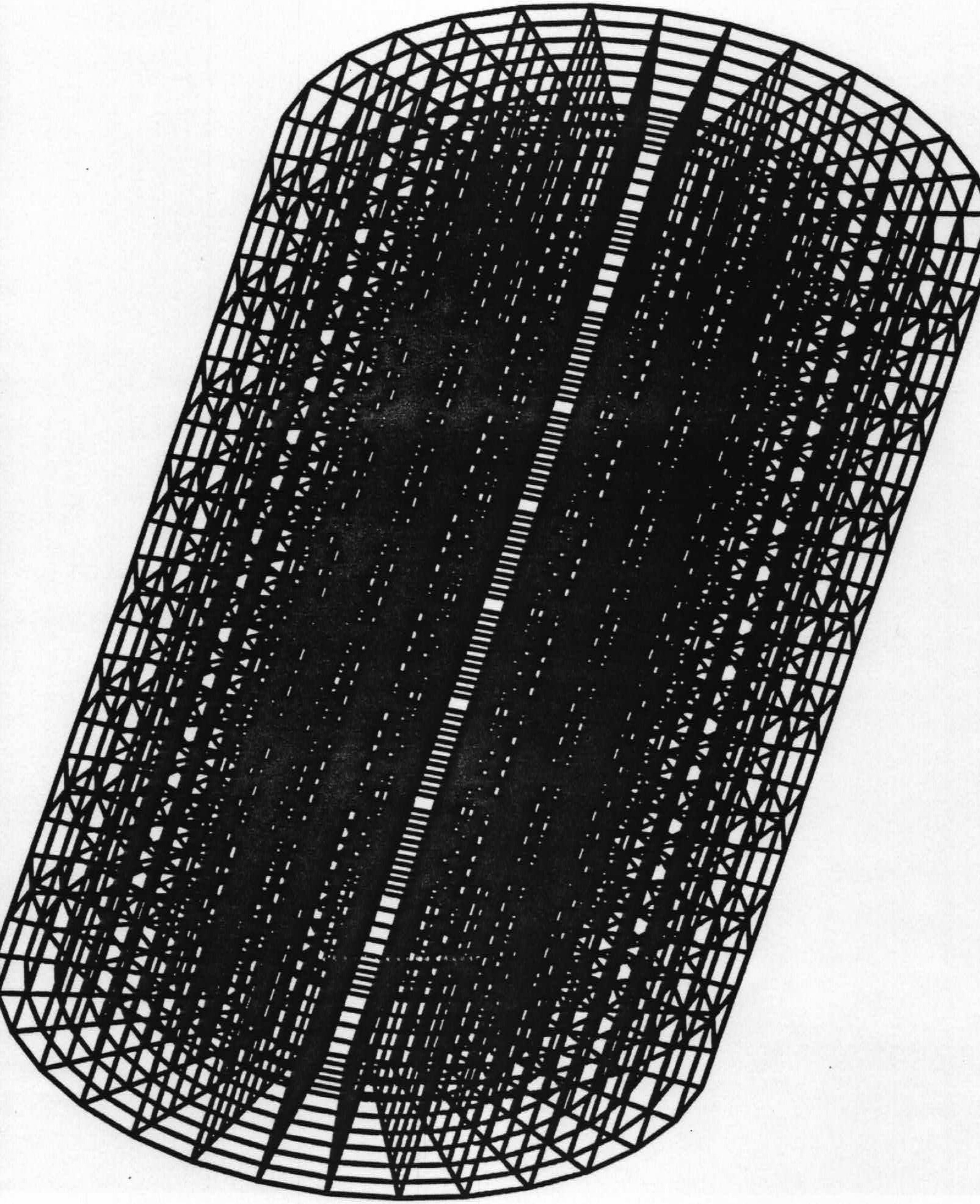
A Data Parallel Implementation of the Finite Element Method

Kapil K. Mathur
S. Lennart Johnsson

Discretized Equations

- $[K]\{u\} = \{f\}$
- $[K] = \sum_{i}^{n} [K^{(i)}]$
- $\{f\} = \sum_{i}^{n} \{f^{(i)}\}$
- $[K]$: Global stiffness matrix.
- $[K^{(i)}]$: Elemental stiffness matrix.
- Characteristics of $[K]$
  - Typical size $\sim 100,000 - 1,000,000$.
  - Sparse often banded.
  - Poorly conditioned, especially for matrices arising from structural applications.



The Finite Element Method

Flowchart

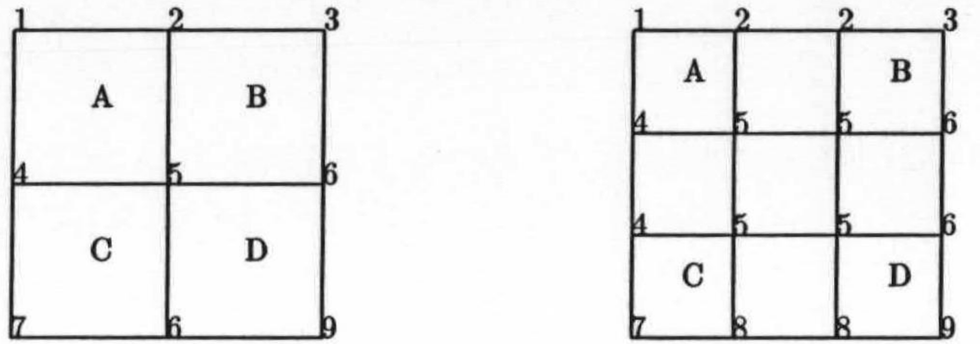
- **Mesh generation** : Discretize the solid into a set of finite elements. This set will in general contain finite elements of different types, for example, bricks, tetrahedrons, and triangular prisms.

- **Local interactions** : Generate the local stiffness matrices corresponding to all elements in the mesh.

- **Global interactions** : Create the global stiffness matrix by assembling the local matrices (if desired).

- **Solution of linear system** : Solve global system of equations
  - Direct solvers (banded LU decomposition).
  - Iterative solvers (conjugate gradient method, multigrid techniques).

Mapping the Computational Domain on the Connection Machine



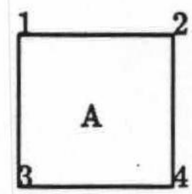
Finite element per processor

Unassembled nodal point per processor

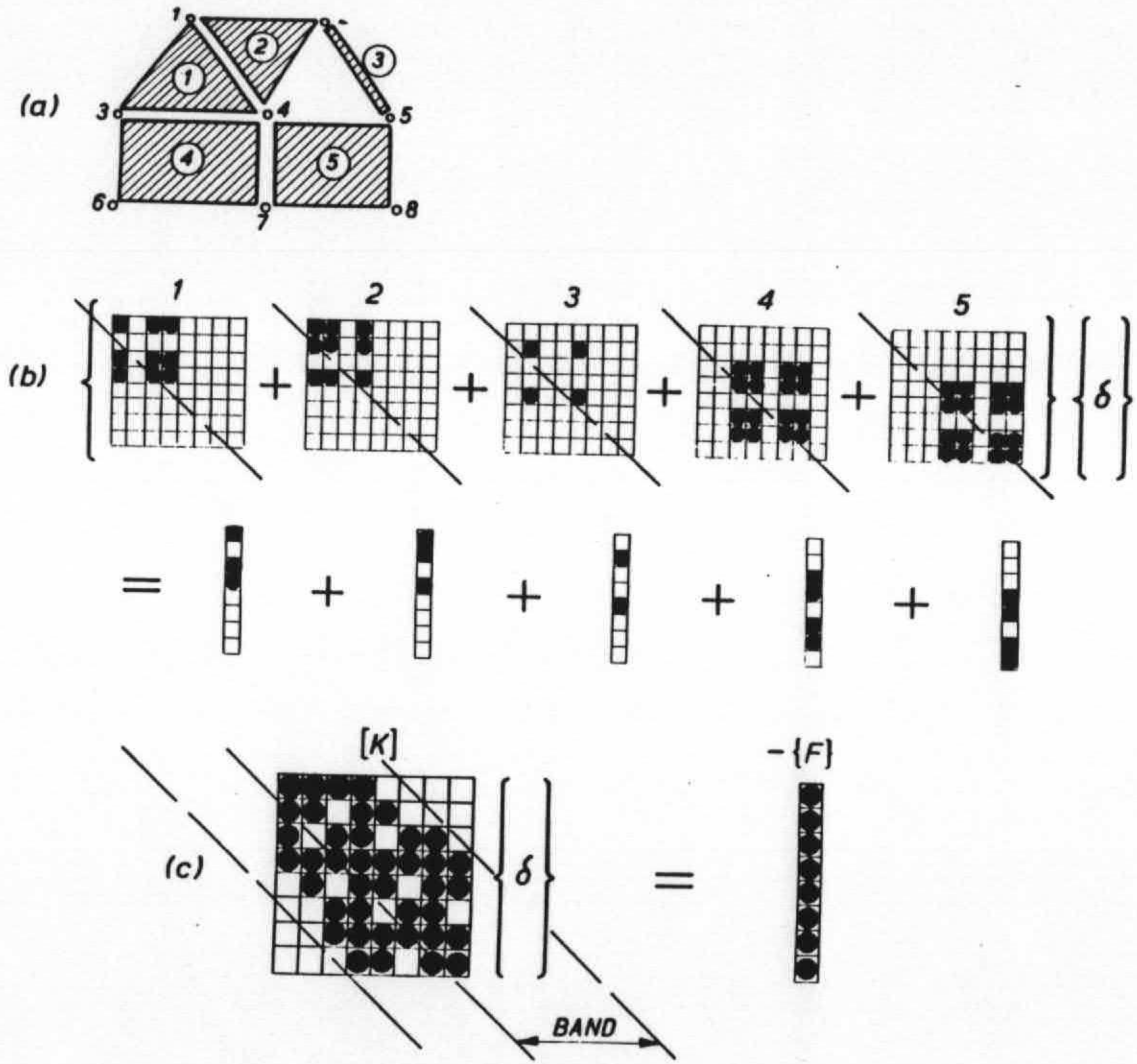
The current implementation uses the second representation.

- Storage requirements are uniform per virtual processor.
- Ensures complete load balance.

Generating the Elemental Stiffness Matrices



- Each elemental stiffness matrix (in 2D) – $k(8, 8)$.
- Four processors share the computational effort for evaluating k.
- Each processor stores and computes 2 rows of k.

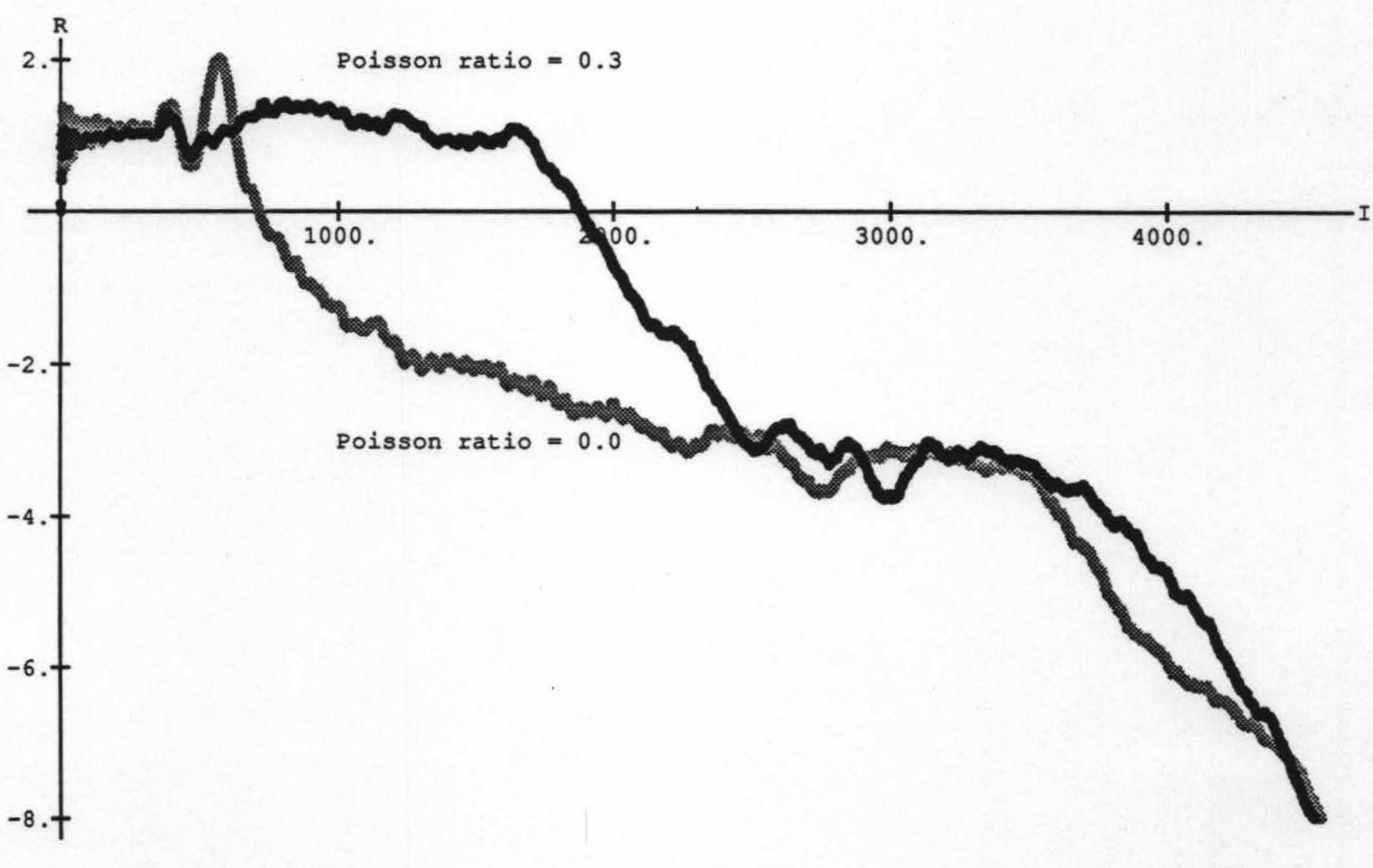

(a)
1
2
3
4
5
6
7
8
(b)
δ
(c)
[K]
δ
-{F}
BAND

Iterative Methods

Conjugate Gradient Method

initialize x
loop until convergence
  compute residual : $r = b - Ax$
  compute acceleration parameters
  evaluate new estimate for x
end loop

A typical iteration process



Data Level Parallelism and The Finite Element Method

- Data level programming is very efficient for creating the local data structures.
  - Nonlinear finite element simulations spend > 70% of the computational effort in creating the local data structures.
  - A data level programming environment has great advantages in creating the local data structures.
- Solution of the linear system by either a band solver or an iterative solver are communications intensive.
- With a good preconditioner an iterative solver can be a big win.

Performance

Generating elemental stiffness matrices

Clock rate 7MHz; virtual processor ratio = 1

Interpolation Order	Number of nodes per element	Quadrature Order	CM time Sun–4	CM time Symbolics
$1 \times 1 \times 1$	8	$2 \times 2 \times 2$	0.233	0.231
$2 \times 2 \times 2$	27	$2 \times 2 \times 2$	0.634	0.726
$2 \times 2 \times 2$	27	$3 \times 3 \times 3$	2.641	2.441
$3 \times 3 \times 3$	64	$3 \times 3 \times 3$	5.297	5.627
$3 \times 3 \times 3$	64	$4 \times 4 \times 4$	12.144	13.445

Performance on a full machine:
$\sim 1.5 - 1.9$ GFlops s^{-1} at vpr = 1

Iterative solver

	Time (milli–second)	%
"all–to–all" broadcasting	9.3	40.8
Local matrix vector product	3.8	16.7
Assembly	5.2	22.8
Acceleration parameters	1.9	8.3
Update displacement vector	2.6	11.4
Time per iteration	22.8	100.0

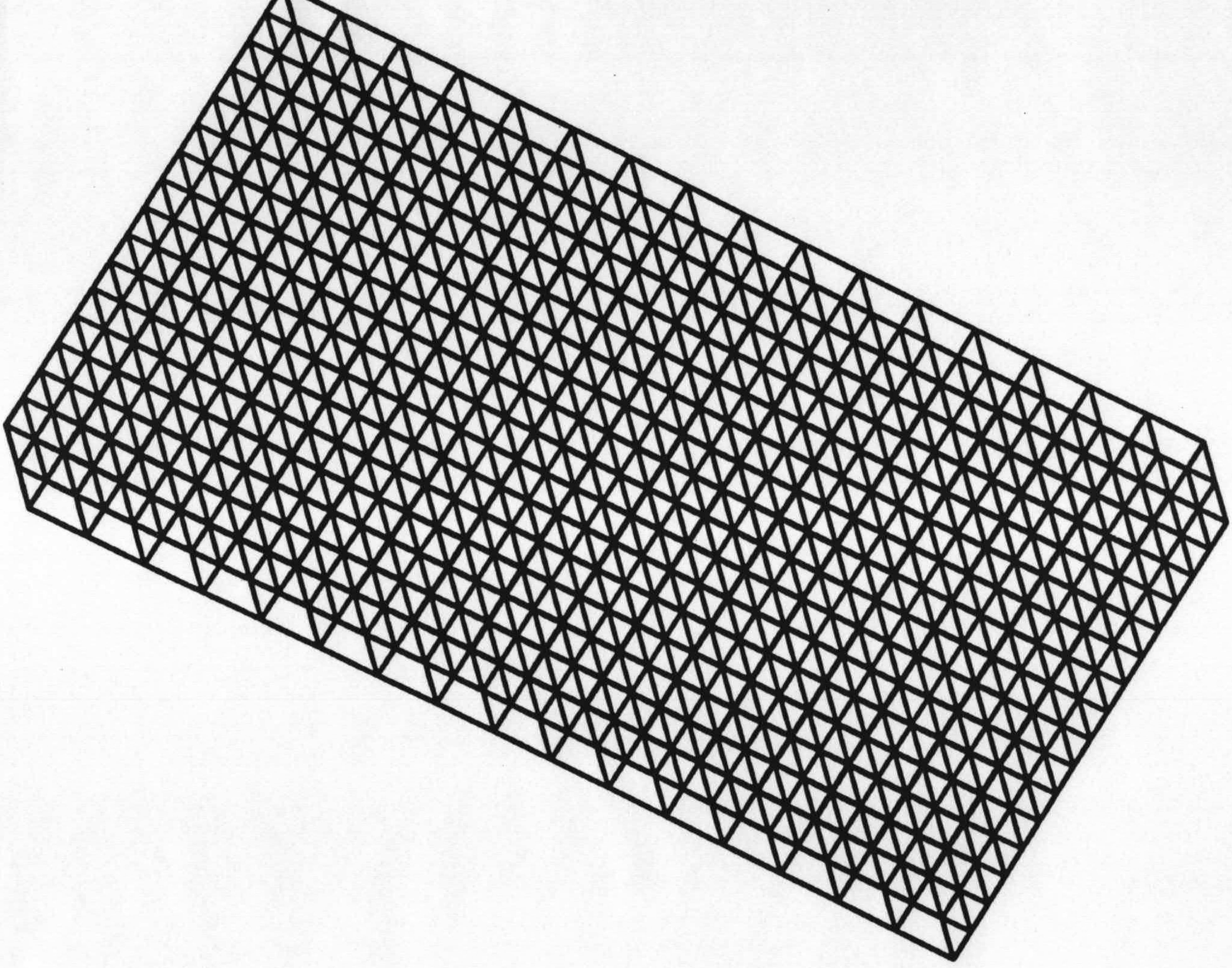
CMSSL Primitives

Communication:

- *All to all broadcast* : Communication among subset of processors representing nodal points on a finite element.
- *Assembly*: Reduction over all shared nodes.
- *Global reduction*: Global reduction over all nodal points.

Arithmetic involving multiple occurrences:

- *Matrix vector multiply*: to compute the sparse matrix vector product in the iterative solver.
- *Matrix matrix multiply*: to evaluate the transformation matrices.
- *Matrix inversion*: to evaluate the transformation matrices.



Performance comparisons

- Domain and boundary conditions:

  Cantilevered plate simulation.

  8–node 3–dimensional solid isoparametric elements.

  Force on the free end of the plate.

- Discretization:

  10 elements along the length.

  400 elements along the width.

  1 element through the thickness.

  4,000 elements; 8822 nodes.

  26,466 degrees of freedom.

  24,060 active degrees of freedom.

- 64K CM–2:

  Geometry: $32 \times 1024 \times 2$.

  Virtual processor ratio = 1.

  1. Stiffness generation = 0.23 s.
  2. Estimated solution time = 207 s (iterative solver in double precision).

- Cray XMP/48:

  1. Stiffness generation = 27.20 s.
  2. Estimated solution time = 1100 s (frontal solver).

- IBM–3090 200VF:

  1. Stiffness generation = 243 s
  2. Estimated solution time = 5600 s (frontal solver).