Edge-based graph partitioning

Rob H. Bisseling

Mathematical Institute, Utrecht University

Joint work with Daniël M. Pelt (CWI Amsterdam)

Workshop Sparse Days at St. Girons
June 30, 2015

Introduction

2D matrix partitioning

Heuristic medium-grain partitioning

Medium-grain method

Results

Optimal sparse matrix bipartitioning

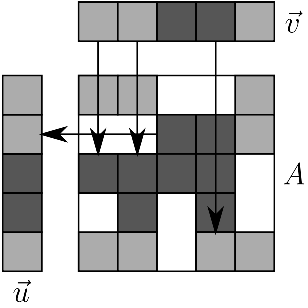
Branch-and-Bound method

Results

Pretty pictures

Conclusion and future work

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
- 4 data words of communication
- Perfect load balance: 8 nonzeros per processor

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 partioning the edges of a graph.
- 2D partitioning splits both rows and columns.
- Partitioning for parallel sparse matrix-vector multiplication (SpMV) can be used in Google PageRank computation.
- Partitioning for SpMV also gives a good partitioning for many graph computations.

Advantage of 2D partitioning

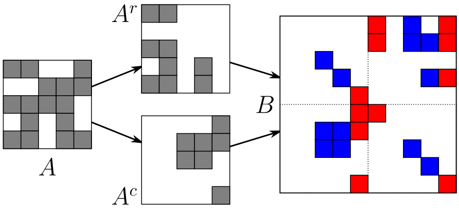
- We can use both dimensions of the matrix to reduce SpMV communication.
- For a $\sqrt{p} \times \sqrt{p}$ block distribution, each matrix row or column is distributed over at most $\sqrt{p}$ processors, instead of p processors for a 1D distribution.
- Relatively dense rows and columns can be split and do not cause load imbalance or memory overflow.

2D (edge-based) parallel matching

Name	*SpMV* 1*D*	*SpMV* 2*D*	*Matching* 1*D*	*Matching* 2*D*
rw9 (af_shell10)	113	105	169	150
rw10 (boneS10)	150	145	228	189
rw11 (Stanford)	340	141	479	234
rw12 (gupta3)	710	44	1,305	61
rw13 (St_Berk.)	716	448	1,152	812
rw14 (F1)	139	130	148	139
sw1 (small world)	1,007	417	2,111	303
sw2	1,957	829	3,999	563
sw3	2,017	832	4,255	528
er1 (random)	1,856	1,133	1,788	1,157
er2	3,451	1,841	3,721	1,635
er3	5,476	2,569	6,350	1,990

Communication volume in parallel sparse matrix–vector multiplication and Karp–Sipser matching.
Source: Patwary, Bisseling, Manne (HLPP 2010).

Medium-grain partitioning method

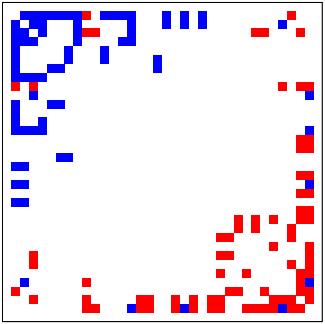


- $m \times n$ matrix A is split by a simple method into $A = A^r + A^c$
- $(m+n) \times (m+n)$ matrix B is formed and partitioned by column using a 1D method

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

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

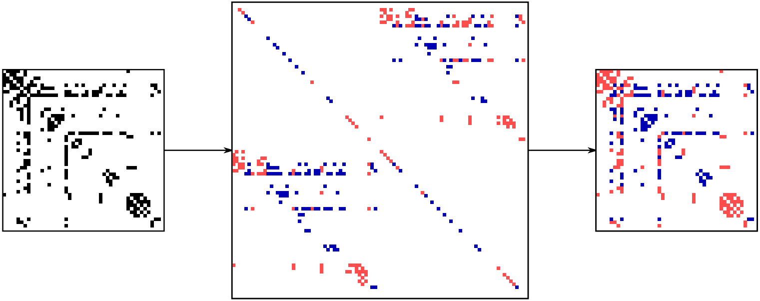
Simple split



34×34 matrix `karate`, $nz(A) = 156$ (Zachary's karate club, 1977), $V = 8$

- Matrix nonzero a_{ij} is assigned to A^c if $nz_c(i) < nz_r(j)$, and to A^r otherwise.
- Fewer nonzeros in a column have more chance to stay together in a good partitioning

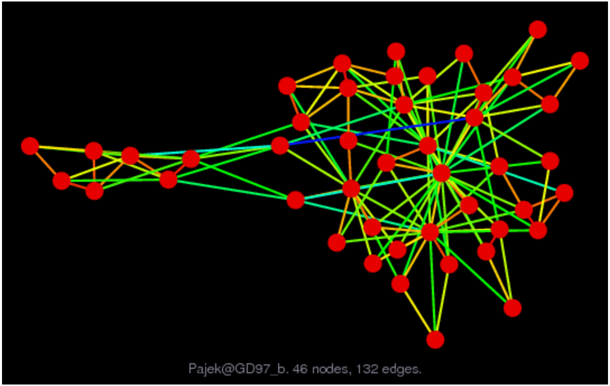
Result for matrix from Graph Drawing contest 1997



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

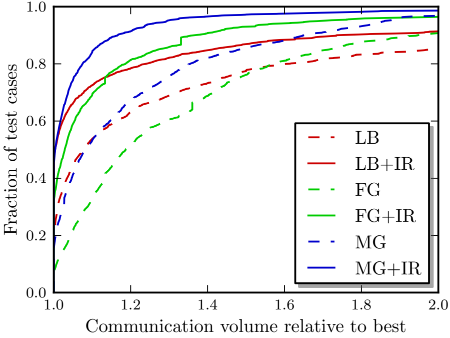


http://www.cise.ufl.edu/research/sparse/matrices/Pajek/GD97_b.html

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

Universiteit Utrecht

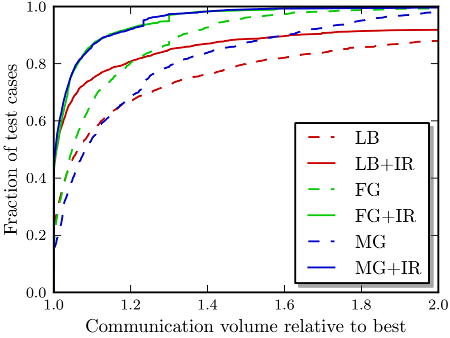
Comparing 3 methods for $p = 2$ using Mondriaan



- 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 $500 \leq nz \leq 5,000,000$

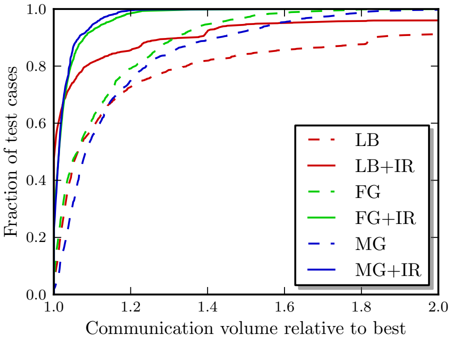
Universiteit Utrecht

Comparing 3 methods for $p = 2$ using PaToH

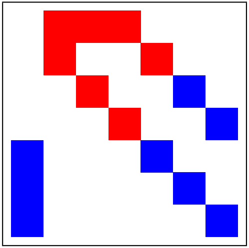


Universiteit Utrecht

Comparing 3 methods for $p = 64$ using PaToH



Optimal bipartitioning



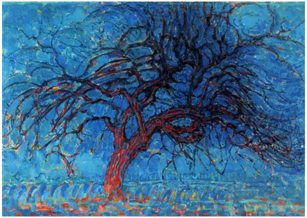
7×7 matrix b1_ss, $nz(A) = 15$

- 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

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

- Rounding enables a feasible solution even for $\varepsilon = 0$ and odd $nz(A)$.

Branch-and-bound method

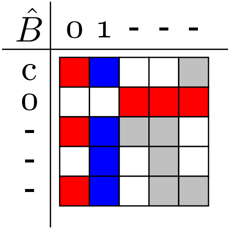


Piet Mondriaan 1908

Evening - the red tree

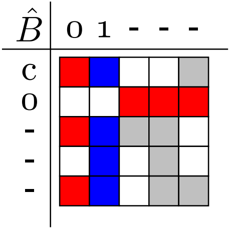
- Construct a ternary tree representing all possible solutions
- Every node in the tree has 3 branches, representing a choice for a matrix row or column:
  - completely assigned to processor $P(0)$
  - completely assigned to processor $P(1)$
  - cut
- The tree is pruned by using lower bounds on the communication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

Lower bounds L_1, L_2 on communication volume



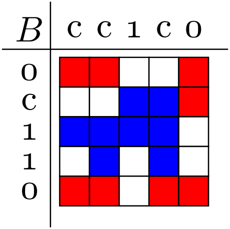
- Partial solution: value 0, 1, or c has been assigned to 2 rows and 2 columns
- Row 0 has been cut: lower bound on volume $L_1 = 1$
- Rows 2 and 4 have been implicitly cut: $L_2 = 2$

Lower bound L_3 on communication volume



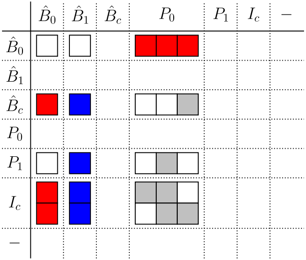
- Columns 3, 4, 5 have been partially assigned to $P(0)$
- They can only be completely assigned to $P(0)$ or cut.
- For perfect load balance ($\varepsilon = 0$), we can assign at most 2 more red nonzeros
- Thus we have to cut column 3, and one more: $L_3 = 2$

Optimal solution



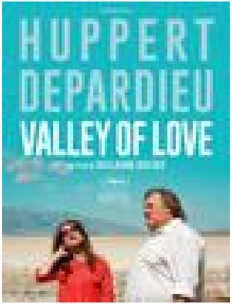
- Optimal solution: volume $= 4$.
- Total lower bound is $LB = L_1 + L_2 + L_3 = 5$.
- Prune partial solution since $LB > UB$.

Lower bound L_4 by conflicting partial assignments



- Permute matrix to create blocks:
  - $\hat{B}_0$: completely assigned to processor $P(0)$
  - P_0: partially assigned to processor $P(0)$
  - $\hat{B}_c$: cut
  - $\hat{I}_c$: implicitly cut
- Conflict for nonzero in row block $P_1 \cap$ column block P_0: $L_4 = 1$

The perfect match



Coming soon to this theatre!

Maximum bipartite graph matching

- Assume row block $P_0 \cap$ column block P_1 contains several nonzeros.
- Define bipartite graph $G = (V_0 \cup V_1, E)$:
  - vertex set V_0 contains the rows of P_0,
  - vertex set V_1 contains the columns of P_1,
  - edge set E containing edges (i, j) for $a_{ij} \neq 0$.
- Compute a maximum matching $M \subseteq E$. Then $L_4 = |M|$, since every nonzero (edge) in the matching causes at least one cut row or column.
- Two nonzeros from the matching cannot be in the same matrix row or column.

Alternative view: minimum vertex cover

- König's theorem (1931): maximum matching in bipartite graph is equivalent to minimum vertex cover.
- This gives the minimum number of cut rows or columns.

Universiteit Utrecht

Dynamic maximum matching

- The conflict graph is small, because we solve small sparse matrix problems and solutions with many conflicts get pruned early.
- Therefore, we maintain a maximum graph matching as the conflict graph changes.
- We prove, as a direct consequence of Berge's theorem (1957):
  - when adding a vertex i with all its edges: sufficient to search for an augmenting path starting at i;
  - when deleting a matched vertex i with all its edges: sufficient to search for an augmenting path starting at the match of i.

Results for 10 largest matrices solved

Matrix	m	n	nz	V_{LB}	V_{MG}	V_{FG}	V_{Opt}	Time (s)
stoch_air	3754	7517	20267	14	14	13	6	0.39
rosen1	520	1544	23794	8	8	24	8	0.03
add32	4960	4960	23884	40	13	13	4	381.29
mhd4800b	4800	4800	27520	3	2	2	2	161.83
Chebyshev3	4101	4101	36879	4	22	15	4	0.07
rosen2	1032	3080	47536	8	8	33	8	0.05
lp_fit2p	3000	13525	50284	25	25	70	21	0.79
rosen10	2056	6152	64192	8	8	26	8	0.10
c-30	5321	5321	65693	1583	43	790	30	6.07
lp_fit2d	25	10524	129042	25	25	27	21	0.76

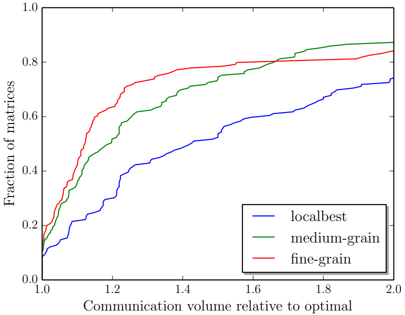
LB = localbest = best of 1D row, 1D column (v1-v3)

MG = medium-grain method (v4.0)

FG = fine-grain model (Çatalyürek and Aykanat 2001)

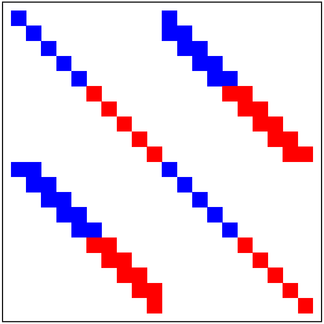
Opt = optimal using MondriaanOpt (Mondriaan v4.1, soon)

Benchmarking 3 methods vs. optimal partitioning



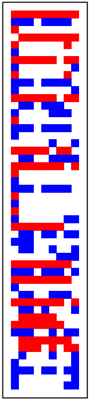
- 217 matrices from U. Florida collection with $nz \leq 1000$
- 85% were solved to optimality for $\varepsilon = 0.03$
- $V_{Opt} = 0$ excluded from test suite
- Medium-grain method solves 87% of test suite within factor 2 of optimal volume.

Matrix steam3



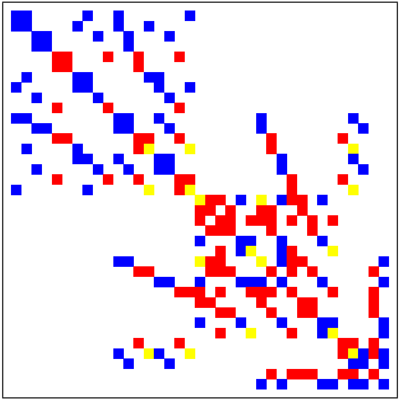
- 80×80 matrix `steam3`, $nz(A) = 928$
- 1D steam model of oil reservoir (Roger Grimes 1983)
- 20 points, 4 degrees of freedom
- $V = 8$, perfect balance

Matrix divorce



- 50×9 matrix `divorce`, $nz(A) = 225$
- Divorce laws in the 50 US states
- Row 0 is Alabama, ..., Row 49 is Wyoming
- Column 0 is Incompatibility, Column 1 is Cruelty, ...
- $V = 8$, imbalance $= nz_{\text{max}} - nz_{\text{min}} = 1$

Matrix cage5



- 37×37 matrix cage5, $nz(A) = 233$
- DNA electrophoresis, 5 monomers in polymer (Alexander van Heukelum 2003)
- $nz_0 = 106$; $nz_1 = 110$; $nz_{free} = 17$
- $V = 14$, imbalance $= nz_{\max} - nz_{\min} = 1$

Conclusion

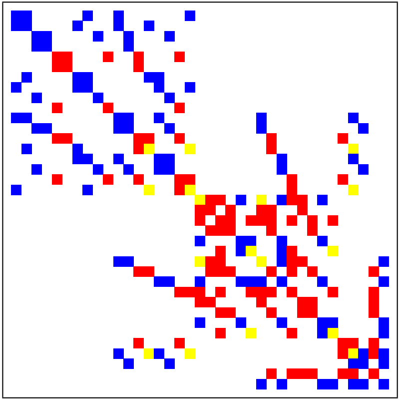
- The medium-grain method is a fast heuristic to compute 2D sparse matrix partitionings with low communication volume at the speed of computing 1D partitionings.
- It can be used to compute edge partitionings of big graphs.
- We also presented an exact branch-and-bound algorithm for computing optimal bipartitionings of small sparse matrices.
- Results are being added, one matrix a day, at `http://www.staff.science.uu.nl/~bisse101/Mondriaan/Opt`

Future work on optimal partitioning

- Expand the MondriaanOpt database.
- Suggestions are welcome.
- Further improve the lower bounds.
- Parallelise the B & B method.
- August 2015: release Mondriaan v4.1 software package, including MondriaanOpt v1.0.

"An exact algorithm for sparse matrix bipartitioning", by Daniël M. Pelt and Rob H. Bisseling, *Journal of Parallel and Distributed Computing*, 2015, in press
http://doi.org/10.1016/j.jpdc.2015.06.005

Thank you!



Universiteit Utrecht