Lecture 18: Sparse Direct Methods

David Bindel

1 Nov 2011

Logistics

- Project 2 in
  - Can submit up to next Monday with 1 point penalty...
  - ... but be careful it doesn't pile up against other work
- Project 3 posted (parallel all-pairs shortest paths)

More life lessons from Project 2?

- Start early so you have time to get stuck and unstuck.
- Understand when rounding is a culprit (and when not).
- Test frequently as you work.
- Check against a slow, naive, obvious calculation.
- Synchronization is expensive!

Enter Project 3

The all pairs shortest path problem:

Input: An adjacency matrix for an unweighted graph:

$$A_{ij} = \begin{cases} 1, & \text{edge between } i \text{ and } j \\ 0, & \text{otherwise} \end{cases}$$

Output: A distance matrix

$$L_{ij} = \text{ length of shortest path from } i \text{ to } j$$

or $L_{ij} = 0$ if i and j are not connected.

Shortest paths and matrix multiply

Two methods look like linear algebra:

- Floyd-Warshall ($O(n^3)$, similar to Gaussian elimination)
- Matrix multiply ($O(n^3 \log n)$, similar to matrix squaring)

Project 3: parallel repeated squaring for all-pairs shortest path

- Given an OpenMP implementation – time it!
- Write naive MPI implementation using `MPI_Allgatherv`
- Write a better version with nonblocking send/receives

The repeated squaring algorithm

- $l_{ij}^s \equiv$ shortest path with at most 2^s hops
- Initial step is almost the adjacency matrix:

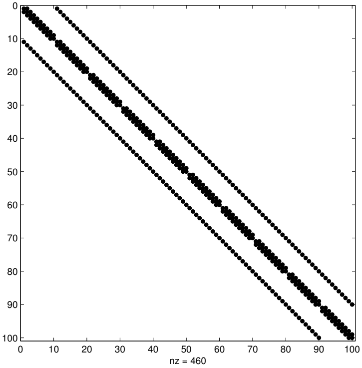
$$l_{ij}^0 = \begin{cases} 1, & \text{edge from } i \text{ to } j \\ 0, & i = j \\ \infty, & \text{otherwise} \end{cases}$$

- Update: $l_{ij}^{s+1} = \min_k \{l_{ik}^s + l_{kj}^s\}$
- Have shortest paths when $L^s = L^{s+1}$ (at most $\lceil \lg n \rceil$ steps)

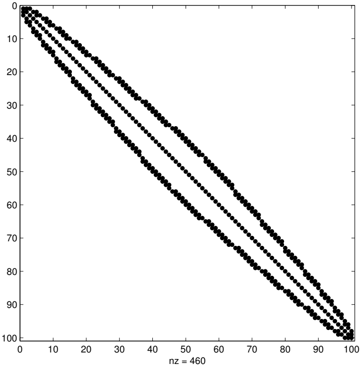
Project 3 logistics

- Goals:
  - Get you some practice with MPI programming
  - And understanding performance tradeoffs!
- May be useful to go back to HW 2 for references
- Please start earlier this time so that you can ask questions!
- If there's a time tradeoff, final project is more important.

Reordering for bandedness



Natural order



RCM reordering

Reverse Cuthill-McKee

- Select “peripheral” vertex v
- Order according to breadth first search from v
- Reverse ordering

From iterative to direct

- RCM ordering is great for SpMV
- But isn't narrow banding good for solvers, too?
  - LU takes $O(nb^2)$ where b is bandwidth.
  - Great if there's an ordering where b is small!

Skylines and profiles

- *Profile* solvers generalize band solvers
- Skyline storage: if storing lower triangle, for each row i:
  - Start and end of storage for nonzeros in row.
  - *Contiguous* nonzero list up to main diagonal.
- In each column, first nonzero defines a profile.
- All fill-in confined to profile.
- RCM is again a good ordering.

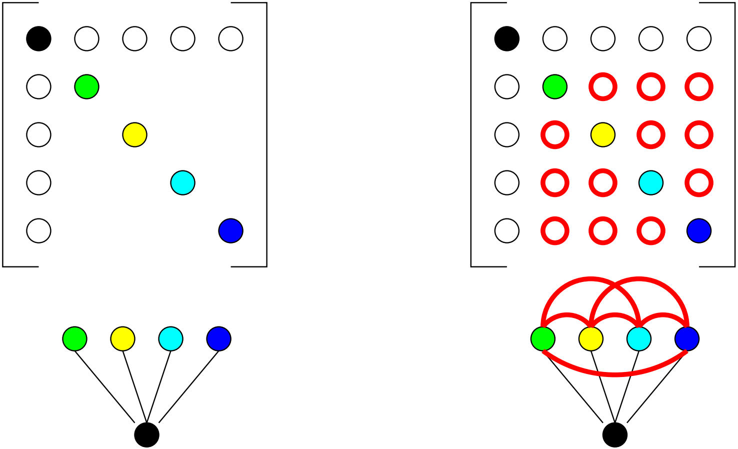
Beyond bandedness

- Bandedness only takes us so far
  - Minimum bandwidth for 2D model problem? 3D?
  - Skyline only gets us so much farther
- But more general solvers have similar structure
  - Ordering (minimize fill)
  - Symbolic factorization (where will fill be?)
  - Numerical factorization (pivoting?)
  - ... and triangular solves

Reminder: Matrices to graphs

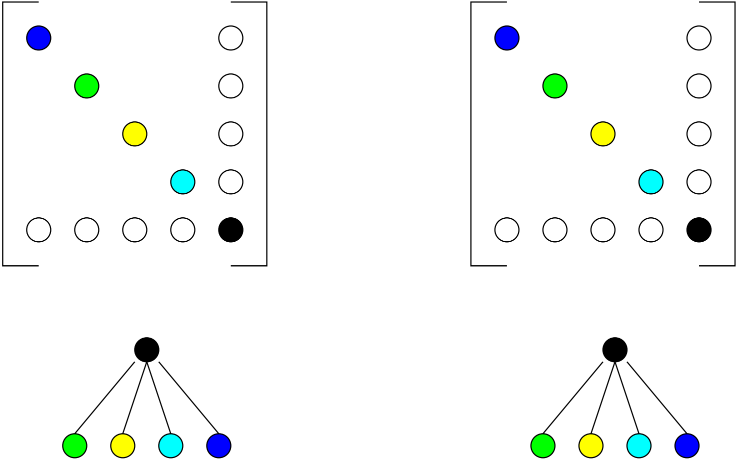
- $A_{ij} \neq 0$ means there is an edge between i and j
- Ignore self-loops and weights for the moment
- Symmetric matrices correspond to undirected graphs

Troublesome Trees



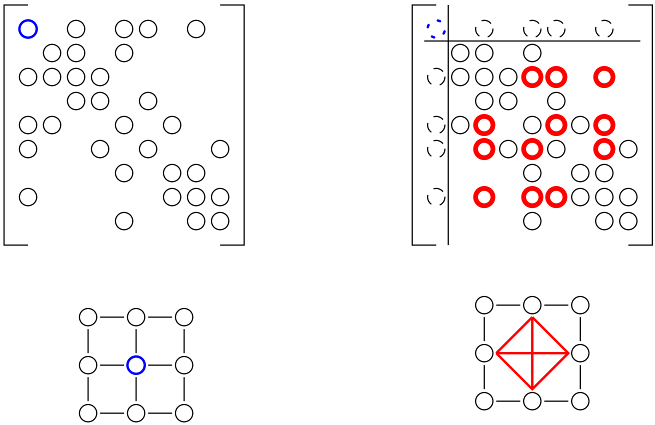
One step of Gaussian elimination *completely* fills this matrix!

Terrific Trees



Full Gaussian elimination generates *no* fill in this matrix!

Graphic Elimination



Eliminate a variable, connect all neighbors.

Graphic Elimination

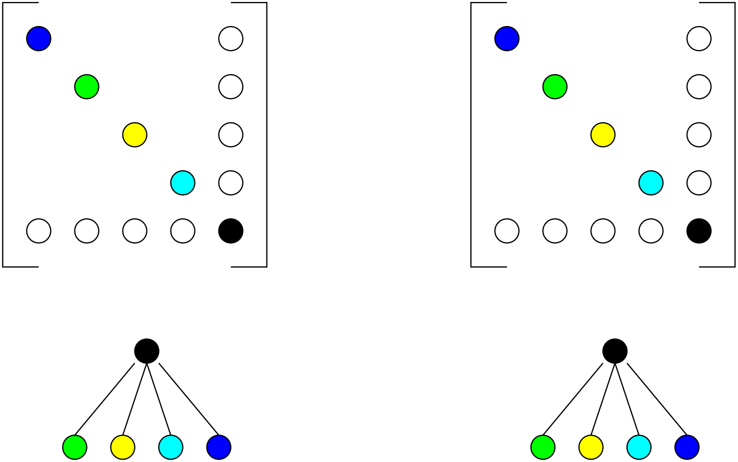
Consider first steps of GE

```
A(2:end,1)       = A(2:end,1)/A(1,1);
A(2:end,2:end) = A(2:end,2:end)-...
                   A(2:end,1)*A(1,2:end);
```

Nonzero in the outer product at (i, j) if `A(i,1)` and `A(j,1)` both nonzero — that is, if i and j are both connected to 1.

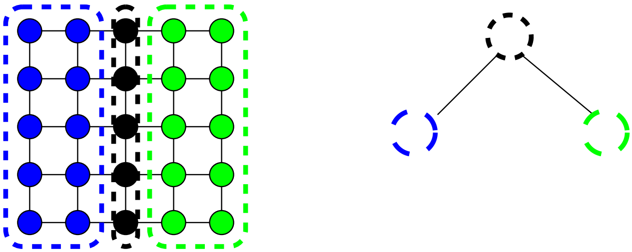
General: Eliminate variable, connect remaining neighbors.

Terrific Trees Redux



Order leaves to root $\implies$
on eliminating i, parent of i is only remaining neighbor.

Nested Dissection



- Idea: Think of *block* tree structures.
- Eliminate block trees from bottom up.
- Can recursively partition at leaves.
- Rough cost estimate: how much just to factor dense Schur complements associated with separators?
- Notice graph partitioning appears again!
  - And again we want small separators!

Nested Dissection

Model problem: Laplacian with 5 point stencil (for 2D)

- ND gives optimal complexity in exact arithmetic (George 73, Hoffman/Martin/Rose)
- 2D: $O(N \log N)$ memory, $O(N^{3/2})$ flops
- 3D: $O(N^{4/3})$ memory, $O(N^2)$ flops

Minimum Degree

- Locally greedy strategy
  - Want to minimize upper bound on fill-in
  - Fill $\leq$ (degree in remaining graph)^2
- At each step
  - Eliminate vertex with smallest degree
  - Update degrees of neighbors
- Problem: Expensive to implement!
  - But better varients via *quotient graphs*
  - Variants often used in practice

Elimination Tree

- Variables (columns) are nodes in trees
- j a descendant of k if eliminating j updates k
- Can eliminate disjoint subtrees in parallel!

Cache locality

Basic idea: exploit "supernodal" (dense) structures in factor

- e.g. arising from elimination of separator Schur complements in ND
- Other alternatives exist (multifrontal solvers)

Pivoting

Pivoting is a tremendous pain, particularly in distributed memory!

- Cholesky — no need to pivot!
- Threshold pivoting — pivot when things look dangerous
- Static pivoting — try to decide up front

What if things go wrong with threshold/static pivoting?
Common theme: Clean up sloppy solves with good residuals

Direct to iterative

Can improve solution by *iterative refinement*:

$$
\begin{aligned}
PAQ &\approx LU \\
x_0 &\approx QU^{-1}L^{-1}Pb \\
r_0 &= b - Ax_0 \\
x_1 &\approx x_0 + QU^{-1}L^{-1}Pr_0
\end{aligned}
$$

Looks like approximate Newton on $F(x) = Ax - b = 0$.
This is just a stationary iterative method!
Nonstationary methods work, too.

Variations on a theme

If we're willing to sacrifice some on factorization,

- Single precision + refinement on double precision residual?
- Sloppy factorizations (marginal stability) + refinement?
- Modify m small pivots as they're encountered (low rank updates), fix with m steps of a Krylov solver?