**Multifrontal methods**

- Start with the frontal method.
- Recall: Finite element matrix:
  \[ A = \sum A^e \]
  \[ A^e \] = element matrix associated with element \( e \).
- An old idea: Execute Gaussian elimination as the elements are being assembled.
- This is called the *frontal method*.
- Very popular among finite element users: saves storage.

**The origin: Frontal method**

- Elimination of \( x_1 \) creates an update matrix.

**Assembly tree:**
- Analogue to elimination tree.

- Can proceed from several incoupled elements at the same time.
  \[ \rightarrow \text{multifrontal technique [Duff & Reid, 1983]} \]
Multifrontal methods: extension to general matrices

- Elimination tree replaces assembly tree
- Proceed in post-order traversal of elimination tree in order not to violate task dependencies.
- When a node is eliminated an update matrix is created.
- This matrix is passed to the parent which adds it to its frontal matrix.
- Requires a stack of pending update matrices
- Update matrices popped out as they are needed
- Often implemented with nested dissection-type ordering
- More complex than a left-looking algorithm
Eliminating nodes 1 and 2: What happens on matrix

\[
\begin{pmatrix}
1 & * & * \\
* & * & 3 \\
* & 4 & * \\
* & * & 6 \\
* & * & 7 \\
* & * & 8 \\
* & * & 9 \\
\end{pmatrix}
\]

← \(U_1(3,:)\) ← \(U_2(3,:)\)

← \(U_1(7,:)\)

← \(U_2(9,:)\)

Supernodes

Contiguous columns tend to inherit the pattern of the columns from they are updated → Many columns will have same sparsity pattern.

A supernode = a set of contiguous columns in the Cholesky factor \(L\) which have the same sparsity pattern.

The set \(\{j, j + 1, \ldots, j + s\}\) is a supernode if

\[
NZ(L_{*,k}) = NZ(L_{*,k+1}) \cup \{k + 1\} \quad j \leq k < j + s
\]

where \(NZ(L_{*,k})\) is nonzero set of column \(k\) of \(L\).

Other terms used: Mass elimination, indistinguishable nodes, active variables in front, subscript compression,...

Idea is old but first suggested by S. Eisenstat for speeding up sparse codes on vector machines.

Beneficial on most machines

Gains come in part from savings in Gather-Scatter operations.