Computational Science on Many-Core Architectures

360.252

Josef Weinbub
Karl Rupp

Institute for Microelectronics, TU Wien
http://www.iue.tuwien.ac.at/

Zoom Channel 621 2711 2607
Wednesday, November 29, 2023

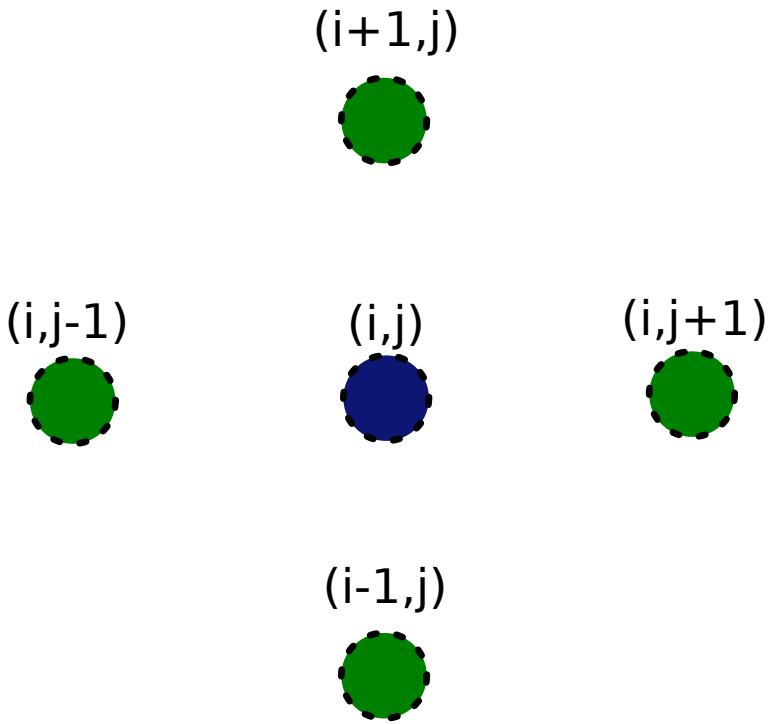
Finite Differences

Discrete Poisson Equation $-\Delta_h u(x,y) = f_h(x,y)$

- Rectangular grid
- Assume homogeneous grid spacing h
- Discretize each coordinate direction separately

$$
\begin{aligned}
\Delta u(x,y) &= u_{xx}(x,y) + u_{yy}(x,y) \\
&\approx \frac{u(x-h,y) - 2u(x,y) + u(x+h,y)}{h^2} + \frac{u(x,y-h) - 2u(x,y) + u(x,y+h)}{h^2} \\
&= \frac{u(x-h,y) + u(x+h,y) - 4u(x,y) + u(x,y-h) + u(x,y+h)}{h^2} \\
&=: \Delta_h u(x,y)
\end{aligned}
$$

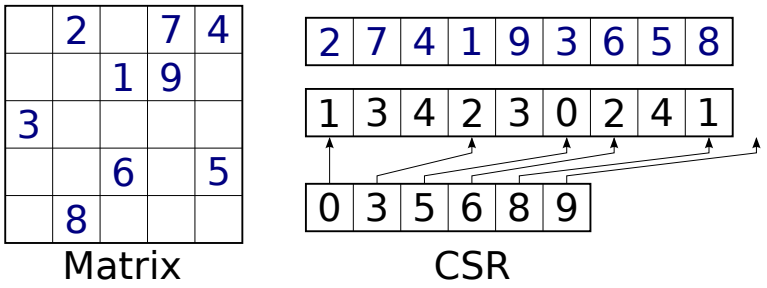
Finite Differences



Finite Differences

System Matrix Structure

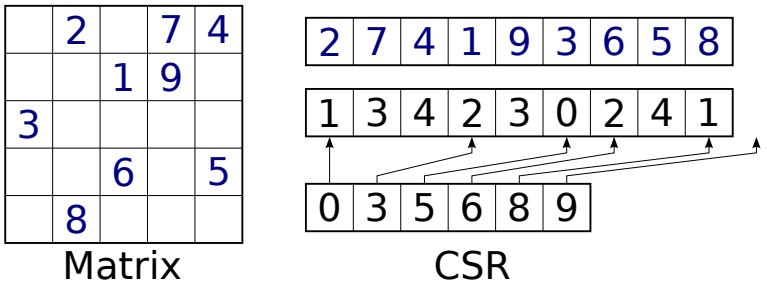
- Interior nodes have 5 nonzero entries
- Boundary nodes have 1 to 4 nonzero entries
- Problem: CSR format requires offset index for each row



Finite Differences

System Matrix Structure

- Interior nodes have 5 nonzero entries
- Boundary nodes have 1 to 4 nonzero entries
- Problem: CSR format requires offset index for each row



Generic GPU matrix assembly skeleton

1. Count the nonzero entries for each row
2. Deduce offsets for CSR
3. Populate column and values arrays

Finite Differences

Step 1: Count Nonzeros for an $N \times M$ grid

```
__kernel__ void count_nnz(int *row_offsets, int N, ...) {
for (int row = blockDim.x * blockIdx.x + threadIdx.x;
          row < N*M;
          row += gridDim.x * blockDim.x)
    {
        int nnz_for_this_node = 1;
        int i = row / N;
        int j = row % N;

        if (i > 0)   nnz_for_this_node += 1;
        if (j > 0)   nnz_for_this_node += 1;
        if (i < N-1) nnz_for_this_node += 1;
        if (j < M-1) nnz_for_this_node += 1;

        row_offsets[row] = nnz_for_this_node;
      }
}
```

Parallel Primitives

Prefix Sum

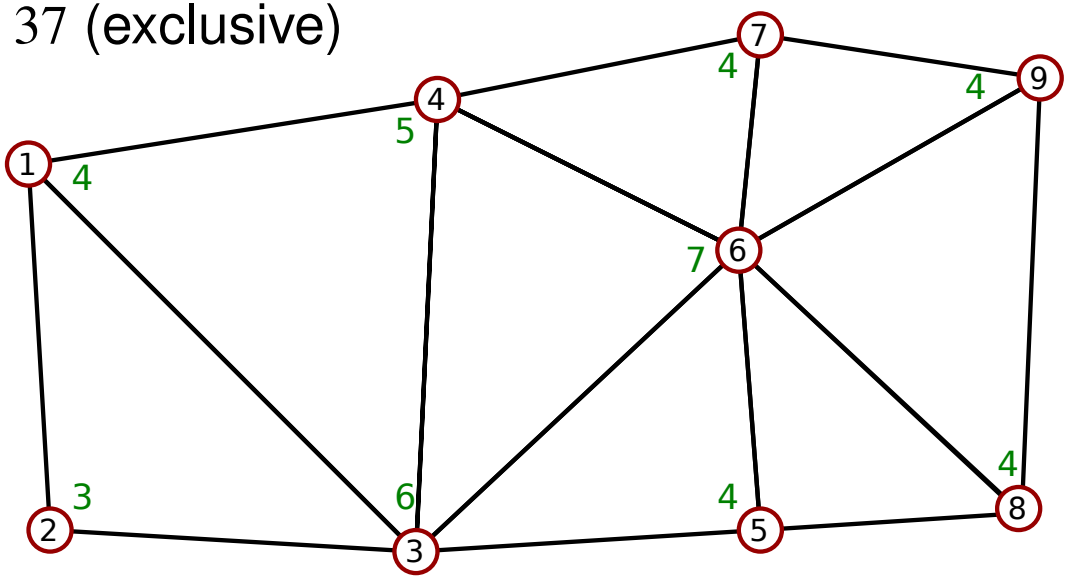
- Inclusive: Determine $y_i = \sum_{k=1}^{i} x_k$
- Exclusive: Determine $y_i = \sum_{k=1}^{i-1} x_k$, $y_1 = 0$

Example

- x: 4, 3, 6, 5, 4, 7, 4, 4, 4
- y: 4, 7, 13, 18, 22, 29, 33, 37, 41 (inclusive)
- y: 0, 4, 7, 13, 18, 22, 29, 33, 37 (exclusive)

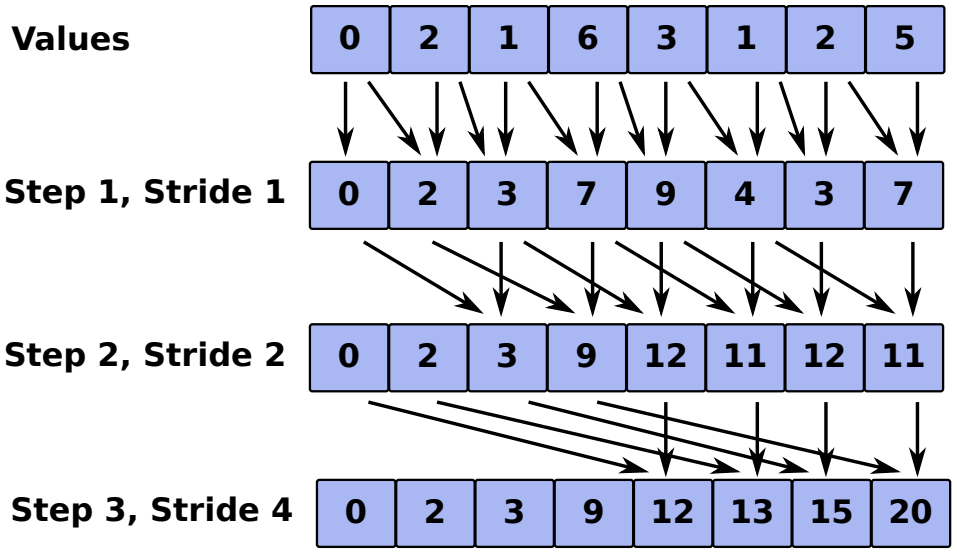
Applications

- Sparse matrix setup
- Graph algorithms



Parallel Primitives

Prefix Sum Implementation




```
for(int stride = 1; stride < blockDim.x; stride *= 2)
{
  __syncthreads();
  shared_m[threadIdx.x] = my_value;
  __syncthreads();
  if (threadIdx.x >= stride)
    my_value += shared_m[threadIdx.x - stride];
}
__syncthreads();
shared_m[threadIdx.x] = my_value;
```

Finite Differences

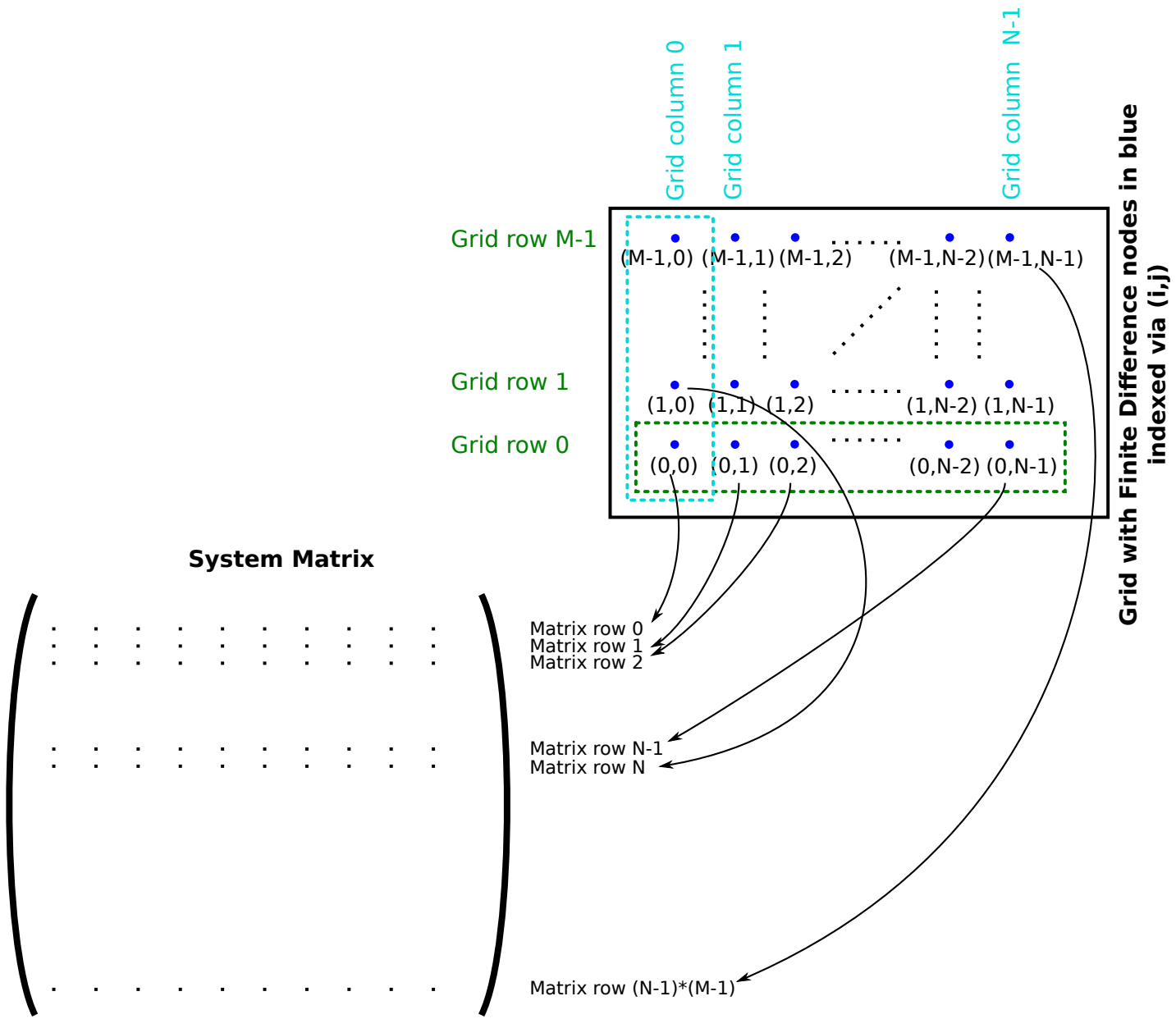
Step 3: Assembly for an $N \times M$ grid

```
__kernel__ void assembleA(int *row_offsets, int N, ...) {
  for (int row = blockDim.x * blockIdx.x + threadIdx.x;
            row < N*M;
            row += gridDim.x * blockDim.x) {
      int i = row / N;
      int j = row % N;
      int this_row_offset = row_offsets[row];

      // diagonal entry
      col_indices[this_row_offset] = i * N + j;
      values[this_row_offset] = 4;
      this_row_offset += 1;

      if (i > 0) { // bottom neighbor
        col_indices[this_row_offset] = (i-1) * N + j;
        values[this_row_offset] = -1;
        this_row_offset += 1;
      }
      if (j > 0)   { /* similarly */ }
      if (i < N-1) { /* similarly */ }
      if (j < M-1) { /* similarly */ }
}   }
```

Finite Differences



Other Discretizations

Finite Volumes

- Iterations over vertices: Similar to finite differences
- Beware of advanced schemes (depends)

Other Discretizations

Finite Volumes

- Iterations over vertices: Similar to finite differences
- Beware of advanced schemes (depends)

Finite Elements

- Iteration over elements (cells)
- BUT: write matrix entries associated with vertices
- $\Rightarrow$ Avoid race conditions
- More preparation required before we can address this