Contemporary Mathematics
Volume **588**, 2013
http://dx.doi.org/10.1090/conm/588/11707

Abusing a hypergraph partitioner for unweighted graph partitioning

B. O. Fagginger Auer and R. H. Bisseling

Abstract. We investigate using the Mondriaan matrix partitioner for unweighted graph partitioning in the communication volume and edge-cut metrics. By converting the unweighted graphs to appropriate matrices, we measure Mondriaan's performance as a graph partitioner for the 10th DIMACS challenge on graph partitioning and clustering. We find that Mondriaan can effectively be used as a graph partitioner: w.r.t. the edge-cut metric, Mondriaan's best results are on average within 13% of the best known results as listed in Chris Walshaw's partitioning archive, but it is an order of magnitude slower than dedicated graph partitioners.

1. Introduction

In this paper, we use the Mondriaan matrix partitioner [**22**] to partition the graphs from the 10th DIMACS challenge on graph partitioning and clustering [**1**]. In this way, we can compare Mondriaan's performance as a graph partitioner with the performance of the state-of-the-art partitioners participating in the challenge.

An *undirected graph* G is a pair (V, E), with vertices V, and edges E that are of the form $\{u, v\}$ for $u, v \in V$ with possibly $u = v$. For vertices $v \in V$, we denote the set of all of v's *neighbours* by

$$V_v := \{u \in V \mid \{u, v\} \in E\}.$$

Note that vertex v is a neighbour of itself precisely when the self-edge $\{v, v\} \in E$.

Hypergraphs are a generalisation of undirected graphs, where edges can contain an arbitrary number of vertices. A *hypergraph* $\mathcal{G}$ is a pair $(\mathcal{V}, \mathcal{N})$, with vertices $\mathcal{V}$, and nets (or hyperedges) $\mathcal{N}$; nets are subsets of $\mathcal{V}$ that can contain any number of vertices.

Let $\epsilon > 0$, $k \in \mathbf{N}$, and $G = (V, E)$ be an undirected graph. Then a valid solution to the *graph partitioning problem* for partitioning G into k parts with imbalance ϵ, is a partitioning $\Pi : V \to \{1, \ldots, k\}$ of the graph's vertices into k parts, each part $\Pi^{-1}(\{i\})$ containing at most

$$|\Pi^{-1}(\{i\})| \leq (1 + \epsilon) \left\lceil \frac{|V|}{k} \right\rceil, \qquad (1 \leq i \leq k) \tag{1.1}$$

vertices.

2010 *Mathematics Subject Classification.* Primary 05C65, 05C70; Secondary 05C85.
Key words and phrases. Hypergraphs, graph partitioning, edge cut, communication volume.

To measure the quality of a valid partitioning we use two different metrics. The *communication volume metric*[1] [**1**] is defined by

$$\mathrm{CV}(\Pi) := \max_{1\leq i\leq k} \sum_{\substack{v\in V \\ \Pi(v)=i}} |\Pi(V_v) \setminus \{\Pi(v)\}|. \tag{1.2}$$

For each vertex v, we determine the number $\pi(v)$ of different parts in which v has neighbours, except its own part $\Pi(v)$. Then, the communication volume is given by the maximum over i, of the sum of all $\pi(v)$ for vertices v belonging to part i.

The *edge-cut metric* [**1**], defined as

$$\mathrm{EC}(\Pi) := |\{\{u, v\} \in E \mid \Pi(u) \neq \Pi(v)\}|, \tag{1.3}$$

measures the number of edges between different parts of the partitioning Π.

TABLE 1. Overview of available software for partitioning graphs (left) and hypergraphs (right), from [**3**, Table 12.1].

Name	Ref.	Sequential/ parallel	Name	Ref.	Sequential/ parallel
Chaco	[**13**]	sequential	hMETIS	[**15**]	sequential
METIS	[**14**]	sequential	ML-Part	[**6**]	sequential
Scotch	[**18**]	sequential	Mondriaan	[**22**]	sequential
Jostle	[**23**]	parallel	PaToH	[**8**]	sequential
ParMETIS	[**16**]	parallel	Par*k*way	[**21**]	parallel
PT-Scotch	[**10**]	parallel	Zoltan	[**12**]	parallel

There exist a lot of different (hyper)graph partitioners, which are summarised in Table 1. All partitioners follow a multi-level strategy [**5**], where the (hyper)graph is coarsened by generating a matching of the (hyper)graph's vertices and contracting matched vertices to a single vertex. Doing this recursively creates a hierarchy of increasingly coarser approximations of the original (hyper)graph. After this has been done, an initial partitioning is generated on the coarsest (hyper)graph in the hierarchy, i.e. the one possessing the smallest number of vertices. This partitioning is subsequently propagated to the finer (hyper)graphs in the hierarchy and refined at each level (e.g. using the Kernighan–Lin algorithm [**17**]), until we reach the original (hyper)graph and obtain the final partitioning.

2. Mondriaan

2.1. Mondriaan sparse matrix partitioner. The Mondriaan partitioner has been designed to partition the matrix and the vectors for a parallel sparse matrix–vector multiplication, where a sparse matrix A is multiplied by a dense input vector $\mathbf{v}$ to give a dense output vector $\mathbf{u} = A\,\mathbf{v}$ as the result. First, the matrix partitioning algorithm is executed to minimise the total communication volume $\mathrm{LV}(\Pi)$ of the partitioning, defined below, and then the vector partitioning algorithm is executed with the aim of balancing the communication among the processors. The matrix partitioning itself does not aim to achieve such balance, but it is not biased in favour of any processor part either.

[1]We forgo custom edge and vertex weights and assume they are all equal to one, because Mondriaan's hypergraph partitioner does not support net weights.

TABLE 2. Available representations of an $m \times n$ matrix $A = (a_{ij})$ by a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{N})$ in Mondriaan.

Name	Ref.	$\mathcal{V}$	$\mathcal{N}$
Column-net	[7]	$\{r_1, \ldots, r_m\}$	$\{\{r_i \mid 1 \le i \le m, a_{ij} \neq 0\} \mid 1 \le j \le n\}$
Row-net	[7]	$\{c_1, \ldots, c_n\}$	$\{\{c_j \mid 1 \le j \le n, a_{ij} \neq 0\} \mid 1 \le i \le m\}$
Fine-grain	[9]	$\{v_{ij} \mid a_{ij} \neq 0\}$	$\underbrace{\{\{v_{ij} \mid 1 \le i \le m, a_{ij} \neq 0\} \mid 1 \le j \le n\}}_{\text{column nets}}$ $\cup \underbrace{\{\{v_{ij} \mid 1 \le j \le n, a_{ij} \neq 0\} \mid 1 \le i \le m\}}_{\text{row nets}}$

Mondriaan uses recursive bipartitioning to split the matrix or its submatrices repeatedly into two parts, choosing the best of the row or column direction in the matrix. The current submatrix is translated into a hypergraph by the column-net or row-net model, respectively (see Table 2). Another possibility is to split the submatrix based on the fine-grain model, and if desired the best split of the three methods can be chosen. The outcome of running Mondriaan is a two-dimensional partitioning of the sparse matrix (i.e., a partitioning where both the matrix rows and columns are split). The number of parts is not restricted to a power of two, as Mondriaan can split parts according to a given ratio, such as 2:1. After each split, Mondriaan adjusts the weight balancing goals of the new parts obtained, as the new part that receives the largest fraction of the weight will need to be stricter in allowing an imbalance during further splits than the part with the smaller fraction.

The total communication volume of the parallel sparse matrix–vector multiplication is minimised by Mondriaan in the following manner. Because the total volume is simply the sum of the volumes incurred by every split into two by the recursive bipartitioning [**22**, Theorem 2.2], the minimisation is completely achieved by the bipartitioning. We will explain the procedure for splits in the column direction (the row direction is similar). When using Mondriaan as a hypergraph partitioner, as we do for the DIMACS challenge, see Section 2.2, only the column direction is used.

First, in the bipartitioning, similar columns are merged by matching columns that have a large overlap in their nonzero patterns. A pair of columns j, j' with similar pattern will then be merged and hence will be assigned to the same processor part in the subsequent initial partitioning, thus preventing the communication that would occur if two nonzeros a_{ij} and $a_{ij'}$ from the same row were assigned to different parts. Repeated rounds of merging during this coarsening phase result in a final sparse matrix with far fewer columns, and a whole multilevel hierarchy of intermediate matrices.

Second, the resulting smaller matrix is bipartitioned using the Kernighan–Lin algorithm [**17**]. This local-search algorithm with so-called hill-climbing capabilities starts with a random partitioning of the columns satisfying the load balance constraints, and then tries to improve it by repeated moves of a column from its current processor part to the other part. To enhance the success of the Kernighan–Lin algorithm and to prevent getting stuck in local minima, we limit the number of columns to at most 200 in this stage; the coarsening only stops when this number has been reached. The Kernighan–Lin algorithm is run eight times and the best solution is taken.

Third, the partitioning of the smaller matrix is propagated back to a partitioning of the original matrix, at each level unmerging pairs of columns while trying to refine the partitioning by one run of the Kernighan–Lin algorithm. This further reduces the amount of communication, while still satisfying the load balance constraints.

If the input and output vector can be partitioned independently, the vector partitioning algorithm usually has enough freedom to achieve a reasonable communication balancing. Each component v_i of the input vector can then be assigned to any of the processors that hold nonzeros in the corresponding column, and each component u_i of the output vector to any of the processors that hold nonzeros in the corresponding row. If the matrix is square, and both vectors must be partitioned in the same way, then there is usually little freedom, as the only common element of row i and column i is the diagonal matrix element a_{ii}, which may or may not be zero. If it is zero, it has no owning processor, and the set of processors owning row i and that owning column i may be disjoint. This means that the total communication volume must be increased by one for vector components v_i and u_i. If the matrix diagonal has only nonzero elements, however, the vector partitioning can be achieved without incurring additional communication by assigning vector components v_i and u_i to the same processor as the diagonal matrix element a_{ii}. More details on the matrix and vector partitioning can be found in [**22**]; improved methods for vector partitioning are given in [**4**], see also [**2**].

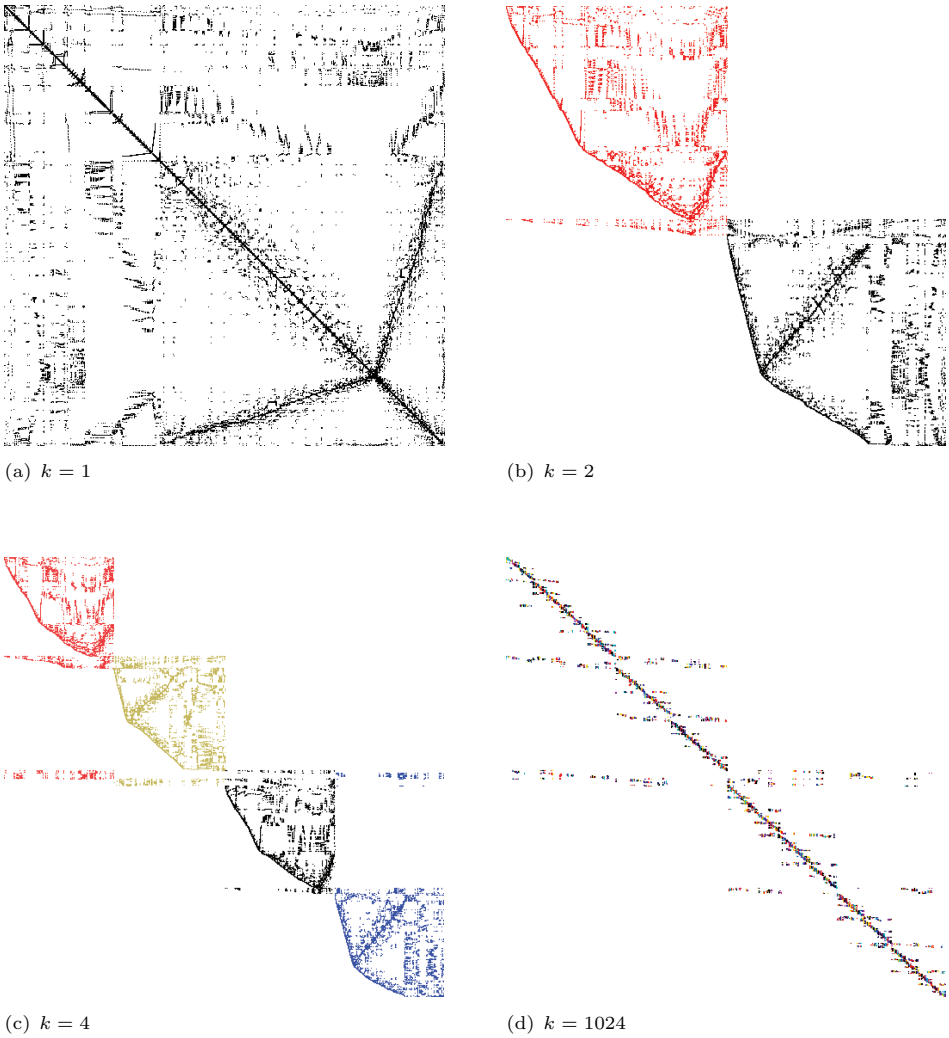
2.2. Mondriaan hypergraph partitioner. Here, we will use Mondriaan as a hypergraph partitioner, which can be done by choosing the column direction in all splits, so that columns are vertices and rows are nets. This means that we use Mondriaan in one-dimensional mode, as only rows will be split. Figure 1 illustrates this splitting procedure. Mondriaan has the option to use its own, native hypergraph bipartitioner, or link to the external partitioner PaToH [**8**]. In the present work, we use the native partitioner.

For the graph partitioning challenge posed by DIMACS, we try to fit the existing software to the aims of the challenge. One could say that this entails abusing the software, as it was designed for a different purpose, namely matrix and hypergraph partitioning. Using a hypergraph partitioner to partition graphs will be at the cost of some additional, unnecessary overhead. Still, it will be interesting to see how the Mondriaan software performs in this unforeseen mode, and to compare the quality of the generated partitionings to the quality of partitionings generated by other software, in particular by graph partitioning packages.

In the situation of the challenge, we can only use the matrix partitioning of Mondriaan and not the vector partitioning, as the vertex partitioning of the graph is already completely determined by the column partitioning of the matrix. The balance of the communication will then solely depend on the balance achieved by the matrix partitioning.

Internally, Mondriaan's hypergraph partitioner solves the following problem. For a hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{N})$ with vertex weights $\zeta : \mathcal{V} \to \mathbf{N}$, an imbalance factor $\epsilon > 0$, and a number of parts $k \in \mathbf{N}$, Mondriaan's partitioner produces a partitioning $\Pi : \mathcal{V} \to \{1, \ldots, k\}$ such that

$$\zeta(\Pi^{-1}(\{i\})) \leq (1+\epsilon) \left\lceil \frac{\zeta(\mathcal{V})}{k} \right\rceil, \qquad (1 \leq i \leq k), \tag{2.1}$$



(a) $k = 1$ (b) $k = 2$

(c) $k = 4$ (d) $k = 1024$

FIGURE 1. Mondriaan 1D column partitioning of the graph `fe_tooth`, modelled as a sparse matrix (cf. Theorem 2.1), into $k = 1$, 2, 4, and 1024 parts with imbalance $\epsilon = 0.03$. The rows and columns of the matrices have been permuted for $k > 1$ to Separated Block Diagonal form, see [**24**].

where the partitioner tries to minimise the $(\lambda - 1)$-*volume*

$$\mathrm{LV}(\Pi) := \sum_{n \in \mathcal{N}} (|\Pi(n)| - 1). \tag{2.2}$$

We will now translate the DIMACS partitioning problems from Section 1 to the hypergraph partitioning problem that Mondriaan is designed to solve, by creating a suitable hypergraph $\mathcal{G}$, encoded as a sparse matrix A in the row-net model.

2.3. Minimising communication volume. Let $G = (V, E)$ be a given graph, $k \in \mathbf{N}$, and $\epsilon > 0$. Our aim will be to construct a matrix A from G such that minimising (2.2) subject to (2.1) enforces minimisation of (1.2) subject to (1.1).

To satisfy (1.1), we need to create one column in A for each vertex in V, such that the hypergraph represented by A in the row-net model will have $\mathcal{V} = V$. This is also necessary to have a direct correspondence between partitionings of the vertices V of the graph and the vertices $\mathcal{V}$ of the hypergraph. Setting the weights ζ of all vertices/matrix columns to 1 will then ensure that (1.1) is satisfied if and only if (2.1) is satisfied.

It is a little more tricky to match (1.2) to (2.2). Note that because of the maximum in (1.2), we are not able to create an equivalent formulation. However, as

$$\mathrm{CV}(\Pi) \leq \sum_{i=1}^{k} \sum_{\substack{v \in V \\ \Pi(v)=i}} |\Pi(V_v) \setminus \{\Pi(v)\}| = \sum_{v \in V} |\Pi(V_v) \setminus \{\Pi(v)\}|, \tag{2.3}$$

we can provide an upper bound, which we can use to limit $\mathrm{CV}(\Pi)$. We need to choose the rows of A, corresponding to nets in the row-net hypergraph $\mathcal{G} = (\mathcal{V}, \mathcal{N})$, such that (2.3) and (2.2) are in agreement.

For a net $n \in \mathcal{N}$, we have that $n \subseteq \mathcal{V} = V$ is simply a collection of vertices of G, so $|\Pi(n)|$ in (2.2) equals the number of different parts in which the vertices of n are contained. In (2.3) we count, for a vertex $v \in V$, all parts in which v has a neighbour, except $\Pi(v)$. Note that this number equals $|\Pi(V_v) \setminus \{\Pi(v)\}| = |\Pi(V_v \cup \{v\})| - 1$.

Hence, we should pick $\mathcal{N} := \{V_v \cup \{v\} \mid v \in V\}$ as the set of nets, for (2.3) and (2.2) to agree. In the row-net matrix model, this corresponds to letting A be a matrix with a row for every vertex $v \in V$, filled with nonzeros $a_{v\,v}$ and $a_{u\,v}$ for all $u \in V_v \setminus \{v\}$. Then, for this hypergraph $\mathcal{G}$, we have by (2.3) that $\mathrm{CV}(\Pi) \leq \mathrm{LV}(\Pi)$. Note that since the communication volume is defined as a maximum, we also have that $k\,\mathrm{CV}(\Pi) \geq \mathrm{LV}(\Pi)$.

THEOREM 2.1. *Let $G = (V, E)$ be a given graph, $k \in \mathbf{N}$, and $\epsilon > 0$. Let A be the $|V| \times |V|$ matrix with entries*

$$a_{u\,v} := \begin{cases} 1 & \text{if } \{u, v\} \in E \text{ or } u = v, \\ 0 & \text{otherwise,} \end{cases}$$

for $u, v \in V$, and let $\mathcal{G} = (\mathcal{V}, \mathcal{N})$ be the hypergraph corresponding to A in the row-net model with vertex weights $\zeta(v) = 1$ for all $v \in \mathcal{V}$.

Then, for every partitioning $\Pi : V \to \{1, \ldots, k\}$, we have that Π satisfies (1.1) if and only if Π satisfies (2.1), and

$$\frac{1}{k}\, LV(\Pi) \leq CV(\Pi) \leq LV(\Pi). \tag{2.4}$$

2.4. Minimising edge cut. We will now follow the same procedure as in Section 2.3 to construct a matrix A such that minimising (2.2) subject to (2.1) is equivalent to minimising (1.3) subject to (1.1).

As in Section 2.3, the columns of A should correspond to the vertices V of G to ensure that (2.1) is equivalent to (1.1).

Equation (1.3) simply counts all of G's edges that contain vertices belonging to two parts of the partitioning Π. Since every edge contains vertices belonging to

at least one part, and at most two parts, this yields

$$\mathrm{EC}(\Pi) = \sum_{e \in E} (|\Pi(e)| - 1).$$

Choosing $\mathcal{N} := E$ will therefore give us a direct correspondence between (2.2) and (1.3).

THEOREM 2.2. *Let $G = (V, E)$ be a given graph, $k \in \mathbf{N}$, and $\epsilon > 0$. Let A be the $|E| \times |V|$ matrix with entries*

$$a_{e\,v} := \begin{cases} 1 & \text{if } v \in e, \\ 0 & \text{otherwise,} \end{cases}$$

for $e \in E$, $v \in V$, and let $\mathcal{G} = (\mathcal{V}, \mathcal{N})$ be the hypergraph corresponding to A in the row-net model with vertex weights $\zeta(v) = 1$ for all $v \in \mathcal{V}$.

Then, for every partitioning $\Pi : V \to \{1, \ldots, k\}$, we have that Π satisfies (1.1) *if and only if Π satisfies* (2.1)*, and*

$$EC(\Pi) = LV(\Pi). \tag{2.5}$$

With Theorem 2.1 and Theorem 2.2, we know how to translate a given graph G to a hypergraph that Mondriaan can partition to obtain solutions to the DIMACS partitioning challenges.

3. Results

We measure Mondriaan's performance as a graph partitioner by partitioning graphs from the `walshaw/` [**20**] category, as well as a subset of the specified partitioning instances of the DIMACS challenge test bed [**1**], see Tables 3 and 4. This is done by converting the graphs to matrices, as described by Theorem 2.1 and Theorem 2.2, and partitioning these matrices with Mondriaan 3.11, using the `onedimcol` splitting strategy (since the matrices represent row-net hypergraphs) with the `lambda1` metric (cf. (2.2)). The imbalance is set to $\epsilon = 0.03$, the number of parts k is chosen from $\{2, 4, \ldots, 1024\}$, and we measure the communication volumes and edge cuts over 16 runs of the Mondriaan partitioner (as Mondriaan uses random tie-breaking). All results were recorded on a dual quad-core AMD Opteron 2378 system with 32GiB of main memory and they can be found in Tables 5–8 and Figures 2 and 3. None of the graphs from Table 3 or 4 contain self-edges, edge weights, or vertex weights. Therefore, the values recorded in Tables 5–8 satisfy either (1.2) or (1.3) (which both assume unit weights), and can directly be compared to the results of other DIMACS challenge participants.

Tables 5 and 6 contain the lowest communication volumes and edge cuts obtained by Mondriaan in 16 runs for the graphs from Table 3. The strange dip in the communication volume for `finan512` in Table 5 for $k = 32$ parts can be explained by the fact that the graph `finan512` consists exactly of 32 densely connected parts with few connections between them, see the visualisation of this graph in [**11**], such that there is a natural partitioning with very low communication volume in this case.

To determine how well Mondriaan performs as a graph partitioner, we have also partitioned the graphs from Tables 3 and 4 using METIS 5.0.2 [**14**] and Scotch 5.1.12 [**18**]. For METIS we used the high-quality `PartGraphKway` option, while Scotch was invoked using `graphPart` with the `QUALITY` and `SAFETY` strategies enabled. We furthermore compare the results from Table 6 to the lowest known edge cuts

TABLE 3. Graphs $G = (V, E)$ from the `walshaw/` [**1**, **20**] category.

G	$\|V\|$	$\|E\|$	G	$\|V\|$	$\|E\|$
add20	2,395	7,462	bcsstk30	28,924	1,007,284
data	2,851	15,093	bcsstk31	35,588	572,914
3elt	4,720	13,722	fe_pwt	36,519	144,794
uk	4,824	6,837	bcsstk32	44,609	985,046
add32	4,960	9,462	fe_body	45,087	163,734
bcsstk33	8,738	291,583	t60k	60,005	89,440
whitaker3	9,800	28,989	wing	62,032	121,544
crack	10,240	30,380	brack2	62,631	366,559
wing_nodal	10,937	75,488	finan512	74,752	261,120
fe_4elt2	11,143	32,818	fe_tooth	78,136	452,591
vibrobox	12,328	165,250	fe_rotor	99,617	662,431
bcsstk29	13,992	302,748	598a	110,971	741,934
4elt	15,606	45,878	fe_ocean	143,437	409,593
fe_sphere	16,386	49,152	144	144,649	1,074,393
cti	16,840	48,232	wave	156,317	1,059,331
memplus	17,758	54,196	m14b	214,765	1,679,018
cs4	22,499	43,858	auto	448,695	3,314,611

TABLE 4. Graphs $G = (V, E)$ from the 10th DIMACS challenge [**1**] partitioning instances.

	G	$\|V\|$	$\|E\|$
1	delaunay_n15	32,768	98,274
2	kron_g500-simple-logn17	131,072	5,113,985
3	coAuthorsCiteseer	227,320	814,134
4	rgg_n_2_18_s0	262,144	1,547,283
5	auto	448,695	3,314,611
6	G3_circuit	1,585,478	3,037,674
7	kkt_power	2,063,494	6,482,320
8	M6	3,501,776	10,501,936
9	AS365	3,799,275	11,368,076
10	NLR	4,163,763	12,487,976
11	hugetric-00000	5,824,554	8,733,523
12	great-britain.osm	7,733,822	8,156,517
13	asia.osm	11,950,757	12,711,603
14	hugebubbles-00010	19,458,087	29,179,764

with 3% imbalance for graphs from the `walshaw/` category, available from `http://staffweb.cms.gre.ac.uk/~wc06/partition/` [**20**]. These data were retrieved on May 8, 2012 and include results from the KaFFPa partitioner, contributed by Sanders and Schulz [**19**], who also participated in the DIMACS challenge. Results for graphs from the DIMACS challenge, Tables 7 and 8, are given for the number of parts k specified in the challenge partitioning instances, for a single run of the Mondriaan, METIS, and Scotch partitioners.

TABLE 5. Minimum communication volume, (1.2), over 16 Mondriaan runs, for graphs from the `walshaw/` category, Table 3, divided into $k = 2, 4, \ldots, 64$ parts with imbalance $\epsilon = 0.03$. A '-' indicates that Mondriaan was unable to generate a partitioning satisfying the balancing requirement, (1.1).

G	2	4	8	16	32	64
add20	74	101	118	141	159	-
data	63	84	80	78	65	-
3elt	45	65	59	65	53	49
uk	19	27	36	33	31	24
add32	9	21	29	24	20	22
bcsstk33	454	667	719	630	547	449
whitaker3	64	130	104	98	77	60
crack	95	97	123	100	78	64
wing_nodal	453	593	523	423	362	256
fe_4elt2	66	94	97	85	69	60
vibrobox	996	1,080	966	887	663	482
bcsstk29	180	366	360	336	252	220
4elt	70	90	86	89	88	71
fe_sphere	193	213	178	139	107	83
cti	268	526	496	379	295	200
memplus	2,519	1,689	1,069	720	572	514
cs4	319	492	409	311	228	161
bcsstk30	283	637	611	689	601	559
bcsstk31	358	492	498	490	451	400
fe_pwt	120	122	133	145	148	132
bcsstk32	491	573	733	671	561	442
fe_body	109	143	173	171	145	133
t60k	71	141	154	139	129	96
wing	705	854	759	594	451	324
brack2	231	650	761	635	562	458
finan512	75	76	137	141	84	165
fe_tooth	1,238	1,269	1,282	1,066	844	703
fe_rotor	549	1,437	1,258	1,138	944	749
598a	647	1,400	1,415	1,432	1,064	871
fe_ocean	269	797	1,002	1,000	867	647
144	1,660	2,499	2,047	1,613	1,346	1,184
wave	2,366	2,986	2,755	2,138	1,640	1,222
m14b	921	2,111	2,086	2,016	1,524	1,171
auto	2,526	4,518	4,456	3,982	3,028	2,388

TABLE 6. Minimum edge cut, (1.3), over 16 Mondriaan runs, for graphs from the `walshaw/` category, Table 3, divided into $k = 2, 4, \ldots, 64$ parts with imbalance $\epsilon = 0.03$. A '-' indicates that Mondriaan was unable to generate a partitioning satisfying the balancing requirement, (1.1).

G	2	4	8	16	32	64
add20	680	1,197	1,776	2,247	2,561	-
data	195	408	676	1,233	2,006	-
3elt	87	206	368	639	1,078	1,966
uk	20	43	98	177	299	529
add32	21	86	167	247	441	700
bcsstk33	10,068	21,993	37,054	58,188	82,102	114,483
whitaker3	126	385	692	1,172	1,825	2,769
crack	186	372	716	1,169	1,851	2,788
wing_nodal	1,703	3,694	5,845	8,963	12,870	17,458
fe_4elt2	130	350	616	1,091	1,770	2,760
vibrobox	10,310	19,401	28,690	37,038	45,877	53,560
bcsstk29	2,846	8,508	16,714	25,954	39,508	59,873
4elt	137	335	543	1,040	1,724	2,896
fe_sphere	404	822	1,258	1,972	2,857	4,082
cti	318	934	1,786	2,887	4,302	6,027
memplus	5,507	9,666	12,147	14,077	15,737	17,698
cs4	389	1,042	1,654	2,411	3,407	4,639
bcsstk30	6,324	16,698	35,046	77,589	123,766	186,084
bcsstk31	2,677	7,731	14,299	25,212	40,641	65,893
fe_pwt	347	720	1,435	2,855	5,888	9,146
bcsstk32	4,779	9,146	23,040	41,214	66,606	102,977
fe_body	271	668	1,153	2,011	3,450	5,614
t60k	77	227	506	952	1,592	2,483
wing	845	1,832	2,843	4,451	6,558	8,929
brack2	690	2,905	7,314	12,181	19,100	28,509
finan512	162	324	891	1,539	2,592	10,593
fe_tooth	3,991	7,434	12,736	19,709	27,670	38,477
fe_rotor	1,970	7,716	13,643	22,304	34,515	50,540
598a	2,434	8,170	16,736	27,895	43,192	63,056
fe_ocean	317	1,772	4,316	8,457	13,936	21,522
144	6,628	16,822	27,629	41,947	62,157	86,647
wave	8,883	18,949	32,025	47,835	69,236	94,099
m14b	3,862	13,464	26,962	46,430	73,177	107,293
auto	9,973	27,297	49,087	83,505	132,998	191,429

TABLE 7. Communication volume, (1.2), for graphs from Table 4, divided into k parts with imbalance $\epsilon = 0.03$ for one run of Mondriaan, METIS, and Scotch. The numbering of the graphs is given by Table 4.

G	k	Mon.	MET.	Sco.	G	k	Mon.	MET.	Sco.
1	8	228	238	250	8	2	1,392	1,420	1,416
	16	180	169	202		8	2,999	2,242	2,434
	32	154	134	137		32	1,852	1,497	1,611
	64	110	112	94		128	1,029	783	814
	128	94	72	88		256	737	553	606
2	2	38,565	46,225	49,273	9	64	1,375	1,099	1,266
	4	38,188	61,833	56,503		128	1,037	814	837
	8	73,739	62,418	60,600		256	761	555	639
	16	82,356	47,988	61,469		512	552	419	481
	32	88,273	43,990	74,956		1024	374	299	330
3	4	11,063	10,790	20,018	10	8	2,508	2,707	3,104
	8	9,652	9,951	14,004		32	1,659	1,620	1,763
	16	7,216	6,507	9,928		128	1,056	820	895
	32	4,732	4,480	6,684		256	728	624	713
	64	3,298	3,111	4,273		512	596	464	478
4	8	749	710	837	11	2	1,222	1,328	1,408
	16	522	640	665		4	2,536	2,668	2,693
	32	524	437	455		32	1,175	1,224	1,168
	64	342	359	348		64	1,022	985	893
	128	285	238	326		256	594	467	510
5	64	2,423	2,407	2,569	12	32	235	214	191
	128	1,774	1,634	1,766		64	228	133	149
	256	1,111	1,120	1,248		128	194	130	138
	512	786	717	824		256	135	95	115
	1024	552	519	540		1024	102	78	83
6	2	1,219	1,267	1,308	13	64	139	53	84
	4	1,887	1,630	2,144		128	139	58	73
	32	1,304	1,285	1,291		256	145	65	104
	64	1,190	1,111	1,228		512	157	110	90
	256	668	566	702		1024	127	124	109
7	16	6,752	9,303	36,875	14	4	3,359	3,283	3,620
	32	7,057	9,123	20,232		32	2,452	2,139	2,462
	64	7,255	9,244	10,669		64	1,864	1,592	1,797
	256	4,379	4,198	4,842		256	1,143	847	1,040
	512	3,280	2,589	3,265		512	737	621	704

TABLE 8. Edge cut, (1.3), for graphs from Table 4, divided into k parts with imbalance $\epsilon = 0.03$ for one run of Mondriaan, METIS, and Scotch. The numbering of the graphs is given by Table 4.

G	k	Mon.	MET.	Sco.	G	k	Mon.	MET.	Sco.
1	8	1,367	1,358	1,386	8	2	2,949	2,869	2,827
	16	2,164	2,170	2,121		8	15,052	14,206	14,622
	32	3,217	3,267	3,283		32	39,756	35,906	36,795
	64	4,840	4,943	4,726		128	81,934	78,824	80,157
	128	7,134	6,979	7,000		256	117,197	114,413	114,800
2	2	208,227	1,972,153	773,367	9	64	56,009	53,557	54,835
	4	835,098	2,402,130	2,614,571		128	81,768	78,055	79,193
	8	1,789,048	2,988,293	3,417,254		256	119,394	113,171	114,758
	16	2,791,475	3,393,061	3,886,568		512	167,820	163,673	165,078
	32	3,587,053	3,936,154	4,319,148		1024	239,947	234,301	234,439
3	4	37,975	37,151	67,513	10	8	16,881	16,992	17,172
	8	54,573	53,502	81,556		32	42,523	40,130	40,967
	16	67,308	66,040	92,992		128	90,105	86,332	86,760
	32	77,443	75,448	104,050		256	129,635	124,737	126,233
	64	85,610	84,111	111,090		512	186,016	178,324	179,779
4	8	4,327	4,381	4,682	11	2	1,345	1,328	1,408
	16	7,718	7,107	7,879		4	4,197	3,143	3,693
	32	13,207	10,386	11,304		32	16,659	13,981	14,434
	64	20,546	16,160	16,630		64	24,031	20,525	21,597
	128	32,039	24,644	25,749		256	50,605	44,082	44,634
5	64	192,783	188,424	196,385	12	32	2,213	1,622	1,770
	128	266,541	257,800	265,941		64	3,274	2,461	2,891
	256	359,123	346,655	366,258		128	5,309	3,948	4,439
	512	475,284	455,321	479,379		256	8,719	6,001	6,710
	1024	621,339	591,928	629,085		1024	19,922	14,692	15,577
6	2	1,370	1,371	1,339	13	64	1,875	623	1,028
	4	3,174	3,163	3,398		128	3,246	1,106	1,637
	32	14,326	14,054	14,040		256	5,381	2,175	2,938
	64	24,095	22,913	25,434		512	9,439	4,157	5,133
	256	58,164	57,255	60,411		1024	15,842	7,987	9,196
7	16	136,555	132,431	279,808	14	4	6,290	5,631	6,340
	32	204,688	219,370	370,494		32	29,137	25,049	27,693
	64	339,620	351,913	462,030		64	43,795	38,596	41,442
	256	653,613	662,569	694,692		256	90,849	82,566	86,554
	512	774,477	755,994	814,142		512	131,481	118,974	124,694

TABLE 9. Comparison of the minimum communication volume, (1.2), and edge cut, (1.3), for graphs from Table 3 (`walshaw/` collection) and Table 4 (DIMACS challenge collection). We compare the Mondriaan, METIS, and Scotch partitioners using (3.1) with $\mathcal{X}$ consisting of the graphs from either Table 3 or 4 and using either the communication volume or the edge cut metric.

	Communication volume				Edge cut			
Walshaw		Mon.	MET.	Sco.		Mon.	MET.	Sco.
	Mon.	-	0.98	0.95	Mon.	-	1.02	1.01
	MET.	1.02	-	0.98	MET.	0.98	-	1.00
	Sco.	1.05	1.02	-	Sco.	0.99	1.00	-
DIMACS		Mon.	MET.	Sco.		Mon.	MET.	Sco.
	Mon.	-	1.15	0.99	Mon.	-	1.08	0.98
	MET.	0.87	-	0.86	MET.	0.93	-	0.91
	Sco.	1.01	1.16	-	Sco.	1.02	1.10	-

Table 9 gives a summary of each partitioner's relative performance with respect to the others. To illustrate how we compare the quality of the partitionings generated by Mondriaan, METIS, and Scotch, consider the following example. Let $\mathcal{X}$ be a collection of graphs (e.g. the graphs from Table 3) on which we would like to compare the quality of the Mondriaan and METIS partitioners in the communication volume metric. Let Π_G^{Mon} and Π_G^{MET} denote the partitionings found for the graph $G \in \mathcal{X}$ by Mondriaan and METIS, respectively. Then, we determine how much better Mondriaan performs than METIS by looking at the average logarithm of the ratios of the communication volumes for all partitionings of graphs in $\mathcal{X}$,

$$\kappa_{\text{Mon,MET}}(\mathcal{X}) := \exp\left(\frac{1}{|\mathcal{X}|}\sum_{G\in\mathcal{X}} \log\frac{\text{CV}(\Pi_G^{\text{Mon}})}{\text{CV}(\Pi_G^{\text{MET}})}\right), \tag{3.1}$$

which is equal to 0.98 in Table 9 for $\mathcal{X} = \{\text{graphs from Table 3}\}$. If the value from (3.1) is smaller than 1, Mondriaan outperforms METIS, while METIS outperforms Mondriaan if it is larger than 1. We use this quality measure instead of simply calculating the average of all $\text{CV}(\Pi_G^{\text{Mon}})/\text{CV}(\Pi_G^{\text{MET}})$ ratios, because it gives us a symmetric comparison of all partitioners, in the following sense:

$$\kappa_{\text{Mon,MET}}(\mathcal{X}) = 1/\kappa_{\text{MET,Mon}}(\mathcal{X}).$$

Scotch is unable to optimise for the communication volume metric directly and therefore it is not surprising that Scotch is outperformed by both Mondriaan and METIS in this metric. Surprisingly, Mondriaan outperforms Scotch in terms of edge cut for the graphs from Table 4. The more extreme results for the graphs from Table 4 could be caused by the fact that they have been recorded for a single run of the partitioners, while the results for graphs from Table 3 are the best in 16 runs. METIS yields lower average communication volumes and edge cuts than both Mondriaan and Scotch in almost all DIMACS cases.

If we compare the edge cuts for graphs from Table 3 to the best-known results from [**20**], we find that Mondriaan's, METIS', and Scotch's best edge cuts obtained in 16 runs are on average 13%, 10%, and 10% larger, respectively, than those from [**20**].

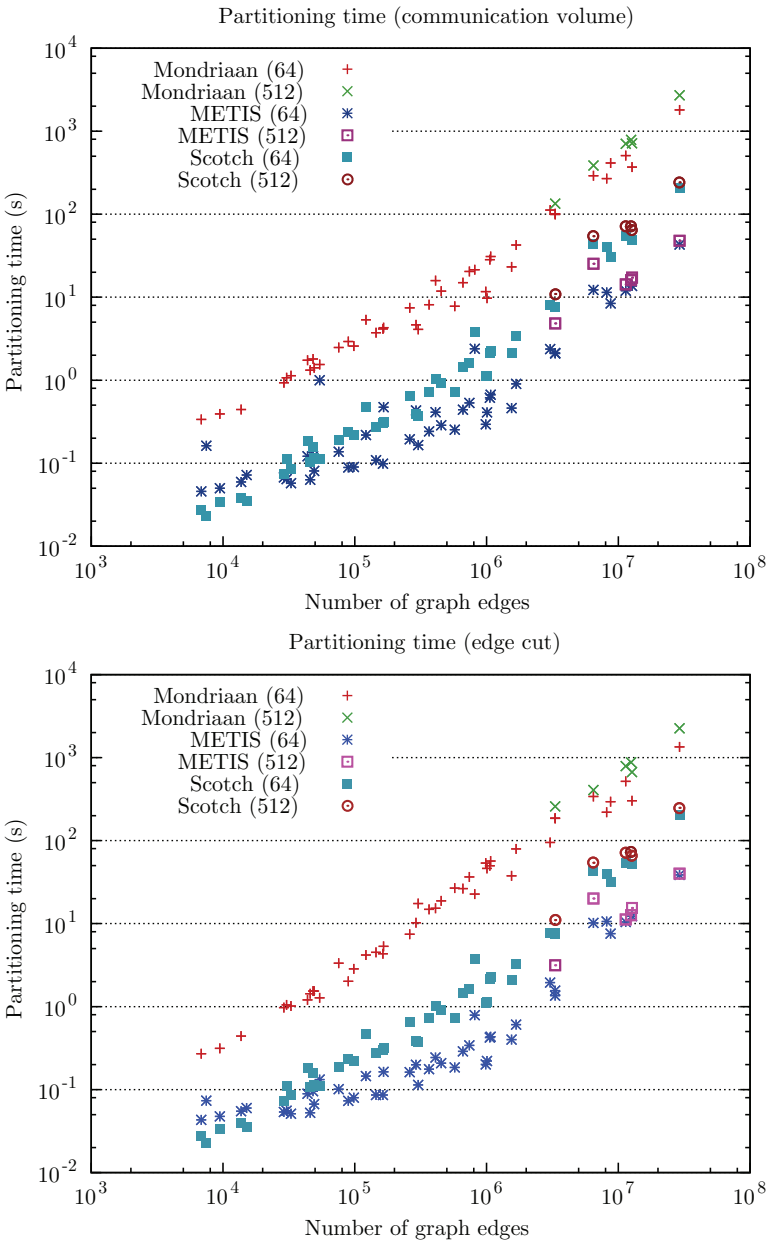


FIGURE 2. The average partitioning time required by the Mondriaan, METIS, and Scotch partitioners to generate the partitionings from Table 5–8 (for 64 and 512 parts).

In Figure 2, we plot the time required by Mondriaan, METIS, and Scotch to create a partitioning for both communication volume and edge cut. Note that the partitioning times are almost the same for both communication volume and edge cut minimisation. METIS is on average 29× faster than Mondriaan for 64 parts

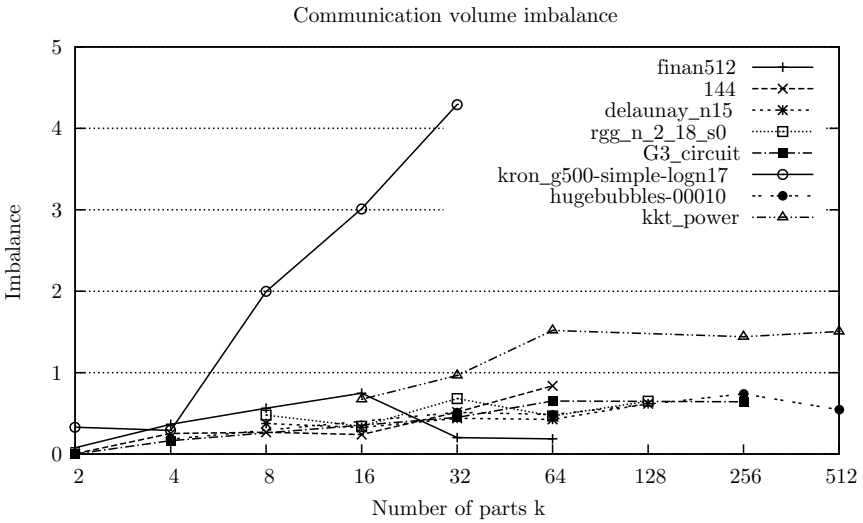


FIGURE 3. The communication volume imbalance given by (3.2), plotted for several graphs.

and Scotch is 12× faster. Note that only six (large) matrices are partitioned into 512 parts.

In the absence of self-edges, the number of nonzeros in the matrices from Theorem 2.1 and Theorem 2.2 equals $2|E| + |V|$ and $2|E|$, respectively. However, the matrix sizes are equal to $|V| \times |V|$ and $|E| \times |V|$, respectively. Therefore, the number of nonzeros in matrices from Theorem 2.2 is smaller, but the larger number of nets (typically $|E| > |V|$, e.g. `rgg_n_2_18_s0`) will lead to increased memory requirements for the edge-cut matrices.

We have also investigated Mondriaan's communication volume imbalance, defined for a partitioning Π of G into k parts as

$$\frac{\mathrm{CV}(\Pi)}{\mathrm{LV}(\Pi)/k} - 1. \tag{3.2}$$

This equation measures the imbalance in communication volume and can be compared to the factor ϵ for vertex imbalance in (1.1). We plot (3.2) for a selection of graphs in Figure 3, where we see that the deviation of the communication volume $\mathrm{CV}(\Pi)$ from perfect balance, i.e. from $\mathrm{LV}(\Pi)/k$, is very small compared to the theoretical upper bound of $k - 1$ (via (2.4)), for all graphs except `kron_g500-simple-logn17`. This means that for most graphs, at most a factor of 2–3 in communication volume per processor can still be gained by improving the communication balance. Therefore, as the number of parts increases, the different parts of the partitionings generated by Mondriaan are not only balanced in terms of vertices, cf. (1.1), but also in terms of communication volume.

4. Conclusion

We have shown that it is possible to use the Mondriaan matrix partitioner as a graph partitioner by constructing appropriate matrices of a given graph for either the communication volume or edge-cut metric. Mondriaan's performance was

measured by partitioning graphs from the 10th DIMACS challenge on graph partitioning and clustering with Mondriaan, METIS, and Scotch, as well as comparing obtained edge cuts with the best known results from [**20**]: here Mondriaan's best edge cut in 16 runs was, on average, 13% higher than the best known. Mondriaan is competitive in terms of partitioning quality (METIS' and Scotch's best edge cuts are, on average, 10% higher than the best known), but it is an order of magnitude slower (Figure 2). METIS is the overall winner, both in quality and performance. In conclusion, it is possible to perform graph partitioning with a hypergraph partitioner, but graph partitioners are much faster.

To our surprise, the partitionings generated by Mondriaan are reasonably balanced in terms of communication volume, as shown in Figure 3, even though Mondriaan does not perform explicit communication volume balancing during matrix partitioning. We attribute the observed balancing to the fact that the Mondriaan algorithm performs random tie-breaking, without any preference for a specific part of the partitioning.

Fortunately, for the given test set of the DIMACS challenge, we did not need to consider edge weights. However, for Mondriaan to be useful as graph partitioner also for weighted graphs, we have to extend Mondriaan to take hypergraph net weights into account for the $(\lambda - 1)$-metric, (2.2). We intend to add this feature in a next version of Mondriaan.

References

[1] D. A. Bader, P. Sanders, D. Wagner, H. Meyerhenke, B. Hendrickson, D. S. Johnson, C. Walshaw, and T. G. Mattson, *10th DIMACS implementation challenge - graph partitioning and graph clustering*, 2012; `http://www.cc.gatech.edu/dimacs10`.

[2] Rob H. Bisseling, *Parallel scientific computation: A structured approach using BSP and MPI*, Oxford University Press, Oxford, 2004. MR2059580

[3] Rob H. Bisseling, Bas O. Fagginger Auer, A. N. Yzelman, Tristan van Leeuwen, and Ümit V. Çatalyürek, *Two-dimensional approaches to sparse matrix partitioning*, Combinatorial scientific computing, Chapman & Hall/CRC Comput. Sci. Ser., CRC Press, Boca Raton, FL, 2012, pp. 321–349, DOI 10.1201/b11644-13. MR2952757

[4] Rob H. Bisseling and Wouter Meesen, *Communication balancing in parallel sparse matrix-vector multiplication*, Electron. Trans. Numer. Anal. **21** (2005), 47–65 (electronic). MR2195104 (2007c:65040)

[5] T. Bui and C. Jones, *A heuristic for reducing fill-in in sparse matrix factorization*, Proceedings Sixth SIAM Conference on Parallel Processing for Scientific Computing, SIAM, Philadelphia, PA, 1993, pp. 445–452.

[6] A. E. Caldwell, A. B. Kahng, and I. L. Markov, *Improved algorithms for hypergraph bipartitioning*, Proceedings Asia and South Pacific Design Automation Conference, ACM Press, New York, 2000, pp. 661–666. DOI 10.1145/368434.368864.

[7] Ü. V. Çatalyürek and C. Aykanat, *Hypergraph-partitioning-based decomposition for parallel sparse-matrix vector multiplication*, IEEE Transactions on Parallel and Distributed Systems **10** (1999), no. 7, 673–693. DOI 10.1109/71.780863.

[8] ———, *PaToH: A multilevel hypergraph partitioning tool, version 3.0*, Bilkent University, Department of Computer Engineering, Ankara, 06533 Turkey. PaToH is available at `http://bmi.osu.edu/~umit/software.htm`, 1999.

[9] ———, *A fine-grain hypergraph model for 2D decomposition of sparse matrices*, Proceedings Eighth International Workshop on Solving Irregularly Structured Problems in Parallel (Irregular 2001), IEEE Press, Los Alamitos, CA, 2001, p. 118.

[10] C. Chevalier and F. Pellegrini, *PT-Scotch: a tool for efficient parallel graph ordering*, Parallel Comput. **34** (2008), no. 6-8, 318–331, DOI 10.1016/j.parco.2007.12.001. MR2428880

[11] Timothy A. Davis and Yifan Hu, *The University of Florida sparse matrix collection*, ACM Trans. Math. Software **38** (2011), no. 1, Art. 1, 25pp, DOI 10.1145/2049662.2049663. MR2865011 (2012k:65051)

[12] K. D. Devine, E. G. Boman, R. T. Heaphy, R. H. Bisseling, and U. V. Catalyurek, *Parallel hypergraph partitioning for scientific computing*, Proceedings IEEE International Parallel and Distributed Processing Symposium 2006, IEEE Press, p. 102, 2006. DOI 10.1109/IPDPS.2006.1639359.

[13] Bruce Hendrickson and Robert Leland, *An improved spectral graph partitioning algorithm for mapping parallel computations*, SIAM J. Sci. Comput. **16** (1995), no. 2, 452–469, DOI 10.1137/0916028. MR1317066 (96b:68140)

[14] George Karypis and Vipin Kumar, *A fast and high quality multilevel scheme for partitioning irregular graphs*, SIAM J. Sci. Comput. **20** (1998), no. 1, 359–392 (electronic), DOI 10.1137/S1064827595287997. MR1639073 (99f:68158)

[15] ———— *Multilevel k-way hypergraph partitioning*, Proceedings 36th ACM/IEEE Conference on Design Automation, ACM Press, New York, 1999, pp. 343–348.

[16] ————, *Parallel multilevel k-way partitioning scheme for irregular graphs*, SIAM Review **41** (1999), no. 2, 278–300. DOI 10.1145/309847.309954.

[17] B. W. Kernighan and S. Lin, *An efficient heuristic procedure for partitioning graphs*, Bell System Technical Journal **49** (2) (1970), 291–307.

[18] F. Pellegrini and J. Roman, *Scotch: A software package for static mapping by dual recursive bipartitioning of process and architecture graphs*, Proceedings High Performance Computing and Networking Europe, Lecture Notes in Computer Science, vol. 1067, Springer, 1996, pp. 493–498. DOI 10.1007/3-540-61142-8_588.

[19] Peter Sanders and Christian Schulz, *Engineering multilevel graph partitioning algorithms*, Algorithms—ESA 2011, Lecture Notes in Comput. Sci., vol. 6942, Springer, Heidelberg, 2011, pp. 469–480, DOI 10.1007/978-3-642-23719-5_40. MR2893224 (2012k:68259)

[20] A. J. Soper, C. Walshaw, and M. Cross, *A combined evolutionary search and multilevel optimisation approach to graph-partitioning*, J. Global Optim. **29** (2004), no. 2, 225–241, DOI 10.1023/B:JOGO.0000042115.44455.f3. MR2092958 (2005k:05228)

[21] A. Trifunović and W. J. Knottenbelt, *Parallel multilevel algorithms for hypergraph partitioning*, Journal of Parallel and Distributed Computing **68** (2008), no. 5, 563–581. DOI 10.1016/j.jpdc.2007.11.002.

[22] Brendan Vastenhouw and Rob H. Bisseling, *A two-dimensional data distribution method for parallel sparse matrix-vector multiplication*, SIAM Rev. **47** (2005), no. 1, 67–95 (electronic), DOI 10.1137/S0036144502409019. MR2149102 (2006a:65070)

[23] C. Walshaw and M. Cross, *JOSTLE: Parallel Multilevel Graph-Partitioning Software – An Overview*, Mesh Partitioning Techniques and Domain Decomposition Techniques (F. Magoules, ed.), Civil-Comp Ltd., 2007, pp. 27–58.

[24] A. N. Yzelman and Rob H. Bisseling, *Cache-oblivious sparse matrix-vector multiplication by using sparse matrix partitioning methods*, SIAM J. Sci. Comput. **31** (2009), no. 4, 3128–3154, DOI 10.1137/080733243. MR2529783 (2011a:65111)

Mathematics Institute, Utrecht University, Budapestlaan 6, 3584 CD, Utrecht, the Netherlands

E-mail address: B.O.FaggingerAuer@uu.nl

Mathematics Institute, Utrecht University, Budapestlaan 6, 3584 CD, Utrecht, the Netherlands

E-mail address: R.H.Bisseling@uu.nl