Combinatorial problems in high-performance computing: partitioning (EXTENDED ABSTRACT)

Rob H. Bisseling, Tristan van Leeuwen,
Mathematical Institute, Utrecht University
Ümit Çatalyürek
Department of Biomedical Informatics, The Ohio State University

Partitioning is of fundamental importance in high-performance computing: partitioning the data and the associated computational work in an optimal manner leads to good load balance and minimal communication in parallel computations on modern architectures. Often, the computation is irregular and the data set is described by a sparse matrix, a graph, or a hypergraph. This results in a combinatorial partitioning problem. Here, we will focus on partitioning a sparse matrix for parallel computation as the core problem. We survey various partitioning methods for the parallel computation of a sparse matrix–vector multiplication and study two methods in particular, called fine-grain and Mondriaan, by highlighting their similarities and differences, and combining them in a hybrid method.

Sparse matrices can be partitioned by one-dimensional or two-dimensional methods. In a 1D approach, complete rows (or columns) are assigned to parts (processors). One-dimensional row partitioning has traditionally been used in the sparse matrix–vector multiplication kernel of iterative linear system solvers and eigenvalue solvers. More recently, two-dimensional partitioning of sparse matrices has been shown to be more effective in distributing the work and minimising communication.

Two types of 2D methods are particularly promising: (i) the *fine-grained* approach by Çatalyürek and Aykanat [1] which partitions the nonzero matrix elements looking at individual elements, thereby allowing for the most general solution; (ii) the *Mondriaan approach* by Vastenhouw and Bisseling [2] which recursively bipartitions the sparse matrix into two (not necessarily contiguous) submatrices, trying splits in both the row and column directions and each time choosing the best. Both methods use a multilevel hypergraph partitioner as their splitting engine. This greedily minimises the exact metric of communication volume.

A natural question to ask is whether combining the two methods in a hybrid method would lead to savings in communication volume. We present

Dagstuhl Seminar Proceedings 09061
Combinatorial Scientific Computing
http://drops.dagstuhl.de/opus/volltexte/2009/2081

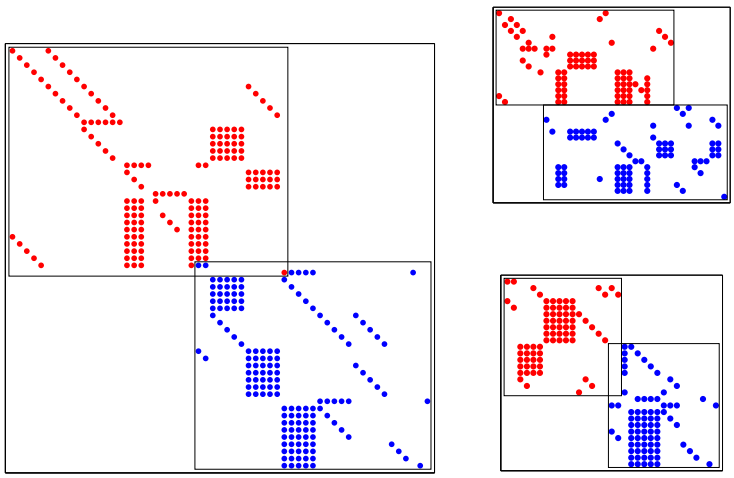

Figure 1: Split of the 59×59 matrix `impcol_b` with 312 nonzeros into four parts. The first split (left) is by the fine-grain method; the second split (top right) is by rows; and the third split (bottom right) is again by the fine-grain method.

a hybrid method which tries to split the current subset in three possible ways: by rows, by columns, and by individual nonzeros. The advantage is that this automatically detects the best method for the type of matrix involved, and adjusts this decision to the current submatrix as the partitioning procedure progresses, at the cost of an additional splitting attempt for each bipartitioning. Fig. 1 illustrates the hybrid method.

Using the fine-grained method to split a submatrix leads to a much larger hypergraph $\mathcal{H}_{\mathrm{f}}$, with nz vertices instead of n, for an $n \times n$ matrix with nz nonzeros. The total amount of information contained in the hypergraph is the same, however, as for a hypergraph $\mathcal{H}_{\mathrm{r}}$ induced by taking rows as vertices, or for a hypergraph $\mathcal{H}_{\mathrm{c}}$ induced by taking columns as vertices. Each vertex of $\mathcal{H}_{\mathrm{f}}$ is contained in only two hyperedges. This special property can be used to speed up the coarsening part of the multilevel bipartitioning of $\mathcal{H}_{\mathrm{f}}$.

We studied the different partitioning methods by numerical experiments on the test set of sparse matrices displayed in Table 1. The results of partitioning by the Mondriaan, fine-grain, and hybrid method are given in Table 2. The results were obtained on a 2.4 GHz AMD Opteron 250 processor with 8 Gbyte RAM. The table shows the quality of the resulting partitioning expressed as the communication volume, i.e., the total number of communicated data words. The allowed load imbalance is 3%. The table also shows the time (in seconds) taken by the partitioning procedure. The statistics given are the average over 100 runs. The software used is version 2.0 of

Name	Rows	Columns	Nonzeros	Application area
dfl001	6071	12230	35632	Linear programming
cre_b	9648	77137	260785	Linear programming
tbdmatlab	19859	5979	430171	Information retrieval
nug30	52260	379350	1567800	Linear programming
c98a	56243	56274	2075889	Cryptology
tbdlinux	112757	20167	2157675	Information retrieval
stanford	281903	281903	2312497	Web searching
polyDFT	46176	46176	3690048	Polymer simulation
cage13	445315	445315	7479343	DNA electrophoresis
stanford-berkeley	683446	683446	7583376	Web searching

Table 1: Properties of the test matrices.

Mondriaan[1] for the sparse matrix partitioner, but with Mondriaan's internal hypergraph bipartitioner replaced by the hypergraph partitioner from version 3.0 of PaToH[2] and used for splitting hypergraphs into two parts. This combination of software was found to perform best.

Table 2 shows that both the Mondriaan and the fine-grain methods have their favourite matrices with regard to communication volume. The fine-grain methods performs better than Mondriaan on 2 matrices (dfl001, cage13); Mondriaan performs better on 6 matrices; the results are inconclusive for 2 matrices (tbdlinux, stanford). In 26 out of 30 problem instances (matrix/p) the hybrid method is better than the best of the two pure methods. In the remaining 4 instances (tbdmatlab/64, tbdlinux/64, polyDFT/16, cage13/16), the hybrid method performs close to the best method. On average, the communication volume of the hybrid method is 3% less than that of the best of fine-grain and Mondriaan. The fine-grain method takes in general 2–4 times longer than the Mondriaan method to compute a partitioning. The time taken by the hybrid method is about equal to the sum of the partitioning times of the fine-grain and Mondriaan methods.

The pure fine-grained method has the advantage of allowing the most general solution and this gives the best result in some cases. Apparently, this advantage is not always realised in practice; the positive effect of forcing the nonzeros of a row (or column) to stay together, which is built into the Mondriaan approach, is in many cases stronger.

We conclude that both 2D approaches, the Mondriaan and the fine-grain approach have their advantages. It is difficult to predict which method is better for a given sparse matrix. The hybrid method automatically finds the best choice at each step of the partitioning and in most cases improves upon both pure methods.

[1] http://www.math.uu.nl/people/bisseling/Mondriaan
[2] http://bmi.osu.edu/~umit/software.html

Name	p	Mondriaan		Fine-Grain		Hybrid	
		vol	time	vol	time	vol	time
dfl001	4	1387	0.2	1379	0.2	1358	0.4
	16	3579	0.4	3563	0.4	3502	0.8
	64	6017	0.7	5973	0.6	5870	1.2
cre-b	4	1183	2.4	1228	4.3	1142	6.3
	16	3490	3.7	3647	6.3	3391	9.5
	64	7812	5.0	8145	7.6	7562	12.2
tbdmatlab	4	10708	4.2	11013	19.5	10392	19.0
	16	26629	7.1	33273	26.9	26319	30.1
	64	49124	9.2	68456	29.4	50653	33.2
nug30	4	55935	20.0	73594	26.1	52580	40.2
	16	115771	36.2	159579	40.0	105325	146.8
	64	194667	48.3	283847	50.9	183660	88.4
c98a	4	100128	36.8	125370	108.0	97188	137.0
	16	227298	65.9	330724	182.6	225418	220.5
	64	417670	84.5	588012	226.5	407192	272.7
tbdlinux	4	33759	40.6	26123	162.9	25734	191.9
	16	77230	66.1	87537	269.4	74637	290.5
	64	145759	82.8	200252	308.2	151205	339.3
stanford	4	886	19.8	935	36.1	845	75.2
	16	3226	33.8	3398	63.0	3039	130.5
	64	9668	45.5	9296	85.4	8307	167.1
polyDFT	4	8772	20.8	8841	99.9	8582	122.8
	16	34099	38.7	36480	143.5	34867	219.1
	64	73337	53.8	82544	188.8	73292	285.5
cage13	4	117124	52.3	89540	110.4	89337	140.5
	16	250480	93.1	189084	171.3	189110	246.2
	64	436944	125.9	333876	228.1	333562	330.1
stanford-berkeley	4	1128	24.7	1482	116.7	1010	134.6
	16	4598	51.9	5161	249.6	4190	259.9
	64	14734	81.1	14739	388.2	13003	383.5

Table 2: Communication volume (in data words) and partitioning time (in seconds) of three different partitioning methods, using the Mondriaan sparse matrix partitioning package with PaToH as hypergraph bipartitioner. The results are given for partitioning into p parts of the 10 test matrices from Table 1.

References

[1] Ümit V. Çatalyürek and Cevdet Aykanat. A fine-grain hypergraph model for 2D decomposition of sparse matrices. In *Proceedings Eighth International Workshop on Solving Irregularly Structured Problems in Parallel (Irregular 2001)*, page 118. IEEE Press, Los Alamitos, CA, 2001.

[2] Brendan Vastenhouw and Rob H. Bisseling. A two-dimensional data distribution method for parallel sparse matrix-vector multiplication. *SIAM Review*, 47(1):67–95, 2005.