Mondriaan, partitioning software for sparse matrix computations

Rob Bisseling and Brendan Vastenhouw
Rob.Bisseling@math.uu.nl
http://www.math.uu.nl/people/bisseling

Department of Mathematics
Utrecht University
Outline

- Mondriaan: sparse matrix-vector multiplication, partitioning matrix and vectors for parallel computations
- Applications in physics: DNA electrophoresis, amorphous silicon
- Software issues: GNU, C, BSP, MPI
Sparse matrix-vector multiplication \( u := Av \)

\( A \) sparse \( m \times n \) matrix
\( u \) dense \( m \)-vector
\( v \) dense \( n \)-vector

- Sequential computation

\[
 u_i := \sum_{j=0}^{m-1} a_{ij} v_j
\]

- Important for iterative solvers: linear systems, eigensystems

- Models interaction \( a_{ij} \) between particles \( i, j \)
Parallel sparse matrix-vector multiplication

Processor \( s \) \((0 \leq s < p)\) participates in four phases:

1. sends its vector components \( v_j \) to processors with a nonzero \( a_{ij} \) in matrix column \( j \);
2. computes products \( a_{ij}v_j \) for its nonzeros \( a_{ij} \) and adds the results into a contribution \( u_{is} \);
3. sends its nonzero contributions \( u_{is} \) to the processor that owns \( u_i \);
4. adds received contributions \( u_i = \sum_{t=0}^{p-1} u_{it} \);
Cartesian matrix partitioning

- Block distribution of $59 \times 59$ matrix \texttt{impcol\_b} with 312 nonzeros, for $p = 4$
- \#nonzeros per processor: 126, 28, 128, 30
Non-Cartesian matrix partitioning

- Block distribution of $59 \times 59$ matrix $\text{impcol}_b$ with 312 nonzeros, for $p = 4$
- #nonzeros per processor: 76, 76, 80, 80
Composition with Red, Yellow, Blue and Black

Piet Mondriaan 1921
Theorem. Given $A$: $m \times n$ matrix, $A_0, \ldots, A_k$ mutually disjoint subsets of $A$ ($k \geq 1$). Then

$$V(A_0, \ldots, A_k) = V(A_0, \ldots, A_{k-2}, A_{k-1} \cup A_k) + V(A_{k-1}, A_k).$$

Here $V(A_0, \ldots, A_k)$ is the matrix-vector communication volume corresponding to the subsets $A_0, \ldots, A_k$.

$\Rightarrow$ each split can be done independently
Recursive bipartitioning algorithm (alternating)

MatrixPartition\((A, \text{sign}, p, \epsilon)\)

*input:*  \(\text{sign} \): direction of first bipartitioning  
\(\epsilon \): allowed load imbalance, \(\epsilon > 0\).

*output:* \(p\)-way partitioning of \(A\) with imbalance \(\leq \epsilon\).

\[
\begin{align*}
\text{if } p & > 1 \text{ then} \\
& \quad q := \log_2 p; \\
& \quad (A_0, A_1) := h(A, \text{sign}, \epsilon/q); \text{ magic bipartitioning} \\
& \quad \text{maxnz} := \frac{\text{nz}(A)}{p} (1 + \epsilon); \\
& \quad \epsilon_0 := \frac{\text{maxnz}}{\text{nz}(A_0)} \cdot \frac{p}{2} - 1; \\
& \quad \epsilon_1 := \frac{\text{maxnz}}{\text{nz}(A_1)} \cdot \frac{p}{2} - 1; \\
& \quad \text{MatrixPartition}(A_0, -\text{sign}, \epsilon_0, p/2); \\
& \quad \text{MatrixPartition}(A_1, -\text{sign}, \epsilon_1, p/2); \\
\text{else} & \text{ output } A;
\end{align*}
\]
Matrix partitioning: try both directions, choose the best

Vector partitioning: $v_j \rightarrow$ one of the owners of a nonzero in matrix column $j$, $u_i \rightarrow$ owner in matrix row $i$
Broadway Boogie Woogie

Piet Mondriaan 1942-43
Local view

- First horizontal split, then two independent vertical splits
- Empty parts: no communication, no further splits
- Submatrix sizes: $27 \times 21$, $26 \times 23$, $27 \times 24$, $24 \times 22$
Application: cage model for DNA electrophoresis


- 3D cubic lattice models a gel
- DNA polymer reptates: kinks, end points move
- DNA sequencing machines: electric field $E$

Our aim: study drift velocity $v(E)$. 
Transition matrix of Markov model

Reduced transition matrix for polymer length $L = 5$.

Polymer state $\sim$ binary number $\sim$ vector component
Nonzero $\sim$ allowed move between two states

Heuristic vector partitioning based on physical structure:
$p = 8$. Induced matrix partitioning into 64 submatrices,
some empty. Assign these to 8 processors.
Partitioning results: Mondriaan vs. heuristic

- Reduced transition matrix for polymer length \( L = 12 \).
  \( n = 130228, \text{nz}(A) = 2032536 \).
  Reduction factor by exploiting symmetries: 2786.
- \( p = 8 \) processors, \( \epsilon = 3\% \) load imbalance.
- Mondriaan version 1.0 (May 10, 2002),
  \( \text{distr}(u) = \text{distr}(v), \text{distr}(a_{ij}) = \text{distr}(a_{ji}) \),
  Total communication volume: 70632 data words.
- Computation balance: \( \text{avg} = 508134 \) max = 523370 flops
  Communication balance: \( \text{avg} = 8829 \) max = 13153 words
- BSP cost:
  \[
  523370 + 13153 \, g + 4l \quad (\text{Mondriaan})
  \]
  \[
  545156 + 64716 \, g + 2l \quad (\text{heuristic})
  \]
  \( g = \) communication time per data word
  \( l = \) synchronisation time
Application: 20000-atom model of amorphous silicon


- Every atom has 4 bonds
- Bond transposition is tried; system relaxed globally until minimum energy achieved. Repeated many times.
Simple Cubic distribution

- Split cubic simulation box into $p = k^3$ subdomains
- Surface-to-volume (S/V) ratio
  = Communication-to-computation ratio = $6p^{1/3}$
Face Centered Cubic sphere packing

Market, San Christóbal de las Casas, Mexico (1993)

- FCC is proven densest sphere packing in 3D (Hales 1998).
BCC is less dense sphere packing in 3D, but best single-cell space partitioning known so far for minimising surface area (Kelvin conjecture 1887)

- Voronoi cell is truncated octahedron. S/V ratio = $5.31 p^{1/3}$.
- Even better: sphere with S/V ratio = $4.83 p^{1/3}$, but can’t fill space!
Body Centered Cubic distribution

- Split cubic simulation box into $p = 2k^3$ subdomains.
  (Can be generalised to $p = 2k_1k_2k_3$.)

Mondriaan - CECAM Workshop Open Source Software, June 2002 – p.20
Partitioning: Mondriaan vs. geometric

- Create particle matrix for 20000 particles:
  - $a_{ij} \neq 0$ if particle $i$ connected to particle $j$.
  - 4 bonds + self-connectivity $\Rightarrow$ 5 nonzeros per row.
  - $n = 20000$, $n\text{z}(A) = 100000$.

- Run 1D Mondriaan version 1.0 with:
  - $\text{distr}(\mathbf{u}) = \text{distr}(\mathbf{v})$, $\text{distr}(a_{ij}) = \text{distr}(a_{ji})$,
  - $p = 16$ processors, $\epsilon = 3\%$ load imbalance.

- Convert vector distribution to particle distribution:
  if $u_i \mapsto P(s)$ then particle $i \mapsto P(s)$
Partitioning results: Mondriaan vs. geometric

- Interior = set of particles inside processor
- Halo = set of particles outside processor, within distance of 2 bonds

<table>
<thead>
<tr>
<th>Partitioning method</th>
<th>interior max</th>
<th>interior avg</th>
<th>halo max</th>
<th>halo avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple cubic</td>
<td>1284</td>
<td>1250</td>
<td>1054</td>
<td>1033</td>
</tr>
<tr>
<td>Mondriaan $A$</td>
<td>1287</td>
<td>1250</td>
<td>1157</td>
<td>1013</td>
</tr>
<tr>
<td>Mondriaan $A^2$</td>
<td>1287</td>
<td>1250</td>
<td>1049</td>
<td>974</td>
</tr>
<tr>
<td>BCC</td>
<td>1277</td>
<td>1250</td>
<td>904</td>
<td>874</td>
</tr>
</tbody>
</table>
Software issues

- Mondriaan version 1.0 released May 10, 2002 under GNU public license. Freedom to adapt to your needs.
- Written in C, in object-oriented style, but without the guarantees of C++. Sequential program.

http://www.math.uu.nl/people/bisseling/Mondriaan
Conclusions and future work

- Mondriaan is a powerful general-purpose partitioner, often performing as well as application-specific partitioners: polymer configurations, many-particle systems.

- Current and future work:
  - Parallel version in BSPlib, MPI.
  - Templates package of iterative solvers in C++, BSPlib, MPI using Mondriaan partitioning.
  - Applications . . .