Accelerating the Mondriaan sparse matrix partitioner

Rob H. Bisseling

Mathematical Institute, Utrecht University

Joint work with Marco van Oort (UU)

Sparse Days at Cerfacs, Toulouse, September 7, 2017

Introduction

Motivation: quality and speed

Medium-grain partitioning

Improvements

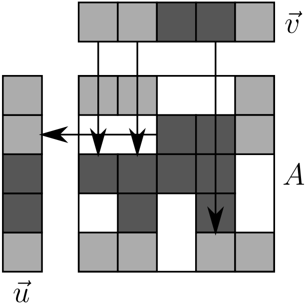
Exploiting free nonzeros

Simplifying the gainbucket data structure

Zero-volume search

Conclusion

Parallel sparse matrix-vector multiplication



- Parallel multiplication of a 5×5 sparse matrix A and a dense input vector $\vec{v}$,

$$\vec{u} = A\vec{v}$$

- 2D matrix distribution over 2 processors
- $V = 4$ data words of communication
- Perfect load balance: 8 nonzeros per processor

Universiteit Utrecht

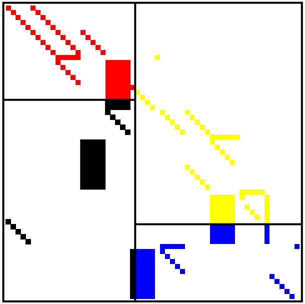
Sparse matrix partitioning and graph partitioning

- A sparse matrix is the adjacency matrix of a sparse graph:

$$a_{ij} \neq 0 \Leftrightarrow (i, j) \in E$$

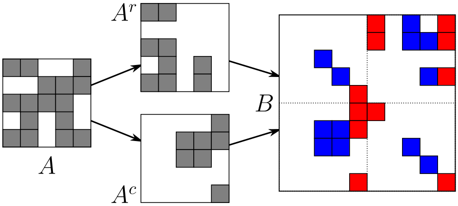
- Partitioning the nonzeros of a matrix is the same as partitioning the edges of a graph.
- 2D partitioning splits both rows and columns.
- Partitioning for parallel sparse matrix-vector multiplication (SpMV) also gives a good partitioning for many graph computations.

Mondriaan v1.0 (2002)



- Non-cartesian block distribution of 59×59 matrix `impcol_b` with 312 nonzeros, for $p = 4$
- #nonzeros per processor: 76, 76, 80, 80
- Iain@oxford, 2002: "Mondriaan is slow!"
- Repaired in v1.02 (2005) with help of Ümit Çatalyürek

Medium-grain partitioning method

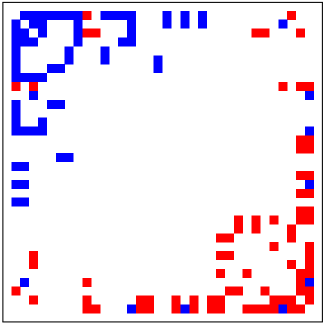


- $m \times n$ matrix A is split by a simple method into $A = A^r + A^c$
- $(m+n) \times (m+n)$ matrix B is formed and partitioned by column using a 1D method

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

"A medium-grain method for fast 2D bipartitioning of sparse matrices", by Daniël M. Pelt and Rob H. Bisseling, Proc. IPDPS 2014, IEEE Press, pp. 529-539.

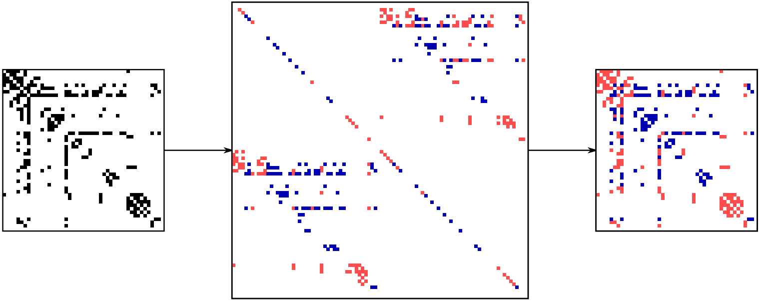
Simple split $A = A^c + A^r$



34×34 matrix `karate`, $nz(A) = 156$ (Zachary's karate club, 1977), $V_{Opt} = 8$

- Observation from optimal partitioning (MondriaanOpt, v4.1): fewer nonzeros in a row or column have more chance to stay together.
- Simple split strategy: matrix nonzero a_{ij} is assigned to A^c if $nz_c(j) < nz_r(i)$, and to A^r otherwise.

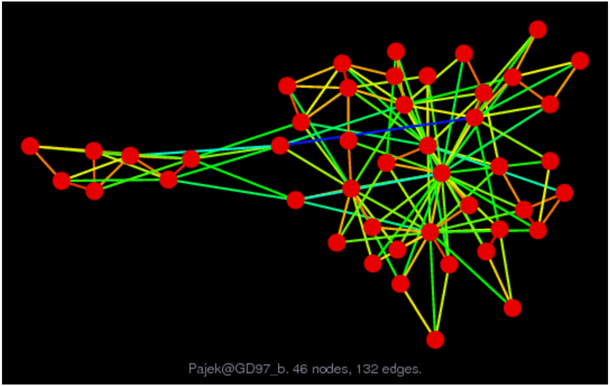
Result for matrix from Graph Drawing contest 1997



47×47 matrix GD97_b, $nz(A) = 264$

- Medium-grain method achieves $V_{Opt} = 11$
- Communication volume of 1D partitioning of $B =$ volume of corresponding 2D partitioning of A

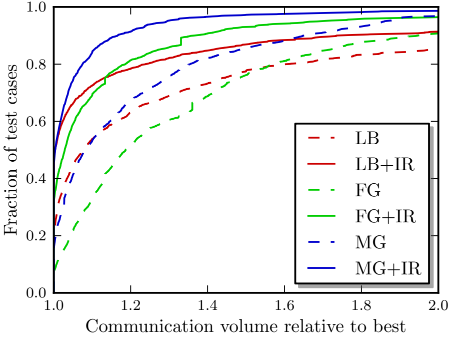
The corresponding graph



`http://www.cise.ufl.edu/research/sparse/matrices/Pajek/GD97_b.html`

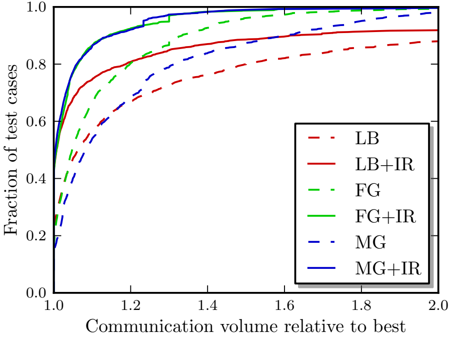
- 46 vertices, 132 edges
- One matrix row and column were empty

Comparing volume for $p = 2$ using Mondriaan



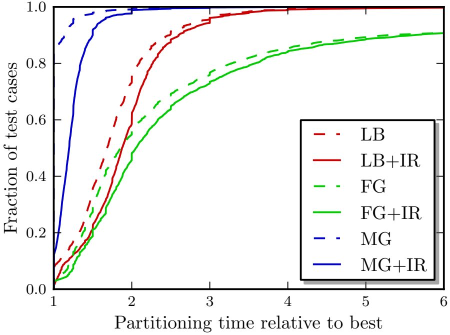
- LB = localbest = best of 1D row, 1D column (v1-v3)
- MG = medium-grain method (v4.0)
- FG = fine-grain model (Çatalyürek and Aykanat 2001)
- IR = iterative refinement, a cheap Kernighan–Lin based postprocessing procedure using the MG idea
- 2267 matrices from U. Florida collection with $500 \leq nz \leq 5,000,000$

Comparing volume for $p = 2$ using PaToH



- LB = localbest = best of 1D row, 1D column (v1-v3)
- MG = medium-grain method (v4.0)
- FG = fine-grain model (Çatalyürek and Aykanat 2001)
- IR = iterative refinement, a cheap Kernighan–Lin based postprocessing procedure using the MG idea
- 2267 matrices from U. Florida collection with $500 \leq nz \leq 5,000,000$

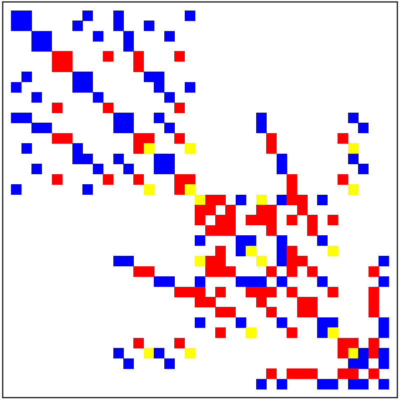
Comparing time for $p = 2$ using Mondriaan



- LB = localbest = best of 1D row, 1D column (v1-v3)
- MG = medium-grain method (v4.0)
- FG = fine-grain model (Çatalyürek and Aykanat 2001)
- IR = iterative refinement, a cheap Kernighan–Lin based postprocessing procedure using the MG idea
- 2267 matrices from U. Florida collection with $500 \leq nz \leq 5,000,000$

Universiteit Utrecht

Free nonzeros (in yellow)



- 37×37 matrix cage5, $nz(A) = 233$
- Markov model of DNA electrophoresis, 5 monomers in polymer (Alexander van Heukelum 2003)
- $nz_0 = 106$; $nz_1 = 110$; $nz_{free} = 17$
- $V = 14$, imbalance $= nz_{max} - nz_{min} = 1$
- Challenge: find optimal bipartitioning of cage6 (93×93)

Trade-off: load imbalance vs. communication

Input: A: $m \times n$ sparse matrix,
p: number of processors, $p = 2^q$ with $q \geq 0$,
ϵ = allowed load imbalance, $\epsilon > 0$.

Output: $(A_0, \ldots, A_{p-1})$: p-way partitioning of A,
satisfying $\max_{0 \leq s < p} nz(A_s) \leq (1+\epsilon)\frac{nz(A)}{p}$.

function $\textsc{MatrixPartition}(A, p, \epsilon)$

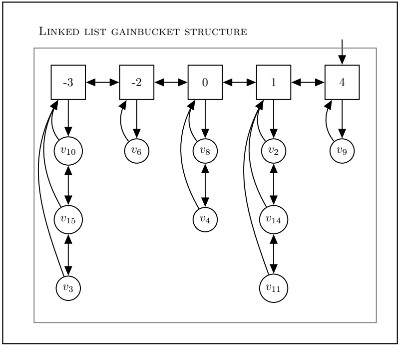
if $p > 1$ **then**
$maxnz := (1+\epsilon)\frac{nz(A)}{p}$;
$(B_0, B_1) := Bipartition(A, \frac{\epsilon}{q})$;
$\epsilon_0 := \frac{maxnz}{nz(B_0)} \cdot \frac{p}{2} - 1$; $\epsilon_1 := \frac{maxnz}{nz(B_1)} \cdot \frac{p}{2} - 1$;
$(A_0, \ldots, A_{p/2-1}) := \mathrm{MatrixPartition}(B_0, \frac{p}{2}, \epsilon_0)$;
$(A_{p/2}, \ldots, A_{p-1}) := \mathrm{MatrixPartition}(B_1, \frac{p}{2}, \epsilon_1)$;
else
$A_0 := A$;

Improvement ratio for moving free nonzeros

P	Matrices	Imbalance	Volume	Time
2	955	0.74 ± 0.35	0.9998 ± 0.0012	1.003 ± 0.023
4	955	0.75 ± 0.33	0.9980 ± 0.0279	-
16	954	0.80 ± 0.26	0.9874 ± 0.0267	1.007 ± 0.022
64	871	0.90 ± 0.19	0.9798 ± 0.0302	-

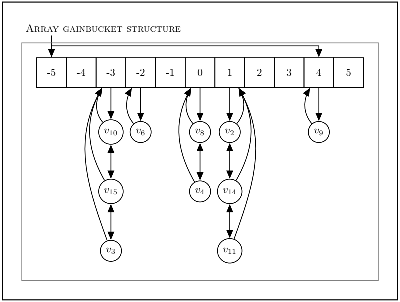
- Mean imbalance, communication volume, and computation time of final partitioning are given compared to the results without the free-nonzero feature.
- All even-numbered matrices from SuiteSparse collection with $10^3 \leq nz \leq 10^7$.
- After every bipartitioning, the load balance is improved by moving free nonzeros.

Gainbucket data structure



- Data structure storing the gains of moving a vertex in the hypergraph to the other part during Kernighan–Lin refinement.
- Currently: a linked list of linked lists.
- Iain@cerfacs, 2010: "Why so complicated?"
- For maximum flexibility as a hypergraph partitioner, allowing widely varying net costs and gains.

Optimise Mondriaan as a sparse matrix partitioner

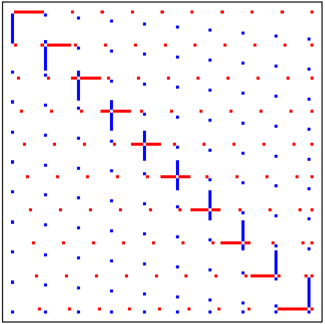


- Use an array of linked lists.
- Range of gains is limited, at most gain 1 per net.
- Mondriaan exploits the 2D nature of matrices to the full.
- Decision 2017: optimise Mondriaan as a sparse matrix partitioner, not as a hypergraph partitioner.

Improvement ratio for simpler gainbuckets

$\log_{10}(nnz)$	Mean time ratio
$3-4$	0.85 ± 0.09
$4-5$	0.89 ± 0.09
$5-6$	0.91 ± 0.08
$6-7$	0.93 ± 0.06
$3-7$	0.89 ± 0.09

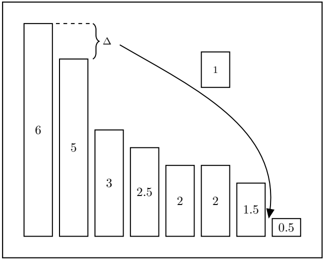
Look mama: no communication!



101×101 matrix GD06_theory, $nz(A) = 380$

- Mondriaan found 116 matrices out of 2409 with 0 communication volume for $p = 2$.
- Bipartite graph with row and column vertices has several connected components with the right sizes.

0-volume search



Karmarkar-Karp

- First, find all c connected components and their sizes by a Breadth-First Search.
- Then, solve a variant of the Subset-Sum problem: try to fit components within the load balance constraints.
- Small c: try all 2^c possibilities
- Larger c: use adaptation of Karmarkar-Karp algorithm (1982).

Volume 0 can happen at every recursion level

	Depth d	$p = 64$
$c = 1$	0	0.0 (0.00%)
	1	6.0 (0.47%)
	2	18.9 (0.73%)
	3	42.3 (0.82%)
	4	94.7 (0.92%)
	5	250.9 (1.22%)
$c > 1$	0	64.0 (23.44%)
	1	58.6 (10.73%)
	2	116.4 (10.66%)
	3	228.3 (10.45%)
	4	415.2 (9.51%)
	5	910.6 (10.42%)

- $c = 1$: 642 test matrices.
- $c > 1$: 273 test matrices.
- Average over 10 runs, divided by 2^d; $\epsilon = 0.03$.

Conclusion

- We have achieved an overall improvement of 25% in computation speed of the Mondriaan package from v4.1 to v4.2.
- Iain has challenged us all along the way.

Dear Iain, happy birthday to you and thank you!