SpMV in Compressed Sparse Row (CSR) Format

CSR format is one of many possibilities

Matrix-vector multiply kernel: \( y(i) \leftarrow y(i) + A(i,j) \cdot x(j) \)

for each row \( i \)
   for \( k=\text{ptr}[i] \) to \( \text{ptr}[i+1] \) do
      \( y[i] = y[i] + \text{val}[k] \cdot x[\text{ind}[k]] \)
Parallel Sparse Matrix-vector multiplication

- \( y = A \times x \), where \( A \) is a sparse \( n \times n \) matrix

**Questions**

- which processors store
  - \( y[i], x[i], \) and \( A[i,j] \)
- which processors compute
  - \( y[i] = \sum_{j=1}^{n} A[i,j] \times x[j] \)
  - \( = (\text{row } i \text{ of } A) \times x \)  
    … a sparse dot product

**Partitioning**

- Partition index set \( \{1, \ldots, n\} = N_1 \cup N_2 \cup \ldots \cup N_p. \)
- For all \( i \) in \( N_k \), Processor \( k \) stores \( y[i], x[i], \) and row \( i \) of \( A \)
- For all \( i \) in \( N_k \), Processor \( k \) computes \( y[i] = (\text{row } i \text{ of } A) \times x \)
  - “owner computes” rule: Processor \( k \) compute the \( y[i] \)s it owns.

May require communication
“Ideal” matrix structure for parallelism: block diagonal
- $p$ (number of processors) blocks, can all be computed locally.
- If no non-zeros outside these blocks, no communication needed

Can we reorder the rows/columns to get close to this?
- Most nonzeros in diagonal blocks, few outside
Goals of Reordering

- **Performance goals**
  - balance load (how is load measured?).
    - Approx equal number of nonzeros (not necessarily rows)
  - balance storage (how much does each processor store?).
    - Approx equal number of nonzeros
  - minimize communication (how much is communicated?).
    - Minimize nonzeros outside diagonal blocks
    - Related optimization criterion is to move nonzeros near diagonal
  - improve register and cache re-use
    - Group nonzeros in small vertical blocks so source (x) elements loaded into cache or registers may be reused (temporal locality)
    - Group nonzeros in small horizontal blocks so nearby source (x) elements in the cache may be used (spatial locality)

- **Other algorithms reorder for other reasons**
  - Reduce # nonzeros in matrix after Gaussian elimination
  - Improve numerical stability
Graph Partitioning and Sparse Matrices

- Relationship between matrix and graph

- Edges in the graph are nonzero in the matrix: here the matrix is symmetric (edges are unordered) and weights are equal (1)
- If divided over 3 procs, there are 14 nonzeros outside the diagonal blocks, which represent the 7 (bidirectional) edges
Graph Partitioning and Sparse Matrices

- Relationship between matrix and graph

- A “good” partition of the graph has
  - equal (weighted) number of nodes in each part (load and storage balance).
  - minimum number of edges crossing between (minimize communication).
  - Reorder the rows/columns by putting all nodes in one partition together.