Implementation of Parallel Sparse Cholesky Decomposition in julia

Omar Mysore

6.338 Project

Presentation Outline

- Project Goal.
- What is Cholesky Decomposition?
- Multifrontal Algorithm.
- Opportunities for Parallelization.
- Implementation in Julia.
- Future Work.

Project Objective

- Build sparse matrix functionality in Julia.
- Currently some basic sparse functions have been implemented in Julia.
- Solving Ax=b for sparse matrices is critical.
- Cholesky decomposition is a useful feature.

What is Cholesky Decomposition?

- It's all about Ax=b.
- When A is symmetric, sparse, and positive definite provides L where $LL^T=A$.
- Faster than LU.

Basic Idea

$$A = \begin{pmatrix} d & v^t \\ v & C \end{pmatrix} = \begin{pmatrix} \sqrt{d} & 0 \\ v/\sqrt{d} & I \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & C - vv^t/d \end{pmatrix} \begin{pmatrix} \sqrt{d} & v^t/\sqrt{d} \\ 0 & I \end{pmatrix}$$

$$A = \begin{pmatrix} B & V^t \\ V & C \end{pmatrix} = \begin{pmatrix} L_B & 0 \\ V L_B^{-t} & I \end{pmatrix} \begin{pmatrix} I & 0 \\ 0 & C - V B^{-1} V^t \end{pmatrix} \begin{pmatrix} L_B^t & L_B^{-1} V^t \\ 0 & I \end{pmatrix}$$

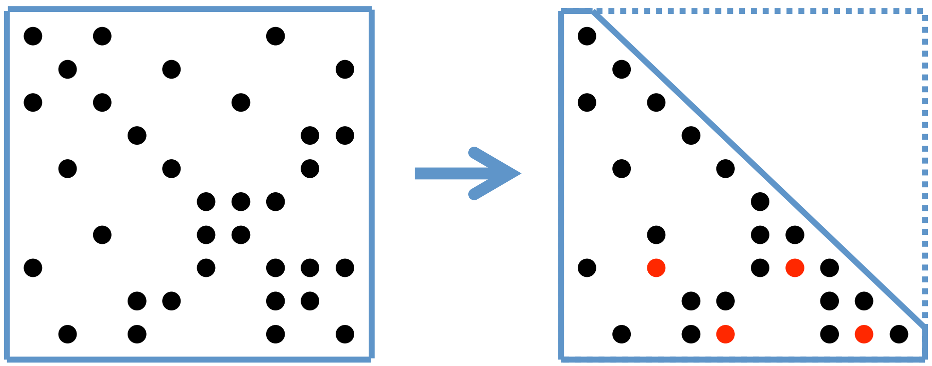
$$B = L_B L_B^t$$

(equations from Liu - see references)

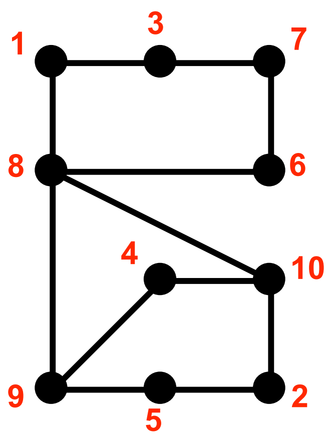
Multifrontal Algorithm Big Picture Overview

- Figure out structure of L.
- Determine dependencies.
- Factor a series of small dense matrices to obtain the values of L.

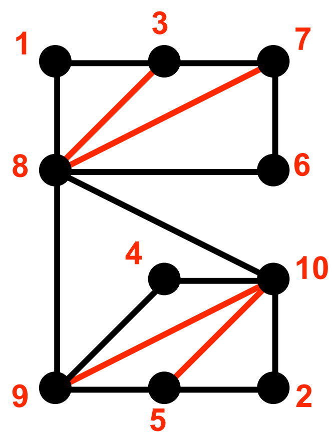
Determining Structure of L



Fill: new nonzeros in factor



G(A)



$G^+(A)$
[chordal]

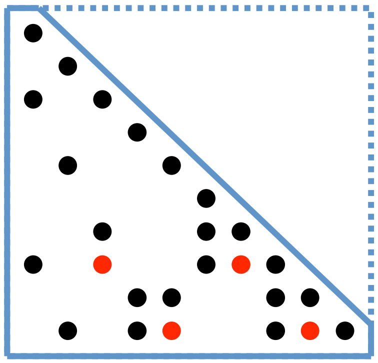
Symmetric Gaussian elimination:

for j = 1 to n

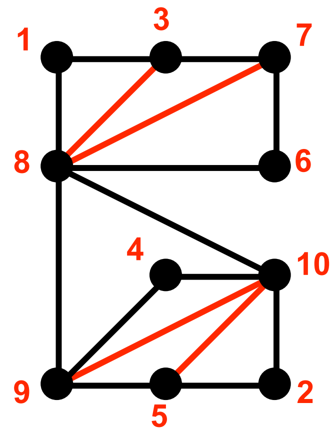
add edges between j's higher-numbered neighbors

(Everything above is from John Gilbert's slides- see references)

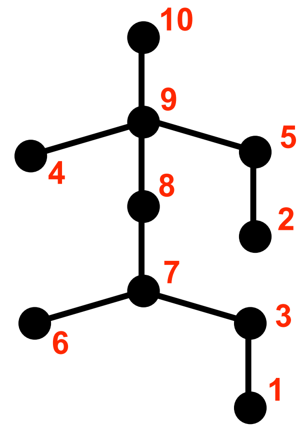
Determining Dependencies



Cholesky factor



$G^+(A)$



T(A)

T(A) : parent(j) = min { i > j : (i,j) in $G^+(A)$ }

(Everything above is from John Gilbert's slides- see references)

Basic Algorithm

- For each column of A:
  - Determine the Frontal Matrix:

$$F_j = \begin{pmatrix} a_{j,j} & a_{j,i_1} & \dots & a_{j,i_r} \\ a_{i_1,j} & & & \\ \vdots & & 0 & \\ a_{i_r,j} & & & \end{pmatrix} + \overline{U}_j.$$

$$\overline{U}_j = -\sum_{k \in T[j]-\{j\}} \begin{pmatrix} \ell_{j,k} \\ \ell_{i_1,k} \\ \vdots \\ \ell_{i_r,k} \end{pmatrix} \begin{pmatrix} \ell_{j,k} & \ell_{i_1,k} & \cdots & \ell_{i_r,k} \end{pmatrix}$$

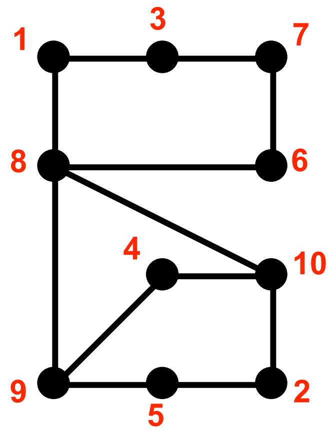
  - Factor F_j to get L_j.

(equations from Liu-see refs.)

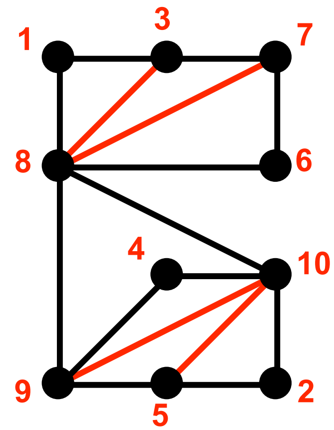
Opportunities for Parallelization

- Determining the structure of L.
- Calculating independent branches of the tree.
- Constructing the frontal matrix.

Parallelizing the Determination of the Structure of L



G(A)

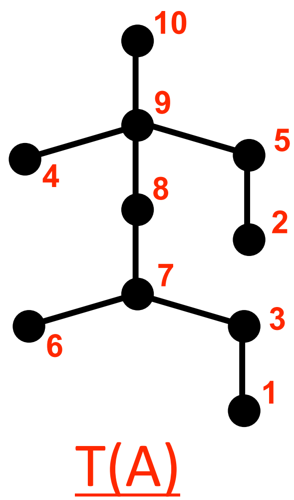


$G^+(A)$
[chordal]

(graphs from Gilbert-see references)

- Each node in G(A) or column of A can be sent to different processors and new edges are returned.

Parallelization from Independent Branches



- Calculate independent branches simultaneously.

(graphs from Gilbert-see refs.)

Parallelizing the Frontal Matrix Calculation

$$F_j = \begin{pmatrix} a_{j,j} & a_{j,i_1} & \dots & a_{j,i_r} \\ a_{i_1,j} & & & \\ \vdots & & 0 & \\ a_{i_r,j} & & & \end{pmatrix} + \overline{U}_j$$

$$\overline{U}_j = - \sum_{k \in T[j]-\{j\}} \begin{pmatrix} \ell_{j,k} \\ \ell_{i_1,k} \\ \vdots \\ \ell_{i_r,k} \end{pmatrix} \begin{pmatrix} \ell_{j,k} & \ell_{i_1,k} & \cdots & \ell_{i_r,k} \end{pmatrix}$$

- Add matrices in pairs on different processors and then combine.

(equations from Liu - see references)

Implementation in julia

- Basic idea for all levels of parallelization is to send vectors/matrices to different processors, let each processor perform the appropriate function, obtain the results, and combine the results.
- Current work involves implementing the parallelizations using the parallel for loop, remote_call and fetch.

Future Work

- Fully implement all parallelization ideas described before.
- Compare the performance of the parallel Julia implementation with serial implementations in other technical computing languages.

References

- John Gilbert's slides from Day 1 of Sparse Matrix Days at MIT. Available at http://www.cs.ucsb.edu/~gilbert/talks/talks.htm.
- Liu, Joseph W. H. "The Multifrontal Method for Sparse Matrix Soluation: Theory and Practice. SIAM Review. Vol. 34, No. 1 (Mar., 1992, pp.82-109.
- Julia Docs from Github.

Acknowledgements

- I would like to thank Alan Edelman, Jeff Bezanson, and Viral Shah.
- Additionally I would like to thank everyone who has been developing the Julia language.