# Sparse matrix partitioning, ordering, and visualisation by Mondriaan 3.0


Rob H. Bisseling, Albert-Jan Yzelman, Bas Fagginger Auer

Mathematical Institute, Utrecht University
Rob Bisseling: also joint Laboratory CERFACS/INRIA, Toulouse, May–July 2010



Albert-Jan


Bas

PMAA'10, Basel, July 1, 2010


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions


Universiteit Utrecht

# Motivation: supercomputer 109/500 (June 2010)

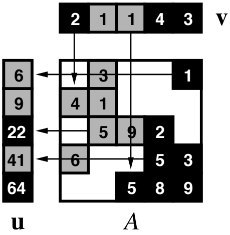
- National supercomputer Huygens named after Christiaan Huygens. Wikipedia: . . . Ausserdem konnte er durch die bessere Auflösung seines Teleskops erkennen, dass das, was Galilei als Ohren des Saturns bezeichnet hatte, in Wirklichkeit die Saturnringe waren."
- Huygens, the machine, has 104 nodes
- Each node has 16 processors
- Each processor has 2 cores and a a shared L3 cache
- Each core has a local L1 and L2 cache

# Parallel sparse matrix–vector multiplication $\mathbf{u} := A\mathbf{v}$



$A$ sparse $m \times n$ matrix, $\mathbf{u}$ dense $m$-vector, $\mathbf{v}$ dense $n$-vector

$$u_i := \sum_{j=0}^{n-1} a_{ij} v_j$$




$p = 2$

4 supersteps: communicate, compute, communicate, compute

# Divide evenly over 4 processors

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Avoid communication completely, if you can

All nonzeros in a row or column have the same colour.

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

Universiteit Utrecht

# Permute the matrix rows/columns

First the green rows/columns, then the blue ones.

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

Universiteit Utrecht

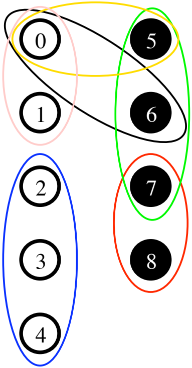
# Combinatorial problem: sparse matrix partitioning

Problem: Split the set of nonzeros $A$ of the matrix into $p$ subsets, $A_0, A_1, \ldots, A_{p-1}$, minimising the communication volume $V(A_0, A_1, \ldots, A_{p-1})$ under the load imbalance constraint

$$nz(A_i) \leq \frac{nz(A)}{p}(1+\epsilon), \quad 0 \leq i < p.$$


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

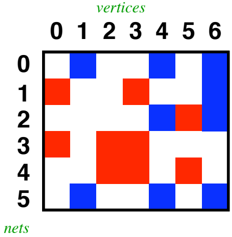
# The hypergraph connection




Hypergraph with 9 vertices and 6 hyperedges (nets), partitioned over 2 processors, black and white

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# 1D matrix partitioning using hypergraphs




- Hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{N}) \Rightarrow$ exact communication volume in sparse matrix–vector multiplication.
- Columns $\equiv$ Vertices: $0, 1, 2, 3, 4, 5, 6$.
  Rows $\equiv$ Hyperedges (nets, subsets of $\mathcal{V}$):

$$n_0 = \{1,4,6\}, \quad n_1 = \{0,3,6\}, \quad n_2 = \{4,5,6\},$$
$$n_3 = \{0,2,3\}, \quad n_4 = \{2,3,5\}, \quad n_5 = \{1,4,6\}.$$

# $(\lambda - 1)$-metric for hypergraph partitioning

- $138 \times 138$ symmetric matrix `bcsstk22`, $nz = 696$, $p = 8$
- Reordered to Bordered Block Diagonal (BBD) form
- Split of row $i$ over $\lambda_i$ processors causes a communication volume of $\lambda_i - 1$ data words


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Cut-net metric for hypergraph partitioning

- Row split has unit cost, irrespective of $\lambda_i$

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Mondriaan 2D matrix partitioning

- $p = 4$, $\epsilon = 0.2$, global non-permuted view

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Fine-grain 2D matrix partitioning

- Each individual nonzero is a vertex in the hypergraph
  Çatalyürek and Aykanat, 2001.

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Mondriaan 2.0, Released July 14, 2008

- New algorithms for vector partitioning.
- Much faster, by a factor of 10 compared to version 1.0.
- 10% better quality of the matrix partitioning.
- Inclusion of fine-grain partitioning method
- Inclusion of hybrid between original Mondriaan and fine-grain methods.
- Can also handle $p \neq 2^q$.

# Matrix lns3937 (Navier–Stokes, fluid flow)

Splitting the sparse matrix lns3937 into 5 parts.

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Recursive, adaptive bipartitioning algorithm

MatrixPartition$(A, p, \epsilon)$

*input:* $p$ = number of processors, $p = 2^q$

$\epsilon$ = allowed load imbalance, $\epsilon > 0$.

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

**if** $p > 1$ **then**

$q := \log_2 p$;

$(A_0^{\mathrm{r}}, A_1^{\mathrm{r}}) := h(A, \mathrm{row}, \epsilon/q)$; hypergraph splitting

$(A_0^{\mathrm{c}}, A_1^{\mathrm{c}}) := h(A, \mathrm{col}, \epsilon/q)$;

$(A_0^{\mathrm{f}}, A_1^{\mathrm{f}}) := h(A, \mathrm{fine}, \epsilon/q)$;

$(A_0, A_1)$ := best of $(A_0^{\mathrm{r}}, A_1^{\mathrm{r}})$, $(A_0^{\mathrm{c}}, A_1^{\mathrm{c}})$, $(A_0^{\mathrm{f}}, A_1^{\mathrm{f}})$;

$maxnz := \frac{nz(A)}{p}(1+\epsilon)$;

$\epsilon_0 := \frac{maxnz}{nz(A_0)} \cdot \frac{p}{2} - 1$; MatrixPartition$(A_0, p/2, \epsilon_0)$;

$\epsilon_1 := \frac{maxnz}{nz(A_1)} \cdot \frac{p}{2} - 1$; MatrixPartition$(A_1, p/2, \epsilon_1)$;

**else** output $A$;


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Mondriaan version 1 vs. 3 (Preliminary)

| Name | $p$ | v1.0 | v3.0 |
|---|---|---|---|
| df1001 | 4 | 1484 | 1404 |
| | 16 | 3713 | 3631 |
| | 64 | 6224 | 6071 |
| cre_b | 4 | 1872 | 1437 |
| | 16 | 4698 | 4144 |
| | 64 | 9214 | 9011 |
| tbdmatlab | 4 | 10857 | 10041 |
| | 16 | 28041 | 25117 |
| | 64 | 52467 | 50116 |
| nug30 | 4 | 55924 | 47984 |
| | 16 | 126255 | 110433 |
| | 64 | 212303 | 194083 |
| tbdlinux | 4 | 30667 | 29764 |
| | 16 | 73240 | 68132 |
| | 64 | 146771 | 139720 |

Mondriaan split strategy: v1 localbest, v3 hybrid, $\epsilon = 0.03$.

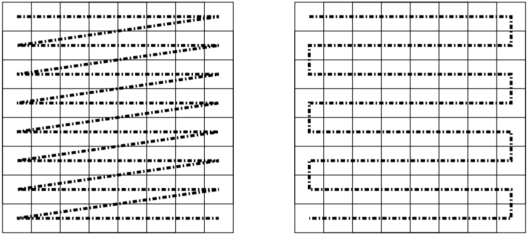

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Mondriaan 3.0 coming soon

- Ordering to SBD and BBD structure: cut rows are placed in the middle, and at the end, respectively.
- Visualisation through Matlab interface, MondriaanPlot, and MondriaanMovie
- Metrics: $\lambda - 1$ for parallelism, and cut-net for other applications
- Library-callable, so you can link it to your own program
- Interface to PaToH hypergraph partitioner


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

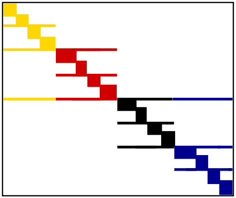
# Ordering a sparse matrix to improve cache use



- Compressed Row Storage (CRS, left) and zig-zag CRS (right) orderings.
- Zig-zag CRS avoids unnecessary end-of-row jumps in cache, thus improving access to the input vector in a matrix–vector multiplication.
- Yzelman and Bisseling, *SIAM Journal on Scientific Computing* 2009.


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Separated block-diagonal (SBD) structure



- SBD structure is obtained by recursively partitioning the columns of a sparse matrix, each time moving the cut (mixed) rows to the middle. Columns are permuted accordingly.
- Mondriaan is used in one-dimensional mode, splitting only in the column direction.
- The cut rows are sparse and serve as a gentle transition between accesses to two different vector parts.


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Partition the columns till the end, $p = n = 59$

- The recursive, fractal-like nature makes the ordering method work, irrespective of the actual cache characteristics (e.g. sizes of L1, L2, L3 cache).
- The ordering is cache-oblivious.

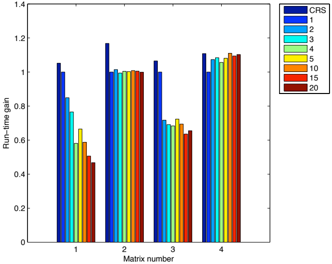
Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Try to forget it all

- Ordering the matrix in SBD format makes the matrix-vector multiplication cache-oblivious. Forget about the exact cache hierarchy. It will always work.
- We also like to forget about the cores: core-oblivious. And then processor-oblivious, node-oblivious.
- All that is needed is a good ordering of the rows and columns of the matrix, and subsequently of its nonzeros.

Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Wall clock timings of SpMV on Huygens



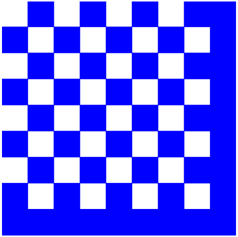

Splitting into 1–20 parts

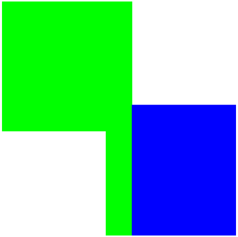
- Experiments on 1 core of the dual-core 4.7 GHz Power6+ processor of the Dutch national supercomputer Huygens.
- 64 kB L1 cache, 4 MB L2, 32 MB L3.
- Test matrices: 1. `stanford`; 2. `stanford_berkeley`; 3. `wikipedia-20051105`; 4. `cage14`


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions


Universiteit Utrecht

# Doubly Separated Block-Diagonal structure

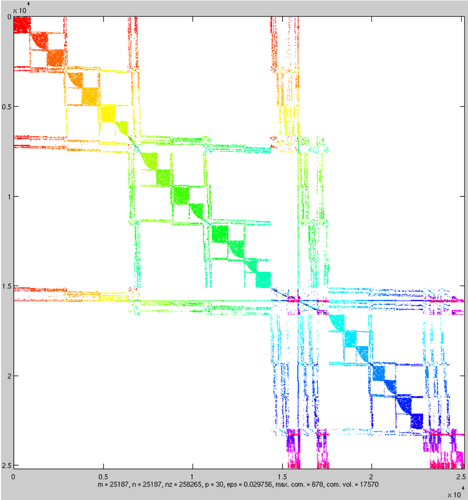




- $9 \times 9$ chess-arrowhead matrix, $nz = 49$, $p = 2$, $\epsilon = 0.2$.
- DSBD structure is obtained by recursively partitioning the sparse matrix, each time moving the cut rows and columns to the middle.
- The nonzeros must also be reordered by a Z-like ordering.
- Mondriaan is used in two-dimensional mode.


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

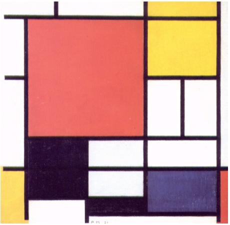
# Screenshot of Matlab interface




- Matrix `rhpentium`, split over 30 processors


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions

# Conclusions



- We have presented two combinatorial problems: partitioning and ordering. Solution of these is an enabling technology for high-performance computing.
- Reordering is a promising method for oblivious computing. We have shown its utility in enhancing cache performance.
- Mondriaan 3.0, to be released soon, provides new reordering methods, based on hypergraph partitioning.
- Visualisation can help in designing new algorithms!


Outline
Partitioning
Matrix-vector
Movies
Hypergraphs
Ordering
SBD
Conclusions


Universiteit Utrecht