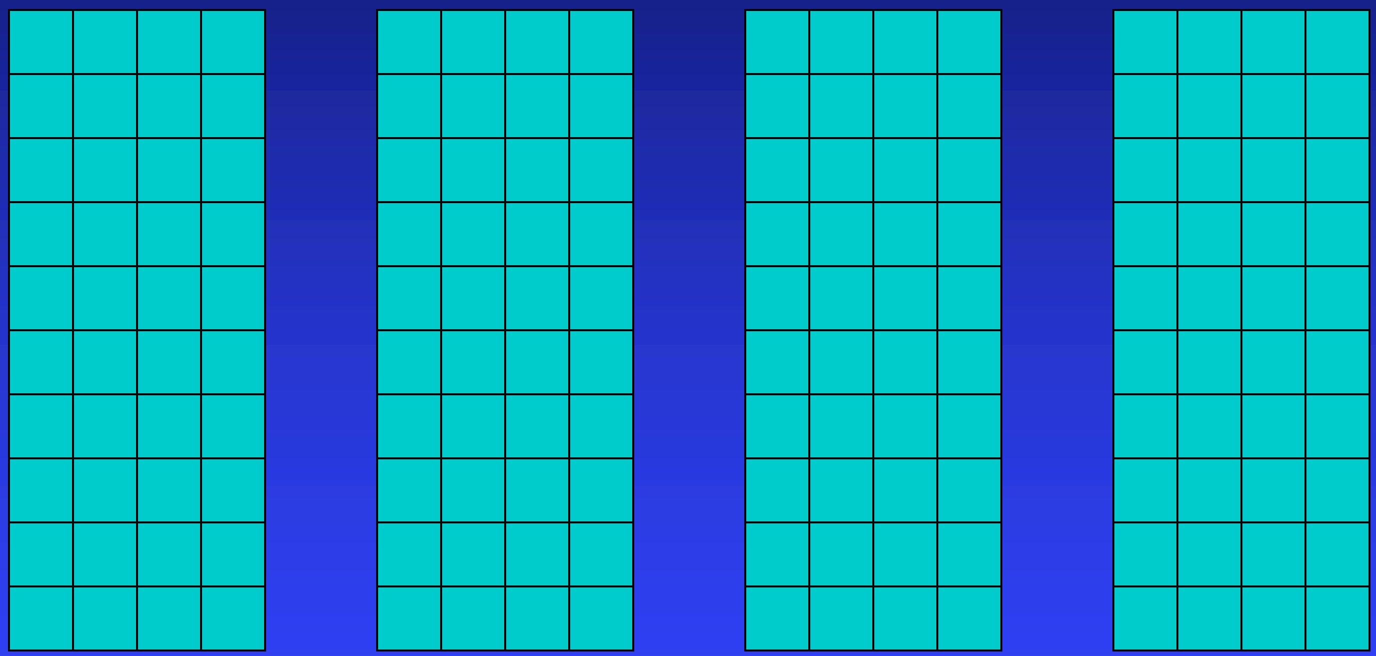
```
u[j+1][i] = 2.0*(1.0-L)*u[j][i] +
    L*(u[j][i+1] + u[j][i-1]) - u[j-1][i];
```



Parallel Program Design

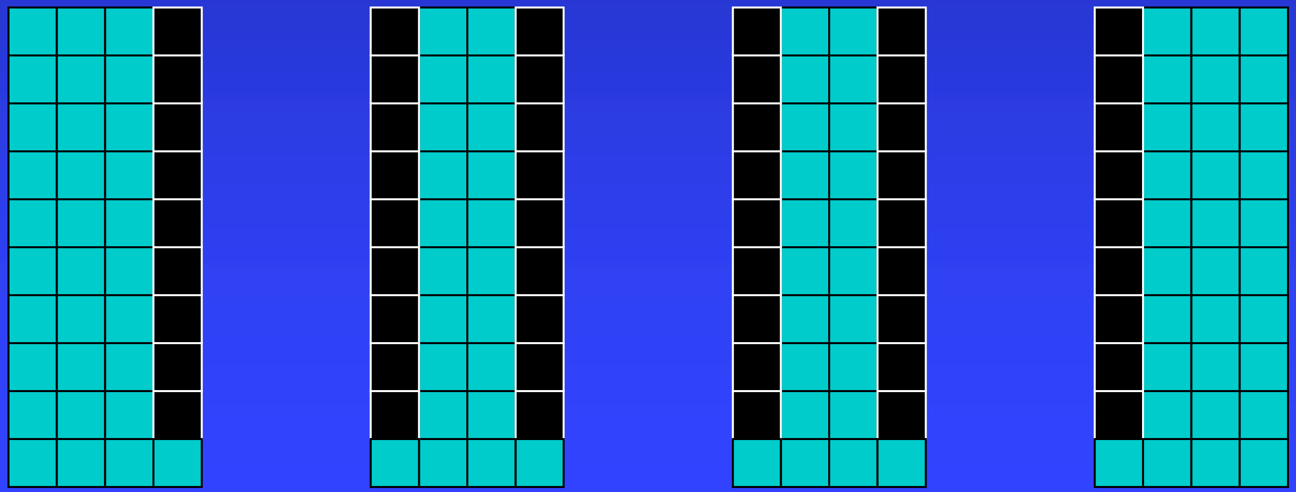
- Associate primitive task with each element of matrix
- Examine communication pattern
- Agglomerate tasks in same column
- Static number of identical tasks
- Regular communication pattern
- Strategy: agglomerate columns, assign one block of columns to each task

Result of Agglomeration and Mapping



Communication Still Needed

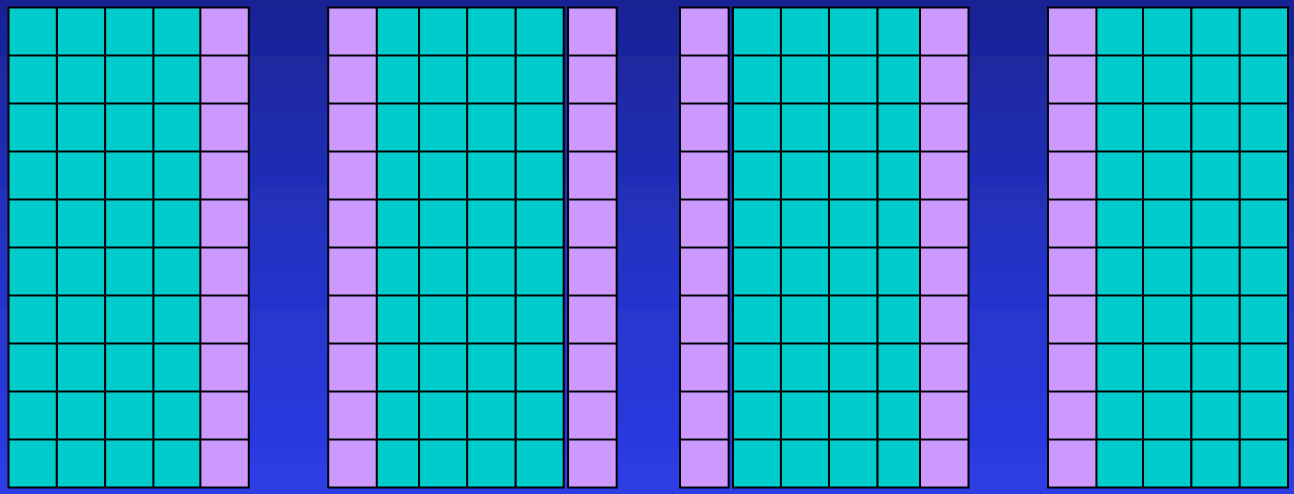
- Initial values (in lowest row) are computed without communication
- Values in black cells cannot be computed without access to values held by other tasks



Ghost Points

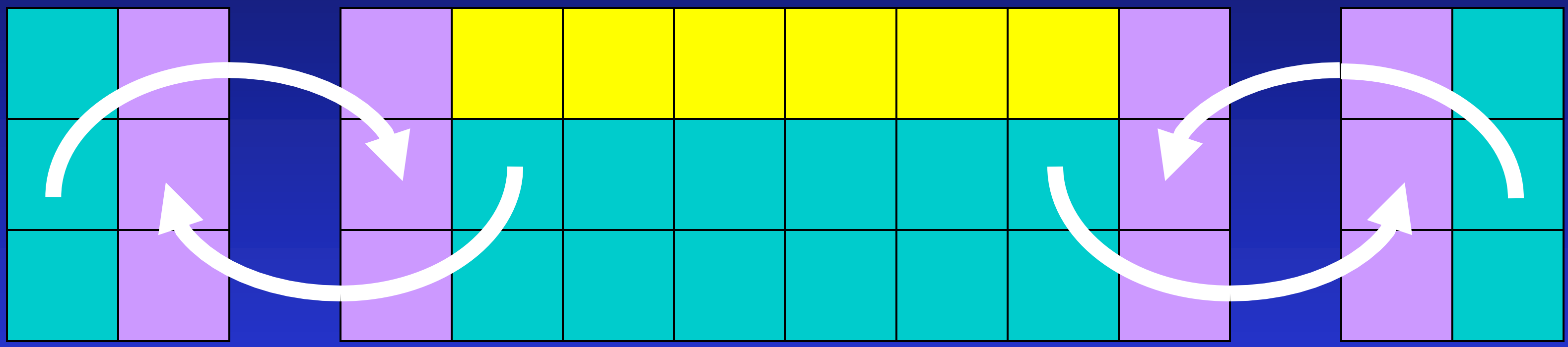
- Ghost points: memory locations used to store redundant copies of data held by neighboring processes
- Allocating ghost points as extra columns simplifies parallel algorithm by allowing same loop to update all cells

Matrices Augmented with Ghost Points



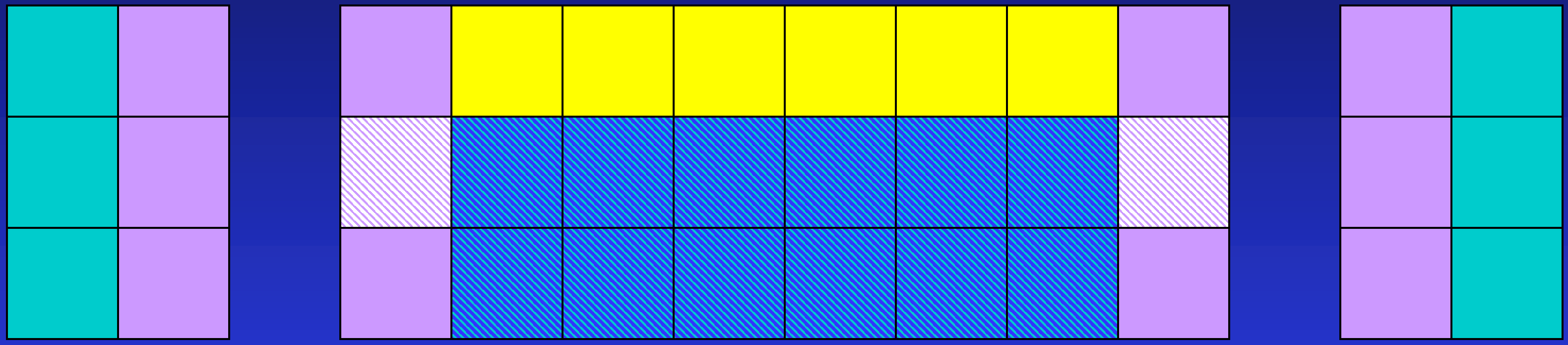
Lilac cells are the ghost points.

Communication in an Iteration



This iteration the process is responsible for computing the values of the yellow cells.

Computation in an Iteration



This iteration the process is responsible for computing the values of the yellow cells. The striped cells are the ones accessed as the yellow cell values are computed.

Complexity Analysis

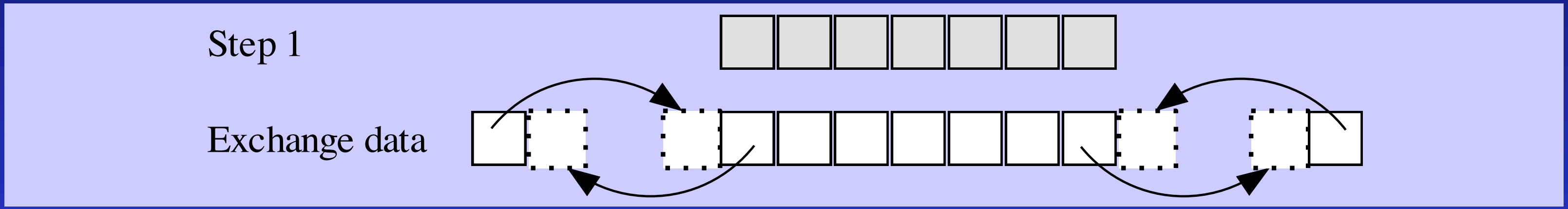
- Computation time per element is constant, so sequential time complexity per iteration is $\Theta(n)$
- Elements divided evenly among processes, so parallel computational complexity per iteration is $\Theta(n / p)$
- During each iteration a process with an interior block sends two messages and receives two messages, so communication complexity per iteration is $\Theta(1)$

Replicating Computations

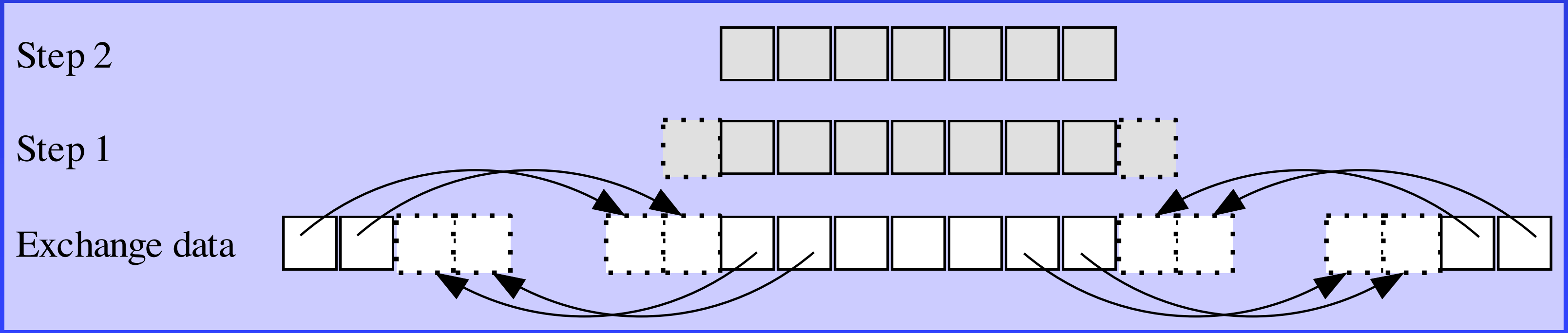
- If only one value is transmitted, the execution time is dominated by communication (message latency)
- We can reduce the number of communications by replicating computations
- If we send k values instead of one, we can advance the simulation k time steps before another communication
- We call these k values Ghost or Halo elements

Replicating Computations

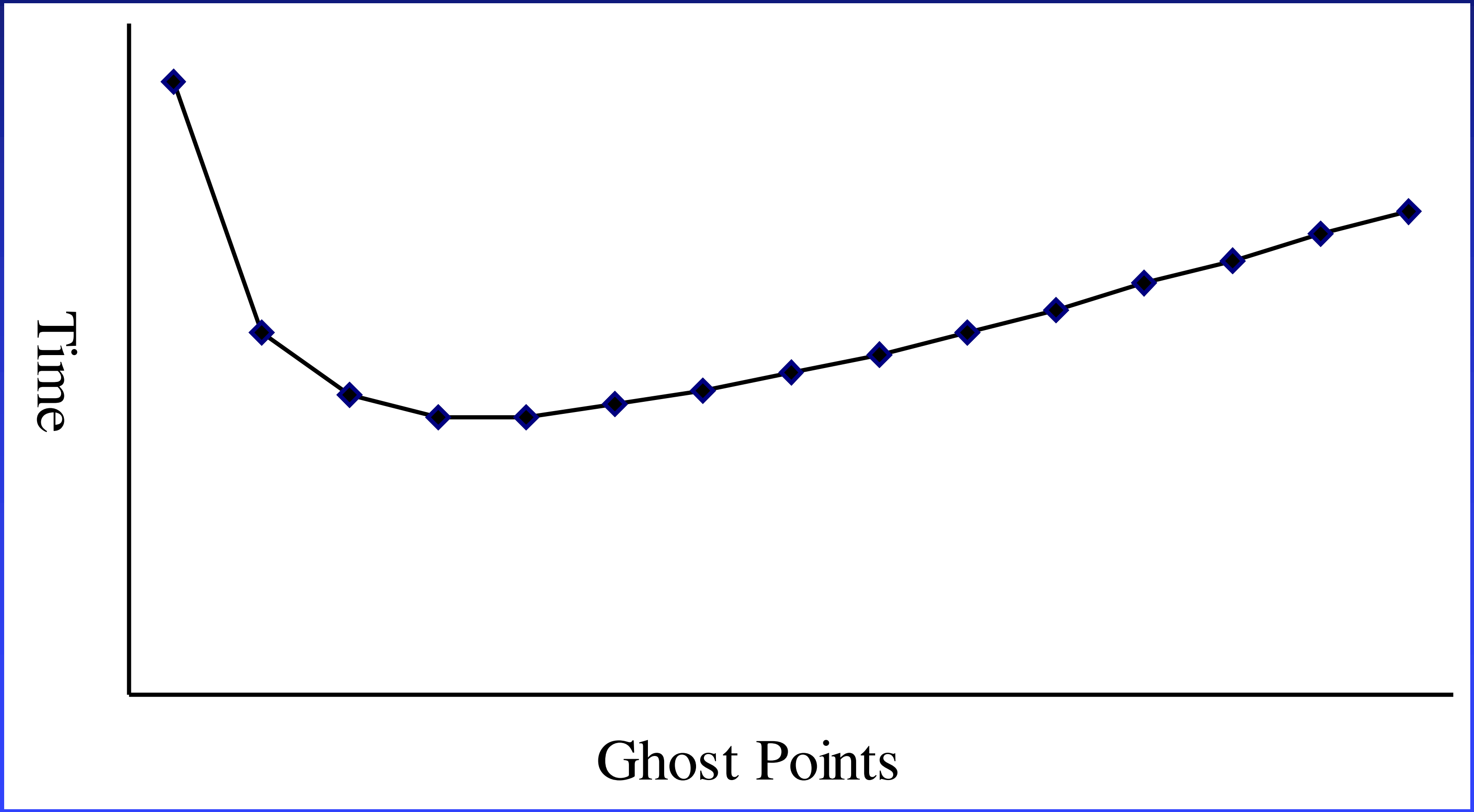
Without replication:



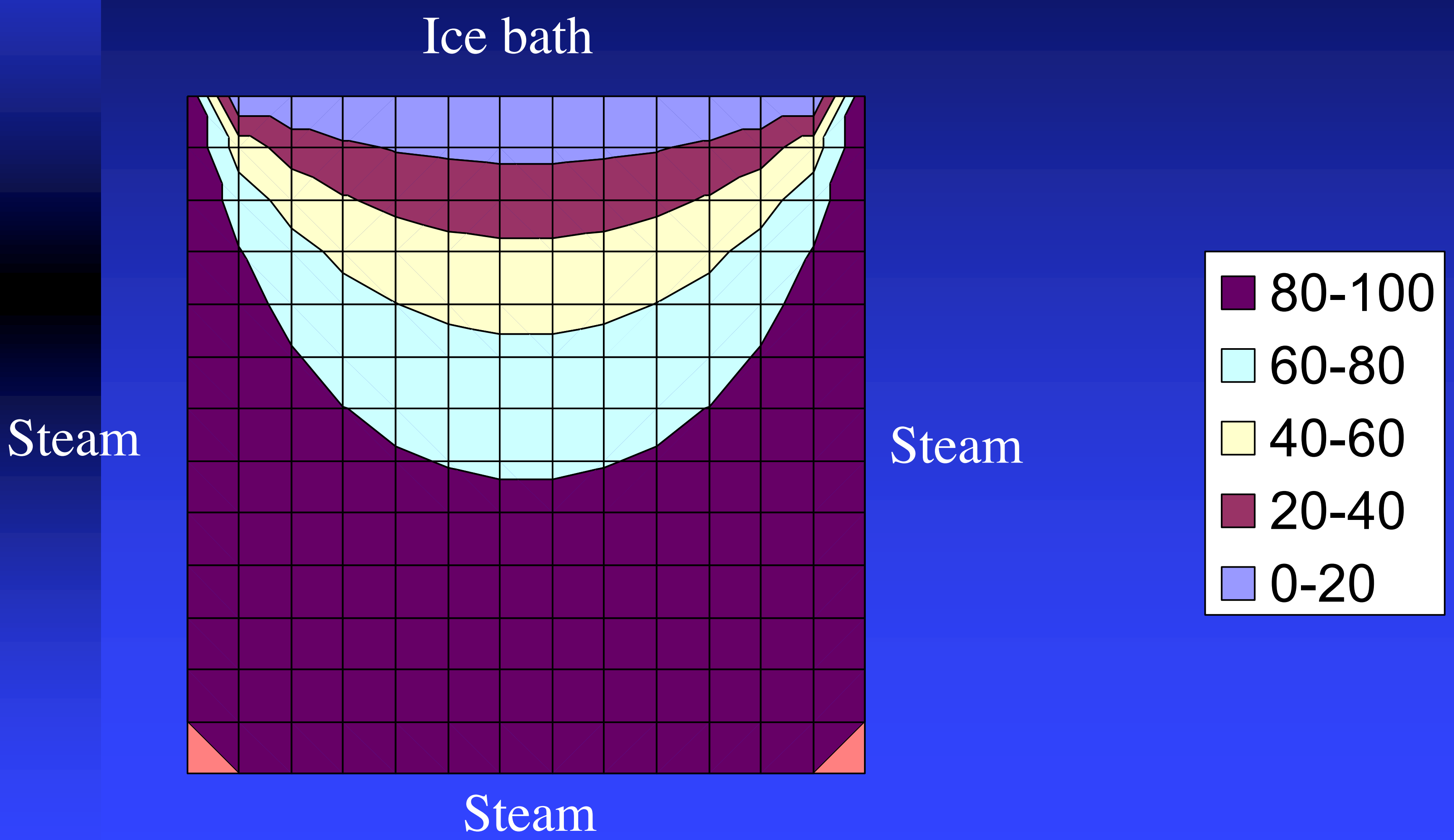
With replication:



Communication Time vs. Number of Ghost Points

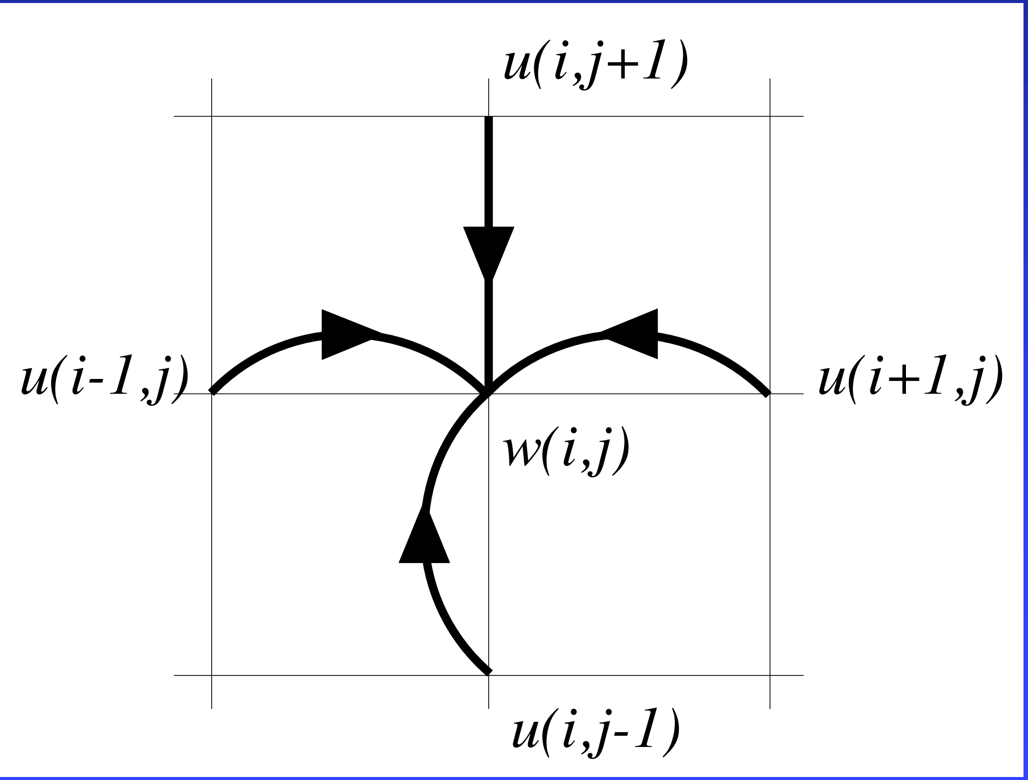


Steady State Heat Distribution Problem



Heart of Sequential C Program

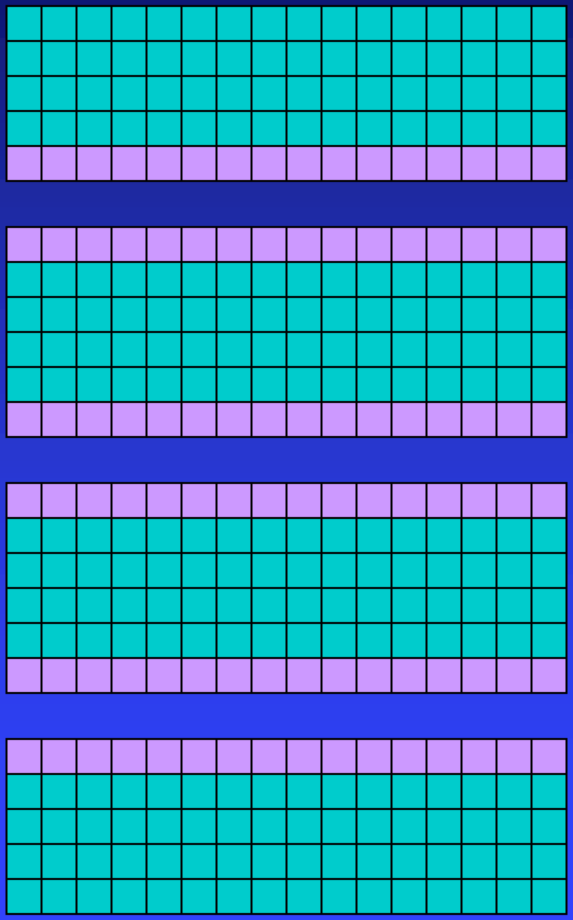
```
w[i][j] = (u[i-1][j] + u[i+1][j] +
           u[i][j-1] + u[i][j+1]) / 4.0;
```



Parallel Algorithm 1

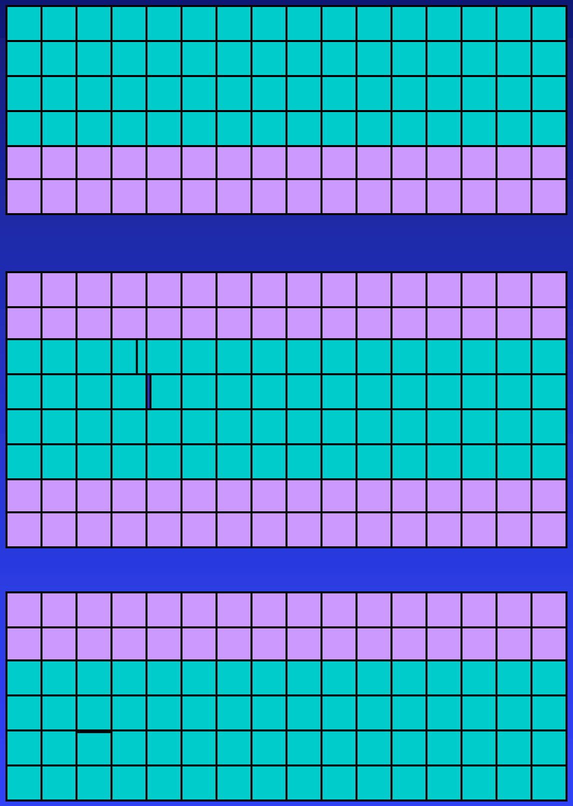
- Associate primitive task with each matrix element
- Agglomerate tasks in contiguous rows (rowwise block striped decomposition)
- Add rows of ghost points above and below rectangular region controlled by process

Example Decomposition



16 × 16 grid
divided among 4 processors

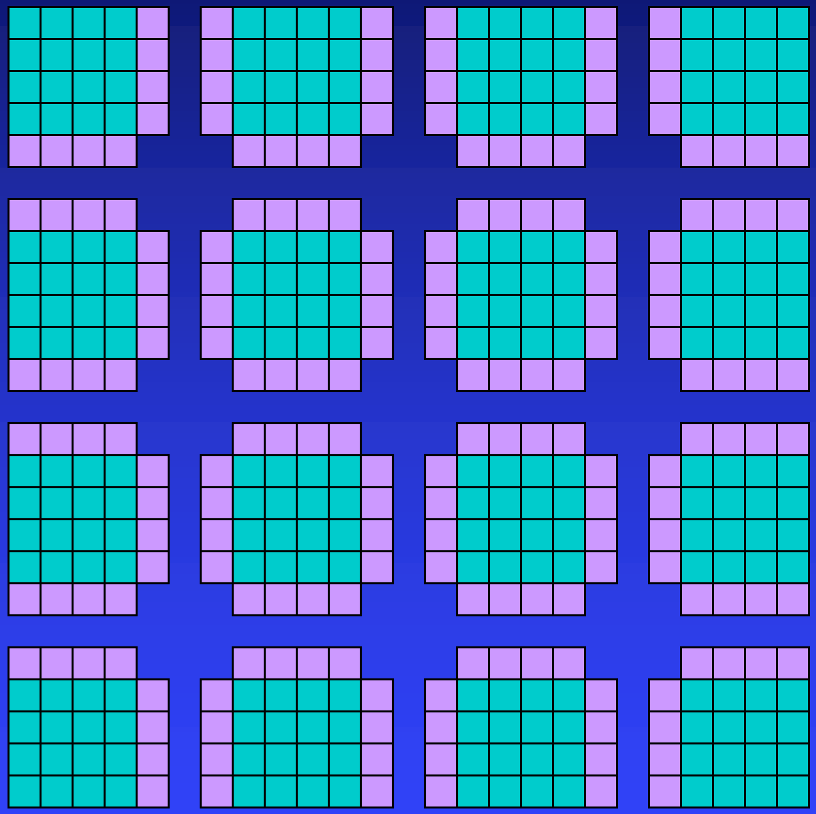
Example Decomposition



16 × 16 grid
divided among 4 processors

Here we can also avoid communication by replicated computation (larger ghost buffers)

Example Decomposition



16 × 16 grid divided among 16 processors

Implementation Details

- Using ghost points around 2-D blocks requires extra copying steps
- Ghost points for left and right sides are not in contiguous memory locations
- An auxiliary buffer must be used when receiving these ghost point values
- Similarly, buffer must be used when sending column of values to a neighboring process

Summary (1)

- PDEs used to model behavior of a wide variety of physical systems
- Realistic problems yield PDEs too difficult to solve analytically, so scientists solve them numerically
- Two most common numerical techniques for solving PDEs
  - finite element method
  - finite difference method

Summary (2)

- Finite different methods
  - Matrix-based methods store matrix explicitly
  - Matrix-free implementations store matrix implicitly
- We have designed and analyzed parallel algorithms based on matrix-free implementations

Summary (3)

- Ghost points store copies of values held by other processes
- Explored increasing number of ghost points and replicating computation in order to reduce number of message exchanges
- Optimal number of ghost points depends on characteristics of parallel system