Parallel Tomographic Reconstruction – Where Combinatorics Meets Geometry

Rob H. Bisseling

Mathematical Institute, Utrecht University

SIAM Conf. on Parallel Processing for Scientific Computing
SIAM Workshop on Combinatorial Scientific Computing
Seattle, February 12, 2020

Thanks to: Jan-Willem Buurlage, Timon Knigge, Daniël Pelt

Universiteit Utrecht

Introduction: computed tomography

- X-rays
- Sparse tomography matrix

Small problems: exact sparse matrix partitioning

- Branch-and-bound bipartitioner MondriaanOpt
- Improved bipartitioner MP
- Database of optimally bipartitioned matrices

Medium-size problems: heuristic sparse matrix partitioning

- Medium-grain partitioning
- Iterative refinement: chicken-or-egg
- How far from optimal?

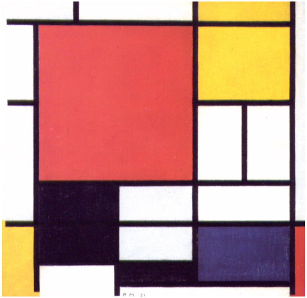
Large tomography problems: geometric data partitioning

- Bipartitioning
- High resolution
- Parallel speedup on GPUs

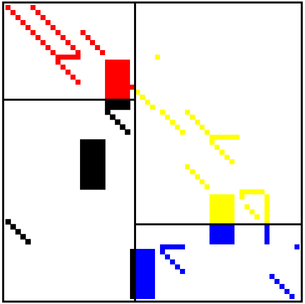
Conclusion and outlook

Universiteit Utrecht

Mondriaan sparse matrix partitioning



Composition with red, yellow, blue, and black
Piet Mondriaan, 1921



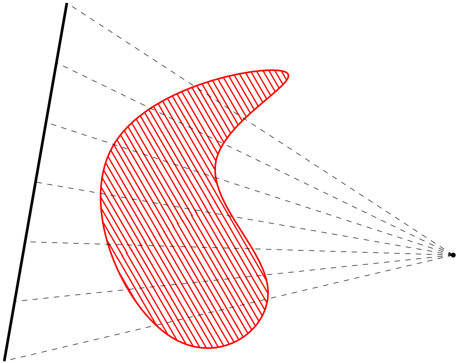
4-way partitioning of matrix `impcol_b`

- Mondriaan is an open-source software package for sparse matrix partitioning.
- Version 1.0, May 2002. Version 4.2.1, August 2019.

Introduction: computed tomography

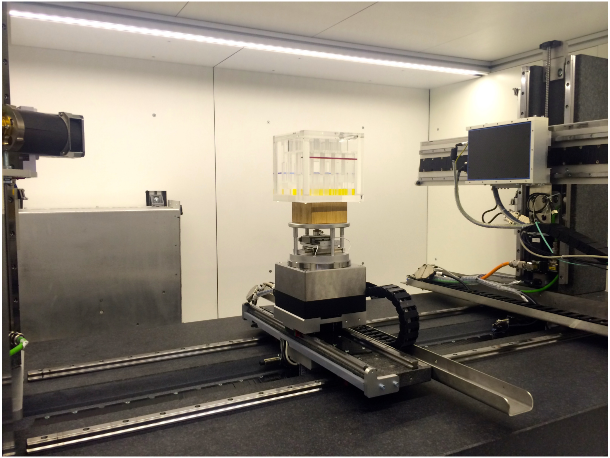
Universiteit Utrecht

Tomography setup



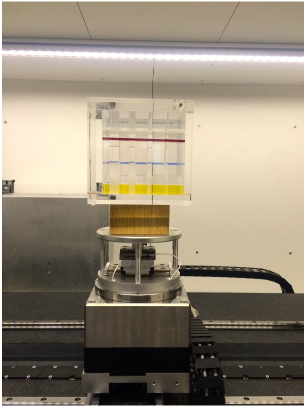
- One projection from the point source to a detector.
- 7 X-rays penetrating the object.

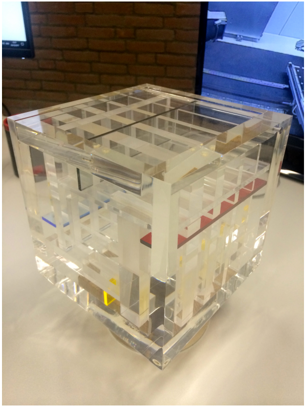
Flexible CT scanner at CWI Amsterdam



Universiteit Utrecht

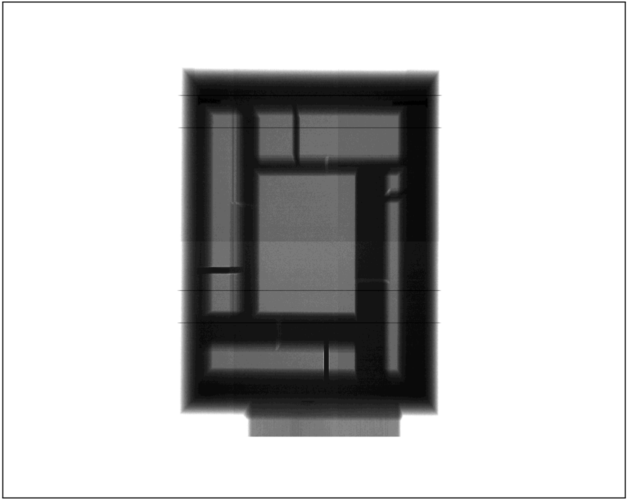
Modern art object in the scanner





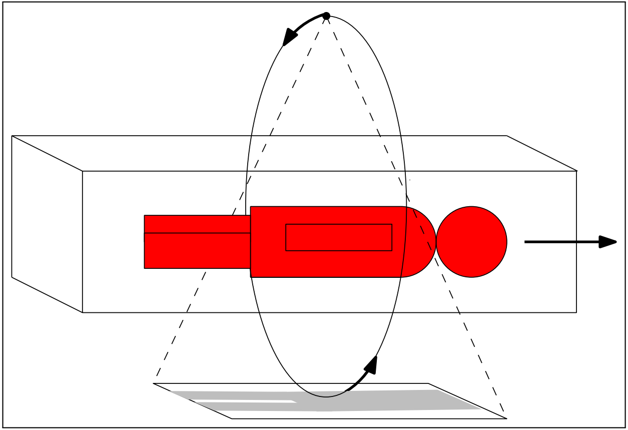
- Nel Haringa and Fred Olijve: Homage to De Stijl, 2004. Acrylic and perspex.

One projection of the art object



Universiteit Utrecht

Helical cone beam

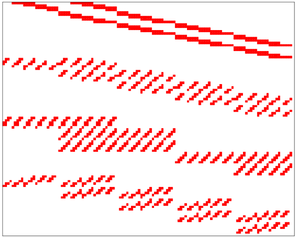


- Scanner and detector move in a circle around the object.
- Object (or scanner) moves along the rotation axis.

Acquisition geometries and their application field

- Helical cone beam: medical imaging, rock samples
- Parallel beam: electron microscopy, synchrotrons
- Laminography: inspection of flat objects
- Tomosynthesis: mammography, airport security screening

Solving a sparse linear system



4 projections
5×5 detector pixels
$5 \times 5 \times 5$ object voxels

$m = 100$, $n = 125$
1394 nonzeros

$$b_i = \sum_{j=0}^{n-1} a_{ij} x_j, \quad 0 \leq i < m.$$

- a_{ij} is the weight of ray i in voxel j,
- x_j is the density of voxel j,
- b_i is the detector measurement for ray i.
- Not every ray hits every voxel: the system is sparse.
- Usually $m < n$: the system is underdetermined.

Universiteit Utrecht

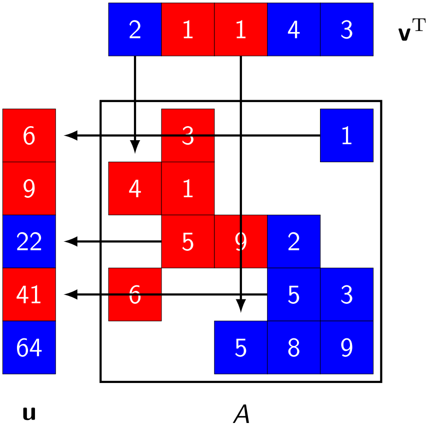
Simultaneous Iterative Reconstruction Technique

- SIRT repeatedly multiplies the sparse matrices A and A^{T} with a vector until convergence.
- For low resolutions, A is small and it can be stored.
- However, for a high resolution of $4000^3 = 64 \times 10^9$ voxels, A has 256×10^{12} nonzeros, so we have Petabytes of data.
- For large problem sizes, implementations are matrix-free: A is too big to store, and too big to partition by a combinatorial method.
- We can regenerate the matrix easily row by row.

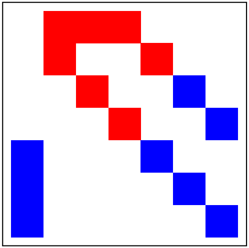
Small problems:
exact sparse matrix partitioning

Universiteit Utrecht

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



Optimal bipartitioning by MondriaanOpt



7×7 matrix `b1_ss`
$nz(A) = 15,\ V = 3$

- Benchmark $p = 2$ because heuristic partitioners are often based on recursive bipartitioning.
- Problem $p = 2$ is easier to solve than $p > 2$.
- Load balance criterion is

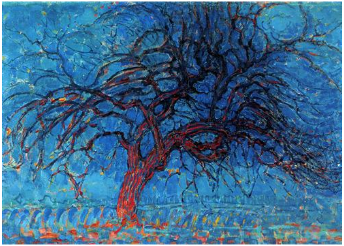
$$nz(A_i) \leq (1 + \varepsilon) \left\lceil \frac{nz(A)}{2} \right\rceil, \quad i = 0, 1,$$

where $\varepsilon \in [0, 1)$ is the allowed load imbalance fraction.

D. M. Pelt and R. H. Bisseling, "An exact algorithm for sparse matrix bipartitioning", *JPDC* **85** (2015) pp. 79–90.

Universiteit Utrecht

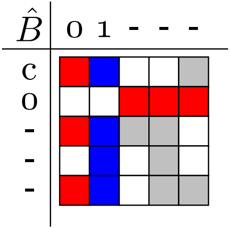
Branch-and-bound method



Evening – the red tree
Piet Mondriaan, 1908

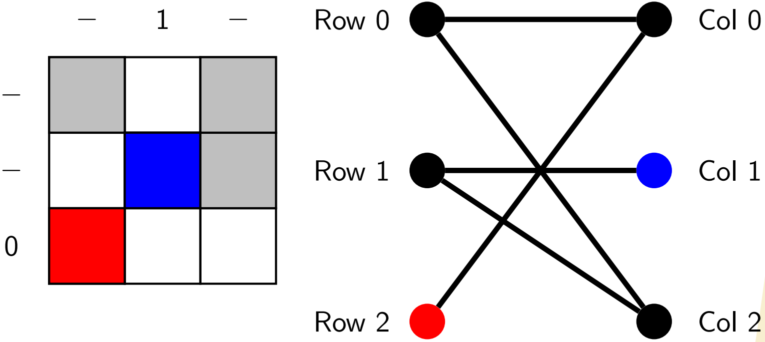
- Construct a ternary tree representing all possible solutions
- Every node in the tree has 3 branches, representing a choice for a matrix row or column:
  - completely assigned to processor $P(0)$
  - completely assigned to processor $P(1)$
  - cut: assigned to processors $P(0)$ and $P(1)$
- The tree is pruned by using lower bounds on the communication volume or number of nonzeros

Packing bound on communication volume



- Columns 3, 4, 5 have been partially assigned to $P(0)$.
- They can only be completely assigned to $P(0)$ or cut.
- For perfect load balance ($\varepsilon = 0$), we can pack at most 2 more red nonzeros into $P(0)$.
- Thus we have to cut column 3, and one more column, giving 2 communications.
- We call the resulting lower bound a packing bound.

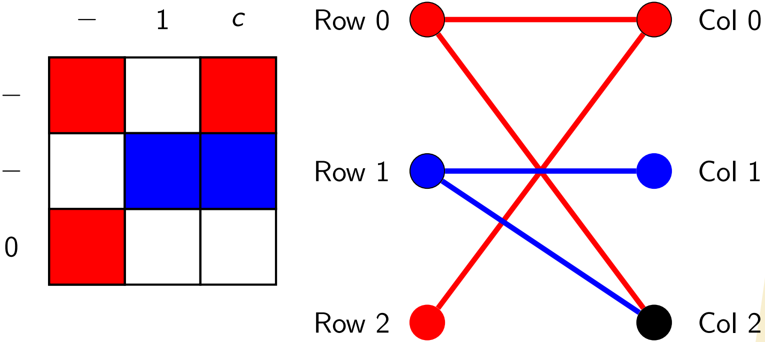
From sparse matrix to bipartite graph



Row 2 has been assigned to part 0 and column 1 to part 1.

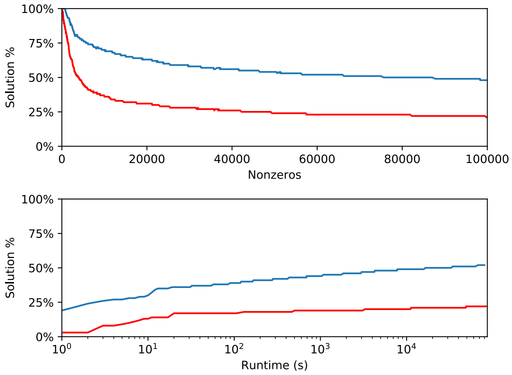
T. E. Knigge and R. H. Bisseling, "An improved exact algorithm and an NP-completeness proof for sparse matrix bipartitioning", submitted.
https://github.com/TimonKnigge/matrix-partitioner

Flow bound on communication



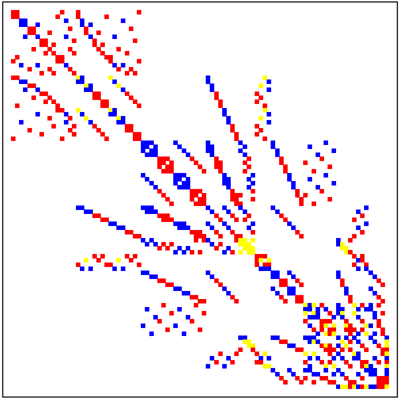
Along the path from row 2 to column 1, at least one row or column must be cut. We can model the problem with multiple paths as a maximum-flow problem.

Test set of 1602 SuiteSparse matrices



- Top: solution % of MondriaanOpt and MP within 24 hours CPU-time as a function of *nz*.
- Bottom: solution % as a function of the runtime.
- MP solved 839 matrices, each within 24 hours.

Sparse matrix cage6 from DNA electrophoresis



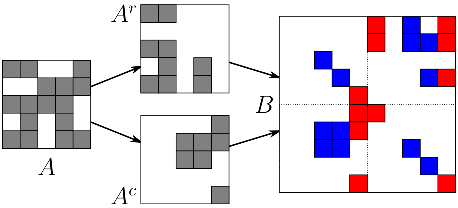
93×93, $nz = 785$

- The smallest matrix that could not be solved within 1 day; it needed 3 days.
- Communication volume $V = 38$.
- 397 red, 316 blue, and 72 yellow (free) nonzeros.
- The yellow nonzeros can be painted blue to give a load imbalance of only 1%.

Medium-size problems: heuristic sparse matrix partitioning

Universiteit Utrecht

Medium-grain partitioning method

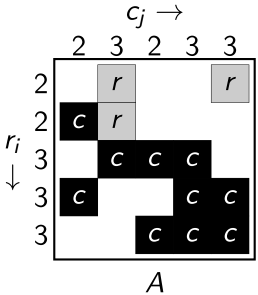


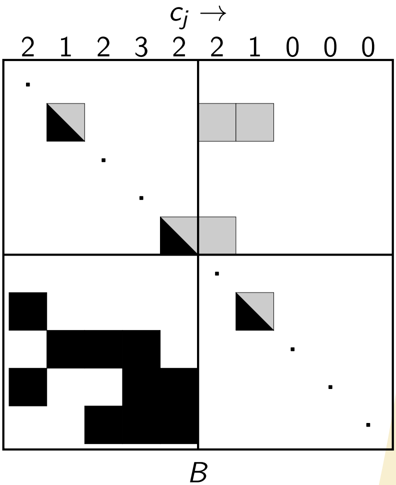
- $m \times n$ matrix A is split by a simple method into $A = A^r + A^c$
- $(m+n) \times (m+n)$ matrix B is formed and partitioned by column using a 1D method

$$B = \begin{bmatrix} I_n & (A^r)^{\mathrm{T}} \\ A^c & I_m \end{bmatrix}$$

D. M. Pelt and R. H. Bisseling, "A medium-grain method for fast 2D bipartitioning of sparse matrices", Proc. IPDPS 2014, pp. 529–539.

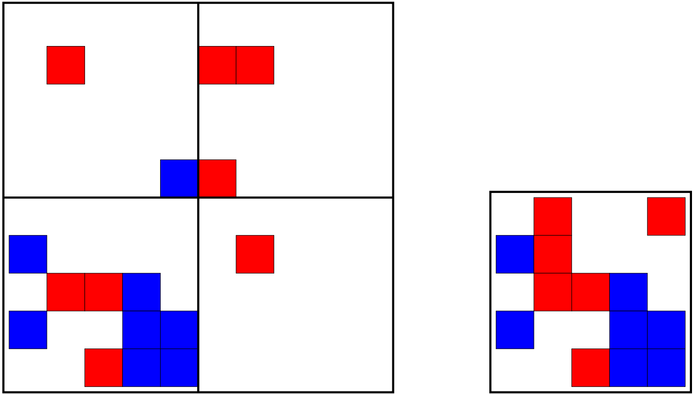
From A to B: the medium-grain method





If $r_i < c_j$, the nonzero goes to the row part A^r, otherwise to the column part A^c.

1D column partitioning of B yields a 2D partitioning of A



B A

Communication volume $V = 4$

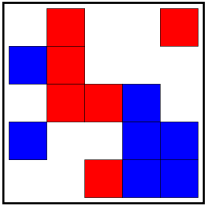
Universiteit Utrecht

Chicken-or-egg problem: which one was first?

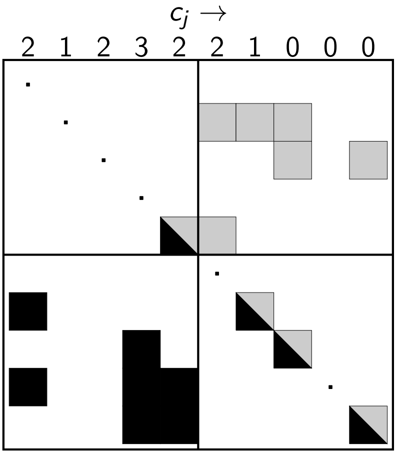
- To partition the matrix A, we first form a matrix B.
- To form a matrix B, we need a partitioning of A.
- That's why we start with a simple partitioning.

Universiteit Utrecht

Iterative refinement: repeated partitioning



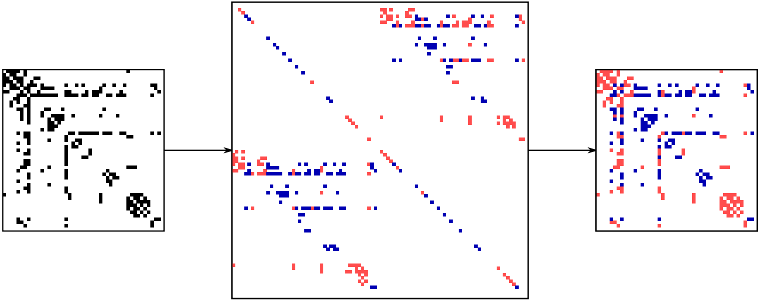
$A = A^r + A^c$



B

Iterative refinement is combinatorial, not numerical.
It uses Kernighan–Lin refinement, 1 level.

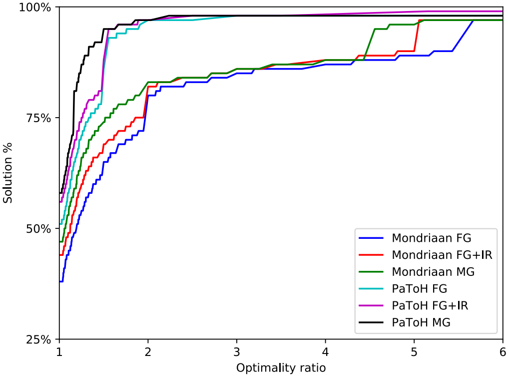
Result for matrix from Graph Drawing contest 1997



47×47 matrix gd97_b, $nz(A) = 264$

- Medium-grain method achieves optimal $V = 11$
- Communication volume of 1D partitioning of B = volume of corresponding 2D partitioning of A

Performance plot comparing volume to optimal



- IR = iterative refinement
- FG = fine-grain partitioning
- MG = medium-grain partitioning (including IR)
- PaToH = combination of Mondriaan sparse matrix partitioner and PaToH hypergraph bipartitioner

Geometric average of runtime and optimality ratio

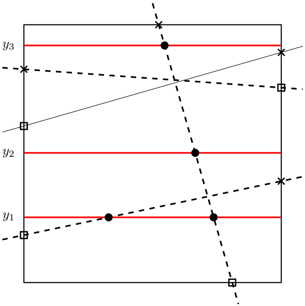
Partitioner	Method	Runtime (in ms)	Optimality ratio
Mondriaan	FG	51.5	1.63
	FG+IR	53.9	1.53
	MG+IR	29.9	1.46
Mondriaan+PaToH	FG	13.9	1.19
	FG+IR	15.2	1.16
	MG+IR	9.2	1.10

- Optimality ratio is ratio of communication volume and optimal volume computed by MP.
- Based on 839 matrices with $nz \leq 100,000$.

Ü. V. Çatalyürek and C. Aykanat, "A Fine-Grain Hypergraph Model for 2D Decomposition of Sparse Matrices", Proc. Irregular 2001, pp. 118.

Large problems: geometric data partitioning

Geometric bipartitioning of a voxel block $\mathcal{V}$



- 2D: line sweep along each coordinate. (3D: plane sweep.)
- Sort the points of entrance (□) and exit (×) of a ray.
- Cut as few rays as possible. Keep the work load balanced (based on line densities).

Theorem on greedy p-way recursive bipartitioning

Theorem

Let $\mathcal{V} = \mathcal{V}_0 \cup \ldots \cup \mathcal{V}_{p-1}$ be a block partitioning. Then, for any acquisition geometry, the communication volume V satisfies:

$$V(\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_{p-1}) = V(\mathcal{V}_0, \mathcal{V}_1, \ldots, \mathcal{V}_{p-2} \cup \mathcal{V}_{p-1}) + V(\mathcal{V}_{p-2}, \mathcal{V}_{p-1}).$$

- Same theorem as with sparse matrix partitioning for parallel SpMV.

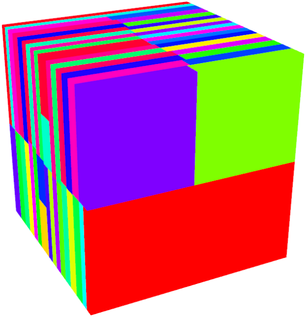
J. W. Buurlage, R. H. Bisseling, K. J. Batenburg, "A geometric partitioning method for distributed tomographic reconstruction", *Parallel Computing* **81** (2019) pp. 104–121.

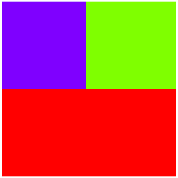
Communication volume: geometric vs. combinatorial partitioning

	geometric (voxels)		combinatorial (Mondriaan)		
p	Slab	GRCB	1D col	1D row	2D MG
16	111,248	111,207	108,741	139,216	101,402
32	233,095	216,620	210,330	292,833	188,294
64	3,928,222	2,505,646	2,604,930	3,987,888	2,210,671

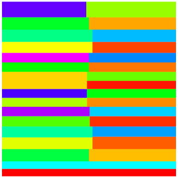
- 64^3 voxels, 64 projections. Narrow cone angle.
- Slab = standard geometric partitioning into slabs
- GRCB = geometric recursive coordinate bisection
- MG = medium-grain with iterative refinement
- Partitioning voxels (1D col) has 35% lower communication volume than partitioning rays (1D row).
- 2D MG is 15% better than GRCB, but not practical.

Partitioning for helical cone beam, 64 processors



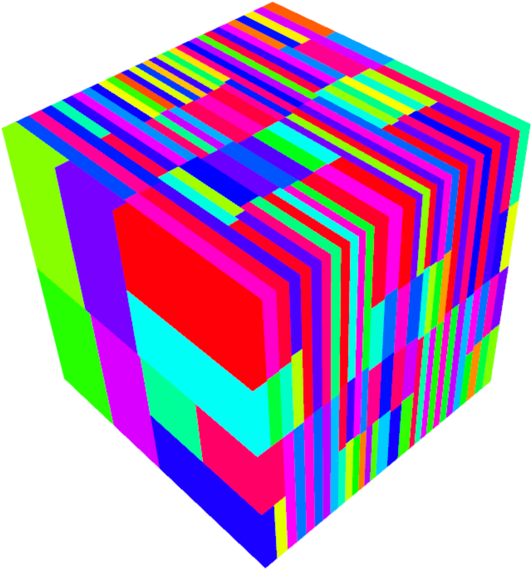


front



bottom

Partitioning for helical cone beam, 256 processors

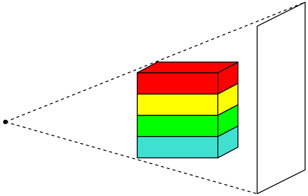


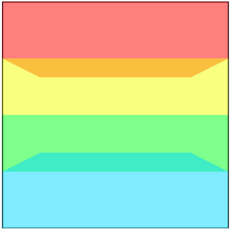
Universiteit Utrecht

Partitionings for various acquisition geometries

Universiteit Utrecht

Projection-based partitioning for high resolution

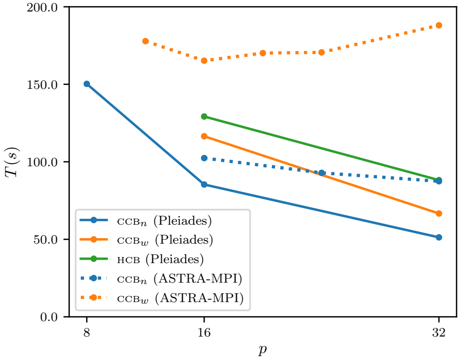




- For a given split of the object volume, the total area of overlapping shadows gives the communication volume.
- Fast overlap computations are based on geometric algorithms.

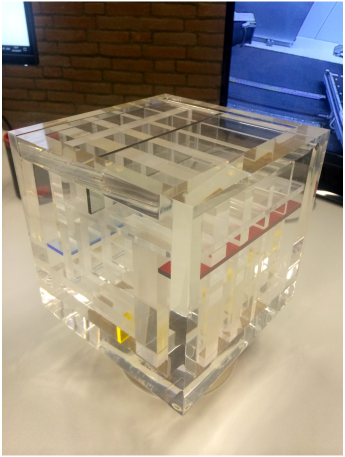
J. W. Buurlage, R. H. Bisseling, W. J. Palenstijn, K. J. Batenburg, "A projection-based data partitioning method for distributed tomographic reconstruction", Proc. SIAMPP 2020, pp. 58-68.
Talk by Jan-Willem Buurlage in CP7, Feb. 13, 3.45 PM.

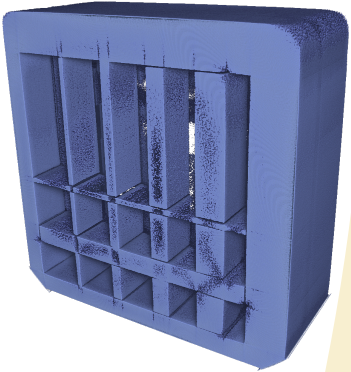
Scalability on 32 GPUs



- 2048^3 voxels, 1024 projections. Time of 3 iterations.
- ASTRA toolbox: state-of-the-art, slab partitioning, only for circular cone beam (CCB). MPI for communication.
- Pleiades extension of ASTRA: projection-based partitioning, for any acquisition geometry. BSP/C++ library Bulk for communication.

Reconstructed art object Homage to De Stijl





A slab of the reconstruction. Thanks to: Sophia Coban.

Conclusion and outlook

- We presented a method for exact matrix bipartitioning that solved 839 out of 2833 SuiteSparse matrices optimally.
- The best heuristic partitioner, a combination Mondriaan+PaToH, is within 10% of optimal for $p = 2$.
- Targeting $p > 2$, we still want to improve the bipartitioner: for $p = 256$, a factor of $(1.10)^8 \approx 2.14$ from optimal.
- We presented a geometric method for partitioning the object space of a flexible CT scanner.
- The method can handle XL problems in a real production environment.

Thank you!

Universiteit Utrecht