Advances in Row Ordering for Frontal Solvers in Process Engineering

Jennifer A. Scott
Computational Science and Engineering Department
Rutherford Appleton Laboratory
Oxon, England

Mark A. Stadtherr
Department of Chemical Engineering
University of Notre Dame
Notre Dame, IN, USA

SIAM Annual Meeting, Atlanta, GA
May 1999

Outline

- Background
  - Process Engineering Problems
  - Frontal Method
- Row Ordering Methods
- Results
- Concluding Remarks

Process Engineering Problems

- Realistically complex process simulation and optimization problems typically require large-scale computation.
- When an equation-based problem formulation is used, a key computational bottleneck is often the solution of large, sparse linear equation systems (may be as much as 80-90% of total simulation time).
- Properties of process engineering matrices:
  - Very sparse
  - Very unsymmetric (structurally)
  - Numerically indefinite
  - Not diagonally dominant
  - May be ill-conditioned

Process Engineering Problems (continued)

- Solve $A\mathbf{x} = \mathbf{b}$, where A is large, sparse and has highly asymmetric structure.
- General-purpose direct solvers (e.g., MA48) typically used
  - Factor: $PAQ = LU$ (P and Q represent row and column permutations)
  - Solve: $L\mathbf{y} = P\mathbf{b}$

    $U\mathbf{z} = \mathbf{y}$

    $\mathbf{x} = Q\mathbf{z}$

  using row- or column-oriented Gaussian elimination with threshold pivoting to obtain LU factors.
- Frontal elimination is an attractive alternative for a wide range of modern computer architectures.

Frontal Method

- Basic idea: Restrict computations to a relatively small *front* (or *frontal matrix*) and exploit efficient dense matrix kernels (high level BLAS).
- Originally developed for banded matrices to solve large finite element problems in limited core (Irons, 1970; Hood, 1976).
- Duff (1979) first suggested using frontal method to exploit vector computing in solving finite element problems, implementing it in the Harwell Subroutine Library (HSL) code MA32 (Duff, 1980).
- Applied to process engineering problems on vector/parallel machines by Vegeais and Stadtherr (1985,1990).
- FAMP code (Zitney and Stadtherr, 1993) used in CRAY versions of commercial process simulation codes (e.g. SPEEDUP, ASPEN PLUS).
- Today, the HSL provides MA42 (Duff and Scott, 1992), a general-purpose frontal solver for elements or assembled problems.

Frontal Method (continued)

- Basic factorization steps:
  - Assemble a row into the frontal matrix (beginning with row 1 and proceeding sequentially).
  - Determine if any columns are fully summed (have all their nonzero entries in the frontal matrix).
  - If enough fully-summed columns, perform partial pivoting in those columns and do partial factorization to eliminate them (outer product update).
  - Repeat until all columns have been eliminated.
- Frontal matrix sizes, and thus computational performance, depend on row ordering.

Example: Frontal Method

$$\begin{array}{c|cccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & X & & & X & X & \\
2 & X & & & X & X & X \\
3 & X & X & X & & & \\
4 & X & X & X & & X & \\
5 & & X & X & & X & X \\
6 & & & & & X & X \\
\hline
\end{array}$$

Assemble row 1

$$\begin{array}{c|ccc|}
 & 1 & 4 & 5 \\
\hline
1 & X & X & X \\
\hline
\end{array}$$

no variables fully summed

Assemble row 2

$$\begin{array}{c|cccc|}
 & 1 & 4 & 5 & 6 \\
\hline
1 & X & X & X & \\
2 & X & X & X & X \\
\hline
\end{array}$$

variable 4 fully summed

select pivot from column 4

(say in row 2)

Example: Frontal Method
(continued)

$$\begin{array}{c|cccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 \\ \hline
1 & X & & & X & X & \\
2 & X & & & X & X & X \\
3 & X & X & X & & & \\
4 & X & X & X & & X & \\
5 & & X & X & & X & X \\
6 & & & & & X & X \\ \hline
\end{array}$$

pivot on element (2,4)

$$\begin{array}{c|c|ccc|}
 & 4 & 1 & 5 & 6 \\ \hline
2 & U & U & U & U \\ \hline
1 & L & X & X & X \\ \hline
\end{array}$$

---

updated frontal matrix:

$$\begin{array}{c|ccc|}
 & 1 & 5 & 6 \\ \hline
1 & X & X & X \\ \hline
\end{array}$$

assemble row 3:
no variables fully summed
assemble row 4:
variable 1 fully summed

Example: Frontal Method (continued)

$$\begin{array}{c|cccccc|}
 & 1 & 2 & 3 & 4 & 5 & 6 \\
\hline
1 & X & & & X & X & \\
2 & X & & & X & X & X \\
3 & X & X & X & & & \\
4 & X & X & X & & X & \\
5 & & X & X & & X & X \\
6 & & & & & X & X \\
\hline
\end{array}$$

using element (4,1) as pivot:

$$\begin{array}{c|c|cccc|}
 & 1 & 5 & 6 & 2 & 3 \\
\hline
4 & U & U & & U & U \\
\hline
1 & L & X & X & X & X \\
3 & L & X & & X & X \\
\hline
\end{array}$$

updated frontal matrix:

$$\begin{array}{c|cccc|}
 & 5 & 6 & 2 & 3 \\
\hline
1 & X & X & X & X \\
3 & X & & X & X \\
\hline
\end{array}$$

continue until LU factors are complete

Row Ordering Methods

- RMCD (Camarda, 1997; Camarda and Stadtherr, 1998)
- NMNC (Camarda, 1997)
- MSRO (Scott, 1998)

RMCD Row Ordering

- Local ordering, based on bipartite graph model of unsymmetric matrix.
- Uses concept of a *net*
  - Net j comprises column vertex j and all adjacent row vertices (corresponding to rows with nonzeros in column j).
- Basic ideas:
  - Find column of minimum degree, giving priority to partially summed columns.
  - Put next in the row ordering the rows in the corresponding net.
  - Remove net from graph and update column degrees.
- When each net is assembled, there is at least 1 fully summed column.
- Restricts growth in row dimension ($frow_i$)of frontal matrix, but not in column dimension ($fcol_i$)

NMNC Row Ordering

- Global ordering, based on bipartite graph model of unsymmetric matrix.
- Uses concept of a net.
- Basic step is a graph bisection into two subgraphs
  - Seek to minimize (approximately) the number of nets cut in the bisection.
  - Seek to keep the subgraphs (approximately) the same size.
  - Heuristic approach used, based on min-net-cut method of Coon and Stadtherr (1995).
- Apply bisection step recursively.
- Restricts growth in both $frow_i$ and $fcol_i$.

MSRO Row Ordering

- Two-phase ordering, global then local.
- Global ordering uses the concept of a *row graph* G_R.
  - The row graph of A is the undirected graph of the symmetric matrix $B = A * A^T$, where * indicates matrix multiplication without accounting for numerical cancellations.
  - The nodes of G_R are the rows of A.
  - There is a edge between nodes i and j if and only if there is at least one column in which both row i and row j have a nonzero entry.
- Global ordering methods used by MSRO:
  - Pseudodiameter approach (e.g. Gibbs et al., 1976) applied to G_R.
  - Spectral method (e.g., Barnard et al., 1995) applied to G_R.
  - NMNC method (G_R not used).

MSRO Row Ordering (continued)

- Local ordering is based on a *priority function* P_i.
- Basic ideas
  - Select the next row in the reordering by choosing, from a set of eligible rows, a row i that minimizes P_i.
  - Eligible rows are *active* rows and their neighbors in G_R.
  - An unordered row is active if it is adjacent in G_R to a row that has already been ordered.
- The priority function is the weighted average of a global priority (determined in global ordering phase) and a local priority (based on increases to $frow_i$ and $fcol_i$ caused by ordering row i next).

Results: Test Problems

- Row ordering methods were tested on a set of 22 matrices drawn from chemical process engineering problems.
- Applications include several multiunit flowsheets, many involving multiple separation columns.
- Application codes from which matrices were drawn include:

  SPEEDUP (Aspen Technology, Inc.), ASPEN PLUS (Aspen Technology, Inc.), NOVA (DOT Products, Inc.), SEQUEL (University of Illinois), ASCEND (Carnegie-Mellon University)
- Matrix sizes range from n = 1048 to n = 70304. All are highly asymmetric.
- Complete, detailed results given in: J. A. Scott, “Row ordering for frontal solvers in chemical process engineering,” RAL Technical Report RAL-TR-1999-035 (submitted to *Comput. Chem. Eng*.)

Results: Highlights I

- Using average front size $f_{ave} = (1/n)\ \Sigma_i\ frow_i * fcol_i$ as criterion, the number of problems on which each ordering was best (or tied for best):

MSRO + spectral (MSRO/spec)	13
MSRO + pseudodiameter (MSRO/pd)	5
MSRO + NMNC (MSRO/NMNC)	4
RMCD	2
NMNC	1

- Due to limitations of package used to obtain the spectral ordering, MSRO/spec was not applied to the four largest problems. So MSRO/spec was best on 13 of the 18 problems for which it was used.

Results: Highlights II

- MSRO algorithms usually provide dramatic reductions in f_{ave}.
- Examples:

Problem:		hydr1 (n = 5308)	lhr14c (n = 14270)
Original Ordering	f_{ave} / 100 =	310	1076
MSRO/spec		3	134
MSRO/pd		10	170
MSRO/NMNC		58	224
NMNC		197	266
RCMD		231	7645

Results: Highlights III

- MSRO/spec and MSRO/pd do not perform as well when there is a very high degree of connectivity in the row graph (average number of neighbors more than about 100).
- Examples:

Problem:	ethylene-1 (n = 10673)	meg1 (n = 2904)
Original Ordering	f_{ave} / 100 = 1452	11823
MSRO/spec	2449	1015
MSRO/pd	3910	1837
MSRO/NMNC	213	1781
NMNC	573	3068
RCMD	11249	461

- Average number of neighbors in G_R is 190.7 for ethylene-1, and 128.1 for meg1. Meg1 has a much shorter pseudodiameter (7) than other problems.

Results: Highlights IV

- Factorization times (t_F) using frontal solver MA42 reflect improved row orderings, but improvement is not as dramatic as in average front size.
- Examples:

Problem:		hydr1 (n = 5308)	lhr14c (n = 14270)
Original Ordering	t_F (sec.) =	1.7	23.9
MSRO/spec		0.7	8.1
MSRO/pd		0.8	9.2
MSRO/NMNC		1.7	12.6
NMNC		1.4	13.1

- CPU times on Sun Ultra 1/140.
- For most problems, savings from MSRO row orderings are at least 50% and as much as 80%.

Results: Highlights V

- MA42 with new row ordering was compared with MA48 on the Sun Ultra 1/140.
- MA48 is a widely-used, general-purpose sparse solver for asymmetric systems. It is based on Gaussian elimination with Markowitz pivoting for sparsity and threshold partial pivoting for numerical stability.
- For half of the test problems, row ordering time plus MA42 factor time was less than MA48 analyze plus factor time.
- For several problems, MA42 factor time was less than MA48 factor time.
- For a few problems, MA42 factor time was also less than MA48 fast factor time.

Results: Highlights VI

- Examples (times in seconds on Sun Ultra 1/140):

	Problem:	lhr34c (n = 35152)	10cols (n = 29496)
MA42:	row ordering	21.6	2.1
	factor	158	7.6
	solve	2.09	0.91
MA48:	analyze	148	15.0
	factor	231	4.7
	fast factor	225	3.2
	solve	0.77	0.20

- Solve-only times for single right-hand side are always less with MA48.
- For solving with multiple right-hand sides simultaneously, use of BLAS3 in MA42 may overcome sparser LU factors of MA48.

Results: Highlights VII

- Numerical experiments were also run on a CRAY J932.
- FAMP (Zitney and Stadtherr, 1993) was used as the frontal solver instead of MA42, since FAMP is highly tuned for the CRAY architecture.
- Row ordering performance is poor on the CRAY due to its slow integer arithmetic. Row orderings can be done faster on the Sun and passed to the CRAY.
- On all problems (except one tie), FAMP factor time is less than MA48 factor time.
- For many problems (10), FAMP factor time is also less than MA48 fast factor time.
- Solve-only times (one right-hand side) are always less with MA48.

Results: Highlights VIII

- Examples (times in seconds on CRAY J932):

	Problem:	lhr34c (n = 35152)	10cols (n = 29496)	bayer04 (n = 20545)
FAMP:	factor	8.8	2.41	2.18
	solve	0.59	0.42	0.27
MA48:	factor	34.8	9.03	4.33
	fast factor	16.1	3.47	1.59
	solve	0.29	0.16	0.11

Concluding Remarks

- The MRSO algorithm (Scott, 1998) is available in the new code MC62, which will be included in next release of Harwell Subroutine Library (HSL 2000).
- MC62 also offers RMCD reordering (Camarda and Stadtherr, 1998), since it can outperform MRSO when pseudodiameter of row graph is short.
- MRSO usually provides a substantial improvement over original ordering and earlier RMCD and NMNC orderings.
- With a good row ordering, frontal solvers can provide a powerful and competitive alternative to general-purpose sparse solvers for chemical process applications.