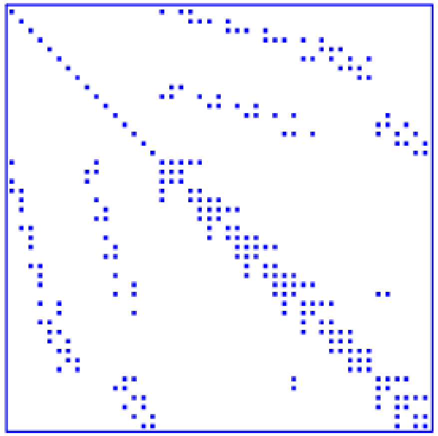
SPARSE MATRICES

- **Introduction to sparse matrices**
- **General intro: solving linear systems**
- **Graph representation of sparse matrices**
- **Computing with sparse matrices: Matrix-vector produts**
- **Graph partitioning, Graph Laplaceans**

What are sparse matrices?



Pattern of a small sparse matrix

➤ For all practical purposes: an $m \times n$ matrix is sparse if it has $O(\min(m, n))$ nonzero entries.

➤ This means roughly a constant number of nonzero entries per row and column. Issue: This definition excludes a large class of matrices that have $O(\log(n))$ nonzero entries per row.

➤ Other definitions use a slow growth of nonzero entries with respect to n or m.

"..matrices that allow special techniques to take advantage of the large number of zero elements." (J. Wilkinson)

A few applications which lead to sparse matrices:

Structural Engineering, Computational Fluid Dynamics, Reservoir simulation, Electrical Networks, optimization, Google Page rank, information retrieval (LSI), circuit similation, device simulation, ...

Goal of Sparse Matrix Techniques

➤ To perform standard matrix computations economically i.e., without storing the zeros of the matrix.

Example: To add two square dense matrices of size n requires $O(n^2)$ operations. To add two sparse matrices A and B requires $O(nnz(A) + nnz(B))$ where $nnz(X)$ = number of nonzero elements of a matrix X.

➤ For typical Finite Element /Finite difference matrices, number of nonzero elements is $O(n)$.

Remark: A^{-1} is usually dense, but L and U in the LU factorization may be reasonably sparse (if a good technique is used)

Graph Representations of Sparse Matrices

➤ Graph theory is a fundamental tool in sparse matrix techniques.

DEFINITION. A graph G is defined as a pair of sets $G = (V, E)$ with $E \subset V \times V$. So G represents a binary relation. The graph is undirected if the binary relation is reflexive. It is directed otherwise. V is the vertex set and E is the edge set.

Example: Given the numbers 5, 3, 9, 15, 16, show the two graphs representing the relations

R1: Either $x < y$ or y divides x.

R2: x and y are congruent modulo 3. [mod(x,3) = mod(y,3)]

➤ Adjacency Graph $G = (V, E)$ of an $n \times n$ matrix A :

- Vertices $V = \{1, 2, ..., n\}$.
- Edges $E = \{(i, j) | a_{ij} \neq 0\}$.

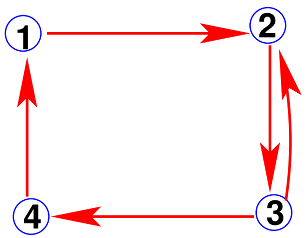
➤ Often self-loops (i, i) are not represented [because they are always there]

➤ Graph is undirected if the matrix has a symmetric structure:

$$a_{ij} \neq 0 \quad \text{iff} \quad a_{ji} \neq 0.$$

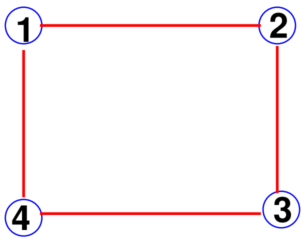
Example: (directed graph)

$$\begin{bmatrix} & \star & & \\ & & \star & \\ & \star & & \star \\ \star & & & \end{bmatrix}$$



Example: (undirected graph)

$$\begin{bmatrix} & \star & & \star \\ \star & & \star & \\ & \star & & \star \\ \star & & \star & \end{bmatrix}$$



✍1 Adjacency graph of:

$$A = \begin{pmatrix} \star & \star & & & \star & \\ \star & \star & \star & & & \star \\ & \star & \star & & & \\ & & & \star & \star & \\ \star & & & \star & \star & \star \\ & \star & & & \star & \star \end{pmatrix}.$$

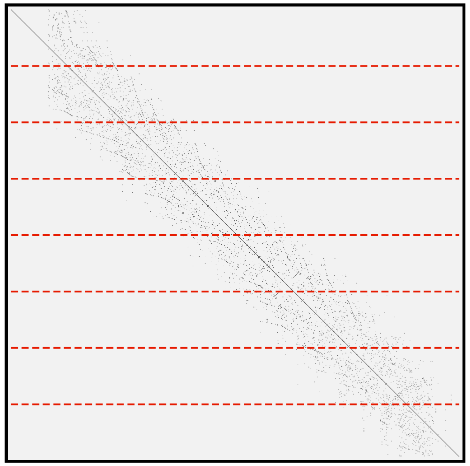
✍2 Graph of a tridiagonal matrix? Of a dense matrix?

✍3 Recall what a star graph is. Show a matrix whose graph is a star graph. Consider two situations: Case when center node is labeled first and case when it is labeled last.

Distributed Sparse Systems

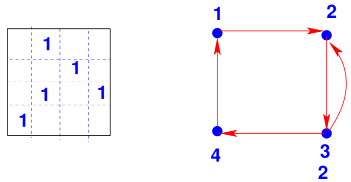
➤ Simple illustration: Block assignment. Assign equation i and unknown i to a given 'process'

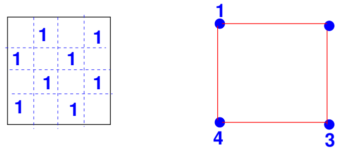
➤ Naive partitioning - won't work well in practice



➤ Best idea is to use the adjacency graph of A:

Vertices $= \{1, 2, \cdots, n\}$;
Edges: $i \rightarrow j$ iff $a_{ij} \neq 0$





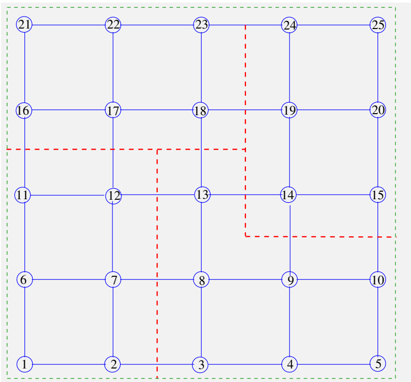
Graph partitioning problem:

- Want a partition of the vertices of the graph so that

(1) partitions have $\sim$ the same sizes

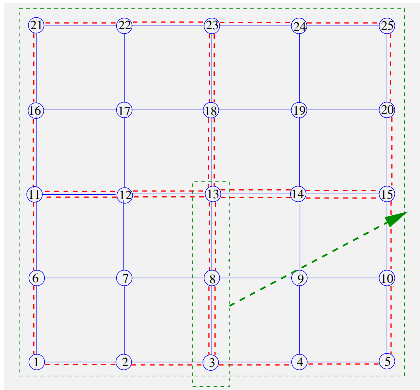
(2) interfaces are small in size

General Partitioning of a sparse linear system



$S_1 = \{1, 2, 6, 7, 11, 12\}$: This means equations and unknowns 1, 2, 3, 6, 7, 11, 12 are assigned to Domain 1.
$S_2 = \{3, 4, 5, 8, 9, 10, 13\}$
$S_3 = \{16, 17, 18, 21, 22, 23\}$
$S_4 = \{14, 15, 19, 20, 24, 25\}$

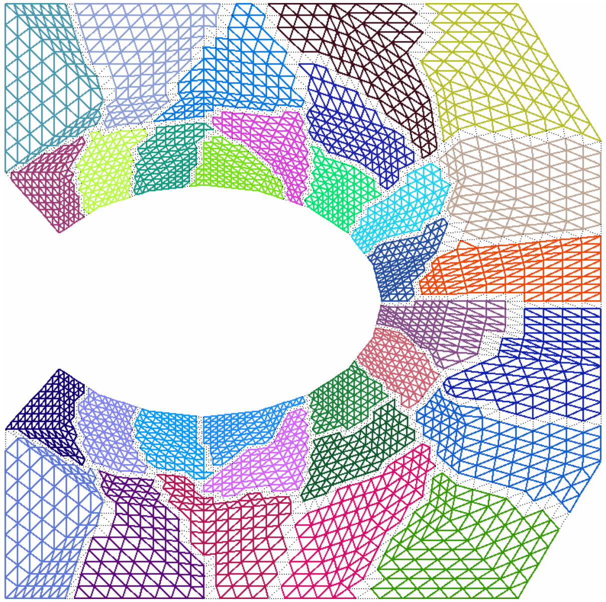
Alternative: Map elements / edges rather than vertices



Equations/unknowns 3, 8, 13 shared by 2 domains. From distributed sparse matrix viewpoint this is an overlap of one layer

➤ Partitioners : Metis, Chaco, Scotch, Zoltan, H-Metis, PaToH, ..

➤ Standard dual objective: "minimize" communication + "balance" partition sizes

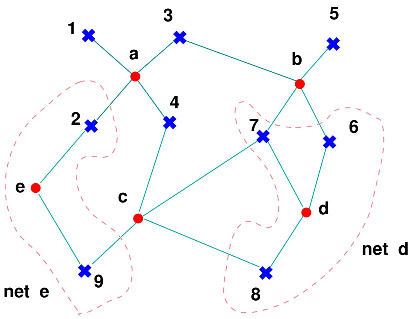


A few words about hypergraphs

- Hypergraphs are very general.. Ideas borrowed from VLSI work
- Main motivation: to better represent communication volumes when partitioning a graph. Standard models face many limitations
- Hypergraphs can better express complex graph partitioning problems and provide better solutions.
- Example: completely nonsymmetric patterns ... Even rectangular matrices

Main idea: an edge now can consist of a small set of more than 2 vertices. Most common example: edge = column indices of nonzero entries of a row of a matrix. See next example.

Example: $V = \{1, \ldots, 9\}$ and $E = \{a, \ldots, e\}$ with $a = \{1, 2, 3, 4\}$, $b = \{3, 5, 6, 7\}$, $c = \{4, 7, 8, 9\}$, $d = \{6, 7, 8\}$, and $e = \{2, 9\}$



Boolean matrix:

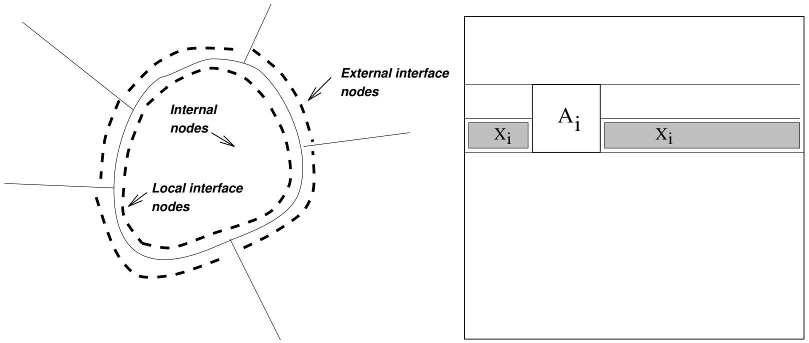
$$A = \begin{array}{c} \begin{array}{ccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 \end{array} \\ \left[\begin{array}{ccccccccc} 1 & 1 & 1 & 1 & & & & & \\ & & 1 & & 1 & 1 & 1 & & \\ & & & 1 & & & 1 & 1 & 1 \\ & & & & & 1 & 1 & 1 & \\ & 1 & & & & & & & 1 \end{array}\right] \end{array} \begin{array}{c} \\ a \\ b \\ c \\ d \\ e \end{array}$$

Distributed Sparse matrices (continued)

- Once a good partitioning is found, questions are:

1. How to represent this partitioning?
2. What is a good data structure for representing distributed sparse matrices?
3. How to set up the various "local objects" (matrices, vectors, ..)
4. What can be done to prepare for communication that will be required during execution?

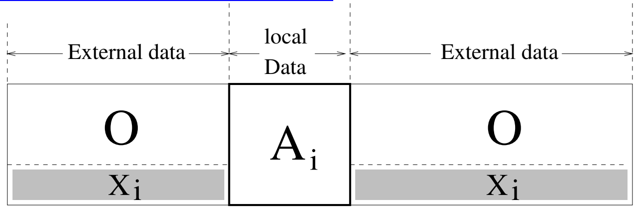
Two views of a distributed sparse matrix



➤ Local interface variables always ordered last.

➤ Need: 1) to set up the various "local objects". 2) Preprocessing to prepare for communications needed during iteration?

Local view of distributed matrix:

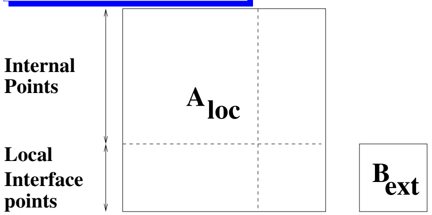


The local system:

$$\underbrace{\begin{pmatrix} B_i & F_i \\ E_i & C_i \end{pmatrix}}_{A_i} \begin{pmatrix} u_i \\ y_i \end{pmatrix} + \underbrace{\begin{pmatrix} 0 \\ \sum_{j \in N_i} E_{ij} y_j \end{pmatrix}}_{y_{ext}} = \begin{pmatrix} f_i \\ g_i \end{pmatrix}$$

➤ u_i : Internal variables; y_i : Interface variables

The local matrix:



The local matrix consists of 2 parts: a part ('A_{loc}') which acts on local data and another ('B_{ext}') which acts on remote data.

➤ Once the partitioning is available these parts must be identified and built locally..

➤ In finite elements, assembly is a local process.

➤ How to perform a matrix vector product? [needed by iterative schemes?]

Distributed Sparse Matrix-Vector Product Kernel

Algorithm:

1. Communicate: exchange boundary data.

   Scatter x_{bound} to neighbors - Gather x_{ext} from neighbors

2. Local matrix – vector product

$$\boxed{y = A_{loc} x_{loc}}$$

3. External matrix – vector product

$$\boxed{y = y + B_{ext} x_{ext}}$$

NOTE: 1 and 2 are independent and can be overlapped.

Distributed Sparse Matrix-Vector Product

Main part of the code:

```
  call MSG_bdx_send(nloc,x,y,nproc,proc,ix,ipr,ptrn,ierr)
c
c do local matrix-vector product for local points
c
  call amux(nloc,x,y,aloc,jaloc,ialoc)
c
c receive the boundary information
c
  call MSG_bdx_receive(nloc,x,y,nproc,proc,ix,ipr,
     *     ptrn,ierr)
c
c do local matrix-vector product  for external points
c
  nrow = nloc - nbnd + 1
  call amux1(nrow,x,y(nbnd),aloc,jaloc,ialoc(nloc+1))
c
  return
```


The local exchange information

➤ List of adjacent processors (or subdomains)

➤ For each of these processors, lists of boundary nodes to be sent / received to /from adj. PE's.

➤ The receiving processor must have a matrix ordered consistently with the order in which data is received.

Requirements

➤ The 'set-up' routines should handle overlapping

➤ Should use minimal storage (only arrays of size nloc allowed).

Main Operations in a typical iterative solver

1. Saxpy's – local operation – no communication
2. Dot products – reduction operation (global)
3. Matrix-vector products – local operation – local communication
4. Preconditioning operations – locality varies.

Distributed Dot Product

```
/*-------------------- call blas1 function   */
   tloc = DDOT(n, x, incx, y, incy);
/*-------------------- call global reduction */
   MPI_Allreduce(&tloc,&ro,1,MPI_DOUBLE,MPI_SUM,comm);
```


Graph Laplaceans - Definition

➤ "Laplace-type" matrices associated with general undirected graphs – useful in many applications

➤ Given a graph $G = (V, E)$ define

- A matrix W of weights w_{ij} for each edge
- Assume $\boxed{w_{ij} \geq 0,, \;\; w_{ii} = 0, \text{ and } w_{ij} = w_{ji} \; \forall (i, j)}$
- The diagonal matrix $D = diag(d_i)$ with $d_i = \sum_{j \neq i} w_{ij}$

➤ Corresponding *graph Laplacean* of G is:

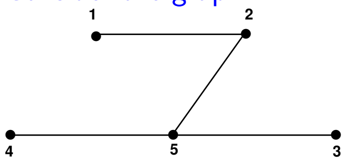
$$L = D - W$$

➤ Gershgorin's theorem $\rightarrow$ L is positive semidefinite.

- Simplest case:

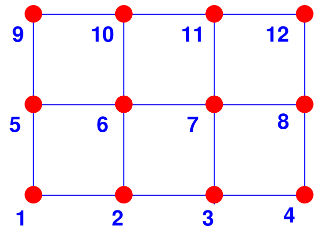
$$w_{ij} = \begin{cases} 1 \text{ if } (i,j) \in E \& i \neq j \\ 0 \quad \text{else} \end{cases} \qquad D = \text{diag}\left[d_i = \sum_{j \neq i} w_{ij}\right]$$

Example:
Consider the graph



$$L = \begin{pmatrix} 1 & -1 & 0 & 0 & 0 \\ -1 & 2 & 0 & 0 & -1 \\ 0 & 0 & 1 & 0 & -1 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & -1 & -1 & -1 & 3 \end{pmatrix}$$

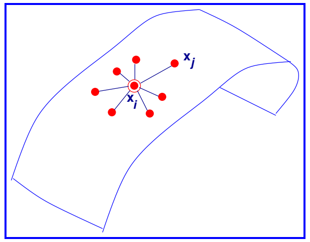
✍D4 Define the graph Laplacean for the graph associated with the simple mesh shown next. [use the simple weights of 0 or 1]. What is the difference with the discretization of the Laplace operator for case when mesh is the same as this graph?



Proposition:
(i) L is symmetric semi-positive definite.
(ii) L is singular with $\mathbf{1}$ as a null vector.
(iii) If G is connected, then $\text{Null}(L) = \text{span}\{\mathbf{1}\}$
(iv) If G has $k > 1$ connected components $G_1, G_2, \cdots, G_k$, then the nullity of L is k and $\text{Null}(L)$ is spanned by the vectors $z^{(j)}$, $j = 1, \cdots, k$ defined by:

$$(z^{(j)})_i = \begin{cases} 1 \text{ if } i \in G_j \\ 0 \text{ if not.} \end{cases}$$

A few properties of graph Laplaceans



Strong relation between $x^T L x$ and local distances between entries of x

- Let L = any matrix s.t. $L = D - W$, with $D = diag(d_i)$ and

$$w_{ij} \geq 0, \qquad d_i = \sum_{j \neq i} w_{ij}$$

Property : for any $x \in \mathbb{R}^n$:

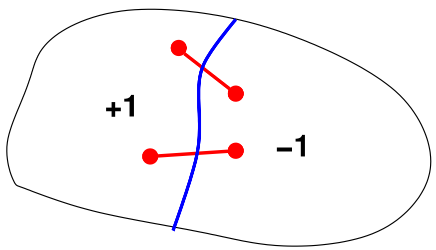
$$x^\top L x = \frac{1}{2} \sum_{i,j} w_{ij} |x_i - x_j|^2$$

Property : (Graph partitioning) Consider situation when $w_{ij} \in \{0, 1\}$. If x is a vector of signs (± 1) then

$$x^\top L x = 4 \times (\text{'number of edge cuts'})$$

edge-cut = pair (i, j) with $x_i \neq x_j$

- Consequence: Can be used to partition graphs



Goal: *"minimze number of edge-cuts while domain sizes are (about) the same"*

- Would like to minimize (Lx, x) subject to $x \in \{-1, 1\}^n$ and $e^T x = 0$ [balanced sets]
- WIl solve a relaxed form of this problem

✍5 What if we replace x by a vector of ones (representing one partition) and zeros (representing the other)?

✍6 Let x be any vector and $y = x + \alpha \mathbf{1}$ and L a graph Laplacean. Compare $x^T L x$ with $y^T L y$).

- Consider any symmetric (real) matrix A with eigenvalues $\lambda_1 \le \lambda_2 \le \cdots \le \lambda_n$ and eigenvectors $u_1, \cdots, u_n$
- Recall that: (Min reached for $x = u_1$)

$$\min_{x \in \mathbb{R}^n} \frac{(Ax, x)}{(x, x)} = \lambda_1$$

- In addition: (Min reached for $x = u_2$)

$$\min_{x \perp u_1} \frac{(Ax, x)}{(x, x)} = \lambda_2$$

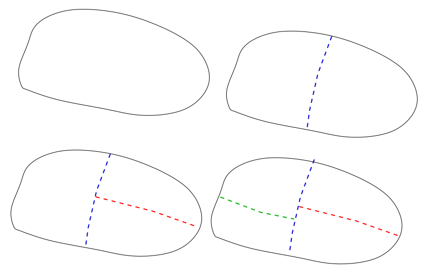
- For a graph Laplacean $u_1 = \mathbf{1}$ = vector of all ones and
- ...vector u_2 is called the Fiedler vector. It solves a relaxed form of the problem -

$$\min_{x \in \{-1,1\}^n;\ \mathbf{1}^T x = 0} \frac{(Lx, x)}{(x, x)} \to \min_{x \in \mathbb{R}^n;\ \mathbf{1}^T x = 0} \frac{(Lx, x)}{(x, x)}$$

- Define $v = u_2$ then $lab = sign(v - med(v))$

Recursive Spectral Bisection

1. Form graph Laplacean
2. Partition graph in 2 based on Fielder vector
3. Partition largest subgraph in two recursively ...
4. ... Until the desired number of partitions is reached



Three approaches to graph partitioning:

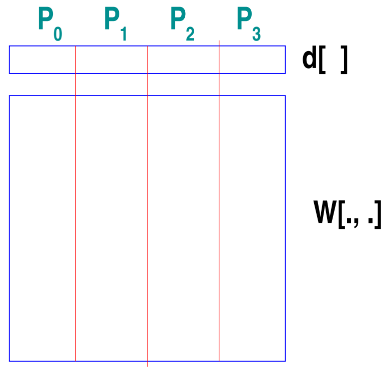
1. Spectral methods - Just seen + add Recursive Spectral Bisection.
2. Geometric techniques. Coordinates are required. [Houstis & Rice et al., Miller, Vavasis, Teng et al.]
3. Graph Theory techniques – multilevel,... [use graph, but no coordinates]
   - Currently best known technique is Metis (multi-level algorithm)
   - Simplest idea: Recursive Graph Bisection; Nested dissection (George & Liu, 1980; Liu 1992]
   - Advantages: simplicity – no coordinates required

Application: Back to Dijkstra's algorithm

➤ Recall the following picture – (go back to Graphs)

Decomposition:

➤ Split Distance array in p parts, uniformely.
➤ Split weight matrix column-wise in p blocks
➤ First: Use the exact same partitioning (naive) for simplicity.
➤ Then use a Domain-Decomposition –type partitioning.



✍D7 Figure out how to adapt the algorithm to the sparse case. [Hint: the update operation is very similar to a parallel sparse matvec]