ELIMINATION TREES

Akhil Vidwans

OUTLINE

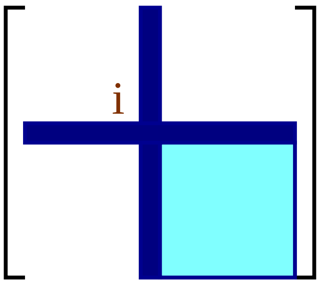
- ‘‘Road-map’’ or ‘‘blueprint’’ for the factorization process
- Motivation : column dependencies
- Derivations
- Properties.....e.g. conditions for fill
- Construction
- Applications....e.g. symbolic factorization

DENSE CHOLESKY

- Outer Product Update:

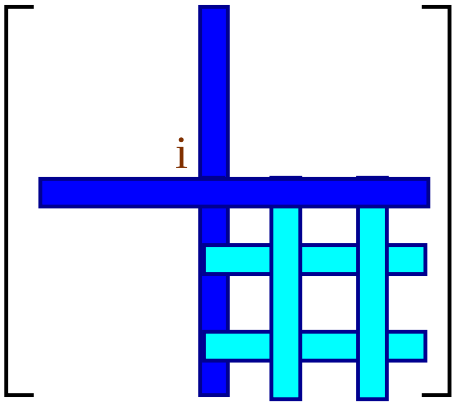
$$A\ (n \times n) = \begin{bmatrix} d & v^T \\ v & B \end{bmatrix} = \begin{bmatrix} \sqrt{d} & 0 \\ v/\sqrt{d} & I \end{bmatrix} \begin{bmatrix} 1 & 0 \\ 0 & B - vv^T/d \end{bmatrix} \begin{bmatrix} d & v^T/\sqrt{d} \\ 0 & I \end{bmatrix}$$

- Column dependencies : i -> [i+1,n]



- Order of factorization: 1, 2, 3.....

SPARSE CHOLESKY

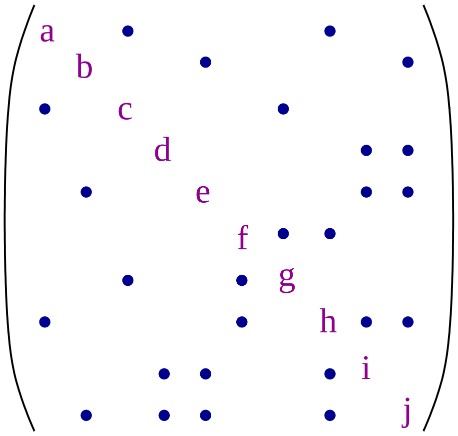


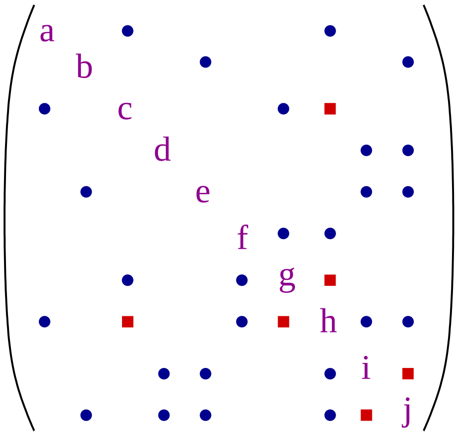
- Update matrix is sparse
- Column dependencies : i -> subset of [i+1,n]
- Updated rows/columns correspond to fill entries

COLUMN DEPENDENCE GRAPH

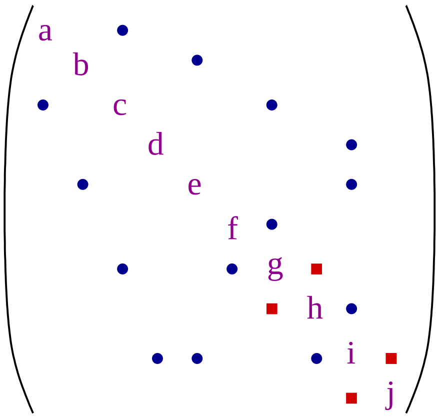
- Nodes correspond to columns
- Edges indicate dependencies
- Each edge maps to a fill edge
- Identical to the graph of the Cholesky factor
- Impractical
- Can less information suffice?

TRANSITIVE REDUCTION





Unnecessary edges:
a-h, b-j, c-h, d-j,
e-j, f-h, h-j.

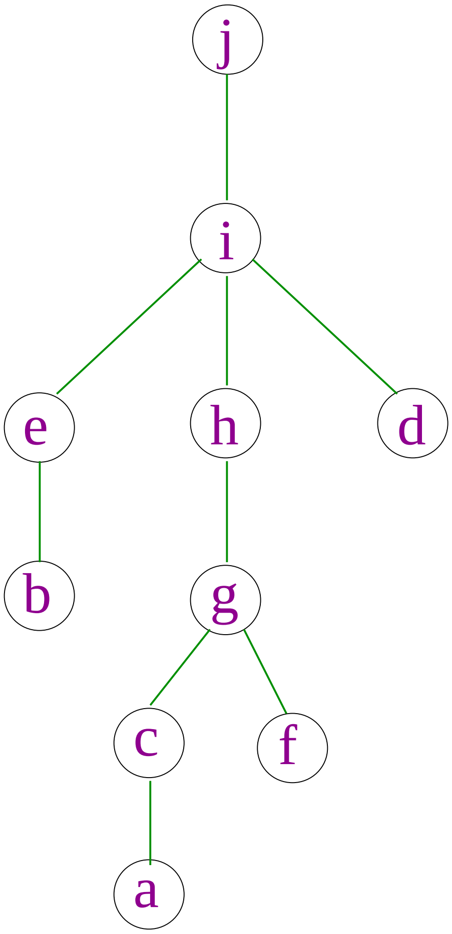


One non-zero entry per column below main diagonal

Graph is:

- acyclic
- undirected

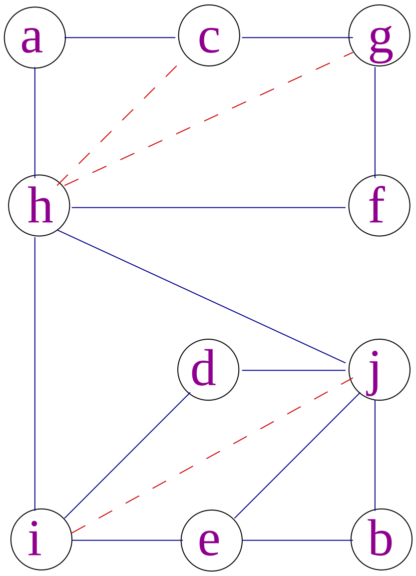
THE ELIMINATION TREE



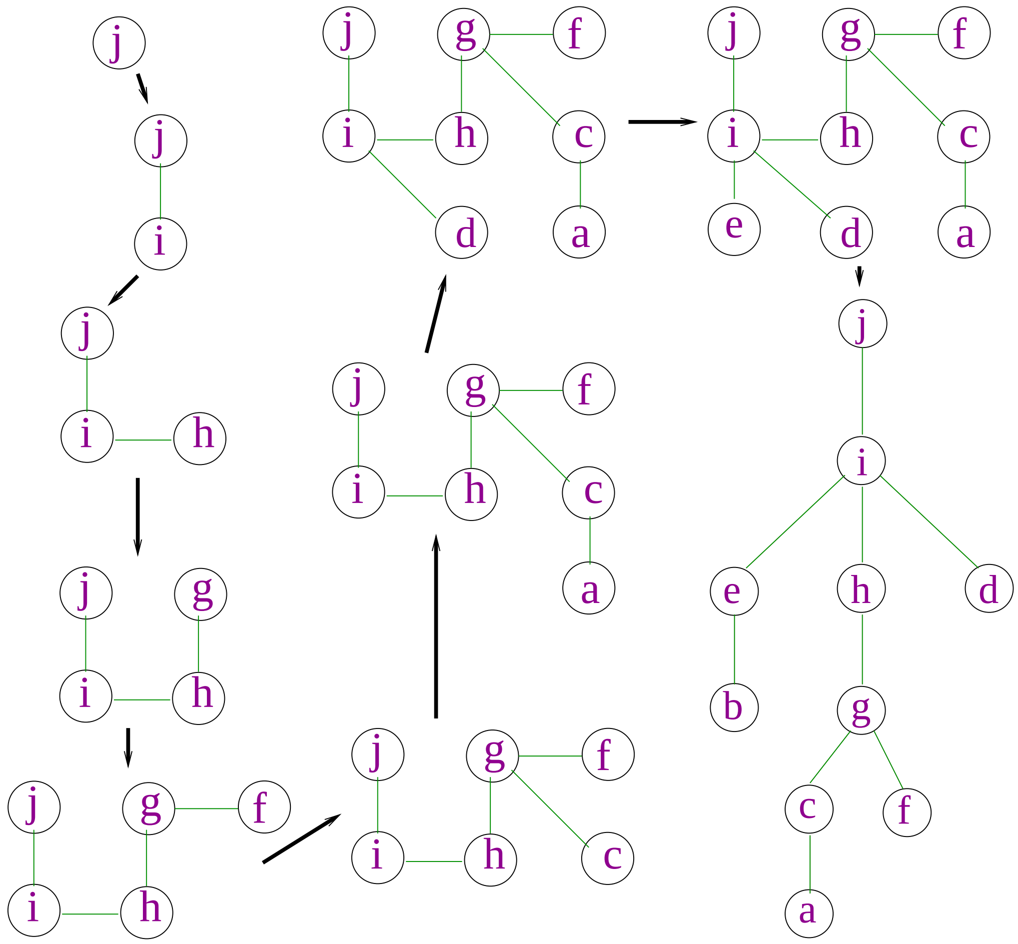
- Root is the highest numbered node
- Parent is numbered higher than children
- Nodes in a subtree are numbered lower than the root of the subtree
- Order of factorization:
  - children before parents

DEPTH FIRST SEARCH

FILLED GRAPH



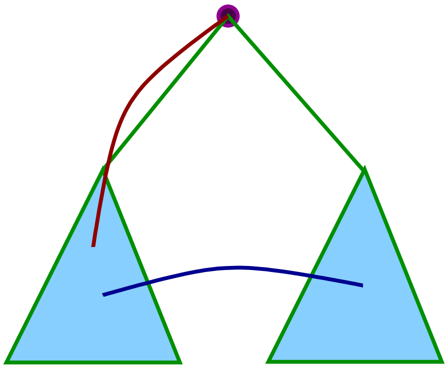
- Start at the root
- Visit neighbors
- Break ties by node number



PROPERTIES

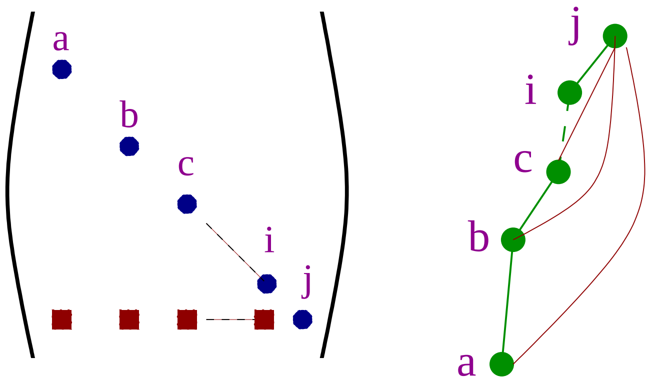
- No ‘‘cross’’ fill edges
- Connectivity property
- Characterization of fill:
  - Subtree characterization
  - Path characterization
- Characterization of row structure
- Characterization of column structure

NO CROSS EDGES



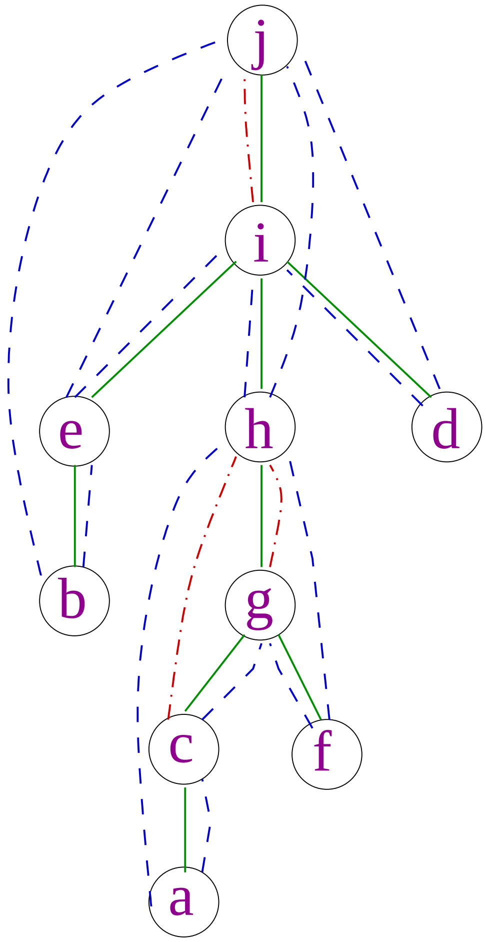
- Blue edge not allowed, red OK.

- Evident from property of DFS



- Fill entry (a,j) gets ‘‘carried’’ up the tree

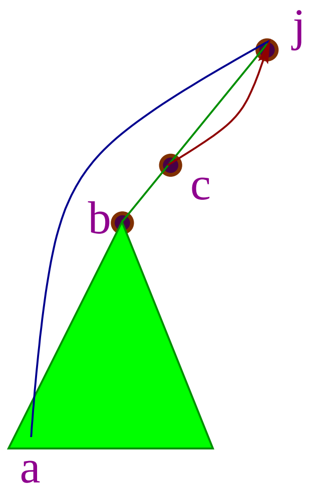
NO CROSS EDGES: EXAMPLE



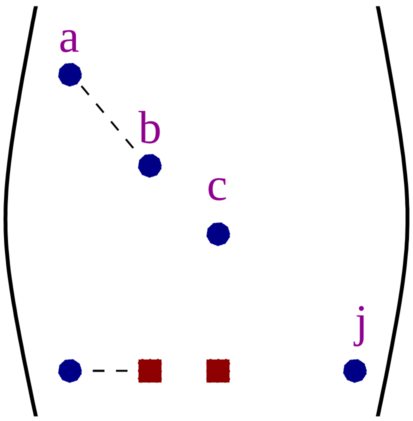
Edges (b,j) and (a,h) are ‘‘back’’ edges of DFS tree

No edge crosses two disjoint subtrees

FILL CHARACTERIZATION I

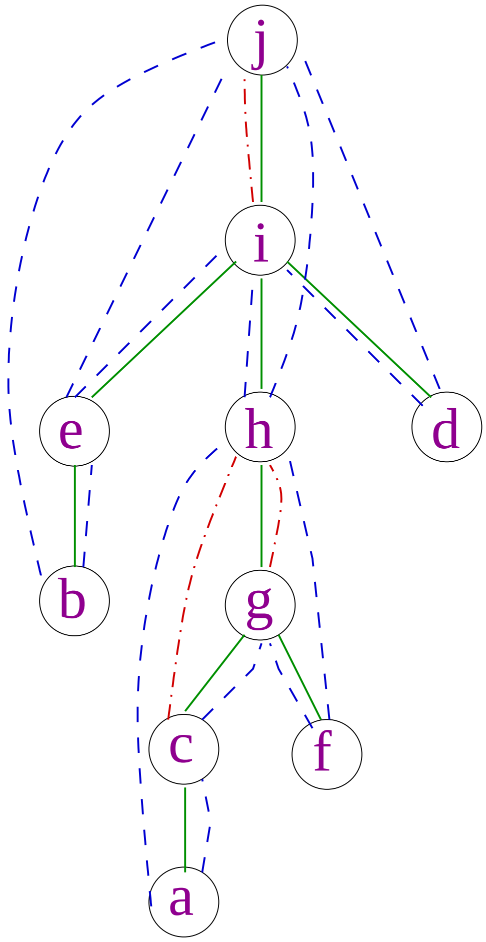


- Fill edge (c,j) exists iff there is some edge (a,j) in the original matrix



- Original matrix edge has to exist in order to get ‘‘carried’’ up

FILL CHARACTERIZATION I : EXAMPLE



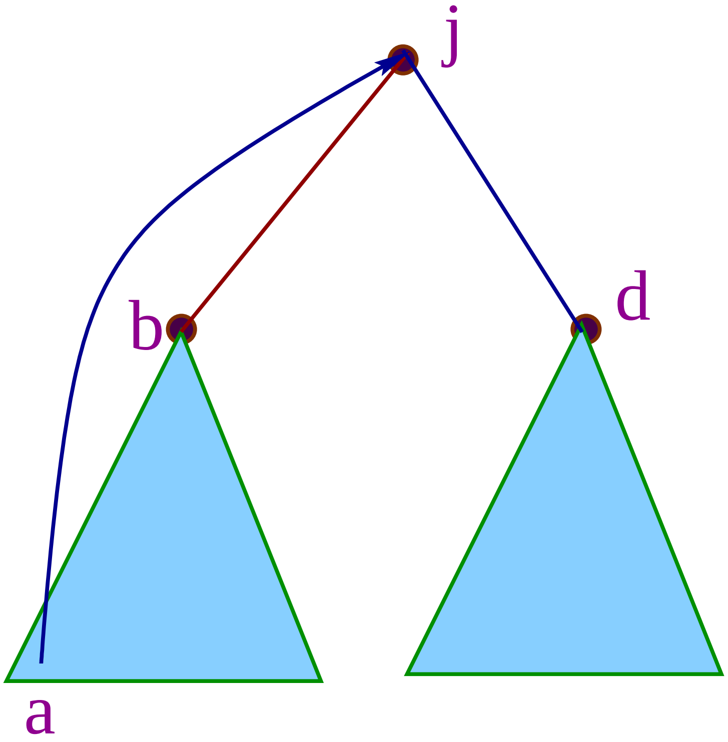
Fill edge (g,h) caused by matrix edge (f,h)

Fill edge (c,h) caused by matrix edge (a,h)

Fill edge (i,j) caused by matrix edge (h,j)

CONNECTIVITY PROPERTY

- Nodes in any subtree of the elimination tree are connected in the original graph



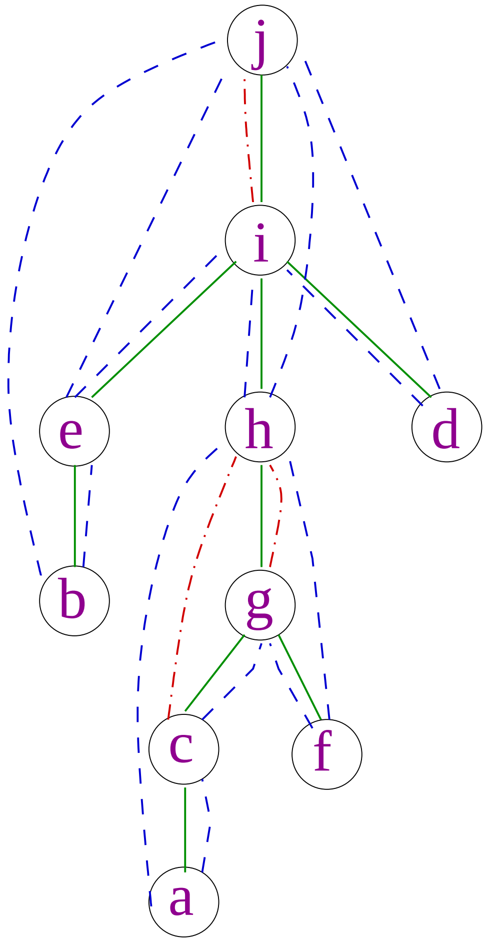
INDUCTION:

Assume b and d subtrees are connected.

(j,d) is a matrix edge

(j,b) is a fill edge, but it requires some matrix edge (a,j)

CONNECTIVITY PROPERTY : EXAMPLE



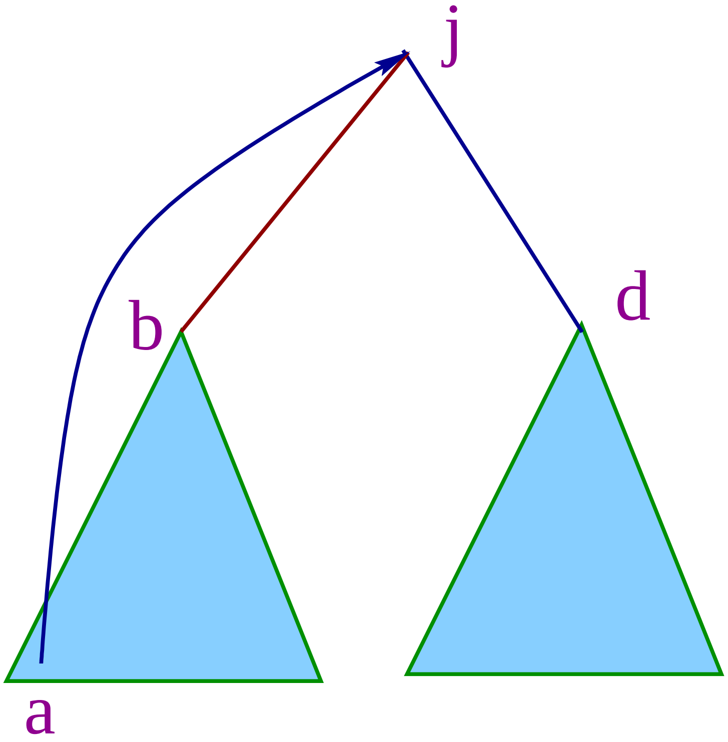
Subtree of h is connected via h-f-g-c-a-h

Subtree of i is connected via d-i-subtree of h-i-e-b

Subtree of j is connected via j-d-subtree of i-e-b

FILL CHARACTERIZATION II

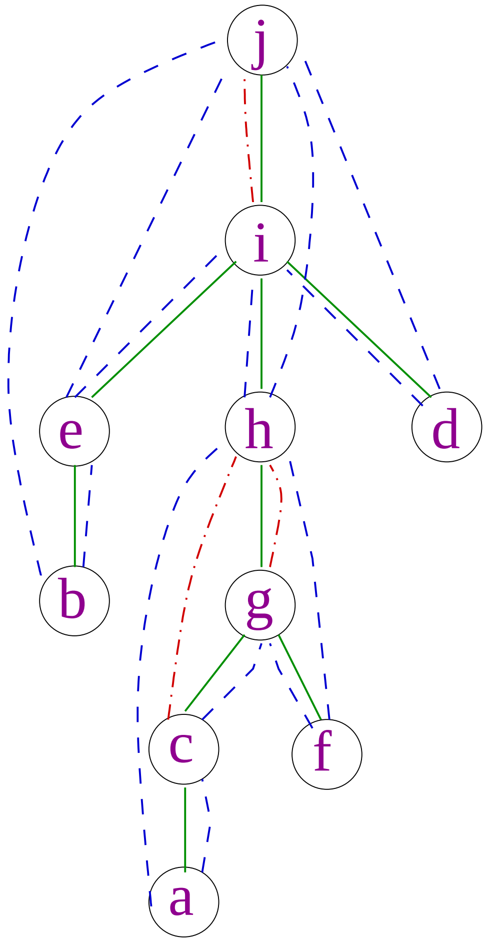
- Fill edge (i,j) exists iff there is a path from i to j in the original graph via nodes numbered less than min(i,j)



Trivially true for (j,d)

True for (j,b) due to (j,a) and connectivity of the subtree of b

FILL CHARACTERIZATION II : EXAMPLE



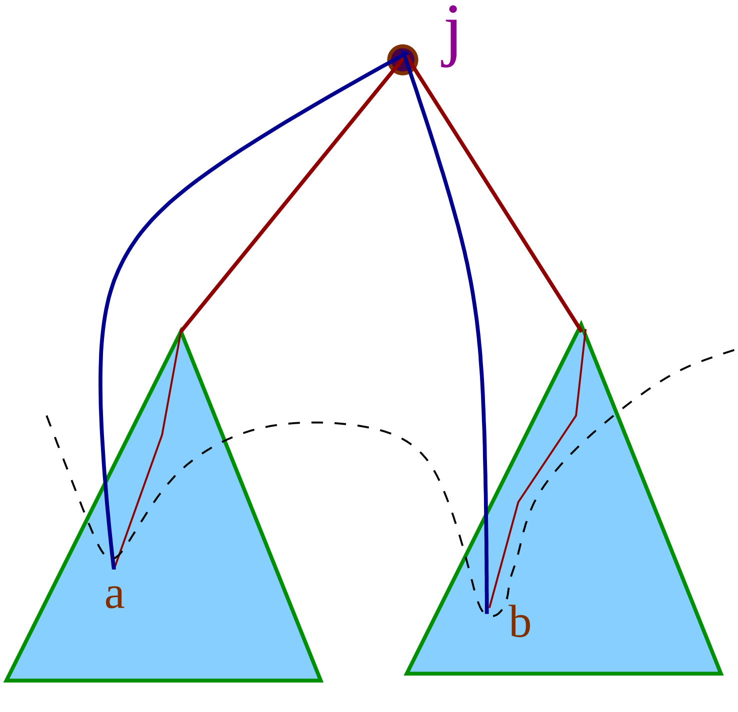
Fill edge (c,h) caused by path c-a-h

Fill edge (g,h) caused by path g-f-h

Fill edge (i,j) caused by path i-d-j

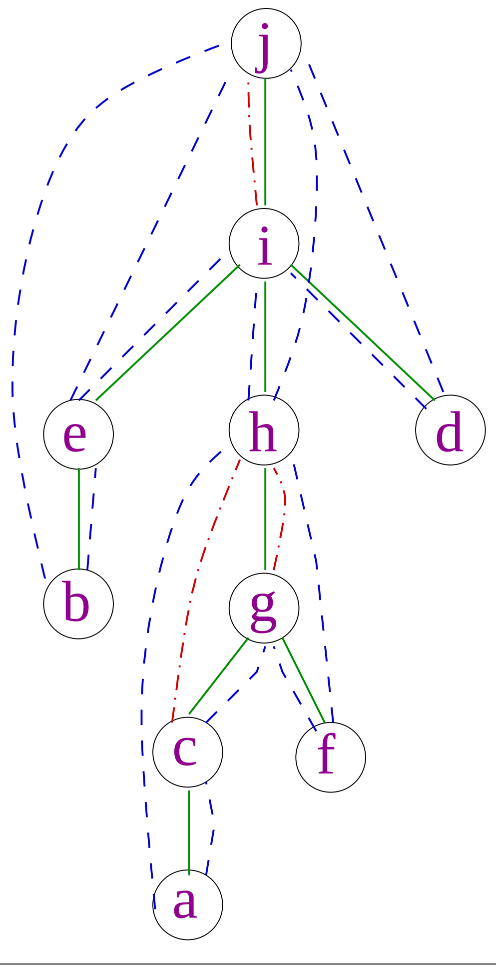
STRUCTURE OF A ROW

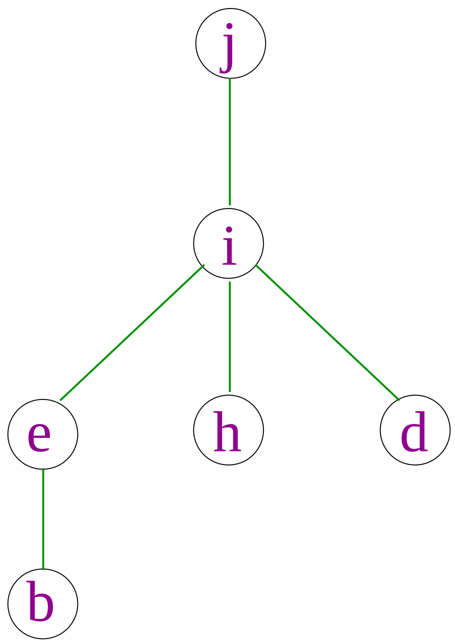
- Row structure of a node is defined by a ‘‘pruned’’ subtree of that node



Every node on tree path from a to j and b to j lies in the row structure of j.

STRUCTURE OF A ROW : EXAMPLE

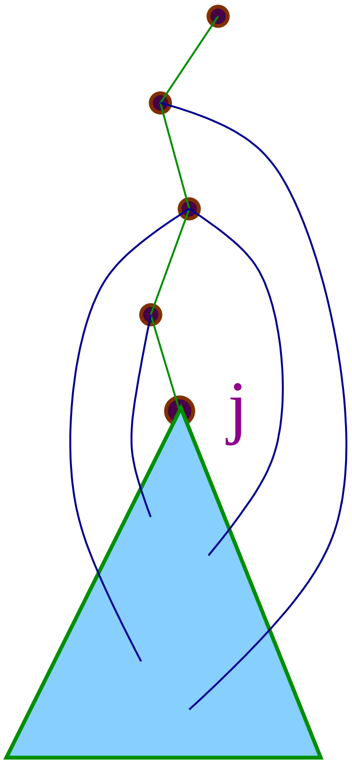




Row structure of node j

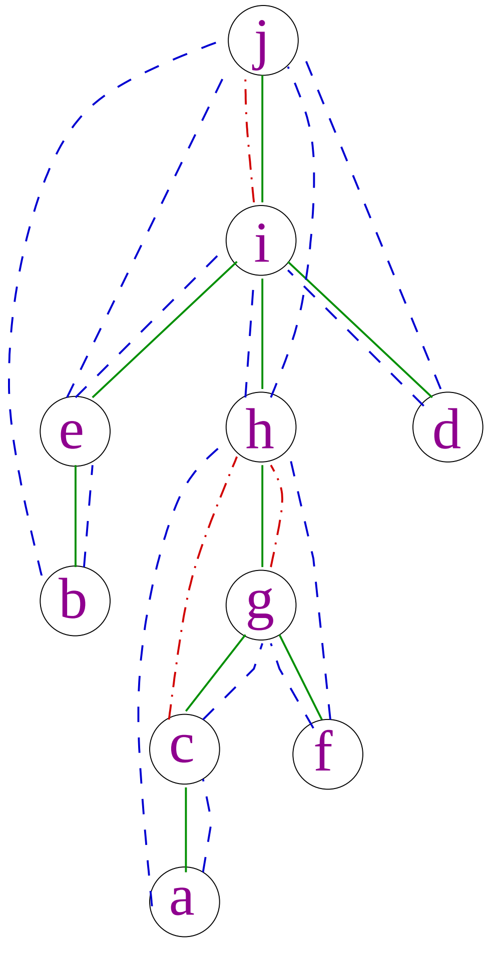
STRUCTURE OF A COLUMN

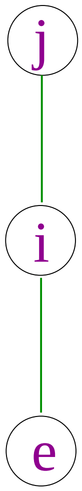
- Column structure of a node is defined by a the neighbourhood of its subtree in the original graph



Adjacency(subtree of j) equals column structure of j.

STRUCTURE OF A COLUMN : EXAMPLE

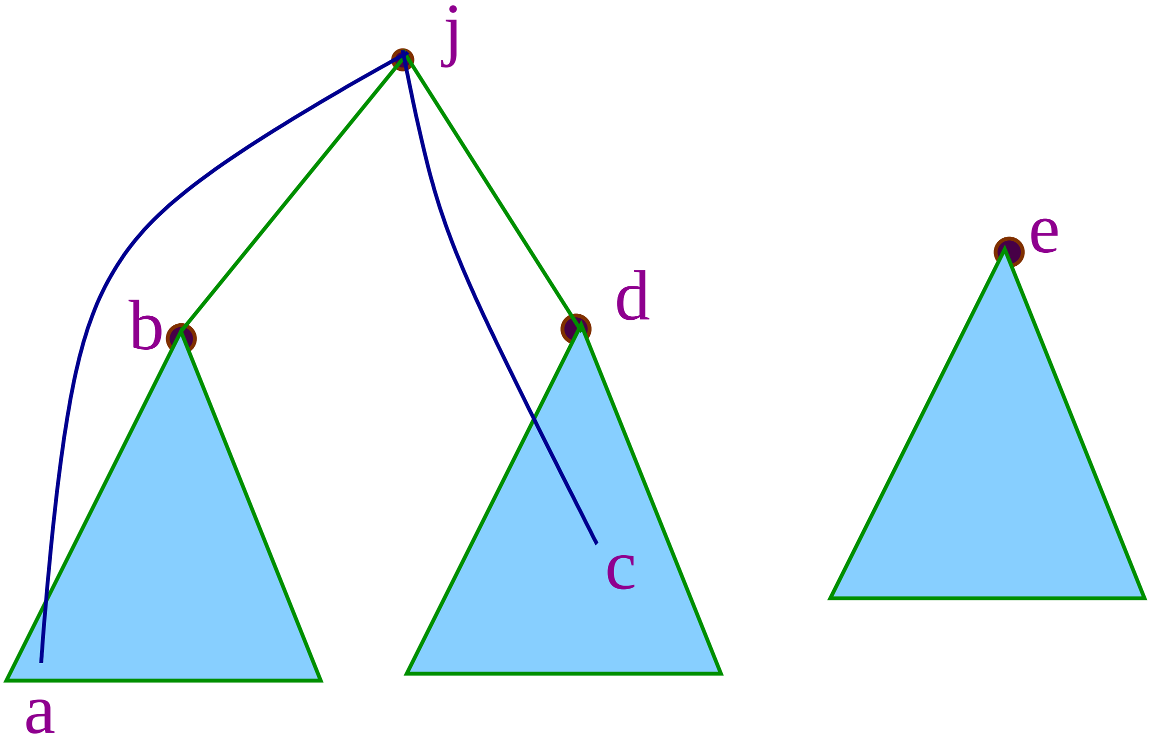




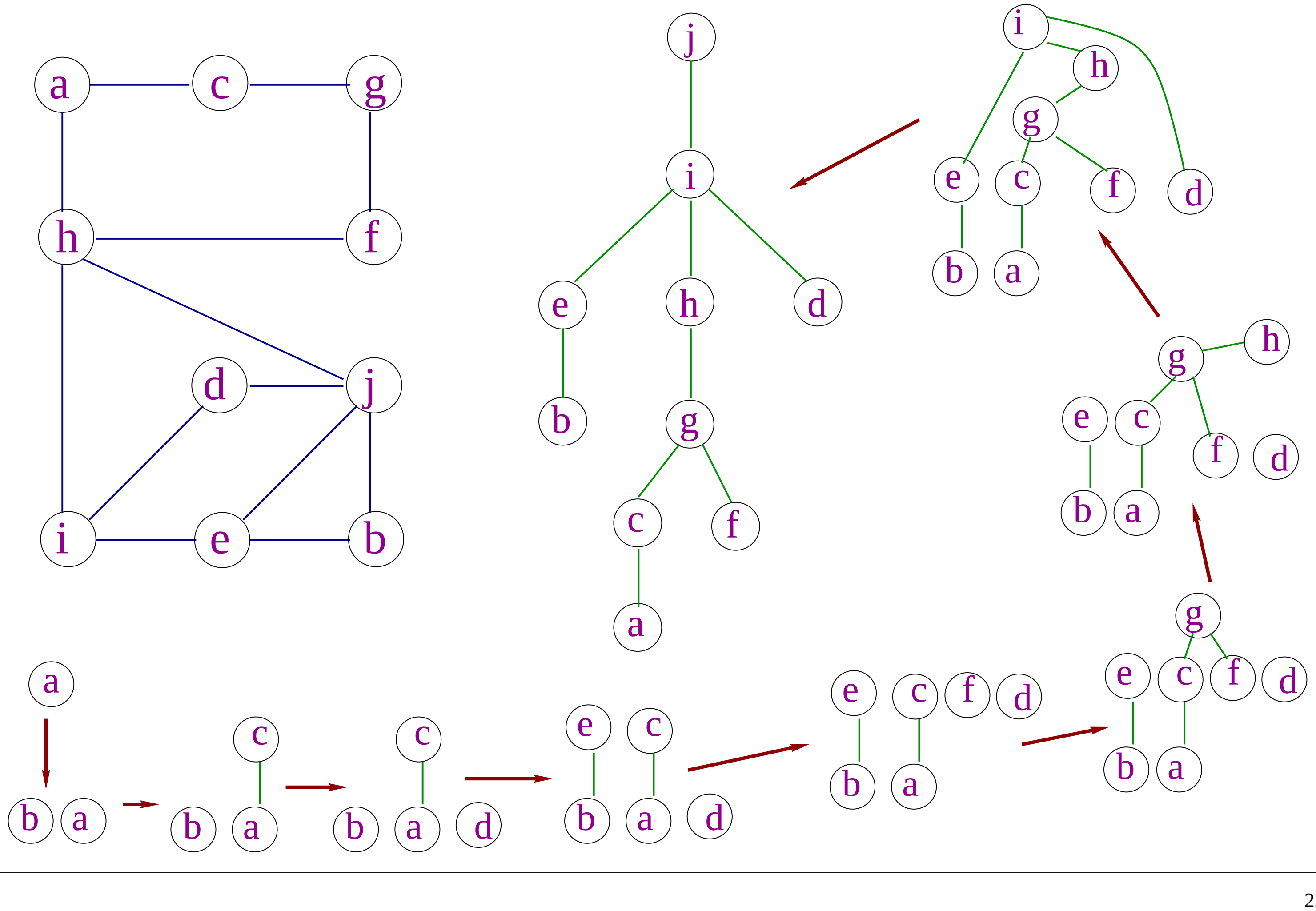
Column structure of node e

CONSTRUCTION OF THE TREE

- Consider nodes from 1 through n, and construct a succession of forests of trees using fill characterization property

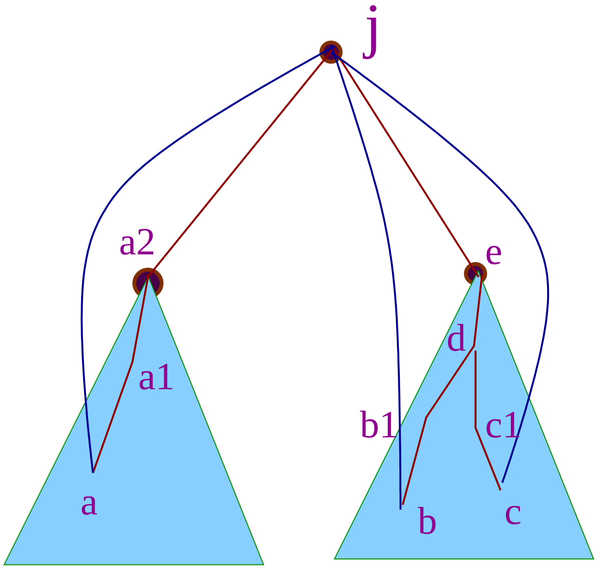


CONSTRUCTION OF THE TREE: EXAMPLE



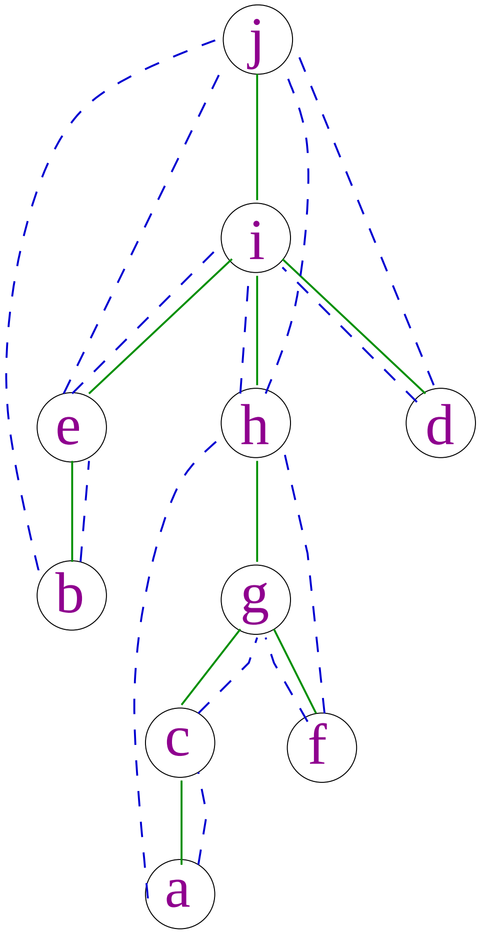
SYMBOLIC FACTORIZATION

- Compute row and/or column structure of Cholesky factor
- Use definitions of row and column structures



- Use: storage allocation

SYMBOLIC FACTORIZATION: EXAMPLE



a :

b :

c : Add a to RS of c
Add c to CS of a

d :

e : Add b to RS of e
Add e to CS of b

f :

g : Add c,f to RS of g
Add g to CS of c,f

h : Add a,c,g,f to RS of h
Add h to CS of a,c,g,f

i : Add e,h,d to RS of i
Add i to CS of e,h,d

j : Add e,b,h,i,d to RS of j
Add j to CS of e,b,h,i,d

SUMMARY

- ‘‘Road-map’’ or ‘‘blueprint’’ for the factorization process
- Motivation : column dependencies
- Derivations: transitive reduction, DFS
- Properties: fill characterization, connectivity and row and column structures
- Construction: bottom up
- Application: symbolic factorization

ERRATUM FOR LIU's PAPER ON ELIMINATION TREES

In the proof for theorem 4.1, the if part is not proved. Instead, the contrapositive of the only if part has been shown. In other words, there are two proofs for the only if part while there is no proof for the if part. This is of significance because the if direction is used in the algorithm to detect perfect elimination orderings later in the same section.

The proof for the if part is as follows (confirmed by the author):

1. First we show that no edge of the elimination tree is a fill edge. This can by proved by considering the first column whose "parental" edge (joining it to its parent in the elimination tree) is a fill edge. By theorem 3.5 and repeated application of the premise for the if direction, it can be easily shown that no such column exists.

   This clearly means that the elimination tree is the same as that defined by the p_j set.

2. The second part is to show that no non-tree edge of the filled graph is a true fill edge. Given that p_j is the elimination tree itself, this is again easy to show by contradiction using theorem 3.5 and repeated application of the premise.

By 1 and 2, no edge of the "filled" graph is a fill edge i.e. the filled graph is the same the original graph i.e. the ordering is a perfect elimination ordering.