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, May–July 2010


Albert-Jan


Bas

Sparse Days, Toulouse, June 17, 2010

Universiteit Utrecht

Partitioning problems

Parallel sparse matrix–vector multiplication

Visualisation by MondriaanMovie

Hypergraphs

2D matrix partitioning

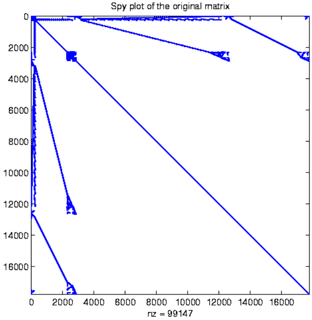
Ordering problems

Separated Block Diagonal structure

Conclusions and future work

Universiteit Utrecht

Motivation: sparse matrix `memplus`



17758×17758 matrix with 126150 nonzeros.
Contributed to MatrixMarket in 1995 by Steve Hamm (Motorola). Represents the design of a memory circuit.
Iterative solver multiplies matrix repeatedly with a vector.

Motivation: supercomputer 109/500 (June 2010)

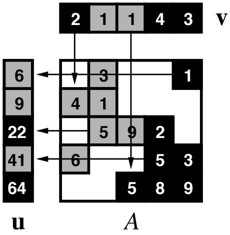
 

- National supercomputer Huygens named after Christiaan Huygens. Wikipédia: "En 1655, Huygens découvrit Titan, la lune de Saturne. Il examina également les anneaux de Saturne et établit qu'il s'agissait bien d'un anneau entourant la planète"
- Huygens, the machine, has 104 nodes
- Each node has 16 processors
- 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

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



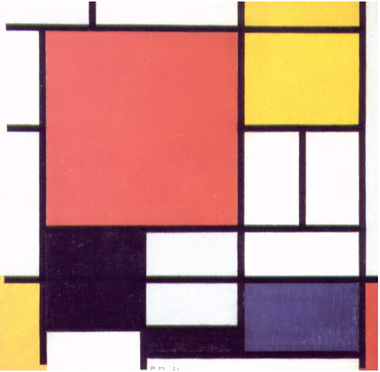
$p = 2$

4 supersteps: communicate, compute, communicate, compute

Universiteit Utrecht

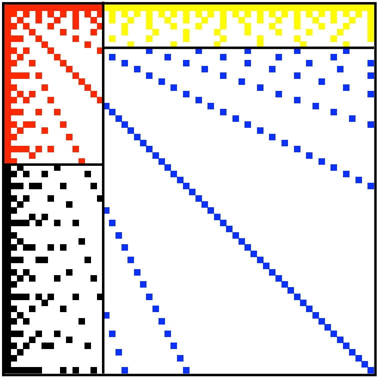
Divide evenly over 4 processors

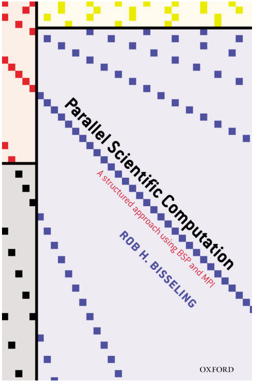
Composition with Red, Yellow, Blue and Black



Piet Mondriaan 1921

Matrix prime60





- Mondriaan block partitioning of 60×60 matrix prime60 with 462 nonzeros, for $p = 4$
- $a_{ij} \neq 0 \iff i|j \text{ or } j|i \qquad (1 \leq i, j \leq 60)$

Universiteit Utrecht

Avoid communication completely, if you can

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

Permute the matrix rows/columns

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

Combinatorial problem: sparse matrix partitioning

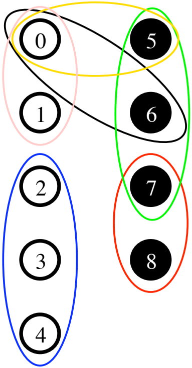
Problem: Split the set of nonzeros A of the matrix into p subsets, $A_0, A_1, \ldots, A_{p-1}$, minimising the communication volume $V(A_0, A_1, \ldots, A_{p-1})$ under the load imbalance constraint

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

The maximum amount of work should not exceed the average amount by more than a fraction ϵ.

- $p = 2$ problem is already NP-complete (Lengauer 1990, circuit layout)

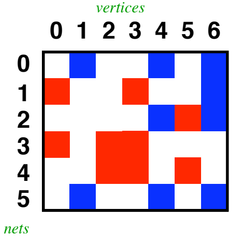
The hypergraph connection



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

Universiteit Utrecht

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 $\mathcal{V}$):

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

- Cut nets n_1, n_2 cause 1 horizontal communication.

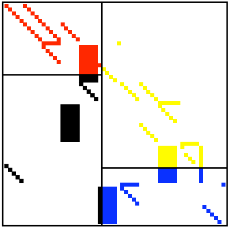
$(\lambda - 1)$-metric for hypergraph partitioning

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

Cut-net metric for hypergraph partitioning

- Row split has unit cost, irrespective of λ_i

Mondriaan 2D matrix partitioning



- Block partitioning (without row/column permutations) of 59×59 matrix `impcol_b` with 312 nonzeros, for $p = 4$
- Mondriaan package v1.0 (May 2002). Originally developed by Vastenhouw and Bisseling for partitioning term-by-document matrices for a parallel web search machine.

Mondriaan 2D matrix partitioning

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

Fine-grain 2D matrix partitioning

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

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.

Universiteit Utrecht

Recursive, adaptive bipartitioning algorithm

MatrixPartition(A, p, ϵ)

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

ϵ = 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;

Mondriaan version 1 vs. 3

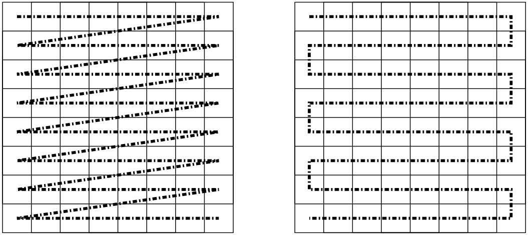
Name	p	v1.0	v3.0
`df1001`	4	1484	1406
	16	3713	3640
	64	6224	6022
`cre_b`	4	1872	1491
	16	4698	4158
	64	9214	9095
`tbdmatlab`	4	10857	10060
	16	28041	24910
	64	52467	50020
`nug30`	4	55924	58770
	16	126255	137200
	64	212303	267200
`tbdlinux`	4	30667	30240
	16	73240	68890
	64	146771	140500

Mondriaan, default values (v1 localbest, v3 hybrid), $\epsilon = 0.03$

Mondriaan 3.0 coming this month

- 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

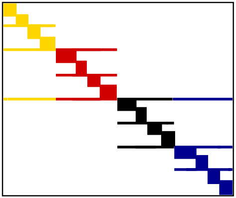
Ordering a sparse matrix to improve cache use



- 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.

Universiteit Utrecht

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.

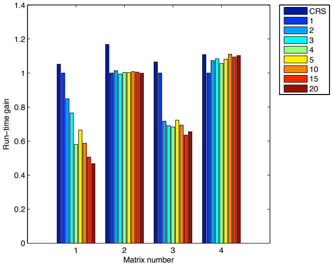
Partition the columns till the end, $p = n = 59$

- 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.

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.

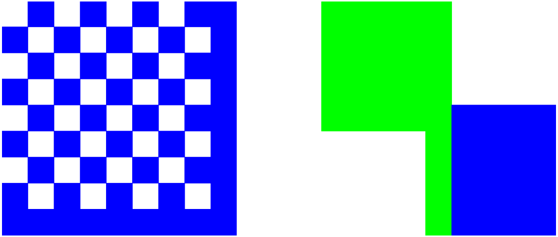
Wall clock timings of SpMV on Huygens



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`

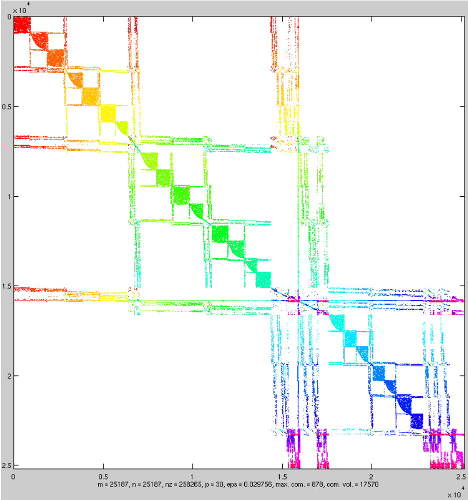
Doubly Separated Block-Diagonal structure



- 9×9 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.

Universiteit Utrecht

Screenshot of Matlab interface



- Matrix `rhpentium`, split over 30 processors

Universiteit Utrecht

Conclusions

- Flop counts become less and less important.
- It's all about restricting movement: moving less data, moving fewer electrons.
- 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!