MODERN ALGORITHMS FOR SUPERCOMPUTING -

SPARSE MATRIX ALGORITHMS

HORST D. SIMON

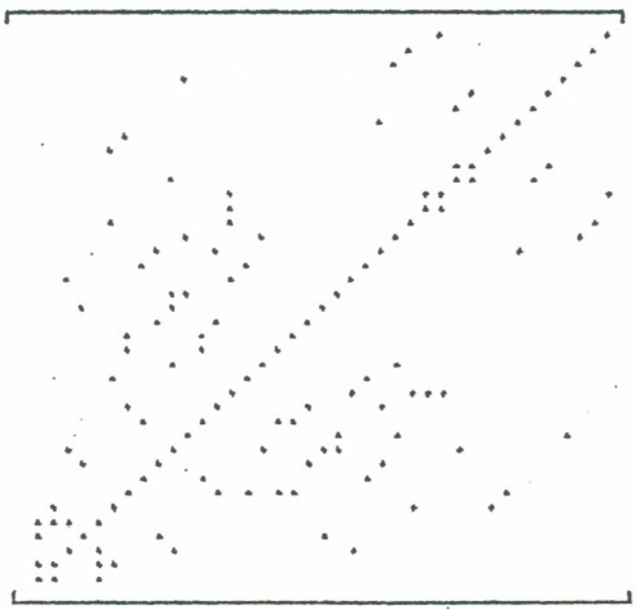
NAS Systems Division

NASA Ames Research Center

Moffett Field, CA 94035

OUTLINE

- Review of terminology and basic concepts
- Progress in ordering methods: multiple minimum degree
- Progress in hardware: impact of gather/scatter
- Progress in out-of-core schemes: Liu's method and multifrontal
- Conclusions



REVIEW: DENSE LINEAR EQUATIONS
Direct Sparse Linear Equations on Supercomputers

- Given an *n by n* dense coefficient matrix, and linear system $A\ x = b$

  -- Factor A into some reduced form

  $(LU, LL^T, LDL^T, UDU^T, QR, LQ, U\Sigma V^T, \ldots)$

  -- Solve sequence of simpler systems

  ($L\ z = b$, then $U\ x = b$)

- Costs

  -- n^3 operations

  -- n^2 storage

SPARSE MATRICES

Direct Sparse Linear Equations on Supercomputers

- "Mostly zero"

  Wilkinson -- "sparse <u>if we can take advantage</u> of the structure of the matrix"

- Goal

  -- much less storage and work than dense

- Problems

  -- A^{-1} and x are dense

  -- L and U are less sparse than A

SPARSE MATRICES (continued)

Direct Sparse Linear Equations on Supercomputers

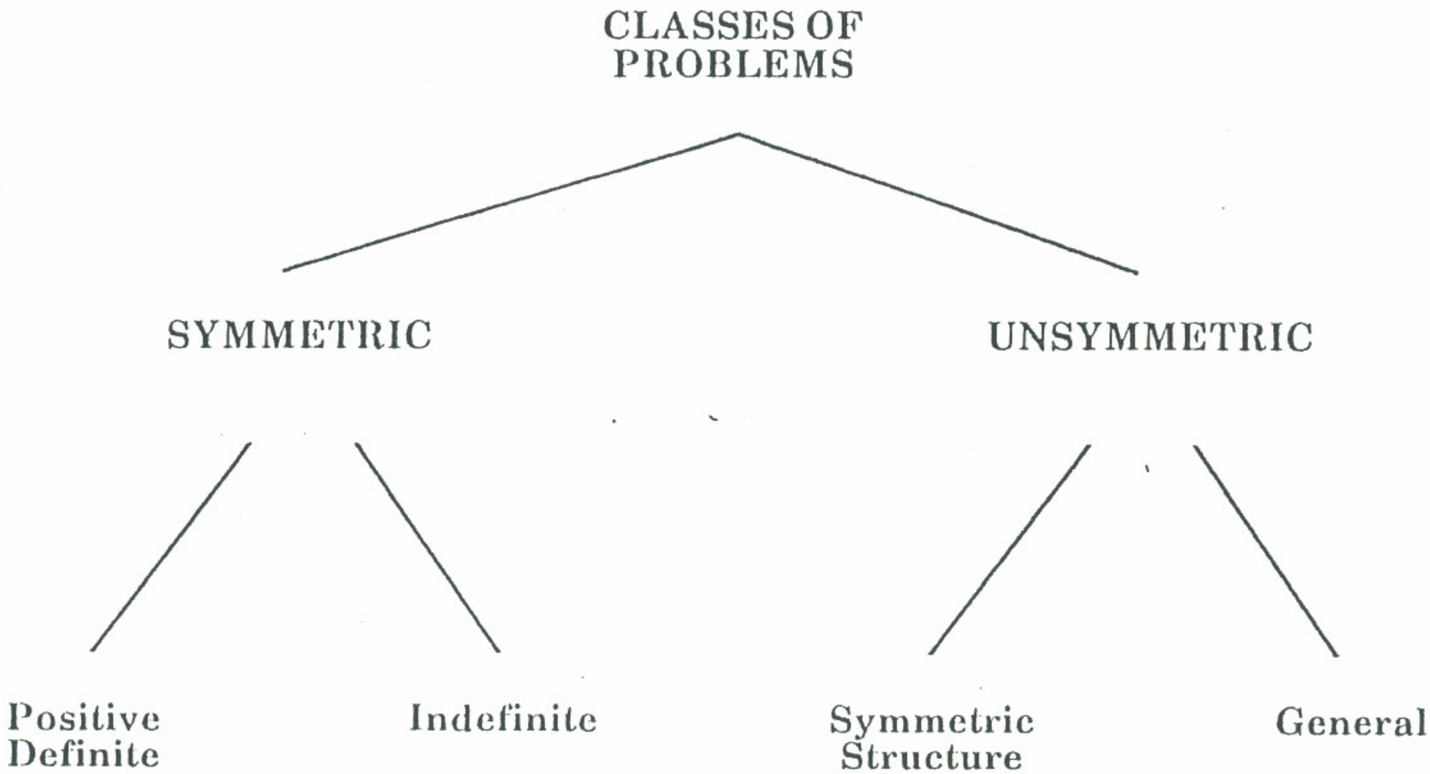
- Fill in

  -- introduction of zeros in *L* or *U* in places where *A* had exact zeros

```
[  x        x    x    x         x   ]

                  +

[ x x         x    x    x        x  ]

                  =

[ x x      + x  + x  + x       + x  ]
```

PROBLEMS

Direct Sparse Linear Equations on Supercomputers



ORDERING

Direct Sparse Linear Equations on Supercomputers

- Dense case

  -- rows and/or columns are interchanged for stability

  -- interchanges do not change the solution

  -- interchanges never necessary for LL^T (Cholesky) decomposition of symmetric positive definite systems

  -- symmetric row and column interchanges preserve positive-definiteness

  -- any reordering of s.p.d. A as good as any other

ORDERING (continued)

Direct Sparse Linear Equations on Supercomputers

- Sparse case

  -- ordering of rows and columns controls fill

$$\begin{bmatrix} * & * & * & * & * \\ * & * & & & \\ * & & * & & \\ * & & & * & \\ * & & & & * \end{bmatrix} \text{ yields full } L \qquad \begin{bmatrix} * & & & & * \\ & * & & & * \\ & & * & & * \\ & & & * & * \\ * & * & * & * & * \end{bmatrix} \text{ has no fill in } L$$

ORDERING (continued)

Direct Sparse Linear Equations on Supercomputers

- Choose some permutation of rows and columns so that reordered system is easier to solve.

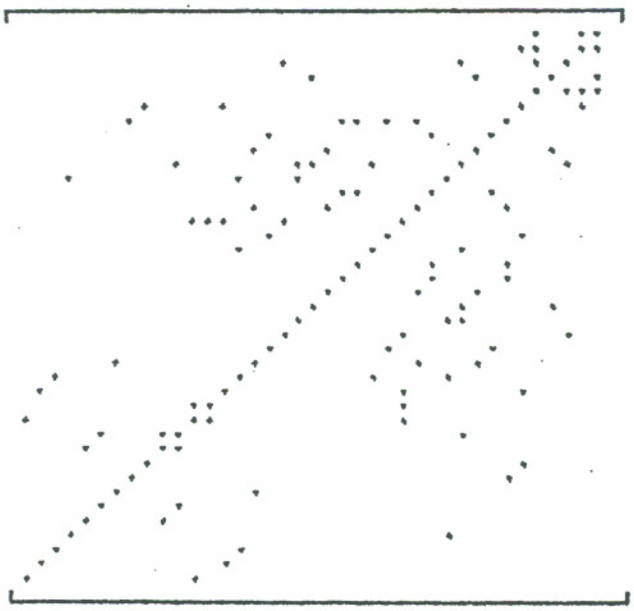
- Obtaining minimum fill is NP-complete.

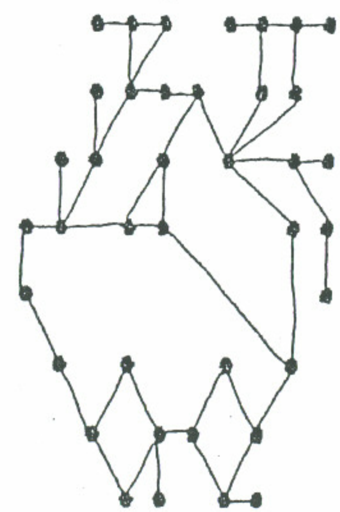
- Also want to consider data structures as well as fill.

- Ordering

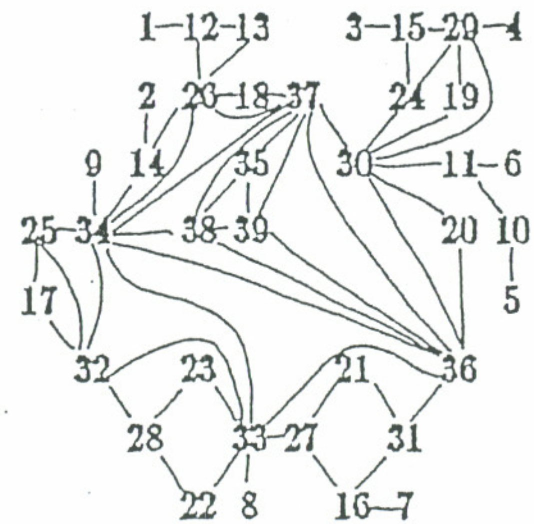
  -- heuristics, often discipline dependent, for obtaining permutation vectors

  -- independent of the numerical values





New England Power System Graph



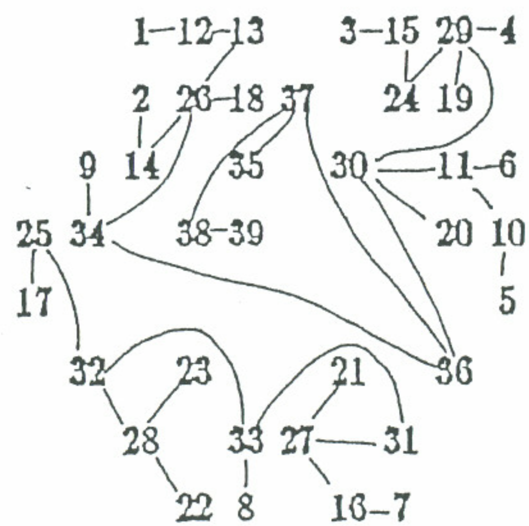
Graph of L, MMD Ordering

Elimination Trees

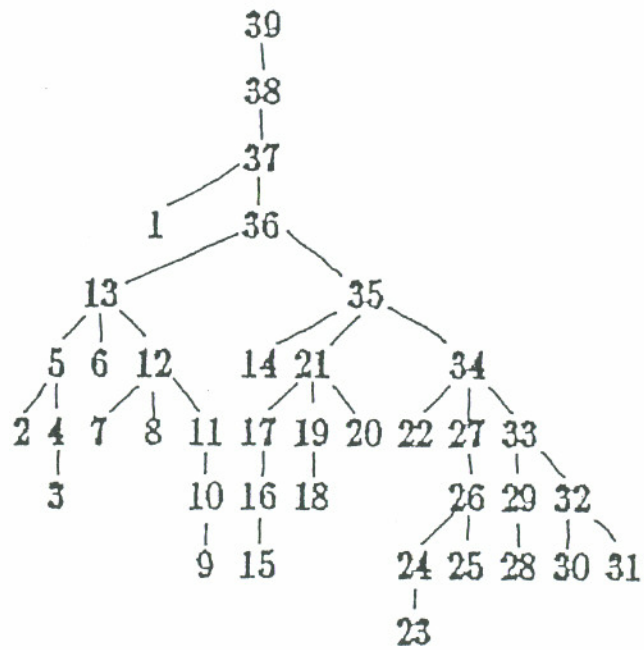
Each ordering implies an elimination tree

Subset of edges in the factored matrix

Equivalence of all traversals numbering child before parent



Elimination tree for L

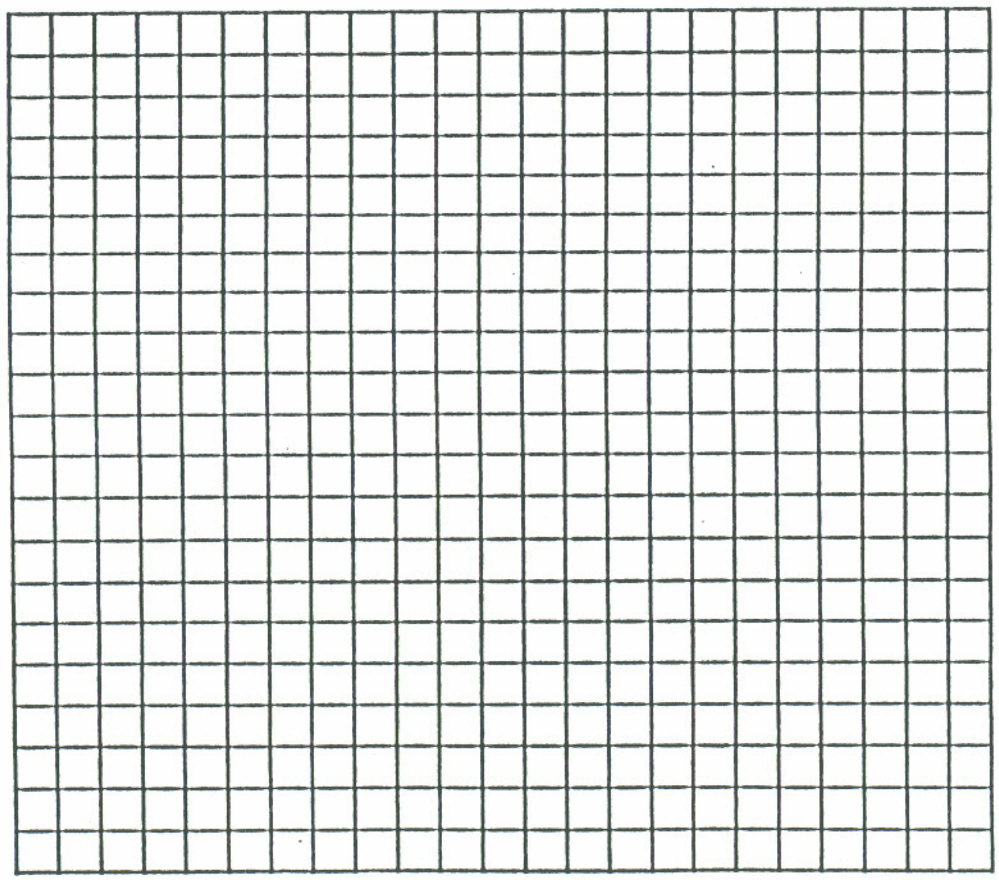


Post-Ordered Elimination tree for L

ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

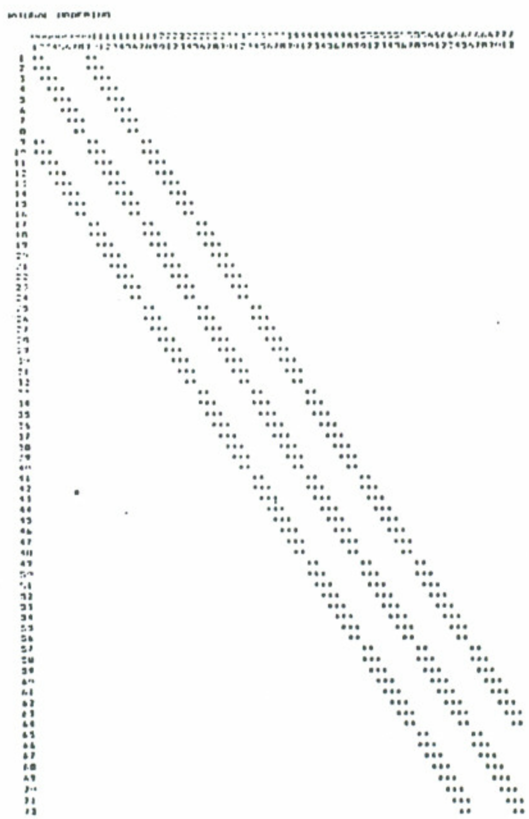
- Attempt to reduce the bandwidth, variable bandwidth (profile, envelope, skyline) or frontwidth.



ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

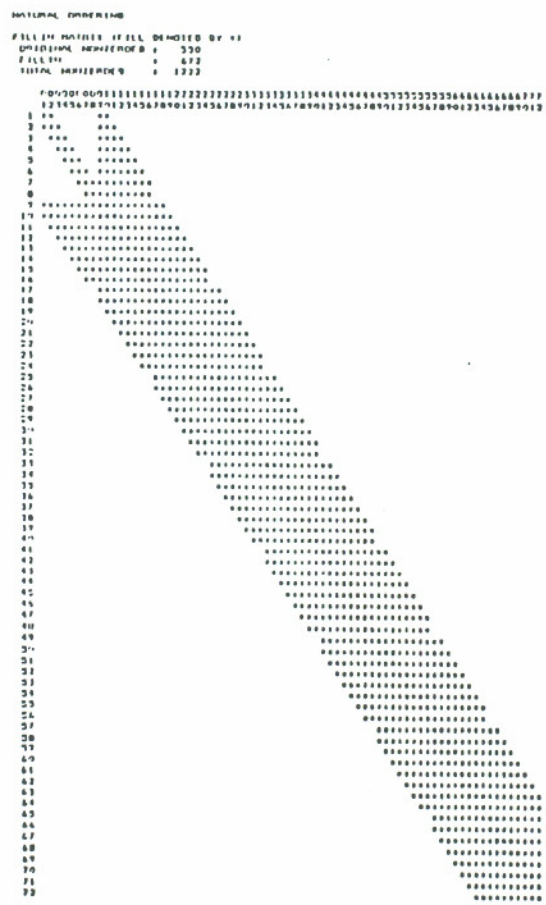
- Attempt to reduce the bandwidth, variable bandwidth (profile, envelope, skyline) or frontwidth.



ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

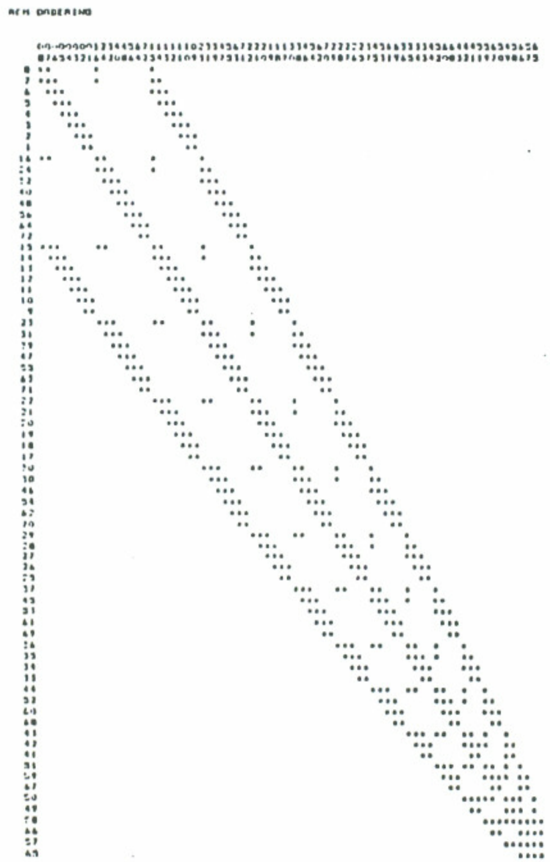
- Attempt to reduce the bandwidth, variable bandwidth (profile, envelope, skyline) or frontwidth.



ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

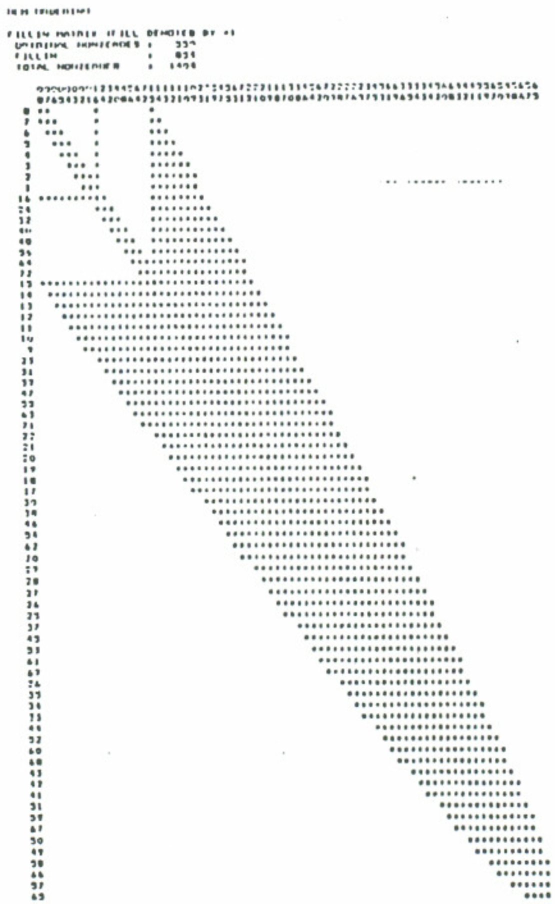
- Attempt to reduce the bandwidth, variable bandwidth (profile, envelope, skyline) or frontwidth.



ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

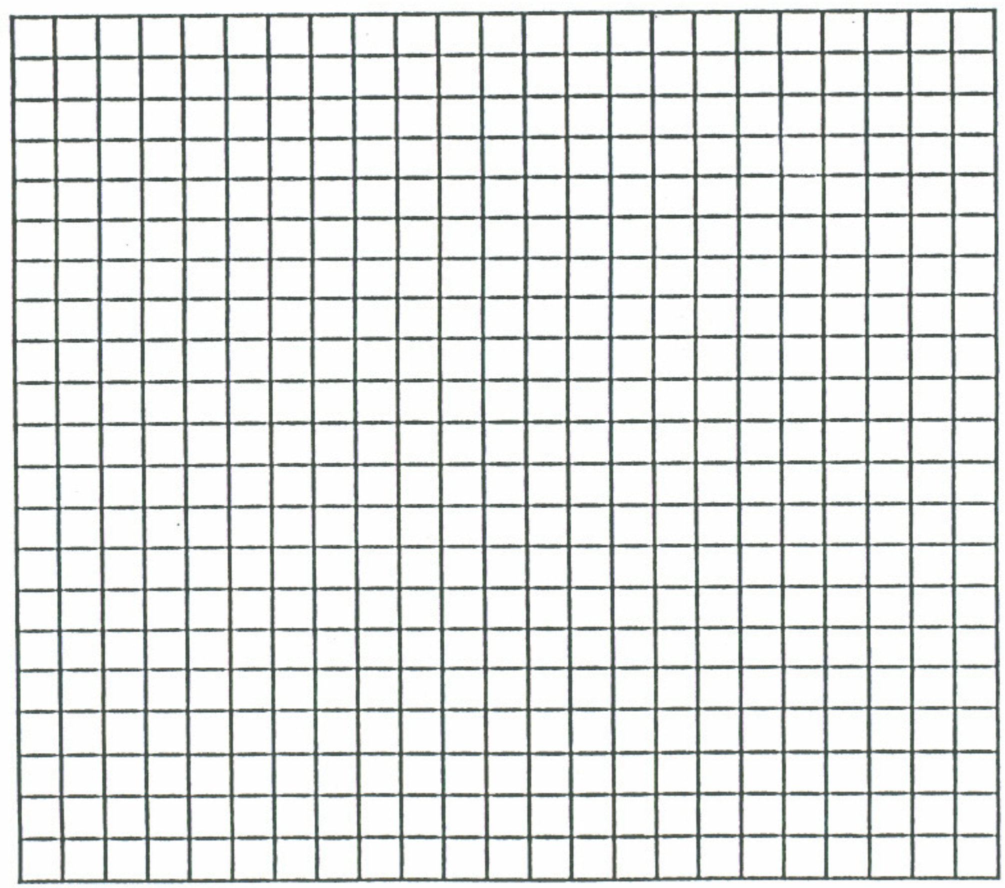
- Attempt to reduce the bandwidth, variable bandwidth (profile, envelope, skyline) or frontwidth.



NESTED DISSECTION REORDERING

Direct Sparse Linear Equations on Supercomputers

- Computer science "divide and conquer" strategy.



NESTED DISSECTION REORDERING
Direct Sparse Linear Equations on Supercomputers

- Computer science "divide and conquer" strategy.

NESTED DISSECTION REORDERING

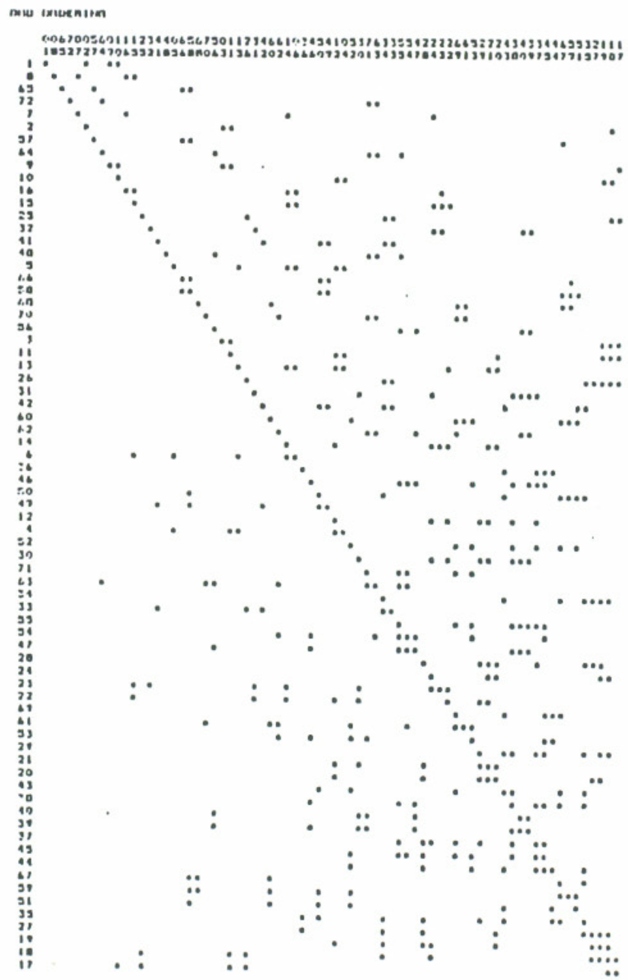
Direct Sparse Linear Equations on Supercomputers

- Computer science "divide and conquer" strategy.

ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

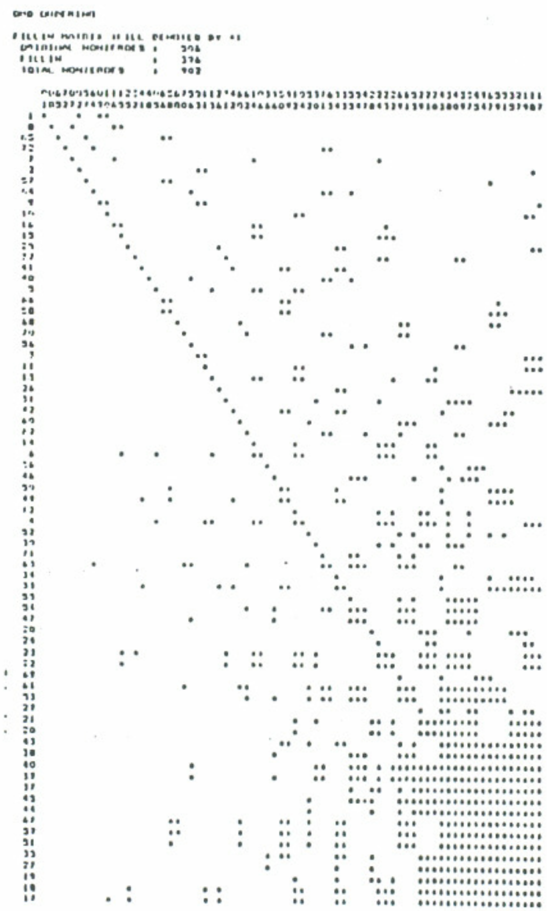
- Minimum Degree Reordering



ENVELOPE REORDERING

Direct Sparse Linear Equations on Supercomputers

- Minimum Degree Reordering



BANDED VERSUS GENERAL SPARSE METHODS IN THE PAST

The Impact of Hardware Gather/Scatter on Sparse Gaussian Elimination

- Banded/Profile/Skyline Schemes:
  - -- more storage
  - -- more operations
  - -- easy ordering
  - -- easy vectorization
  - -- easy out-of-core

- General Sparse Schemes:
  - -- less storage
  - -- less operations
  - -- difficult ordering
  - -- no vectorization
  - --difficult out-of-core

- However . . .

BANDED VERSUS GENERAL SPARSE METHODS (CON'T)

The Impact of Hardware Gather/Scatter on Sparse Gaussian Elimination

- Recent Advances in Algorithm Reseach

  -- multiple minimum degree algorithm (J. Liu, 1985)

  ⇒ easy ordering for general sparse methods

  -- general sparse out-of-core scheme (J. Liu, 1985)

- Recent Advances in Hardware

  -- Hardware gather/scatter on the Cray X-MP allows execution of general sparse schemes at vector speed.

OUTLINE

- Review of terminology and basic concepts
- Progress in ordering methods: multiple minimum degree
- Progress in hardware: impact of gather/scatter
- Progress in out-of-core schemes: Liu's method and multifrontal
- Conclusions

SAMPLE PROBLEMS

Test Problem	DOF
BCSSTK24: Winter Sports Arena	3562
BCSSTK15: Off Shore Platform	3948
BCSSTK28: Solid Element Model	4410
BCSSTK16: Corps of Engineers Dam	4884
BCSSTK18: Pin Boss (Auto Steering Component)	8738
BCSSTK29: Boeing 767 Rear Bulkhead	13992
BCSSTK25: 76 Story Skyscraper (Columbia Center)	15439
BCSSTK30: Generator Platform	28924
BCSSTK31: Automobile Component	35884
BCSSTK32: Automobile Chassis	44608

OUTLINE

- Review of terminology and basic concepts
- Progress in ordering methods: multiple minimum degree
- Progress in hardware: impact of gather/scatter
- Progress in out-of-core schemes: Liu's method and multifrontal
- Conclusions

DATA STRUCTURES FOR GENERAL SPARSE FACTORIZATION

Direct Sparse Linear Equations on Supercomputers

- Matrix entries are scattered, columns of factored matrix are stored in compact form

  $a_1, a_2, a_3, \ldots, a_p$
  $i_1, i_2, i_3, \ldots, i_p$

- Inner loop of factorization

```
      DO  100 I = 1, P
          Y (INDEX (I)) = A * X(I) + Y(INDEX(I))
100   Continue
```

- SAXPYI

  -- 88 mflops at $n = \infty$
  -- 53 mflops at $n = 64$
  -- 14 mflops at $n = 10$
  -- 3 mflops at $n = 2$

SAXPYI SPEED ON THE CRAY X-MP/24 USING 1 CPU (RATES GIVEN IN MEGAFLOPS)

	M = 10	M = infinity
(ignoring the hardware for gather/scatter)		
CFT 1.13	5.0	5.7
VectorPak	6.3	14.5
(using the hardware for gather/scatter)		
CFT 1.14	13.8	66.3
VectorPak	15.5	88.1

EXECUTION TIMES (SEC) FOR FACTORIZATION AND SOLUTION ROUTINES

The Impact of Hardware Gather/Scatter on Sparse Gaussian Elimination

CRAY X-MP • VectorPak

Problem	Factorization Time		Factorization MFLOPS	
	RCM	MD	RCM	MMD
STK3562	3.4	1.2	26	26
STK3948	7.4	4.0	38	41
STK4884	4.0	4.1	38	41
ST10974	*	4.8	*	30
ST11948	*	3.8	*	37
ST15439	17.4	7.7	27	40
LRGPWR	3.6	0.1	27	3

RELATIVE EXECUTION TIMES FOR SPARSE MATRIX FACTORIZATION

The Impact of Hardware Gather/Scatter on Sparse Gaussian Elimination (normalized so that VectorPak with g/s-1.00)

Sports Arena (N = 3562)	CFT 1.14	w/out g/s	6.3
	VectorPak	w/out g/s	2.5
	CFT 1.14	w/ g/s	1.1
	VectorPak	w/ g/s	1.0
Columbia Center (N = [redacted]) 15439	CFT 1.14	w/out g/s	8.7
	VectorPak	w/out g/s	3.1
	CFT 1.14	w/ g/s	1.2
	VectorPak	w/ g/s	1.0

OUTLINE

- Review of terminology and basic concepts
- Progress in ordering methods: multiple minimum degree
- Progress in hardware: impact of gather/scatter
- Progress in out-of-core schemes: Liu's method and multifrontal
- Conclusions

Out-of-core Factorization

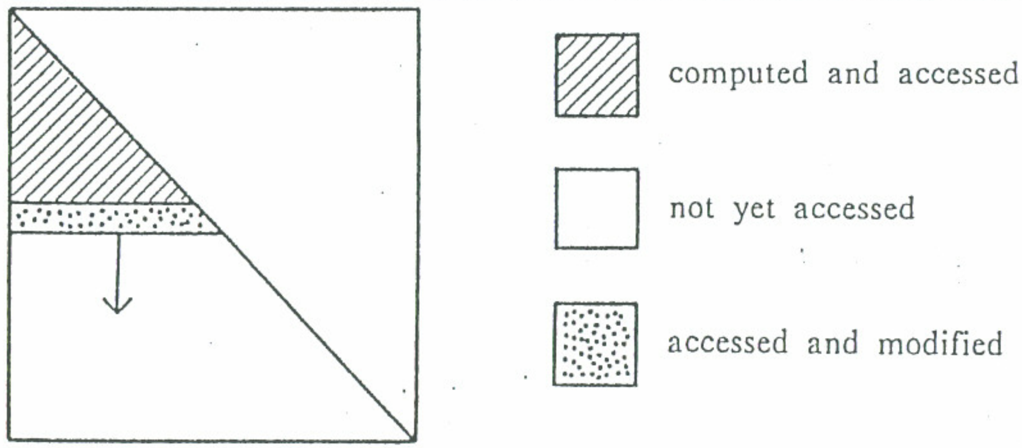
Three ways of organizing the factorization:

bordering method

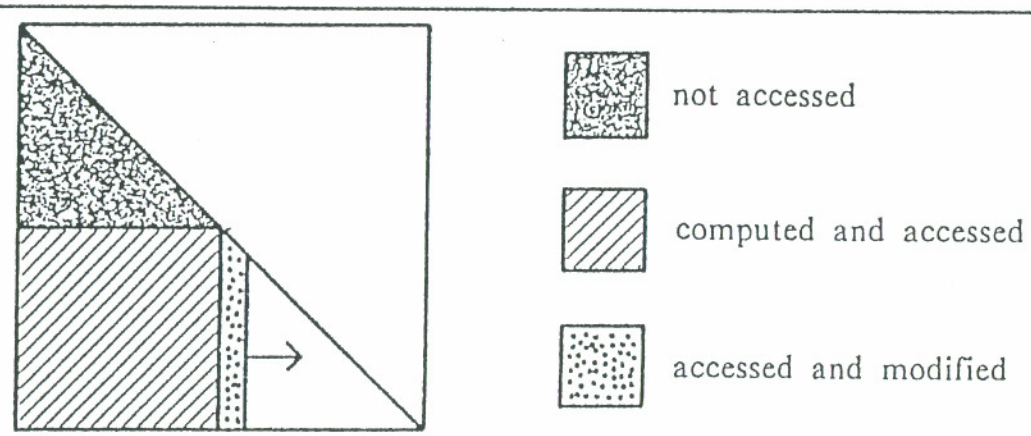
inner product method

outer product method

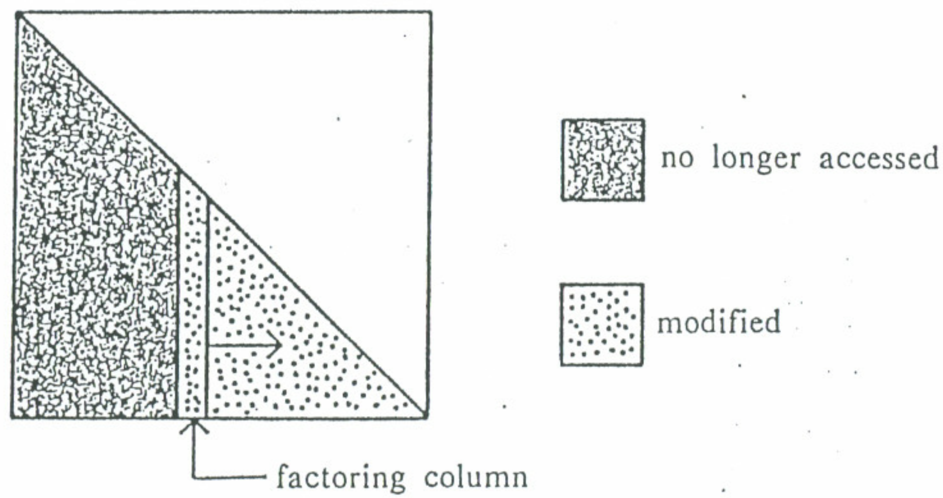
Bordering Method



Inner Product Method



Outer Product Method

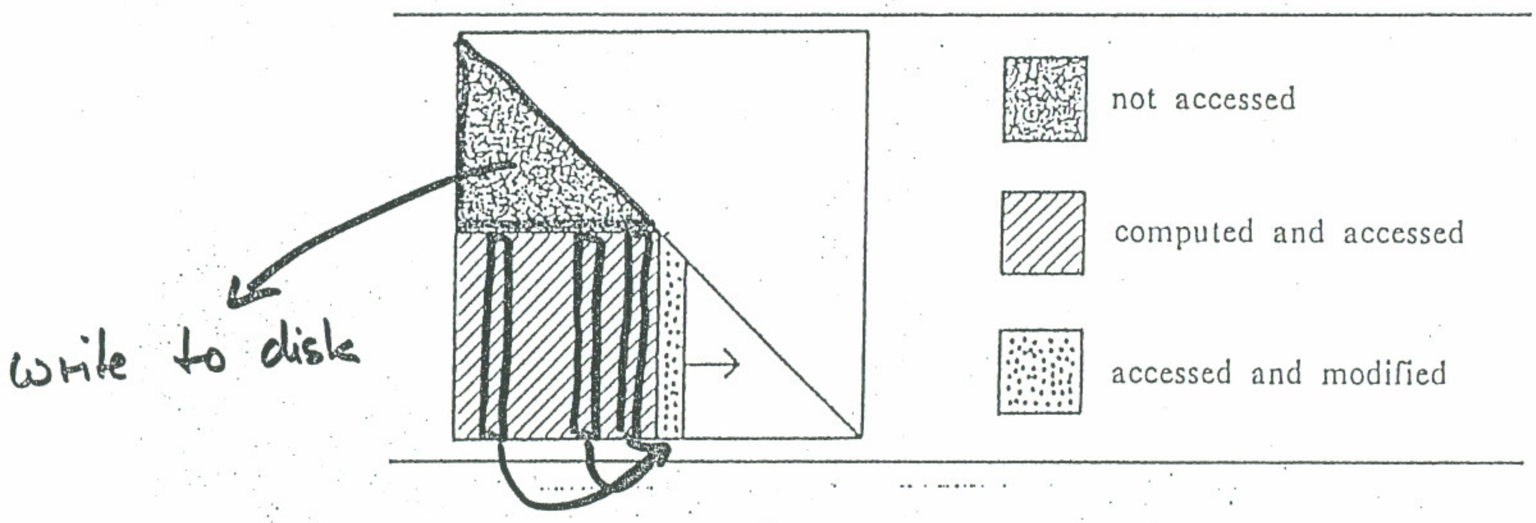


Out-of-core General Sparse Factorization

- Write entries in upper triangle to secondary storage
- Garbage collection as needed
- Slight increase of workspace decreases garbage collections
- Depth first traversal of elimination tree with special heuristic
- Use of SAXPYI
- Subscript compression open problem

Inner Product Method

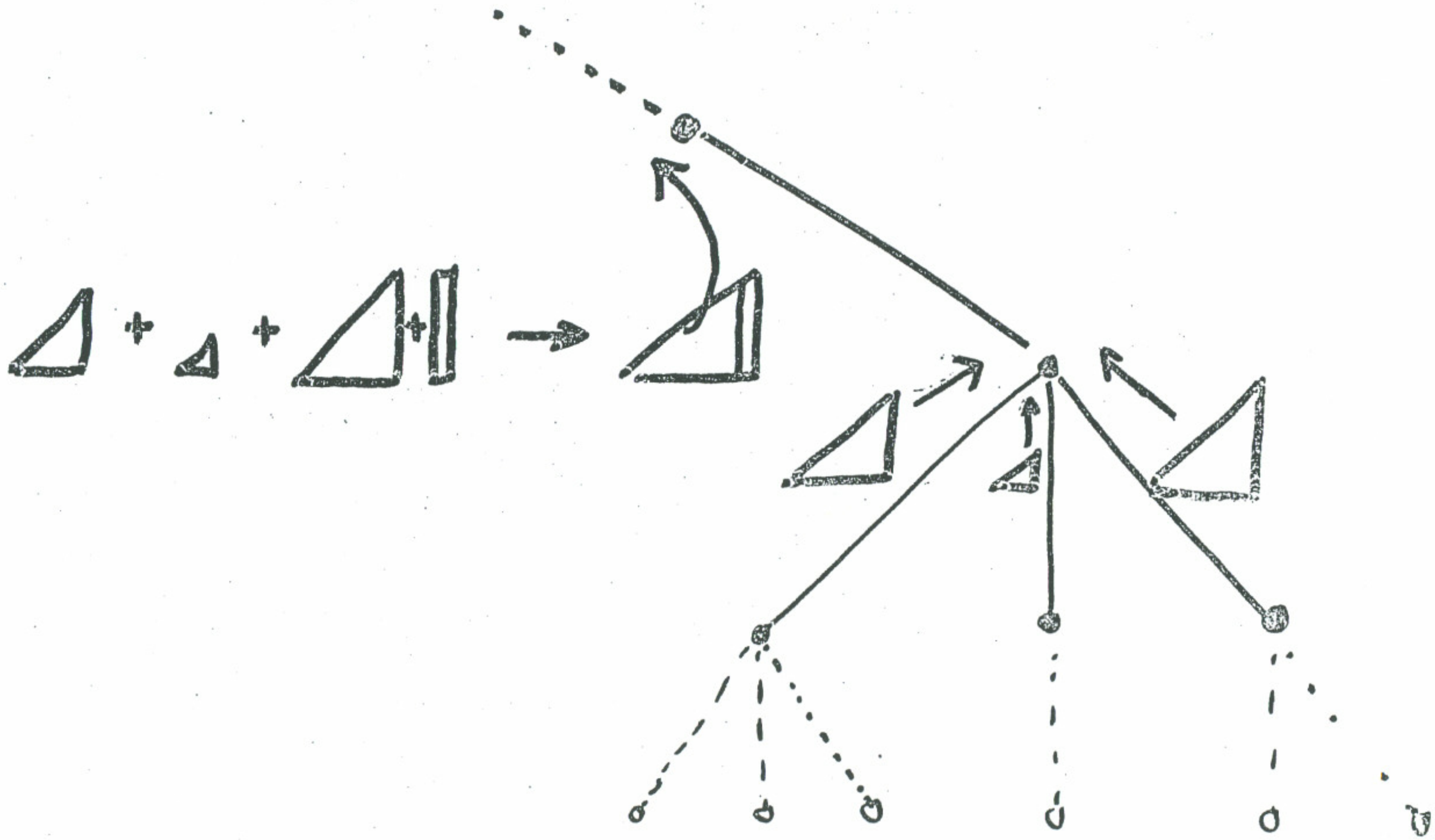
Liu's algorithm



Multifrontal Methods (Duff)

- Derived from frontal methods in finite element codes
- Simultaneous assembly and factorization
- Elimination of a node, when all children have been eliminated
- Update of frontal matrix by merging updates of children
- Compute new frontal matrix by rank one updates
- Keep frontal matrix on stack for parent update

Multifrontal Method



Multifrontal Method

Liu's General Sparse versus Multifrontal

General Sparse	Multifrontal
pull from behind	push forward
heap of columns	stack of frontal matrices
updates from descendants	updates from children
SAXPYI	SAXPYI and SAXPY
explicit indices	implicit indices
vector operations	matrix operations

SAMPLE PROBLEMS

Test Problem	General Sparse	Multifrontal
BCSSTK28	60467	43884
BCSSTK29	301065	223251
BCSSTK30	410095	354271
BCSSTK31	1278817	1099804
BCSSTK32	658654	418329