An Effective Model to Decompose Linear Programs for Parallel Solution*

Ali Pınar and Cevdet Aykanat
Computer Engineering Department
Bilkent University, Ankara, Turkey

Abstract. Although inherent parallelism in the solution of block angular Linear Programming (LP) problems has been exploited in many research works, the literature that addresses decomposing constraint matrices into block angular form for parallel solution is very rare and recent. We have previously proposed hypergraph models, which reduced the problem to the hypergraph partitioning problem. However, the quality of the results reported were limited due to the hypergraph partitioning tools we have used. Very recently, multilevel graph partitioning heuristics have been proposed leading to very successful graph partitioning tools; Chaco and Metis. In this paper, we propose an effective graph model to decompose matrices into block angular form, which reduces the problem to the well-known graph partitioning by vertex separator problem. We have experimented the validity of our proposed model with various LP problems selected from `NETLIB` and other sources. The results are very attractive both in terms of solution quality and running times.

1 Introduction

Coarse grain parallelism inherent in the solution of block angular *Linear Programming* (LP) problems has been exploited in recent research works [5, 10]. However, these approaches suffer from *inscalability* and *load imbalance*, since they exploit only the existing block angular structure of the constraint matrix. This work focuses on the problem of decomposing irregularly sparse constraint matrices of large LP problems to obtain block angular structure for scalable parallelization. The objective in the decomposition is to minimize the size of the master problem—the sequential component of the overall parallel scheme—while maintaining computational balance among subproblem solutions.

The literature that addresses this problem is extremely rare and very recent. Ferris and Horn [3] model the constraint matrix as a bipartite graph, and use graph partitioning heuristics for decomposition. However, this model is not suitable for the existing graph partitioning heuristics and tools. In our previous work [12], we have proposed two hypergraph models which reduce the decomposition problem to the well-known hypergraph partitioning problem.

Very recently, multilevel graph partitioning heuristics have been proposed leading to very successful graph partitioning tools; Chaco [6] and Metis [7]. This

* This work is partially supported by the Commission of the European Communities, Directorate General for Industry under contract ITDC 204–82166 and The Scientific and Technical Research Council of Turkey under grant no EEEAG 160.

$$A_B^p = \begin{pmatrix} B_1 & & & \\ & B_2 & & \\ & & \ddots & \\ & & & B_k \\ R_1 & R_2 & \dots & R_k \end{pmatrix} \quad A_B^d = \begin{pmatrix} B_1 & & & & C_1 \\ & B_2 & & & C_2 \\ & & \ddots & & \vdots \\ & & & B_k & C_k \end{pmatrix} \quad A_{DB} = \begin{pmatrix} B_1 & & & & C_1 \\ & B_2 & & & C_2 \\ & & \ddots & & \vdots \\ & & & B_k & C_k \\ R_1 & R_2 & \dots & R_k & D \end{pmatrix}$$

Fig. 1. Primal (A_B^p), dual (A_B^d) and doubly-bordered (A_{DB}) block angular forms of an LP constraint matrix A

work proposes a new graph model—Row-Interaction Graph (RIG)—for decomposing the constraint matrices. In RIG, each row is represented by a vertex, and there is an edge between two vertices if there exists at least one column which has nonzeros in both respective rows. This model reduces the decomposition problem into the graph partitioning by vertex separator problem. Vertices in part P_i of a partition correspond to the rows in block B_i, and vertices in the separator correspond to the coupling rows. Hence, minimizing the number of vertices in the separator corresponds to minimizing the size of the master problem.

We have experimented the validity of the proposed graph model with various LP constraint matrices selected from `NETLIB` and other sources. We have used *Metis* tool for multi-way partitioning of sample RIGs by edge separators. Then, we have used various proposed heuristics for refining the edge-based partitions found by *Metis* to partitions by vertex separators. Our results are much better than those of previous methods. We were able to decompose a matrix with 10099 rows, 11098 columns, 39554 nonzeros into 8 blocks with only 517 coupling rows in 1.9 seconds and a matrix with 34774 rows, 31728 columns, 165129 nonzeros into 8 blocks with only 1029 coupling rows in 10.1 seconds. The solution times with *LOQO*[14] are 907.6 seconds and 5383.3 seconds, respectively.

2 Previous Work

2.1 Bipartite Graph Model

Ferris and Horn [3] model the sparsity structure of the constraint matrix as a bipartite graph. In this model (BG), each row and each column is represented by a vertex, and the sets of vertices representing rows and columns form the bipartition. There exists an edge between a row vertex and a column vertex if and only if the respective entry in the constraint matrix is nonzero. This graph is partitioned using Kernighan-Lin [8] heuristic. Then, vertices are removed until no edges remain among different parts. This enables permutation of the matrix into a doubly-bordered form (Fig. 1). Out of the vertices removed, the ones representing columns constitute the row-coupling columns, and the ones representing the rows constitute the column-coupling rows. This doubly-bordered matrix A_{DB} is transformed into a block angular matrix A_B^p by *column splitting*[3].

2.2 Hypergraph Models

In our previous study [12], we have proposed two hypergraph models for the decomposition. A hypergraph $\mathcal{H} = (\mathcal{V}, \mathcal{N})$ is defined as a set of vertices and a set of nets (hypergedges) among these vertices. Each net is a subset of vertices of the hypergraph. In a partition, a net is cut (external), if it has vertices in more than one parts, and uncut (internal), otherwise.

In the first model, namely the *row–net* (RN) model, each row is represented by a net, whereas each column is represented by a vertex. The set of vertices connected to a net corresponds to the set of columns which have a nonzero entry in the row represented by this net [12]. In this model, the decomposition problem reduces to the well-known *hypergraph partitioning* problem. Hypergraph partitioning tries to minimize the number of cut nets, while maintaining balance between the parts. Maintaining balance corresponds to balancing among block sizes in the block angular matrix A_B^p (Fig. 1), and minimizing the number of cut nets corresponds to minimizing the number of coupling rows in A_B^p.

The second model, namely the *column–net* (CN) model, is the dual of the RN model, so partitioning this hypergraph gives dual block angular matrix A_B^d.

3 Graph Partitioning by Vertex Separator

We say that $\Pi^k = (P_1, P_2, \ldots, P_k; S)$ is a k-way vertex separation of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ if the following conditions hold: each part P_i, for $1 \leq i \leq k$, is a nonempty subset of $\mathcal{V}$; all parts and the vertex separator $S \subset \mathcal{V}$ are mutually disjoint; union of k parts and the separator is equal to $\mathcal{V}$; and there does not exist an edge between two parts P_i and P_j for any $i \neq j$. We also restrict our separator definition as follows: each vertex in the separator S is adjacent to vertices of at least two different parts. Balance criterion for part sizes is defined as: $(W_{max} - W_{avg})/W_{max} \leq \epsilon$ where W_{max} is the size of the part with maximum size, W_{avg} is the average part size, and ϵ is a predetermined imbalance ratio.

Using these definitions, the problem of partitioning by vertex separators can be stated as: "*finding a balanced vertex partition with desired number of parts which minimizes the cardinality of the set S*".

4 Row Interaction Graph

In this section, we present a new graph model, namely the *row interaction graph* (RIG), for the decomposition. In RIG, each row is represented by a vertex, and there exists an edge between two vertices if and only if there exists at least one column which has nonzeros in both respective rows. So formally:

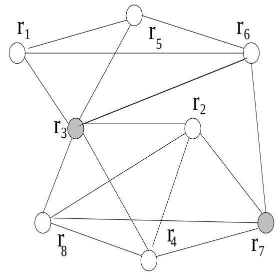
Definition 1 *A graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a RIG representation of a sparse matrix $A = (a_{ij})$ iff the following conditions are satisfied.*

- $\mathcal{V} = \{r_1, r_2, \ldots, r_i, \ldots, r_M\}$, *where r_i represent the ith row of matrix A.*
- $e = (r_i, r_j) \in \mathcal{E} \iff \exists k \;\; 1 \leq k \leq N \ni a_{ik} \neq 0$ *and* $a_{jk} \neq 0$

A sample sparse matrix A, and the associated RIG are presented in Fig. 2. In this graph, edge (r_1, r_5) is because of rows 1 and 5 having a nonzero in column 3. However, there does not exist an edge between r_1 and r_2, because there does not exist a column in which both row 1 and row 2 have a nonzero.

A k-way vertex separation $\Pi^k = (P_1, P_2, \ldots, P_k; S)$ of RIG induces a row and column permutation for matrix A transforming it into a block angular form

$$
A = \begin{array}{c} \begin{array}{ccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 \end{array} \\ \left(\begin{array}{ccccccc}
x & & x & & & & \\
 & x & & & & & x \\
x & x & x & x & & & \\
 & x & & & x & & x \\
 & & x & x & & & \\
 & & x & x & & x & \\
 & & & & x & x & x \\
 & x & & & x & &
\end{array}\right) \end{array}
\begin{array}{c} \\ 1 \\ 2 \\ 3 \\ 4 \\ 5 \\ 6 \\ 7 \\ 8 \end{array}
$$



$$
A_B^p = \begin{array}{c} \begin{array}{ccccccc} 1 & 3 & 4 & 6 & 2 & 5 & 7 \end{array} \\ \left(\begin{array}{ccccccc}
x & x & & & & & \\
 & x & x & & & & \\
 & x & x & x & & & \\
 & & & & x & & x \\
 & & & & x & x & x \\
 & & & & x & x & \\
x & x & x & & x & & \\
 & & & x & & x & x
\end{array}\right) \end{array}
\begin{array}{c} \\ 1 \\ 5 \\ 6 \\ 2 \\ 4 \\ 8 \\ 3 \\ 7 \end{array}
$$

Fig. 2. A sample 8×7 matrix, its associated RIG and its block angular form A_B^p induced by the vertex separation $\Pi^2 = (\{r_1, r_5, r_6\}, \{r_2, r_4, r_8\}; \{r_3, r_7\})$ on the RIG.

A_B^p with k blocks. In a separation Π^k of RIG, vertices in the separator S correspond to the coupling rows of A, and vertices in part P_i correspond to the rows in block B_i. The permutation of the columns is controlled by the rows. Each column is placed in the same block as the rows it shares non-zero(s). By definition of the vertex separator, there are no edges between vertices in different parts, hence there is no column interaction between rows in different blocks, i.e., there are no columns which have nonzeros in two rows at different parts.

Given $\Pi^2 = (\{r_1, r_5, r_6\}, \{r_2, r_4, r_8\}; \{r_3, r_7\})$ as a separation of the RIG in Fig. 2, the associated permuted matrix A_B^p can be obtained as follows. Vertices r_1, r_5, r_6 (r_2, r_4, r_8) are in part P_1 (P_2), so respective rows will be placed in block B_1 (B_2). Rows 3 and 7 will form the coupling block because of the vertices r_3 and r_7 on the separator. Columns 1,3,4,6 (2,5,7) are placed in block B_1 (B_2) since they share nonzeros with rows placed in this block.

RIG model reduces the decomposition problem to a well-known problem, graph partitioning by vertex separators. The problem of graph partitioning by vertex separators has two objectives: (i) minimizing the number of vertices in the separator, (ii) maintaining balance between number of vertices in parts other than the separator. The first objective directly corresponds to minimizing the number of coupling rows, since each vertex in the separator of RIG corresponds to a row in the coupling block of A_B^p. The second objective corresponds to maintaining balance among the block sizes in the block angular matrix A_B^p.

5 Finding Vertex Separators

We have adopted commonly used scheme of finding vertex separators from edge separators. Edges in the edge separator are called the *cut edges*. An edge is cut if it is between two different parts. Each edge can be associated with a weight, and *cutsize* of a partition is the sum of weights of cut edges. In the light of these definitions, problem of graph partitioning by edge separator can be stated as: *finding a balanced partition of vertices of the graph which minimizes the cutsize.*

The set of vertices adjacent to the cut edges is called the *wide separator* [9]. We will call the subgraph induced by the wide separator and the cut edges as the *wide-separator* subgraph $\mathcal{G}_{WS}$. A subset of the vertices in the wide separator can be chosen to form a narrow separator, a feasible separator of smaller cardinality.

5.1 Finding Wide Separators

There are no certain metrics for the "goodness" of a wide separator that will lead to a smaller narrow separator. However, two metrics have gained popularity due to their simplicity and availabilty of appropriate software tools. The first one is minimizing the number of cut edges. Minimizing the number of cut edges can give us a good estimate of a vertex separator, since it finds logical clusters on the graph. The second one is minimizing the number of vertices in $\mathcal{G}_{WS}$. Leiserson and Lewis [9] model the graph with a hypergraph, where there exists a vertex for each vertex in the graph, and there exists a net n_i for each vertex v_i which contains v_i and all vertices adjacent to v_i. With this hypergraph, if a net n_i is on the cut, then the vertex v_i should be on the wide separator. Hence, minimizing the number of cut nets on this hypergraph corresponds to minimizing the number of vertices in $\mathcal{G}_{WS}$.

Although, both metrics are valuable assets for the goodness of a wide separator, they do not guarantee a narrow separator of smaller cardinality.

5.2 Edge Weightening for Better Wide Separators

We propose a heuristic model for finding a better wide separator. Our basic observation is that all edges are not of equal importance for the goodness of a wide separator. Edges incident to a vertex with high degree are less important, since this vertex has a higher probability to be moved to the separator. Here, degree $deg(u)$ of a vertex $u \in \mathcal{V}$ refers to the number of edges incident to u in RIG. So, we can assign weights to the edges inversely proportional to the degrees of its end-vertices. We propose the following weight function:

$$weight((u,v)) = \frac{1}{max(deg(u), deg(v))}$$

Minimizing the cutsize of this edge-weighted RIG is expected to yield good wide separators for refining to narrow separators.

5.3 From Wide Separators to Narrow Separators

This part of the problem is equivalent to finding a minimum vertex cover on $\mathcal{G}_{WS}$. This problem can be solved optimally in polynomial time for two way partitions, by finding maximum matchings on bipartite graphs [13]. However, we need to resort to heuristics for the solution of this problem for multi-way partitions.

We have experimented the greedy heuristics, *maximum-inclusion* (MI) and, *minimum-removal* (MR) proposed in[9]. In this work, we also propose a new heuristic, namely, *one-max-inclusion* (OMI) heuristic which is presented in Figure 3. Our heuristic is similar to MI with the following enhancement: OMI starts with including the vertices adjacent to a vertex of degree 1 to the vertex cover (narrow separator), since this does not destroy our chance to find an optimal solution. When there are no vertices of degree 1, we take a greedy decision similar to that of MI and include the vertex with the highest degree to the separator. Then, we again seek for vertices with degree 1, and repeat this process until all edges are adjacent to a vertex in the separator.

INPUT: Wide Separator subgraph: $\mathcal{G}_{WS} = (\mathcal{V}_{WS}, \mathcal{E}_{WS})$
OUTPUT: Narrow Separator: $S \subset \mathcal{V}_{WS}$

```
repeat
    for each v ∈ V  deg(v) = 1 do
        u ← only neighbor of v
        S ← S ∪ {u};
        for each x ∈ Adj(u) do E ← E − {(u, x)} endfor;
    endfor
    if (E ≠ ∅) then
        v ← vertex with maximum degree ;
        for each u ∈ Adj(v) do S ← S ∪ {u} endfor;
        for each x ∈ Adj(u) do E ← E − {(u, x)} endfor;
until E = ∅
```

Fig. 3. A greedy heuristic for finding a narrow separator from the wide-separator subgraph of a partition of RIG by edge separator

In our experiments, OMI heuristic, overperformed the other two, MI and MR [11]. We have compared the performance of OMI heuristic with optimal solutions obtained by matchings, for bisections. We have seen that, average difference for 27 different data set after 20 runs was only 0.11%, and the peak difference was only 0.58% for one data set.

6 Experimental Results

We have experimented the validity of the model on various LP matrices selected from the `Netlib` suite [4], Kennington problems [1], and collection of Gondzio[2]. The properties of these problems are presented in Table 1. In this table, M, N, Nz, and D, columns represent the number of rows, columns, nonzeros, and density of the respective constraint matrices, respectively. Here, D is computed as $Nz/(M\times N)$. In Table 1, Nz/N and Nz/M columns denote average number of nonzeros per row and column, respectively, and $|\mathcal{E}/\mathcal{V}|$ column denotes average vertex degree of the associated RIGs. All experiments have been performed on a *SUN Sparc 5* workstation. We have used Metis [7] for graph partitioning, an FM-variant [2] for hypergraph partitioning, and OMI heuristic implemented in C for finding narrow separators from wide separators. For each experiment, partitioning heuristic has been run 20 times with random seeds. Following tables and figures display the averages of these runs.

Figure 4 shows the relative performance of the edge-weighted graph (W-RIG) model and hypergraph(H-RIG) model compared to unweighted graph (U-RIG) model in finding narrow separators for 8-way partitioning. W-RIG and U-RIG models correspond to running Metis on weighted and unweighted RIG, respectively, and then refining the resulting wide-separators to narrow separators with OMI. H-RIG corresponds to running the FM-variant [2] on the hypergraph

[2] These problems are available by anonymous ftp from IOWA Optimization Center ftp col.biz.uiowa.edu:pub/testprob/lp/gondzio

Table 1. Properties of the Constraint Matrices and their associated RIGs

Problem	Constraint Matrix Properties						RIG Properties	
Name	M	N	Nz	$D\%$	Nz/N	Nz/M	$\lvert\mathcal{E}\rvert$	$\lvert\mathcal{E}\rvert/\lvert\mathcal{V}\rvert$
80bau3b	2262	9799	21002	0.09	2.14	9.28	10074	8.91
bnl2	2324	3489	13999	0.17	4.01	6.02	13457	11.58
cycle	1903	2857	20720	0.38	7.25	10.89	27714	29.13
czprob	929	3523	10669	0.33	3.03	11.48	7072	15.22
d2q06c	2171	5167	32417	0.29	6.27	14.93	26991	24.87
ganges	1309	1681	6912	0.31	4.11	5.28	7656	11.70
greenbea	2392	5405	30877	0.24	5.71	12.91	33841	28.30
sctap3	1480	2480	8874	0.24	3.58	6.00	7386	9.98
ship12l	1151	5427	16170	0.26	2.98	14.05	10673	18.55
stocfor2	2157	2031	8343	0.19	4.11	3.87	12738	11.81
woodw	1098	8405	37474	0.41	4.46	34.13	20421	37.20
cre-a	3516	4067	14987	0.10	3.69	4.26	51015	10.10
cre-c	3068	3678	13244	0.12	3.60	4.32	49025	13.93
cre-d	8926	69980	242646	0.04	3.47	27.18	285068	16.40
osa-07	1118	23949	143694	0.54	6.00	128.53	273779	15.87
CO9	10789	14851	101578	0.06	6.84	9.41	20748	11.80
CQ9	9278	13778	88897	0.07	6.45	9.58	18905	12.32
GE	10099	11098	39554	0.04	3.56	3.92	181670	40.71
NL	7039	9718	41428	0.06	4.26	5.89	52466	93.86
mod2	34774	31728	165129	0.01	5.20	4.75	119208	22.10
world	34506	32734	164470	0.01	5.02	4.77	106156	22.88

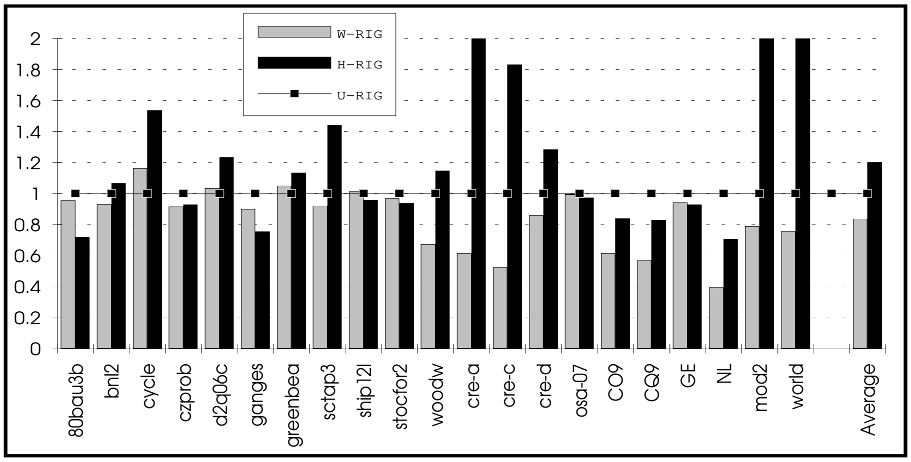


Fig. 4. Narrow separator quality of edge-weighted graph (W-RIG) model and hypergraph (H-RIG) model compared to unweighted graph (U-RIG) model in finding narrow separators of test RIGs for 8-way partitioning. Bars under the baseline indicate that the respective model performs better than the U-RIG model.

representations of the RIGs (as discussed in Section 5.1), and refining the results with OMI. W-RIG model produces 20% better results on the average than U-RIG model. The difference becomes more significant for larger problems. Although H-RIG model is worse than U-RIG model on the average, it produces better results for many of the problems.

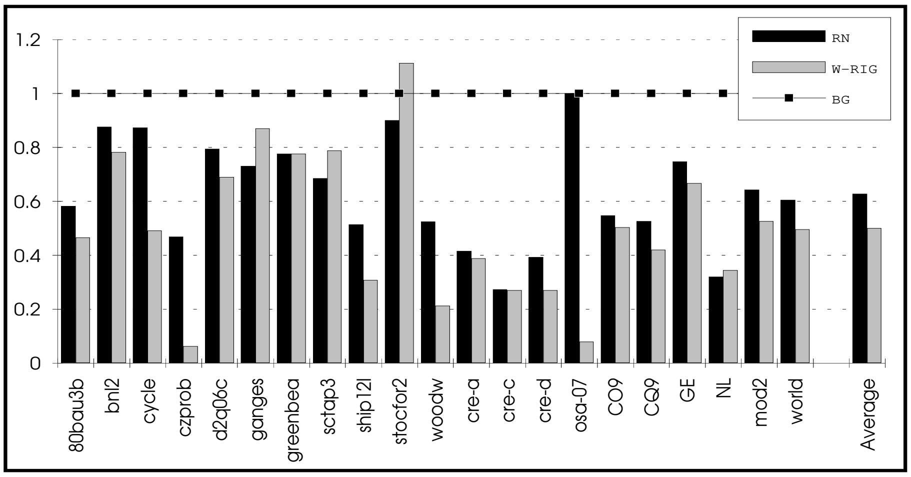


Fig. 5. Quality of edge-weighted RIG model (W-RIG) and RN hypergraph model compared to the BG bipartite graph model for 8-way block angular decomposition of test matrices. Bars under the baseline indicate that the respective model performs better than BG model.

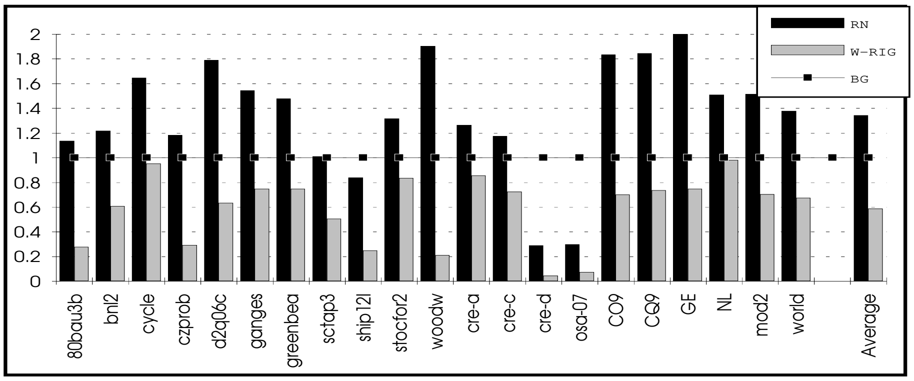


Fig. 6. Execution times of (W-RIG) model and RN model compared to (BG) model for 8-way block angular decomposition of test matrices.

Figures 5 and 6 illustrate quality and execution times of W-RIG model and row-net (RN) hypergraph model compared to the bipartite graph (BG) model for 8-way block angular decomposition of test matrices. W-RIG model overperforms BG model in all problems except for `stocfor2`. W-RIG results are twice better than BG on the average. The difference becomes drastic for `osa-07`, `czprob`, and `woodw`. The common point in these matrices is the large number of columns with respect to rows. Relative performance of BG deteriorates for matrices with $N \gg M$, since BG treats both rows and columns as decision variables. The difference between run times of BG and W-RIG becomes very significant for `80bau3b`,`czprob`, `woodw`, `cre-d`, `osa-07`, all of which has $N \gg M$. Recall that, we have used the same partitioning tool for both BG and RIG models, hence

Table 2. The effectivity of RIG Model

Problem			k	# Coup. Rows		t_{part}	
		LOQO		abs.	rel.	abs.	rel.
Name	Rows	$t_{sol}(secs)$			%	secs.	%
cycle	1903	110.8	4	64	3.36	0.87	0.79
			8	100	5.25	1.05	0.95
d2q06c	2171	400.0	4	223	10.27	0.96	0.24
			8	293	13.50	1.17	0.29
ganges	1309	21.9	4	68	5.19	0.32	1.46
			8	128	9.78	0.41	1.87
greenbea	2392	166.3	4	125	5.23	1.34	0.81
			8	231	9.66	1.63	0.98
ship12l	1151	20.5	4	49	4.26	0.43	2.10
			8	78	6.78	0.54	2.63
stocfor2	2157	24.8	4	44	2.04	0.53	2.14
			8	120	5.56	0.66	2.66
woodw	1098	80.7	4	68	6.19	0.74	0.92
			8	160	14.57	0.86	1.07
cre-a	3516	40.8	4	112	3.19	1.03	2.52
			8	141	4.01	1.27	3.11
cre-c	3068	40.7	4	102	3.32	0.89	2.19
			8	127	4.14	1.08	2.65
cre-d	8926	6719.9	4	913	10.23	6.12	0.09
			8	1117	12.51	6.73	0.10
osa-07	1118	398.7	4	80	7.16	3.39	0.85
			8	80	7.16	4.05	1.02
CO9	10789	1827.6	4	1099	10.19	4.30	0.24
			8	1363	12.63	4.72	0.26
CQ9	9278	1664.4	4	751	8.09	4.00	0.24
			8	1061	11.44	4.36	0.26
GE	10099	907.6	4	331	3.28	1.71	0.19
			8	517	5.12	1.93	0.21
NL	7039	699.2	4	547	7.77	2.82	0.40
			8	633	8.99	3.22	0.46
mod2	34774	5383.3	4	559	1.61	9.44	0.18
			8	1029	2.96	10.07	0.19
world	34506	25819.7	4	615	1.78	9.24	0.04
			8	1074	3.11	10.02	0.04
Average			4		5.48		0.90
			8		8.06		1.1

the difference is directly due to the effectiveness of the models.

The performances of W-RIG and RN model are quite competitive. W-RIG is better on the average. However, the difference is not too large, and may be due to the partitioning tool used. But a careful observation reveals that the performance of RN model becomes poor for problems with $N \gg M$. This is simply because RN works on too many vertices.

Table 2 shows the overall effectiveness of the proposed model. The number of coupling rows and the percent ratio of the number of coupling rows to the total number of rows, the actual partitioning times and percent ratio of partitioning times to solution times of the problems with *LOQO* [14] are presented. On the overall average, only 5.48% and 8.06% of the rows are on the coupling block for 4 and 8 block decompositions, respectively. The partitioning times are negligible compared to *LOQO* solution times (0.9% for 4 blocks, and 1.1% for 8 blocks). Another remarkable point in this table is that partitioning times grow slowly with the problem size, although solution times rapidly increase. This makes decomposition very practical for large problems.

7 Conclusion

We have proposed an effective graph model to decompose LP matrices to block angular form for scalable parallelization. The new model reduced the problem to the well-known graph partitioning by vertex separator problem. The validity of the model has been experimented with various LP matrices, and its performance has been compared with bipartite graph [3] and hypergraph models [12]. The proposed model overperformed the previous two models on the existing graph/hypergraph partitioning tools. The new model is very effective and enables us to decompose a matrix with 10099 rows, 11098 columns, 39554 nonzeros into 8 blocks with only 517 coupling rows in 1.9 seconds and a matrix with 34774 rows, 31728 columns, 165129 nonzeros into 8 blocks with only 1029 coupling rows in 10.1 seconds. The solution times with $LOQO$ are 907.6 seconds for the former and 5383.3 seconds for the latter.

References

1. W. J. Carolan, J. E. Hill, J. L. Kennington, S. Niemi, S. J. Wichmann An Empirical Evaluation of the KORBX Algorithms for Military Airlift Applications *Operations Research* 38(2):240-248, 1990.
2. U. V. Çatalyurek and C. Aykanat, Decomposing Irregularly Sparse Matrices for Parallel Matrix-Vector Multiplication, *Proc. of Irregular 96*,1996, (to appear).
3. M. C. Ferris, and J. D. Horn. Partitioning mathematical programs for parallel solution. Technical report TR1232, Computer Sciences Department, University of Wisconsin Madison, May 1994.
4. D. M. Gay, "Electronic mail distribution of linear programming test problems" *Mathematical Programming Society COAL Newsletter*, 1985.
5. S. K. Gnanendran and J. K. Ho. Load balancing in the parallel optimization of block-angular linear programs. *Mathematical Programming*, 62:41–67, 1993.
6. B. Hendrickson and R. Leland, A Multilevel Algorithm for Partitioning Graphs, Sandia National Laboratories,SAND93-1301, 1993.
7. G. Karypis and V. Kumar, A Fast and High Quality Multilevel Scheme for Partitioning Irregular Graphs, Dept. of Computer Science, Univ. of Minnesota, 1995, TR 95-035.
8. B.W. Kernighan and S. Lin. An efficient heuristic procedure for partitioning graphs. Technical Report 2, The Bell System Technical Journal, Feb. 1970.
9. C. E. Leiserson and J. G. Lewis, Orderings for parallel sparse symmetric factorization, *3rd SIAM Conf. Parallel Processing for Scientific Comp.*, 27–31, 1987
10. D. Medhi. Bundle-based decomposition for large-scale convex optimization: error estimate and application to block-angular linear programs. *Mathematical Programming*, 66:79–101, 1994.
11. A. Pınar, Decomposing Linear Programs for Parallel Solution M.S. Thesis, Bilkent University, July, 1996.
12. A. Pınar, Ü. V. Çatalyurek, C. Aykanat and M. Pınar, Decomposing Linear Programs for Parallel Solution *Lecture Notes in Computer Science*, 1041:473–482, 1996.
13. A. Pothen and C. J. Fan, Computing the Block Triangular Form of a Sparse Matrix, *ACM. Trans. on Math. Software*,16(4):303–324,1990.
14. R. J. Vanderbei, *LOQO* User's Manual. Princeton University, November 1992.