McGill

ECSE 420
Lab 2: Finite Elements

TA: Loren Lugosch

October 2016

Outline

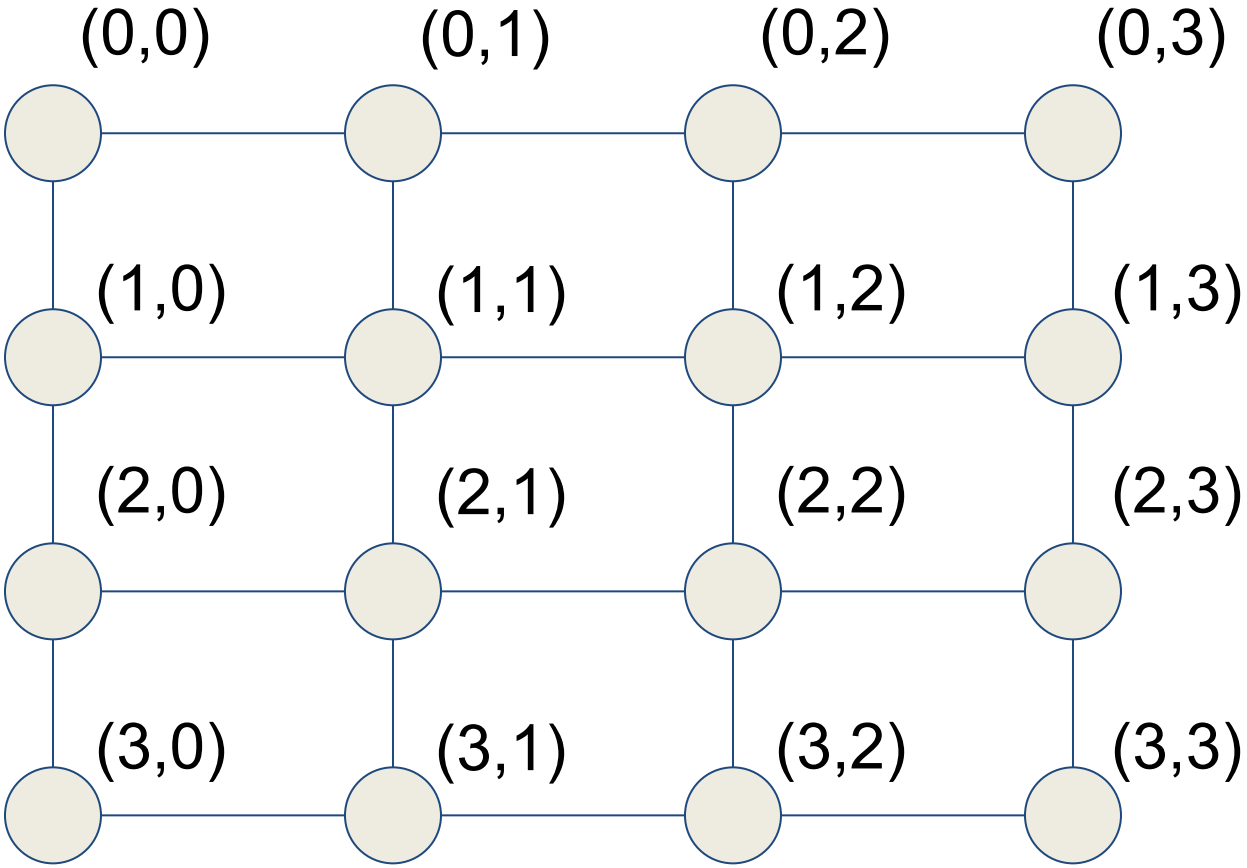
- Overview of finite element method
- Lab 2 info

Finite Elements

- We can simulate a large or complex object by
  - breaking it into smaller, simpler objects (“finite elements”)
  - simulating the finite elements independently
  - sending data between elements as needed to model interaction
- Example: ocean simulator from Assignment 2
- This lab: simulate drum using grid of finite elements

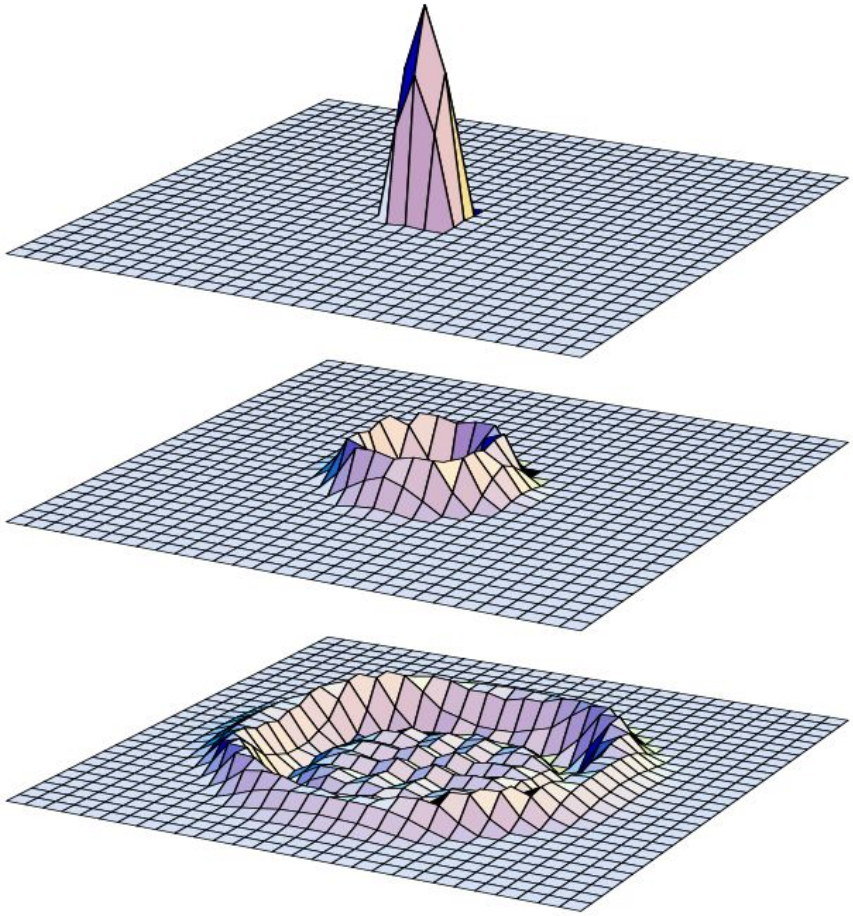
Finite Elements

- We will model a drum as elements of tensile material connected in a square grid



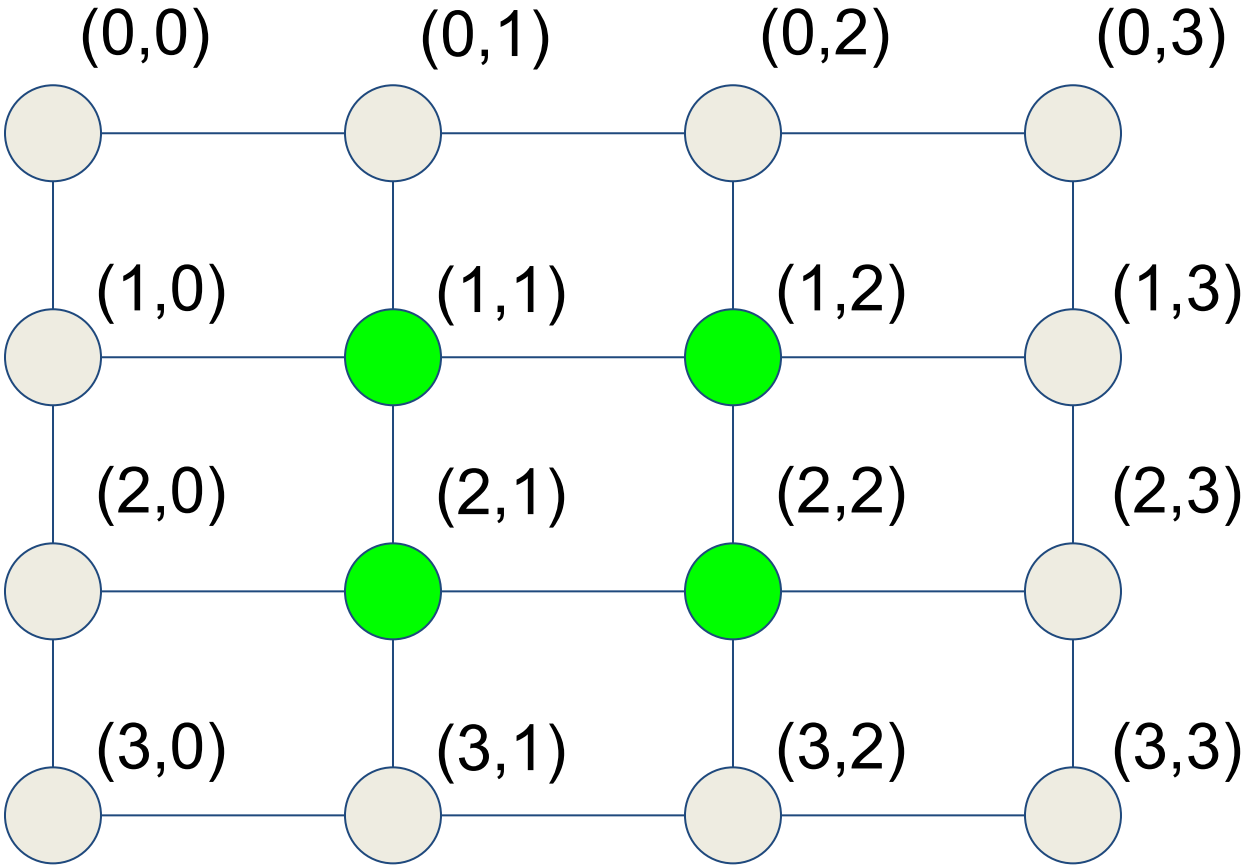
Finite Elements

- Response of grid to impulse at center:



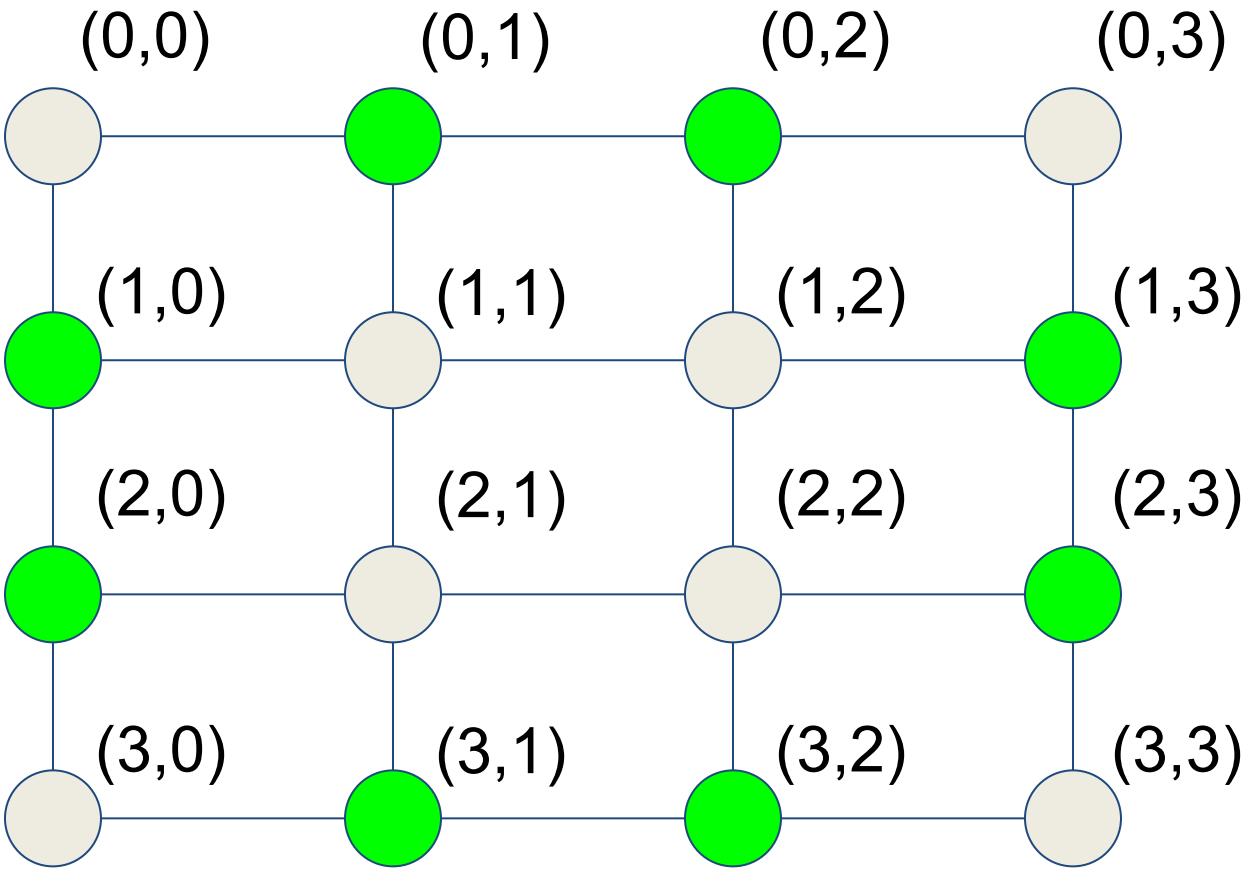
Finite Elements

- At each time step, update vertical displacement of each element in **interior**:



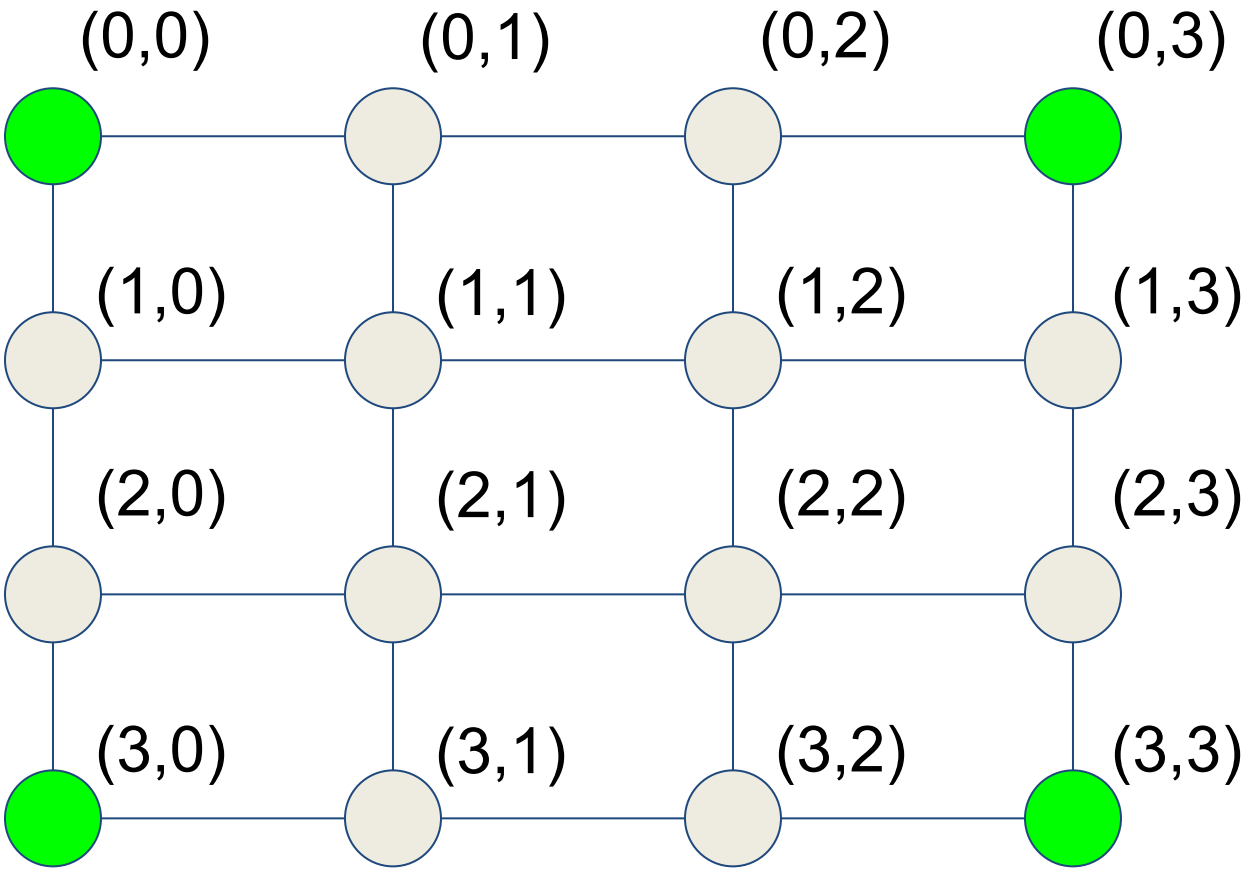
Finite Elements

- Then, update elements in the **sides**:



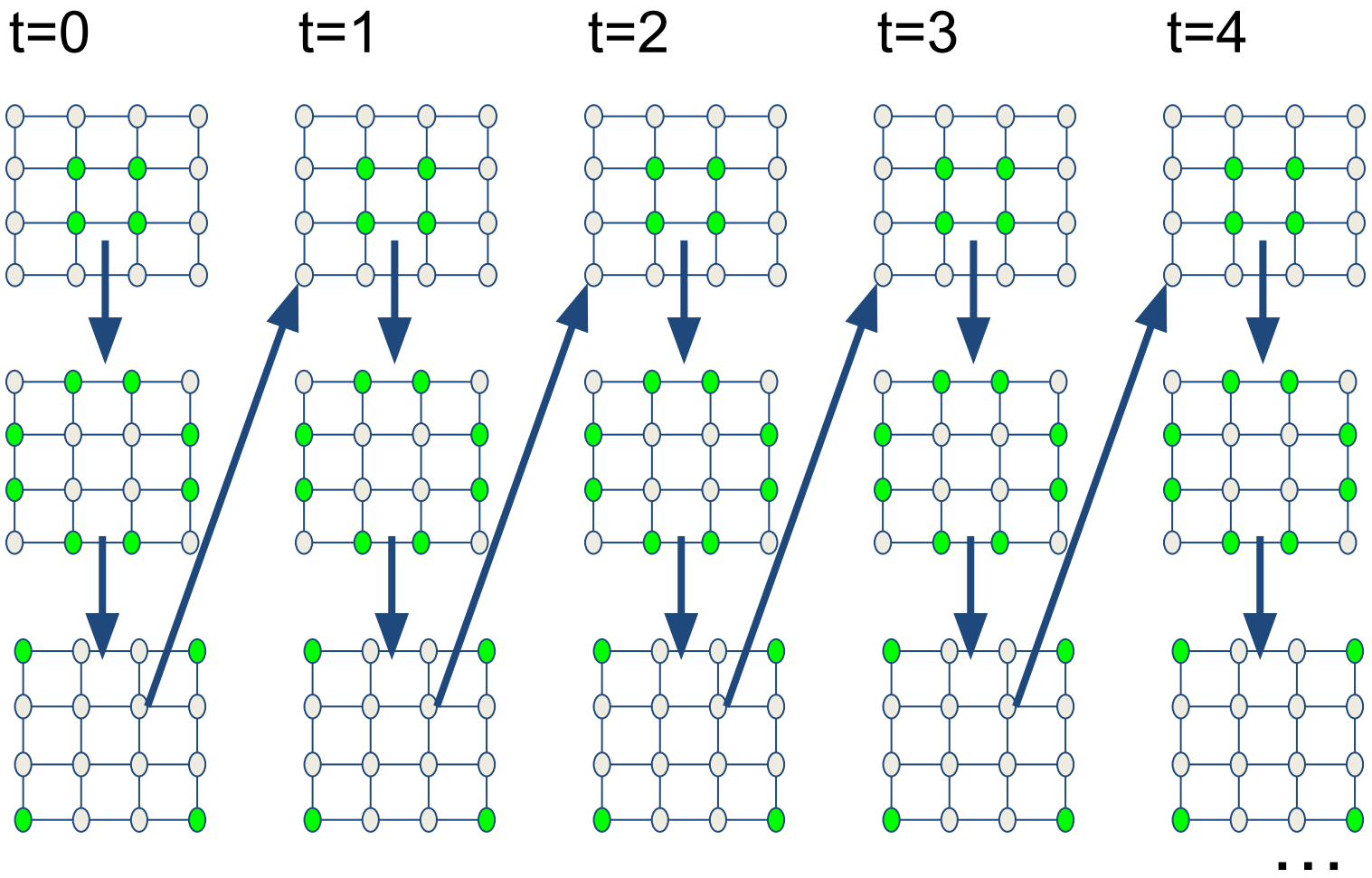
Finite Elements

- Finally, update elements in the **corners**:



Finite Elements

- Once all displacements are updated, repeat:



Finite Elements

- **Interior** update equation:

$$u(i,j) = \frac{\rho(u1(i-1,j) + u1(i+1,j) + u1(i,j-1) + u1(i,j+1) - 4u1(i,j)) + 2u1(i,j) - (1-\eta)u2(i,j)}{1+\eta}$$

$$1 < i < N-1, 1 < j < N-1$$

- where **u** = vertical displacement
- **u1** = previous displacement
- **u2** = previous previous displacement
- **ρ (rho)** = physical constant related to pitch
- **η (eta)** = physical constant related to damping

Finite Elements

- Each element of u depends only on u1 and u2, and not on other elements of u
- Each element can be updated in parallel!

Finite Elements

- Equation breakdown:

$$u(i,j) = \frac{\rho(u1(i-1,j) + u1(i+1,j) + u1(i,j-1) + u1(i,j+1) - 4u1(i,j)) + 2u1(i,j) - (1-\eta)u2(i,j)}{1+\eta}$$

- Sum of (top, bottom, left, right) neighbors' previous displacements and this element's previous displacement multiplied by rho
- This element's previous displacement multiplied by 2
- This element's previous previous displacement multiplied by (1-eta)

Finite Elements

- **Side** update equations:

$$u(0,i) = Gu(1,i)$$

$$u(N-1,i) = Gu(N-2,i)$$

$$u(i,0) = Gu(i,1)$$

$$u(i,N-1) = Gu(i,N-2)$$

$$1 < i < N-1$$

- where **G** = boundary gain (whether the edge of the drum is taut or loose)

Finite Elements

- Note: side u element updates depend on interior u elements (not u1 or u2)
  $\longrightarrow$ must happen after interior points updated

Finite Elements

- **Corner** update equations:

$$u(0,0) = Gu(1,0)$$

$$u(N-1,0) = Gu(N-2,0)$$

$$u(0,N-1) = Gu(0,N-2)$$

$$u(N-1,N-1) = Gu(N-1,N-2)$$

- Must happen after side points updated

Finite Elements

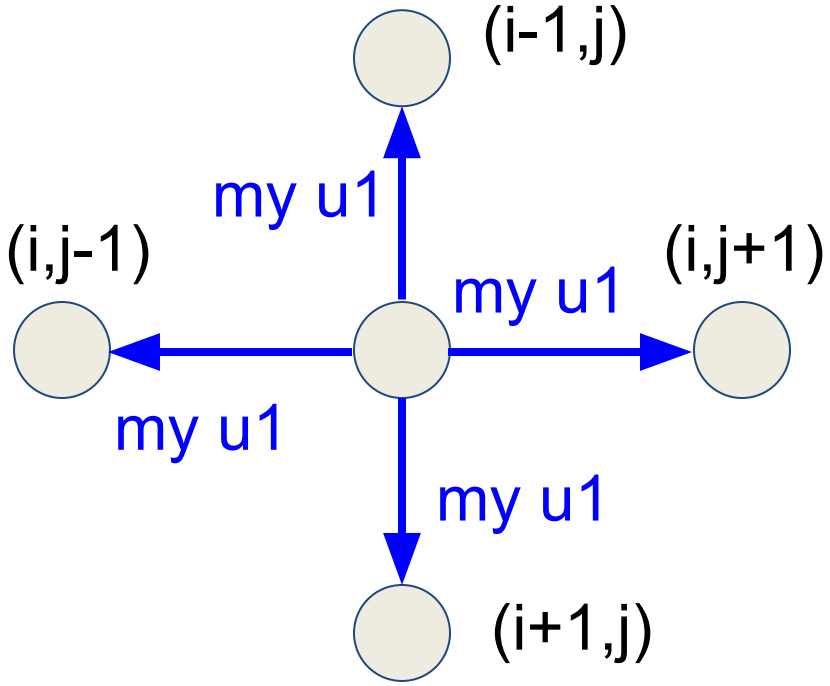
- Example using a 4x4 grid for T=2 iterations:

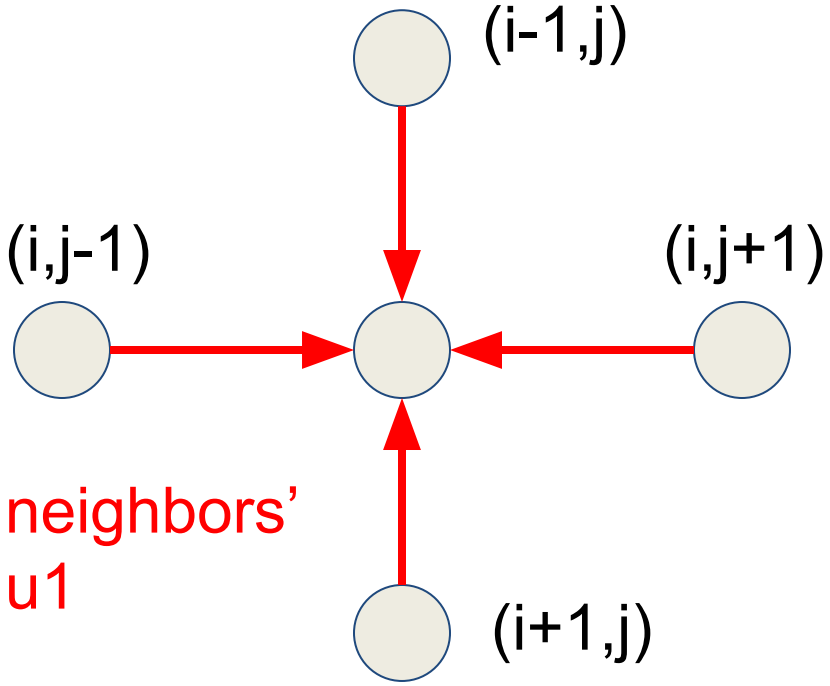
Lab 2

- Use MPI to parallelize computation and send messages between elements
- Part 1:
  - Small grid
  - One MPI process per element
- Part 2:
  - Large grid
  - Multiple elements assigned to each MPI process

Lab 2, Part 1

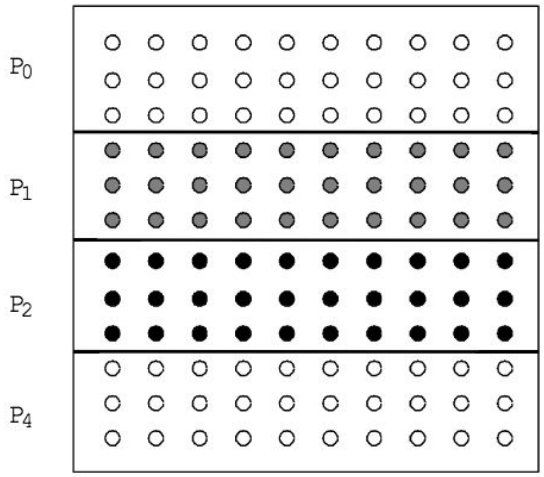
- Use MPI communication to send u1 to neighbors, receive u1, update u, repeat

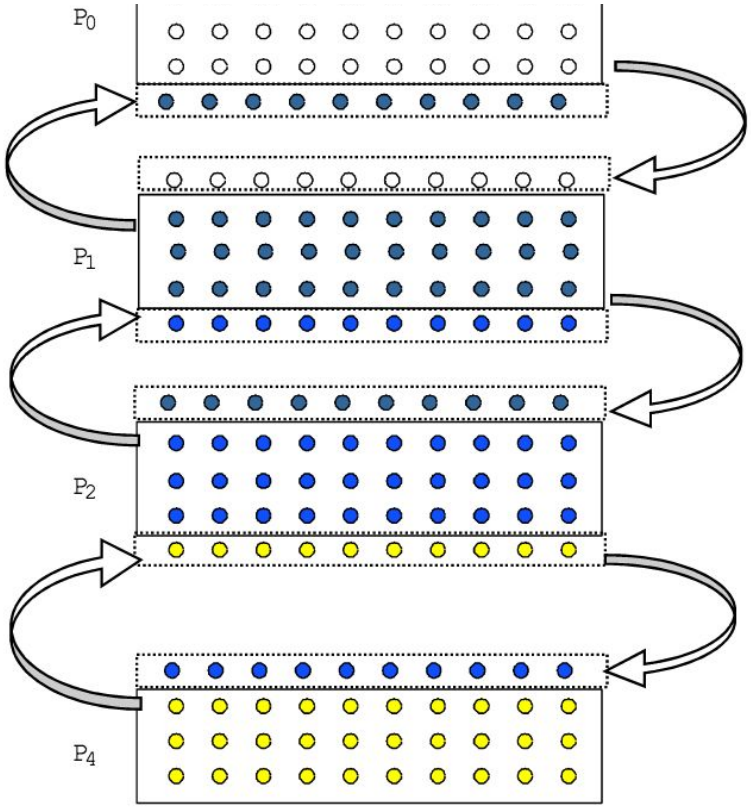




Lab 2, Part 2

- Use row decomposition:





Data partition allocated per processor

Add ghost rows to hold boundary data

Send edges to neighbors

Receive into ghost rows

Compute as in sequential program