Lecture Notes
in Economics and
Mathematical Systems

[illegible]
[illegible]

206

Mark H. Karwan, Vahid Lotfi,
Jan Telgen, Stanley Zionts

[illegible]
[illegible]
[illegible]
[illegible]
[illegible]

Redundancy in Mathematical Programming

A State-of-the-Art Survey

Springer-Verlag
Berlin Heidelberg New York Tokyo

Chapter 12

STRUCTURAL REDUNDANCY IN LARGE-SCALE OPTIMIZATION MODELS

Gordon H. Bradley, Gerald G. Brown
and
Glenn W. Graves

This paper discusses automatic detection and exploitation of structural redundancy in large-scale mathematical programming models. From our perspective, such redundancy represents embedded special structure which can give significant insight to the model proponent as well as greatly reduce solution effort. We report experiments with real-life linear programming (LP) and mixed-integer (MIP) models in which various methods are developed and tested as integral modules in an optimization system of advanced design. We seek to understand the modeling implications of these embedded redundancies as well as to exploit them during actual optimization. The latter goal places heavy emphasis on efficient, as well as effective, identification techniques for economic application to large models. Several (polynomially bounded) heuristic detection algorithms are presented from our work. In addition, bounds are reported for a maximum row dimension of the more complex structures. These bounds are useful for objectively estimating the quality of heuristically derived assessments of structural redundancy. Finally, some additional suggestions are made for analyzing nonlinear programming (NLP) models.

12.1 Introduction

Automatic detection and exploitation of structural redundancy in large-scale linear programming (LP) (as well as mixed integer programming (MIP) and nonlinear) models has been the subject of a continuing research program conducted at the Naval Postgraduate School and UCLA over the past decade. This exposition draws from various results in that effort, and refers (sparingly) to significant work by other researchers. The references contain complete descriptions of these results for the interested reader.

Our scope is intentionally limited to automated methods of sufficient efficiency to enable us to economically apply them to real-world optimization problems. Thus, we consider only those approaches showing greatest promise for immediate practical application. Although the interpretations of embedded model redundancy can lend profound insights in their own right, we are equally interested in detecting errors in data preparation and model generation--mathematically mundane issues of fundamental importance to the practioner.

In this context, our definition of structural redundancy includes not only features which permit reduction of effective problem size, but also those embedded special structures which invite application of special solution methods with enhanced efficiency. This somewhat expanded view of redundancy admits features which yield to, for instance, basis factorization or decomposition.

The sheer size of contemporary large-scale LP models presents significant computational difficulties, even for discovery of otherwise elementary structures (in the sense of formal complexity). Implementation of effective structural analysis procedures is primarily a matter of exercising large-scale data structures efficiently. As we shall see, though, these practical considerations can give significant theoretical guidance in the specification of efficiently achievable classes of model transformations.

That detection of embedded special structure can be of practical importance in actual model solution is undisputed. It is widely known that explicit simplex operations can be materially improved in efficiency by incorporation of basis factorization methods (e.g., McBride (1973) and Graves and McBride (1976)). The details of such modifications of the simplex procedure are not given here. However, the underlying themes of simplex factorization are the substitution of logic for floating point arithmetic, and separation of the apparent problem monolith into more manageable components.

This work deals primarily with row factorizations. The pervasive implied problem for row factorization is the identification of the best embedded structure from all those that may lie at hand in any particular model. Conventional wisdom differs as to the criterion for this discrimination among factorizations of the same class. However, it is generally accepted that the row dimensionality of the factorization serves as an excellent measure of effectiveness. In this sense, embedded special structures fall naturally into a taxonomy implied by the intrinsic complexity of the associated maximum row identification problems.

We proceed with a discussion of several types of embedded special structures detectable by efficient polynomially bounded algorithms. These structures are considered in increasing order of maximum row identification complexity. We emphasize that efficient polynomial algorithms are operationally defined here as low-order polynomial in terms of intrinsic problem dimensions (e.g., number of rows, columns, and non-zero elements), and not in terms of the total volume of model information (e.g., total number of bits in all coefficients, ad nauseam).

12.2 Overview of the Analysis

We are usually faced with the following practical situation. An LP (or MIP) is presented, typically in MPS format on magnetic tape. Some documentation and possibly solution experience is also provided, but the matrix generator and source data elements for the model are rarely at hand. The model is frequently sent to us because of some difficulty encountered in preliminary solution attempts.

Our efforts are devoted to two issues: analysis of the LP, and solving it efficiently. The analysis is initially focused on rather ordinary details, primarily estabishing computer capability, and on review of gross problem statistics such as those shown in Table 12.1 for the same problems we will discuss further. At this stage we attempt to detect outright disasters before investing more time and money.

For instance, we have frequently found that the actual LP bears little resemblance to the intended formulation due to matrix generator difficulties, human error, or data base failures. Even an LP which seems to follow its formulation template may exhibit superfluous objective functions, right-hand sides, and so forth.

After the cursory review, a representative model is achieved, set up and input to an internal data structure for detailed analysis. The input process is relatively expensive for large LP models, involving conversion of thousands of records. Row and column summary statistics are produced and reviewed. At this point, pathological coefficient scaling is frequently revealed; this is sometimes caused by poor modelling, and occasionally by bad source data elements. For example, we have found at this point that unwanted model features (such as, say, production capacity constraints) have be "relaxed" in an ad hoc fashion by providing outrageous dummy coefficients (e.g., essentially infinite capacity or infinitesimal production rates). These redundancies, while innocently intended, can play havoc with LP solution procedures (this is especially true for MIP and nonlinear algorithms).

Next, we apply a set of simple reductions to the LP model. At this stage, we identify redundancies with two goals. First, we want to complete our "bottom-up" analysis of the model. However, we also seek to set the stage for actual model solution.

In this sense, we cannot (ordinarily) afford and we do not (routinely) apply large scale reduction methods which are computationally equivalent to actual solution of the LP. Rather, we discover as much as possible about redundancy in the model by efficient, polynomially bounded static analysis not employing basis exchanges. From this we attempt to infer qualitatively the complete redundancy structure of the model at hand.

Also, we ignore structural features which have little or no bearing on our analysis or solution effort. For instance, we are indifferent about degeneracy, since our solution procedures exploit this delightful property (Graves (1965)).

Similarly, we do seek certain properties which may make our solution much more efficient, but which may not be so attractive to analysts using other methods. Among these properties are embedded Generalized Upper Bounds (GUB) and network, or pure network (NET) rows, for which our interest is particularly keen (Bradley, Brown and Graves (1977a), Brown and Graves (1975), Graves and VanRoy (1979)).

Actual solution of the model follows these analysis efforts unless we find that the model requires significant modification or managerial review. For models that merit solution, we see no reason to impose unreasonable restrictions on the model builder. That is, the responsibility for efficient solution is ours regardless of the redundancy structure found, as long as this redundancy is not in conflict with the intent of the modeler. Thus, all reductions must provide an equivalent solution to the model as originally posed, and cannot require that the model be changed or severely modified externally (e.g., requiring general linear transformation of the formulation to suit the solver is out of the question).

12.3 Details of the Analysis

For analysis, the linear program is stored in a sparse data structure. Nonzero coefficients are stored along with the corresponding row index by column with access via column entry points. Each row and column has associated with it an external label, several coefficient values representing upper and lower bounds and ranges, and a coded tag giving the type of constraint (e.g., equation) or column and its status in the analysis.

Although conversion to a super-sparse data structure is subsequently required for the solution of the model, the sparse structure is much more convenient for analysis of the model composition and for model modifications. Very little auxiliary storage is required for the analysis and practical problems at large scale are routinely analyzed. The system is designed to operate on problems with up to 30,000 rows plus columns (e.g., see Bradley, Brown and Galatas (1980) for examples using FORTRAN).

Our analysis is confined to reductions that do not change the feasible region. The analysis can also be called "orthogonal" in that the reduction tests are made on the current problem with no pivotal transformations actually performed. The reductions may show how to transform the problem by removing columns and constraints and by the elimination of columns (equivalent to pivoting), but the tests are applied only to the current representation of the problem.

The analysis is applied to a fully ranged and bounded linear program.

$$\min \quad \sum_j c_j x_j$$

$$\text{s.t. } r_i \leq \sum_j a_{ij} x_j \leq r^i \qquad \forall\, i \ \text{(ranged constraints)}$$

$$h_j \leq x_j \leq u_j \qquad \forall\, j \ \text{(simple upper bounds).}$$

Some ranges and bounds may be missing (that is, $+\infty$ or $-\infty$).

12.3.1 Simple Reductions

Singleton column. If a column has a single nonzero coefficient, the column can be removed. The ranges of the constraint that contain the nonzero coefficient are modified to construct an equivalent problem.

Fixed column. If a column has been fixed at a certain value, or (equivalently) its upper and lower bounds are equal, it can be removed. The ranges of all constraints that have a nonzero coefficient in the removed column are modified.

Vacuous columns. Columns with all coefficients zero may be removed. The associated variable may assume any value between its upper and lower bounds.

Inconsistent column. Any column with its upper bound strictly less than its lower bound indicates that the linear program has no feasible solution.

Free column. A free column results from modeling a variable that has no upper or lower bound or by analysis that can show that neither the upper nor lower bound is necessary to define the feasible region.

Singleton constraint. If a constraint has a single nonzero coefficient, the constraint may be removed. The bounds of the column that contains the coefficients are modified.

Redundant constraints. A constraint is redundant if its removal does not change the feasible region. An examination of the bounds on columns with nonzero coefficients yields a test for a redundant constraint. For constraint i define

$$R^i = \sum_{a_{ij}>0} a_{ij}u_j + \sum_{a_{ij}<0} a_{ij}h_j ,$$

and

$$R_i = \sum_{a_{ij}>0} a_{ij}h_j + \sum_{a_{ij}<0} a_{ij}u_j .$$

A constraint is redundant if $R^i \leq r^i$ and $R_i \geq r_i$. If only one inequality holds, the corresponding range can be eliminated (that is, set to ∞ or $-\infty$).

Constraints that fix variables. If $R^i = r_i$ or $R_i = r^i$ then each column with a nonzero coefficient in the constraint must be fixed at the appropriate bound in order for the constraint to be satisfied. The constraint can then be removed.

Inconsistent constraints, If $R^i < r_i$ or $R_i > r^i$, then the LP has no feasible solution.

Vacuous constraints. Constraints with no nonzero coefficients may be removed. If $r^i < 0$ or $r_i > 0$, then the LP has no feasible solution.

All of the sets above are applied in a single pass. Since the reductions (if any) may make it possible to identify new reductions, the complete analysis consists of repeated passes until no additional reductions are found. Table 12.1 displays the results for the sample problems. The times given are for execution on an IBM 360/67 using FORTRAN H (Extended) without code optimization.

With real-life LP (and MIP) models, a remarkably large fraction of model constraints can be removed by these simple techniques. For some cases, models have nearly been solved this way.

We have often been surprised at the number of reductions achieved after several passes. The repeated passes can serve to unravel a model and reveal special structure that is quite obscure in a static analysis.

Experiments with some of these reductions have been reported by Brearley, Mitra and Williams (1978). More extensive work at large scale has been done by Bradley, Brown and Graves (1977b) and by Krabek (1979).

Table 12.1

CHARACTERISTICS OF SAMPLE LP (MIP) MODELS

	Constraints		Variables		Nonzero Coefficients	
Model	Total	Equality	Total	Integer	Total	Orders of Magnitude
NETTING	90	36	177	114	375	8
AIRLP	171	170	3,040	0	6,023	3
COAL	171	0	3,753	0	7,506	6
TRUCK	220	0	4,752	4,752	30,074	1
CUPS	361	330	582	145	1,341	7
FERT	606	559	9,024	0	40,484	8
PIES	663	480	2,923	0	13,288	7
PAD	695	160	3,934	0	13,459	7
ELEC	785	462	2,800	0	8,462	6
GAS	799	638	5,536	0	27,474	5
PILOT	976	701	2,172	0	13,057	14
FOAM	1,000	0	4,020	42	13,083	7
LANG	1,236	665	1,425	0	22,028	3
JCAP	2,487	310	3,849	560	9,510	≥ 9
PAPER	3,529	2,456	6,543	0	32,644	6
ODSAS	4,648	4,059	4,683	0	30,520	4

12.3.2 Transformation Reductions

Equations confine the feasible region to an affine subspace of lower dimension. It is possible to eliminate an equation and a column with a nonzero coefficient in the equation. If $a_{ik} \neq 0$ in an equation constraint i, then

$$x_k = (r^i - \sum_{j \neq k} a_{ij} x_j)/a_{ik}$$

can be used to eliminate x_k from the problem. The equation is eliminated but the bounds on x_k generate a constraint.

$$h_k \leq (r^i - \sum_{j \neq k} a_{ij} x_j)/a_{ik} \leq u_k \ .$$

This elimination is equivalent to pivoting on a_{ik} . Although repeated application will remove all equations, this will not in general make the problem easier to solve or reduce the number of constraints.

There are two special cases where the elimination of a column can reduce the number of constraints. If the column x_k has no upper or lower bound (i.e., a "free" variable), then the generated constraint is redundant and thus may be eliminated.

If the equation has only two columns with nonzero coefficients, the generated constraint is a simple bound on a variable and thus may be combined with the existing bounds on the variable.

A particular type of constraint that is common to many models leads to an equation that has a nonzero coefficient for a column with no bounds. Commonly called <u>material balance equations</u>, these constraints set one nonnegative column equal to the sum of several other nonnegative columns.

$$x_k - \sum_{j \in J} x_j = 0 \ ,$$

$$0 \leq x_k \ , \ 0 \leq x_j \ , \ \ j \in J \ .$$

Since the x_j $j \in J$ are all restricted to be nonnegative, it is easy to see that the bound $x_k \geq 0$ is redundant -- that is, if it is eliminated, the feasible region is unchanged. Thus k can be regarded as a column with no bounds and the elimination of x_k will reduce the number of constraints by one.

Notice that although the analysis of the material balance equation results in removing the bound from x_k , when eliminating x_k any single equation that has a nonzero coefficient in column k can be substituted out.

The analysis applies to a generalization of material balance rows:

$$a_{ik}x_k + \sum_{j \in J} a_{ij}x_j = b_i \ ,$$

$$0 \leq x_k \ , \ \ h_j \leq x_j \leq u_j \ ,$$

where $a_{ik} > 0$, b_i and $h_j \geq 0$ and $a_{ij} < 0 \ \forall \ j \in J$. Although this is not the most general form which can be used to designate that the bound(s) on x_k are redundant, this form (and its negative) captures all the cases in the real-world problems that we have analyzed.

Doubleton equations. Equation constraints with exactly two nonzero coefficients can be identified for immediate elimination. The elimination of the equation and one of two columns is accomplished as described above. The generated inequality contains just a single nonzero coefficient so it can be removed by modifying the bounds on the surviving column.

Free column equations. An equation with a nonzero coefficient for a free column can be identified for immediate elimination along with the free column. The transformation to remove the constraint and column is equivalent to pivoting on the nonzero coefficient.

Redundant bounds. Our analysis uses only the generalized material balance equation to identify columns that have no bounds. It is possible to generalize this idea to include the use of all constraints to identify such columns. Each constraint may be used to generate bounds on all the columns with nonzero coefficients. For constraint i with $a_{ik} > 0$

$$-\sum_{\substack{a_{ij}>0 \\ j\neq k}} a_{ij}u_j - \sum_{a_{ij}<0} a_{ij}h_j + r_i \leq a_{ik}x_k \leq r^i - \sum_{\substack{a_{ij}>0 \\ j\neq k}} a_{ij}h_j - \sum_{a_{ij}<0} a_{ij}u_j \; .$$

An analogous result can be constructed for $a_{ik} < 0$.

The intersection of these bounds from all the constraints imply bounds for column k which may reveal h_k and u_k to be redundant and thus permit k to be designated a free column, or which can be used to tighten h_k and u_k .

Although this test can be done efficiently for any particular a_{ik} , there are potentially quite a large number of candidates to test. We have not included this test in our results here, and we further believe that there are few real LP problems for which this test yields significantly more columns without bounds than examination for generalized material balance equations.

One particular situation for which this test may be selectively applied is that in which a coefficient is much larger, or much smaller than its cohorts. In this case the inferred bounds may reveal infinitesimal or (respectively) gigantic bounds for x_k , possibly suggesting deletion of the column.

Table 12.2 shows doubleton equations found after the final simple reduction pass. Note the last model (ODSAS) for which almost all constraints are identified as doubleton equations.

Table 12.2

RESULTS OF SIMPLE REDUCTIONS

Model	Columns Single.	Columns Fixed	Constraints Single.	Constraints Redund.	Reduction Passes	Doubleton Equations	Seconds
NETTING	1	8	29	17	4	7	0.81
AIRLP	0	20	0	0	2	0	1.78
COAL	0	0	0	0	2	0	2.12
TRUCK	0	2	0	1	2	0	5.57
CUPS	49	57	18	55	4	39	1.90
FERT	0	406	0	13	4	0	14.25
PIES	50	183	16	0	3	0	3.32
PAD	30	183	16	0	3	0	3.26
ELEC	56	494	110	14	4	3	8.64
GAS	60	501	31	30	4	0	10.08
PILOT	123	277	12	91	11	36	17.15
FOAM	0	2	36	0	2	0	3.30
LANG	220	105	68	55	20	9	61.45
JCAP	414	6	277	360	3	180	12.16
PAPER	190	145	90	45	5	359	20.61
ODSAS	40	0	0	40	3	3,609	31.00

A sample analysis for material balance equations performed on the PAPER model detected 1,645 such constraints.

It is not always obvious whether actually applying a particular transformation reduction will produce an LP model which is easier to solve. In particular, transformation reductions can generate a "reduced, equivalent LP" which is actually denser, and not necessarily as well-scaled as its progenitor.

On the other hand, some reductions offer a decided advantage for solution efficiency. For constraints like $x_j - x_k = 0$, $x_j, x_k \geq 0$, both variables must be in the basis for them to assume a positive value. For many commercial linear programming systems, partial pricing and the lack of effective mechanisms to cope with degeneracy do not allow the efficient treatment of the special relationship between the variables. Similarly, for material balance equations with nonnegative variables, at least two variables with coefficients of opposite sign must be in the basis before any of the variables can assume a positive value. The transformation reductions eliminate these particular instances where relationships among variables interfere with the solution progress.

For large-scale models, we analyze the reductions carefully, using all information available for the model and the problem it addresses. Numerical and structural consequences of reductions are critically reviewed in concert with the algebraic interpretations and modeling discoveries which they characterize.

Notice that both simple and transformation reductions may be viewed as linear operators that do not change the feasible region of the problem. After the reduced problem is solved, the inverse operators applied to the optimal solution construct an optimal solution to the original problem.

The analysis makes no special use of an objective function. Free rows (i.e., $-r_i = r^i = \infty$) may be included in the problem. The objective function and any free rows designated to be included are only examined in determining if a column has a single nonzero coefficient. Thus the reduction can be done for several different objective functions simultaneously.

12.3.3 Generalized Upper Bounds

Rows for which each column has at most one nonzero coefficient (restricted to those rows) collectively form a generalized upper bound (GUB) set. Usually, we additionally require that the coefficients in these rows be capable of being rendered to ± 1 by simple row or column scaling.

The problem of identifying a GUB set of maximum row dimension is NP-hard, making optimal GUB identification algorithms hopelessly inefficient for our purposes. Heuristics adapted from work by Graves and by Senju and Toyoda (1968) (see also Brearley, Mitra and Williams (1978)) work very effectively and dependably at large-scale to find maximal GUB sets.

Unfortunately, the problem of determining just the size of the maximum GUB set is also NP-hard. However, Brown and Thomen (1980) have developed bounds on the size of the maximum GUB set which are sharp and easily computed. These bounds have been used to show, in some cases, that maximum GUB sets have been achieved via heuristic methods. In any case, the bounds provide an excellent objective measure of the quality of any GUB set, regardless of the means of its derivation. Frequently, manual GUB analysis will suffice for models with amenable structure.

The bounds are developed in terms of the number of distinct conflicts present in the model. Two rows are in conflict if they each have a nonzero element in a common column, making them mutually exclusive in a GUB set. If s_i is the number of rows in conflict with row i, then the total problem conflict count for a model with m rows is

$$c = 1/2 \sum_i s_i < 1/2\, m(m-1) \quad .$$

A problem-independent bound on the size of the maximum GUB set is

$$u_1 = \lfloor (.5 + \sqrt{.25 + m(m-1) - 2c}\,) \quad ,$$

where $\lfloor$ indicates truncation to the next lower integer.

A tighter, problem-dependent bound is

$$u_2 = \begin{cases} m - \lceil c/y \quad , & c \leq (m-y)\, y \\ \lfloor (.5 + \sqrt{.25 + y(2m - y - 1) - 2c}\,) \; , & c > (m-y)\, y \; ; \end{cases}$$

where

$$y = \max_i s_i \quad ,$$

and $\lceil$ indicates rounding to the next higher integer.

Tighter upper bounds have been derived for the size of the maximum GUB set, as well as lower bounds.

Table 12.3 contains the results of automatic GUB identification applied to the benchmark models. Row eligibility is based on the capability to scale the row to contain only 0 , ±1 coefficients. GUB quality is the number of GUB rows found, expressed as a percentage of the best known upper bound on maximum GUB row dimension (actual GUB quality may be greater than this conservative estimate). The results were obtained using FORTRAN H (Extended) with code optimization.

Table 12.3

GUB IDENTIFICATION

Model	Constraints GUB-Eligible	Constraint Conflicts Count	Constraint Conflicts Density	GUB Rows	GUB Quality	GUB Seconds
NETTING	71	46	1.85%	36	78.26%	0.05
AIRLP	170	2,983	20.64%	150	100%	0.65
COAL	170	3,753	26.13%	111	91.74%	0.92
TRUCK	219	10,438	43.53%	29	20.28%	5.00
CUPS	336	744	1.32%	160	66.67%	0.21
FERT	605	16,455	9.01%	559	98.59%	6.73
PIES	662	4,116	1.88%	172	40.76%	2.82
PAD	694	4,416	1.84%	188	41.87%	3.34
ELEC	784	6,167	2.01%	309	62.80%	1.15
GAS	789	22,220	7.15%	608	93.25%	3.79
PILOT	975	12,110	2.55%	255	33.73%	2.75
FOAM	989	8,186	1.67%	917	98.18%	1.73
LANG	1,235	46,424	6.09%	342	35.15%	14.90
JCAP	2,446	16,578	0.55%	529	29.19%	2.23
PAPER	3,528	35,047	2.82%	1041	34.65%	5.77
ODSAS	4,647	5,220	0.05%	749	18.61%	7.12

12.3.4 Implicit Network Rows

Implicit generalized network rows are a set of rows for which each column has at most two nonzero coefficients (restricted to those rows). Such rows in LP are called implicit networks with gains if columns with two nonzero coefficients (in these rows) can be converted by simple row and column scaling such that one nonzero coefficient is $+1$.

Pure network rows (NET) can be converted by simple row and column scaling such that all nonzero coefficients (restricted to those rows) have value ± 1 , and such that columns with two nonzero coefficients (in those rows) have one $+1$ and one -1 . Such rows in LP are called pure networks.

Simple row and column scaling is restricted such that application of each scale factor renders an entire row, or column, to the desired sign (and unit magnitude for pure NET).

The problem of identifying a NET factorization of maximum row dimension is NP-hard (Wright (1980)), making optimal NET identification algorithms unattractive in a practical sense. The problem of determining just the size of the maximum NET set is also NP-hard. Thus, heuristic identification methods are mandated.

An extension of GUB can be used to achieve NET factorizations. First a GUB set is determined by methods mentioned in Section 12.3.3. Then, a second GUB set is found from an eligible subset of remaining rows. The second GUB set is conditioned such that its row members must possess nonzero coefficients of opposite sign in each column for which the prior GUB set has a nonzero coefficient. This double-GUB (DGUB) factorization yields a bipartite NET factorization. Thus, DGUB heuristically seeks the maximum embedded transportation or assignment row factorization. Pure network equivalents derive from proper editing of eligible rows.

Generalizing on the theme of Senju and Toyoda, a method has been developed by Brown and Wright (1980) for direct NET factorization of implicit network rows. Pure NET rows can be identified with the same procedure by simple screening of admissible candidate rows.

This heuristic is designed to perform a network factorization of a signed elementary matrix (0, 1 entries only). It is a deletion heuristic which is feasibility seeking. The measure of infeasibility at any point is a matrix penalty computed as the sum of individual row penalties. The algorithm is two-phased, one pass, and non-backtracking. The first phase yields a feasible set of rows, while the second phase attempts to improve the set by reincluding rows previously excluded. Each iteration in Phase I either deletes a row or reflects it (multiplies it by -1) and guarantees that the matrix penalty will be reduced. Thus, the number of iterations in Phase I is bounded by the initial value of the matrix penalty, which is polynomially bounded.

Let $A = [a_{ij}]$ be an $m \times n$ matrix with $a_{ij} = 0, \pm 1 \; \forall \; i,j$.

Problem: Find a matrix $N = [n_{ij}]$ with $(m - k)$ rows and n columns which is derived from A by

1. Deleting k rows of A where $k \geq 0$,

2. Multiplying zero or more rows of A by -1 , where N has the property that each column of N has at most one $+1$ element and at most one -1 element.

We wish to find a "large" N in the sense of containing as many rows as possible, i.e., minimize k .

Terminology and Notation:

1. E is the set of row indices for rows eligible for inclusion in N and is called the eligible set.

2. C is the set of row indices for rows removed from E in Phase I (Deletion). Some rows in C may be readmitted to E in Phase II. C is called the candidate set.

3. The phrase "reflect row i' of A" means to multiply each element in row i' by -1 , i.e., $a_{i'j} \leftarrow -a_{i'j} \; \forall \; j$.

4. Other notation will be defined in the algorithm itself.

ALGORITHM:

Phase I - <u>Deletion of Infeasible Rows</u>

Step 0: <u>Initialization</u>. Set $E = \{1,2,\ldots,m\}$, $C = \phi$.

For each column j of A compute the $+$ penalty (K_j^+) and the $-$ penalty (K_j^-) as follows:

$$K_j^+ = \Big(\sum_{i \in E : a_{ij} > 0} 1 \Big) - 1, \quad K_j^- = \Big(\sum_{i \in E : a_{ij} < 0} 1 \Big) - 1 \; .$$

These penalties represent the number of excess $+1$ and -1 elements, respectively, in column j which prevent the row whose indices remain in E from forming a valid N matrix. A penalty value of -1 , for $K_j^+(K_j^-)$ indicates that the column does not contain a $+1(-1)$ element.

Step 1: <u>Define Row Penalties</u>. For every $i \in E$, compute a row penalty (p_i) as follows:

$$p_i = \sum_{j:a_{ij}>0} K_j^+ + \sum_{j:a_{ij}<0} K_j^-.$$

This is simply the sum of + penalties for all columns in which row i has a +1 plus the sum of - penalties for all columns in which row i has a -1 .

Step 2: Define Matrix Penalty. Compute the penalty (h) for the matrix by summing the row penalties as follows:

$$h = \sum_{i \in E} p_i .$$

If $h = 0$, then go to Step 7. Otherwise go to Step 3.

Step 3: Row Selection. Find the row $i' \in E$ with the greatest penalty i.e.,

$$\text{Find } i' \in E \text{ such that } p_{i'} = \max_{i \in E} p_i .$$

(If there is a tie, choose i' from among the tied values.) Compute the reflected row penalty $\overline{p}_i$, for i' as follows:

$$\overline{p}_{i'} = \sum_{j:a_{i'j}>0} (K_j^- + 1) + \sum_{j:a_{i'j}<0} (K_j^+ + 1) .$$

This would be the row penalty for row i' if it were to be reflected.

Step 4: Delete, or Reflect Row

Case i) $\overline{p}_{i'} \geq p_{i'}$. Let $E \leftarrow E - \{i'\}$, $C \leftarrow C \cup \{i'\}$. Go to Step 5.
Case ii) $\overline{p}_{i'} < p_{i'}$. Reflect row i' . Go to Step 6.

Step 5: Reduce Column Penalties as follows:

For all j such that $a_{i'j} > 0$, $K_j^+ \quad K_j^+ - 1$.

For all j such that $a_{i'j} < 0$, $K_j^- \quad K_j^- - 1$.

Go to Step 1.

Step 6: Change Column Penalties as follows:

Using the $a_{i'j}$ values after reflection of row i' ,

For all j such that $a_{i'j} > 0$, $K_j^+ \leftarrow K_j^+ + 1$ and $K_j^- \leftarrow K_j^- - 1$.

For all j such that $a_{i'j} < 0$, $K_j^+ \leftarrow K_j^+ - 1$ and $K_j^- \leftarrow K_j^- + 1$.

Go to Step 1.

Phase II - Reinclusion of Rows from C

Step 7: Eliminate Conflicting Rows. The rows with indices in E , some possibly reflected from the original A matrix, form a valid N matrix. However, some of the rows removed from E and placed in C may now be reincluded in E if they do not make $h \quad 0$. Remove from C (and discard) all row indices for rows which, if reincluded in E in present or reflected form, would make $h > 0$. That is remove i from C if

a) $\exists\ j_1$ such that $a_{ij_1} > 0$ and $K_{j_1}^+ = 0$

or $a_{ij_1} < 0$ and $K_{j_1}^- = 0$;

and

b) $\exists\ j_2$ such that $a_{ij_2} > 0$ and $K_{j_2}^- = 0$,

or $a_{ij_2} < 0$ and $K_{j_2}^+ = 0$.

If $C = \phi$, STOP, otherwise go to Step 8.

Step 8: Select Row for Reinclusion. At this point a row from C may be reincluded in E . There are several possible schemes for selecting the row. After the row is reincluded, the column penalties are adjusted. Then go to Step 7.

No dominating rule has been discovered for breaking ties in maximum row penalty encountered in Step 3. The rule used for the computational results presented herein is to select the row with the minimum number of nonzero entries in the network set.

Other possible rules are "first-come, first-served," maximum number of nonzero entries, type of constraint, or modeler preference.

Modifications can be made to Step 0 to allow for (1) matrices including non -0, ±1 entries and/or (2) pre-specified network rows. The modifications are:

1. $E = \{i | a_{ij} = 0, \pm 1 \text{ for all } j\}$

2. Let $P = \{i | \text{row } i \text{ is prespecified}\}$

   $E \leftarrow E - P$

   After computation of K_j^+ and K_j^- find for all j

   if $\exists\, i \in P$ such that $a_{ij} = 1$ then $K_j^+ \leftarrow K_j^+ + 1$,

   if $\exists\, i \in P$ such that $a_{ij} = -1$ then $K_j^- \leftarrow K_j^- + 1$.

At termination of the algorithm, the rows in N are given by $E \cup P$.

One easily obtained upper bound on the maximum row dimension of the network factorization is:

$$u_1 = m - \underset{j}{\mathrm{MAX}}(K_j^+ + K_j^-) \quad .$$

This bound is easily computed and evidently sharp. It can be used to objectively evaluate the quality of a heuristically derived network factorization. The bound may also be used to preemptively terminate factorization effort.

Another generally tighter bound has been developed by Wright (1980) which is based on the reflection and deletion potentials for each row in the eligible set. Using this information it is possible to obtain a lower bound on the number of rows which must be deleted to achieve a feasible network set. The upper bound is then:

$$u_2 = m - \text{lower bound on rows deleted.}$$

This bound is also evidently sharp and is the bound used to compute NET quality in the following table.

Table 12.4

NET IDENTIFICATION

Model	Constraints Net-Eligible	DGUB Rows	DGUB Seconds	NET Rows	NET Quality	NET Seconds
NETTING	59	54	0.07	54	94.74%	0.08
AIRLP	150	150	0.41	150	100%	0.35
COAL	111	111	0.50	111	100%	0.43
TRUCK	219	47	8.40	46	33.58%	19.83
CUPS	300	251	0.29	295	99.33%	0.14
FERT	585	572	6.03	572	100%	6.15
PIES	142	128	0.56	128	96.97%	0.59
PAD	174	160	0.58	160	97.56%	0.59
ELEC	322	272	0.99	286	93.46%	2.07
GAS	752	682	5.00	668	94.08%	9.71
PILOT	109	109	0.92	109	100%	0.36
FOAM	966	951	1.89	951	99.58%	1.16
LANG	850	585	3.74	661	87.20%	14.82
JCAP	1,811	874	2.50	917	83.97%	44.07
PAPER	2,324	1,484	7.24	1,627	78.52%	94.16
ODSAS	410	317	3.39	286	77.51%	14.55

Table 12.4 displays the results of DGUB and NET factorizations of the benchmark models. Row eligibility is determined by the capacity to scale each row, by row scaling alone, to contain only 0, 1 entries. The NET quality is the number of NET rows found, expressed as a percentage of the upper bound on maximum NET row dimension given above (actual NET quality may be considerably better than this estimate).

12.3.5 Hidden Network Rows

Hidden network rows[1] are a set of rows which satisfy NET row restrictions after full linear transformation of the model. That is, realization of these (LNET) rows may require a general linear transformation of the original model.

[1]We have coopted the term hidden from Bixby (1981), but his definition may not superficially appear to be equivalent.

The discrimination between implicit and hidden network rows is not (necessarily) in their use, but rather in their detection. The transformation group associated with implicit network rows involves only permutations and simple scaling of individual rows and columns. The hidden network rows require a completely general linear transformation and partial ordering. Thus, identification of hidden networks requires significant computation just to identify eligible rows, since any given row may conflict with subsets of its cohorts after transformation.

This problem has been solved for entire hidden network factorization, where all rows are shown to be LNET or the algorithm fails. Bixby and Cunningham (1980) and Musalem (1979) have given polynomially complex methods for entire LNET conversion. (The entire GUB problem is polynomial as well.)

Strategically, the entire hidden LNET factorization requires two steps:

DETECTION: necessary conditions for existence of an entire LNET factorization must be established, and

SCALING: a linear transformation to achieve the NET structure must be determined, if one exists.

Cunningham and Bixby attempt detection, followed by scaling. Musalem tries scaling, then detection. This is a crucial difference between methods, since problems which cannot be completely NET factorized may fail in either step.

Briefly, Cunningham and Bixby detect by showing that the incidence matrix of the model rows can be converted to a graphic matroid. They employ a method of Tutte (see references of Bixby and Cunningham, 1980). Given success, the graphic record of the detection is used to attempt to scale the model to NET, or to show that no such scaling exists.

Musalem scales the model to a ± 1 matrix, and then uses a method by Iri (see references of Musalem (1979)) to build a tree, edge by edge, which reveals the partial ordering coincident with entire hidden LNET factorization.

Both methods are polynomially complex. However, entire LNET factorization is relatively expensive by either method in that quite a large amount of real arithmetic and logic is required. Underlying data structures have not been suggested for either method. Both methods fail if complete LNET factorization is impossible, and neither leaves the investigator with much information useful in salvaging a partial LNET factorization. We conjecture that risk of preemptive failure narrowly favors the Musalem approach, since he defers the relatively involved detection step.

Locating a hidden LNET factorization of maximal row dimension has been suggested by Bixby (1981) and by Musalem (1979), but no concrete method is given and no computational testing is reported. Evidently, the maximum LNET problem is NP-hard, and its maximal relaxation remains unsolved in the practical sense of this report.

12.4 Extensions to Mixed Integer and Nonlinear Models

Mixed integer (MIP) and nonlinear (NLP) optimization models present additional challenges, especially at large scale. Our interest in the detection of structural redundancy is intensified since general purpose algorithms for (MIP) and (NLP) normally operate by solving sequences of many embedded LP models. This provides added impetus to the analysis of problems prior to their actual solution, and economically justifies some additional initial investment in problem analysis.

12.4.1. Mixed Integer Extensions

The structural analysis presented for LP is also applicable to MIP. Since most real-world models and many commercial optimization systems have only binary variable capability, our analysis addresses binary variables exclusively. Thus, binary factorization of integer variables is a prerequisite. We also assume that reductions requiring scaling of binary columns are inadmissable.

When a reduction tightens a bound for a discrete variable the bound is rounded to the nearest integer (down for u_j and up for h_j). Any tightening of bounds for a binary variable immediately results in a fixed column or an inconsistent column. The reduction for fixed columns, vacuous columns, inconsistent columns, singleton constraints, vacuous constraints, constraints that fix variables, and redundant constraints are applied exactly as described for LP.

The treatment of doubleton equations requires special consideration to identify inconsistent constraints.

A doubleton equation with both columns binary has either one solution, two solutions or no solutions. All four possible solutions (0,0), (1,0), (0,1) and (1,1) are tried.

If only one solves the equation, the binary variables are fixed at these values and the constraint is removed (the test for constraints that fix variables will also discover the equation with a single solution and accomplish the same reduction).

For the case with two solutions there can only be two situations: consider the doubleton as $a_{i1}x_1 + a_{i2}x_2 = b_i$ with $a_{i1} \neq 0$ and $a_{i2} \neq 0$. Then if (0,0) and (1,1) can both solve the equation, this implies that $b = 0$ and $a_{i1} = -a_{i2}$. If (0,1) and (1,0) are solutions, this implies that $a_{i1} = a_{i2} = b$. Both cases are treated correctly by the transformation described for continuous variables.

If there are no solutions, the constraint is designated as inconsistent.

For a doubleton equation with one continuous and one binary variable, the transformation described for continuous variables is used, but it must be the continuous variable that is eliminated.

The reductions to eliminate singleton columns and to designate a free column equation many not be applied to binary variables. This is ensured by marking the binary columns as ineligible for these reductions. Note that since a binary variable can never be designated as a free column, binary variables cannot be eliminated as a variable in a free column constraint but may be among the other columns in such a constraint.

The redundant bounds test may be used to tighten bounds on binary variables and thus fix them, or show that the MIP has no feasible solution.

12.4.2 Nonlinear Extensions

Large-scale nonlinear optimization, though not yet in wide use, can benefit from the analysis techniques given here for LP, and demands some additional special treatment.

We have experience with only two large-scale, general-purpose optimization systems with full nonlinear capability: our own X-system and MINOS/Augmented (Saunders and Murtagh (1980)). Both of these systems can accept linear problem features and labels in MPS format and nonlinear terms from function-generators. Both systems can also employ several alternate problem generation interface standards.

These systems are each designed to exploit any linearity or near-linearity in the NLP. Given a starting solution, it is of no little interest to analyze the linear portion of the NLP which will, after all, be solved many times--the reason that we support *all* LP features for NLP. Also, any local linearization of the NLP is subject to analysis.

However, it has been our misfortune to have repeatedly discovered that NLP presents us with unique structural curiosities.

We refer to the foremost among these as *function coordination*. There are myriad opportunities with NLP to unwittingly introduce discontinuities and miscellaneous unruliness in functions and derivatives. Whether by programming error, mathematical blunder or numerical difficulty, these errors inflict great vexation and expense.

Detection of such difficulties is quite challenging since, unlike LP, procedures *and* data are used to express the problem at hand. As a bare minimum, we employ a preemptive analysis module that acts as a complete surrogate for the optimizer, employing standard interface conventions and exercising all functions and data.

The starting solution and scaling parameters are used to check analytic gradients (if supplied) with numerical difference approximations. Approximation of functions is then attempted to reveal behavior local to the initial solution such as apparent convexity and degree of nonlinearity. Optionally, the first step of the algorithm is simulated and the same analyses performed.

From the initial results algorithm tolerances may be changed, programming errors detected, and so forth, until acceptable model behavior is observed.

In some cases, suspicious functions may be evaluated at column bounds to see if numerical arithmetic faults occur. Some models require construction and maintenance of a *trust region* for the approximations implied by the NLP algorithm, and prior analysis is absolutely essential in these cases.

Structural analysis of MIP and NLP can frequently--even repeatedly--presage outright failure of the solution algorithms to be employed. For these models, the effects of structural redundancy can be far more significant than for simple LP.

12.5 Conclusion

The techniques reported here have been used with great success on a wide variety of large LP (MIP) models. The context of this research is somewhat atypical in that the models which we work with are often sent to us for analysis and solution precisely because they have already failed elsewhere. In these cases, our motives are to quickly diagnose suspected trouble before optimization, prescribe remedies, and perform the actual optimization reliably and efficiently.

This has undoubtedly biased our view of structural detection methods. Practical considerations arising from turnaround deadlines and the specific advantages of our own optimization system (Brown and Graves (1975))[2] have colored our judgment. Many provocative suggestions for further research have not been pursued, either due to lack of opportunity, to poor intuition, or to simple economics. Whether or not by equivalent prejudice, Krabek (1979) reports some similar methods for detecting redundancy in large-scale MIP.

Various commercial optimization systems support "CRASH", "REDUCE", and other operators which implement some of these reductions automatically during LP solution. These systems are not reviewed here. We stress the value of structural analysis techniques as stand-alone tools, rather than as exclusive features of actual LP-solution algorithms.

A great deal of insight has been gained from these experiments. The cost of analysis is truly insignificant relative to the information and solution efficiency thereby gained. Revelations have ranged from outright rejection of absurd formulations, to subtle inferences on the project management and interpersonal relations among model proponents. Very few models fail to reveal some totally unsuspected structural curiosity. Indeed, it is often some small aberration that proves most revealing. Sometimes, the combined effects of several minor features collectively contribute to a discovery of significant model attributes.

Our general operational guideline has been to avoid heavy computational investment in model analysis. Rather, highly efficient methods are used repeatedly on variations of each model. Manual and intuitive analysis of these results usually reveal much more than could be reasonably expected from any totally automated method of exponential complexity. After all, just the names of rows and columns can be expected to reveal a great deal about the model, but exploiting this mathematically

[2]The X-system (XS) differs in many ways from classical large-scale mathematical programming systems; it simultaneously supports simple and generalized upper bounds, general basis factorization, MIP, nonlinear, and decomposition features. In addition, the fundamental LP algorithm has been enhanced to intrinsically incorporate elastic range restrictions. XS is particularly suited for solution in limited time of large models with complicating features.

virtually defies automation in any general manner; interactive analysis of large-scale models is uncompromisingly challenging in a technical sense and equally rewarding.

Large degrees of structural redundancy are routinely found as intrinsic features in real-life models. However, we feel that it is an abominable practice to proselytize in favor of some particular model structure at the expense of model realism or common sense. For instance, network models have recently received unprecedented attention in the literature. The implication has often been that since networks are usually found in models, networks should be used as the exclusive model. This is, of course, patent nonsense, smacking of a solution in search of a problem. An analyst should view intrinsic redundancy as an interesting feature of models, rather than forcing models to exhibit minimal redundancy, or requiring that they follow some particular structural pattern.

As for automating the discovery of <u>all</u> redundancy in a model, this exercise seems to be almost exclusively academic with large-scale real-life LP projects. In those rare cases for which such extensive analysis is justified, we suggest a straightforward view and a frontal attack with an imbedded LP optimizer.

12.6 Acknowledgments

David Thomen has developed much of the GUB identification material, and William Wright has contributed fundamentally to the network identification research. Both gentlemen have suffered nobly with us the singular exasperation of experimentation at large-scale.

References

Aho, H. V., Hopcroft, J. E., Ullman, J. D., The Design and Analysis of Computer Algorithms, Reading, MA: Addison-Wesley, 1974.

Amerongen, R.A.M. van, Methoden voor vermogensherverdeling in elektrische energievoorzieningssystemen in (potentieel) overbelaste situaties, afstudeerverslag T. H. Delft, afdeling elktrotechniek, 1977.

Antosiewicz, H. A., ed., Proceedings of the Second Symposium in Linear Programming, Washington, D.C., 1955.

Balas, E., "Relati de Dominatie in Probleme de Programare Lineara," Studii se Cercetari Mathematiche, Vol. 13, No. 3, 1962, pp. 511-519.

Balas, E., "An Additive Algorithm for Solving Linear Programs with Zero One Variables," Operations Research, Vol. 13, No. 4, 1965, pp. 517-546.

Balinsky, M. L., "An Algorithm for Finding all Vertices of Convex Polyhedral Sets," Journal of the Society for Industrial and Applied Mathematics, Vol. 9, No. 1, 1961, pp. 72-88.

Bixby, R. E., "Hidden Structure in Linear Programs," Computer Assisted Analysis and Model Simplification, ed. H. Greenberg and J. Maybee, Academic Press, New York, 1981, pp. 327-360.

Bixby, R. E. and Cunningham, W. H., "Converting Linear Programs to Network Problems," Mathematics of Operations Research, Vol. 5, 1980, pp. 321-357.

Bland, R, G., "New Finite Pivoting Rules for the Simplex Method," Mathematics of Operations Research, Vol. 2, 1977, pp. 103-107.

Boneh, A., "Minimal Representation of Nonlinear Inequalities by a Probablistic Set Covering Problem Equivalence," Technical Report TRCS81-05, Computer Science Department, University of California, Santa Barbara, April, 1981.

Boneh, A. and Golan, A., "Constraints Redundancy and Feasible Region Boundedness by Random Feasible Points Generator," Paper presented at EURO III, Amsterdam, April 9-11, 1979.

Boneh, A. & Golan, A., "PREDUCE - A Probabilistic Algorithm Identifying Redundancy," Proceedings of the COAL Conference on Mathematical Programming Testing and Validation, Algorithms and Software, National Bureau of Standards, Boulder, Colorado, January 5-6, 1981.

Boot, J.C.G., "On Trivial and Binding Constraints in Programming Problems," Management Science, Vol. 8, No. 4, 1962, pp. 419-441.

Boot, J.C.G., Quadratic Programming, Amsterdam: North Holland, 1964.

Bowker, A. H. and Lieberman, G. J., Engineering Statistics, Prentice-Hall, Inc., New Jersey, 1972.

Bradley, G., Brown, G., and Galatas, P., "ATHENA: An Interactive System to Analyze Large-Scale Optimization Models," Naval Postgraduate School Technical Report NPS52-80-005, April, 1980.

Bradley, G., Brown, G., and Graves, G., "Design and Implementation of Large-Scale Primal Transshipment Algorithms," Management Science, Vol. 24, No. 1, 1977a, p. 1.

Bradley, G., Brown, G., and Graves, G., "Preprocessing Large-Scale Optimization Models," paper presented at ORSA/TIMS, Atlanta, November, 1977b.

Brearly, A. L., Mitra, G., and Williams, H. P., "Analysis of Mathematical Programming Models Prior to Applying the Simplex Algorithm," Mathematical Programming, Vol. 8, 1978, pp. 54-83.

Brown, G. and Graves, G., "Design and Implementation of a Large-Scale (Mixed Integer) Optimization System," paper presented at ORSA/TIMS, Las Vegas, November, 1975.

Brown, G. and Thomen, D., "Automatic Identification of Generalized Upper Bounds in Large-Scale Optimization Models," Management Science, Vol 26, No. 11, 1980, pp. 1166-1184.

Brown, G. and Wright, W., "Automatic Identification of Network Rows in Large-Scale Optimization Models," (in Proceedings of the Symposium of Computer-Assisted Analysis and Model Simplification, Boulder, March 24, 1980, and forthcoming in a collection of papers from IIASA, Laxenburg, Austria).

Charnes, A. and Cooper, W. W., Management Models and Industrial Applications of Linear Programming, Vol. I and II, New York: Wiley, 1961.

Charnes, A., Cooper, W. W., and Thompson, G. L., "Some Properties of Redundant Constraints and Extraneous Variables in Direct and Dual Linear Programming Problems," Operations Research, Vol. 10, No. 5, 1962, pp. 711-723.

Dantzig, G. B., "Programming in a Linear Structure," Comptroller USAF, Washington, D.C., 1948.

Dantzig, G. B., "Upper Bounds, Secondary Constraints, and Block-triangularity," Econometrica, Vol. 23, No. 2, 1955, pp. 174-183.

Dantzig, G. B., Linear Programming and Extensions, Princeton: Princeton University Press, 1963.

Dantzig, G. B., and Wolfe, P., "The Decomposition Principle for Linear Programs," Operations Research, Vol. 8, 1960, pp. 101-111.

Dyer, M. E. and Proll, L. G., "Vertex Enumeration in Convex Polyhedra - a Comparative Computational Study," in T. B. Boffey, ed, Proceedings of the CP77 Combinatorial Programming Conference.

Eckhardt, U., "Redundante Ungleichungen bei Linearen Ungleichungssystemen," Unternehmensforschung, Vol. 12, 1971, pp. 279-286.

Farkas, J., "Uber die Theorie der Einfachen Ungleichungen," Journal Reine Angew. Math., Vol. 124, 1902, pp. 1-27.

Gal, S., Bachelis, B., and Ben-Tal, A., "On Finding the Maximal Range of Validity of a Constrained System," SIAM Journal of Control and Optimization, Vol. 16, No. 3, 1978, pp. 473-503.

Gal, T., Betriebliche Entscheidungsprobleme, Sensitivitatsanalyse und Parametrische Programmierung," De Gruyter, Berlin/New York, 1973.

Gal, T., "Redundancy Reduction in the Restrictions Set Given in the Form of Linear Inequalities," Progress in Cybernetics and Systems Research, Vol. 1, 1975a, pp. 177-179.

Gal, T., "Zur Identification Redundanter Nebenbedingungen in Linearen Programmen," Zeitschrift for Operations Research, Vol. 19, 1975b, pp. 19-28.

Gal, T., "A Note on Redundancy and Linear Parametric Programming," Operational Research Quarterly, Vol. 26, No. 4, 1975c, pp. 735-742.

Gal, T., "A General Method for Determining the Set of All Efficient Solutions to a Linear Vectormaximum Problem," European Journal of Operational Research Vol. 1, 1977, pp. 307-322.

Gal, T., "Determination of All Neighors of a Degenerate Extreme Point in Polytopes," Working Paper No. 17b, Fernuniversitat, Hagen, 1978.

Gal, T., "Redundancy in Systems of Linear Inequalities Revisited," Discussion Paper No. 19, Fernuniversitat, Hagen, 1978.

Gal, T., "Another Method for Determining Redundant Constraints," this volume, 1982.

Gal, T., and Leberling, H., "Redundant Objective Functions in Linear Vectormaximum Problems and Their Determination," European Journal of Operational Research, Vol. 1, 1977, pp. 176-184.

Gale, D., The Theory of Linear Economic Models, New York: McGraw-Hill, 1960.

Garey, M. R. and Johnson, D. S., Computers and Intractability: A Guide to the Theory of NP-completeners, San Francisco: Freeman, 1979.

Gottlieb, M., "Reduktion Linearer Problemstellungen des Produktionsbereichs," Thesis, Universitat Goltinger, 1979.

Graves, G., "A Complete Constructive Algorithm for the General Mixed Linear Programming Problem," Naval Research Logistics Quarterly, Vol. 12, No. 1, 1965, p. 1.

Graves, G. and McBride, R., "The Factorization Approach to Large-Scale Linear Programming," Mathematical Programming, Vol. 10, No. 1, 1976, p. 91.

Graves, G. and Van Roy, T., "Decomposition for Large-Scale Linear and Mixed Integer Linear Programming," UCLA Technical Report, November, 1979.

Greenberg, H., "An Algorithm for Determining Redundant Inequalities and all Solutions to Convex Polyhedra," Numerische Mathematik, Vol. 24, 1975, pp. 19-26.

Hicks, H. R., Fundamental Concepts in the Design of Experiments, Holt, Rinehart and Winston, New York, 1973.

Hines, W. H. and Montgomery, D. C., Probability and Statistics in Engineering and Management Science, Ronald Press Co., New York, 1972.

Hoffman, A. J., "How to Solve a Linear Programming Problem," in H. A. Antosiewicz, ed., 1955, pp. 397-423.

Holm, S. and Klein, D., "Size Reduction of Linear Programs with Special Structure," Working Paper, Odense University, 1975.

Holm, S. and Klein, D., "Identification of Non-binding Constraints and Zero Variables in Linear Programming," Operations Research Verfahren, Vol. 25, No. 1, 1976, pp. 58-65.

Jeroslow, R. G., "The Simplex Algorithm with the Pivot Rule of Maximizing Criterion Improvement," Discrete Mathematics, Vol. 4, No. 4, 1973, pp. 367-377.

Karwan, M. H., Zionts, S., and Villarreal, B., "An Improved Interactive Multicriteria Integer Programming Algorithm," Working Paper No. 530, School of Management, State University of New York, Buffalo, 1982.

Khachian, L. G., "Polynomial Algorithm for Linear Programming," Doklady Akademika USSR, Mathematica, Vol. 244, No. 5, 1979, pp. 1093-1096.

Klein, D. and Holm, S., "Size Reduction of Linear Programs Using Bounds on Problem Variables," Unpublished Working Paper, Florida International University, September, 1979.

Kotiah, T.C.T. and Steinberg, D. I., "Occurrences of Cycling and Other Phenomena Arising in a Class of Linear Programming Models," Comm of the A.C.M., Vol. 20, No. 2, 1977, pp. 107-112.

Krabek, C., "Some Experiments in Mixed Integer Matrix Reduction," Paper presénted at XXIV International TIMS Meeting, Honolulu, June 18, 1979.

Kuhn, H. W. and Tucker, A. W., ed., "Linear Inequalities and Related Systems," Princeton, 1956.

Kunzi, H. P. and Tschach, H., "Numerische Betrachtungen zur Linearen Optimierung," Operations Research Verfahren, III, 1967, pp. 270-285.

Lemke, C., "Dual Method of Solving the Linear Programming Problems," Naval Research Logistics Quarterly, Vol. 1, No. 1, 1954, pp. 36-47.

Lisy, J., "Metody pro Nalezini Redundantnich Omezeni v Ulohach Linearniho Programovani," Ekonomicko Matematicky Obzor, Vol. 7, No. 3, 1971, pp. 285-298.

Llewellyn, R. W., Linear Programming, New York: Holt, Rinehart and Winston, 1964.

Lotfi, V., A Study of Size-Reduction Techniques in Linear Programming, Ph.D. Dissertation, State University of New York, Buffalo, 1981.

Luenberger, D. G., Introduction to Linear and Non-Linear Programming, Reading, MA: Addison-Wesley, 1973.

Mattheiss, T. H., An Algorithm for Determining Irrelevant Constraints and All Vertices in Systems of Linear Inequalities," Operations Research, Vol. 21, No. 1, 1973, pp. 247-260.

Mattheiss, T. H., "A Method for Finding Redundant Constraints of a System of Linear Inequalities," this volume, 1982.

Mattheiss, T. H., and Schmidt, B. K., "Computational Results on an Algorithm for Finding All Vertices of a Polytope," Mathematical Programming, Vol. 18, 1980, pp. 308-329.

Mattheiss, T. H. and Rubin, D. S., "A Survey and Comparison of Methods for Finding all Vertices of Convex Polyhedral Sets," Mathematics of Operations Research, Vol. 5, No. 2, 1980, pp. 167-185.

Mauri, M., "Vincoli Superflui e Variabili Estranee in Programmarione Lineare," Ricerca Operativa, Vol. 5, No. 1, 1975, Part 1, pp. 21-42, and Vol. 6, No. 2, 1976, Part 2, pp. 3-42.

McBride, R., Factorization in Large-Scale Linear Programming, Ph.D. Dissertation, UCLA, 1973.

Meyerman, B. G., "Some Results of a Reduction Algorithm for Linear Programming Problems," Unpublished Research Paper, Department of Operations Research, University of Groningen, 1979.

Meyerman, G. L., "Betekenis van een Aantal Cultuurtechnische Factoren voor de Ontwikkelingsmogelijkheden van Veenkoloniale Akkerbouwbedrijven," Dissertation, Agricultural University Wageningen, 1966.

Motzkin, T. S., Beitrage zur Theorie der Linearen Ungleichungen, Dissertation, UCLA, December, 1979.

Musalem, S., Converting Linear Models to Network Models, Ph.D. Dissertation, UCLA 1979.

Nijkamp, P. and Spronk, J., "Three Cases in Multiple Criteria Decision Making: An Interactive Multiple Goal Programming Approach," Report 7822, Centre for Research in Business Economics, Erasmus University, Rotterdam, 1978.

O'Neill, R. P. and Layman, C. H., "A Study of the Effect of LP Parameters on Algorithm Performance," Computers and Mathematical Programming, National Bureau of Standards, 1978.

Parzen, E., Stochastic Processes, Holden-Day, Inc., 1962.

Rabin, M. O., "Probabilistic Algorithms," in Algorithms and Complexity edited by J. F. Traub, Academic Press, Inc., 1976, pp. 21-39.

Rinnooy Kan, A.H.G. and Telgen, J., "The Complexity of Linear Programming," Statistica Neerlandica. To appear.

Robinson, S. M., "Stability Theory for Systems of Inequalities," SIAM J. Numer. Anal., Vol. 12, No. 5, 1975, pp. 754-769.

Rubin, D. S., "Redundant Constraints and Extraneous Variables in Integer Programs," Management Science, Vol. 18, No. 7, 1972, pp. 423-427.

Rubin, D. S., "Finding Redundant Constraints in Linear Programs," this volume, 1982.

Saunders, M. and Murtagh, B., "MINOS/Augmented Users Manual," Technical Report SOL 80-14, Stanford, June, 1980.

Senju, S. and Toyoda, R., "An Approach to Linear Programming with 0-1 Variables," Management Science, Vol. 15, No. 4, 1968, p. B196.

Sethi, A. P. and Thompson, G. L., "The Non-candidate Constraint Method for Reducing the Size of a Linear Program," this volume, 1982.

Shefi, A., Reduction of linear inequality constriants and Determination of All Possible Extreme Points, Ph.D. Dissertation, Stanford University, 1969.

Sherman, B. F., "A Counterexample to Greenberg's Algorithm for Solving Linear Inequalities," Numerische Mathematik, Vol. 27, 1977, pp. 491-492.

Smith, R. L., "Efficient Monte-Carlo Procedures for Generating Points Uniformly Distributed Over Bounded Regions," Technical Report 81-1, Department of Statistics and Operations Engineering, University of Michigan, Ann Arbor.

Spronk, J. and Telgen, J., "A Note on Multiple Objective Programming and Redundancy," Report No. 7906, Centre for Research in Business Economics, Erasmus University, Rotterdam, 1979.

Telgen, J., "On Redundancy in Systems of Linear Inequalities," Report 7718, Econometric Institute, Erasmus University, Rotterdam, 1977a.

Telgen, J., "Redundant and Non-binding Constraints in Linear Programming Problems," Report 7720, Econometric Institute, Erasmus University, Rotterdam, 1977b.

Telgen, J., "Overbodige en Niet-bindende Restricties in Lineaire Programmerings Problemen," Bedrijfskunde, Vol. 51, No. 2, 1979a, pp. 168-173.

Telgen, J., "On R. W. Llewellyn's Rules to Identify Redundant Constraints in Systems of Linear Ilequalities," Zeitschrift for Operations Research, Vol. 23, 1979b, pp. 197-206.

Telgen, J., Redundancy and Linear Programs, Doctoral Thesis, Erasmus Universitat, Rotterdam, 1979c.

Telgen, J., "A Note on a Linear Programming Problem that Cycled," COAL Newsletter, Vol. 2, 1980a.

Telgen, J., "Identifying Redundant Constraints and Implicit Equalities in Systems of Linear Constraints," Working Paper 90, College of Business Administration, University of Tennessee, Knoxville, TN 37916, 1980b.

Telgen, J., Private Communication, 1980c.

Telgen, J., "Minimal Representation of Convex Polyhedral Sets," Journal of Optimization, Theory, and Applications, 1981a.

Telgen, J., "Identifying Redundancy in Systems of Linear Constraints," this volume, 1981b.

Thompson, G. L., Tonge, F. M. and Zionts, S., "Techniques for Removing Non-binding Constraints and Extraneous Variables from Linear Programming Problems," Management Science, Vol. 12, No. 7, 1966, pp. 588-608.

Tischer, H. J., Mathematische Verfahren zur Reduzierung der Zeilenund Spaltenzahl Linearer Optimierungsaufgaben, Dissertation, Zentralinstitut fur Fertigungstechnik des Maschinenbaues, Karl Marx Stadt, 1968.

Tocher, K. D., The Art of Simulation, The English Universities Press, 1963, pp. 41-42.

Tucker, A. W., "Dual Systems of Homogeneous Linear Relations," in Kuhn and Tucker, 1955, pp. 3-18.

Tucker, Q. W., "Combinatorial Theory Underlying Linear Programs," in Graves and Wolfe, Recent Advances in Mathematical Programming, New York: McGraw-Hill, 1962.

Vajda, S., "On Recruitment in a Graded Population," paper presented at Institute per le Applicazioni del Calcolo, Rome, April, 1974.

Williams, H. P., "Simplifying Linear Programming Problems," Research Report No. 73-2, University of Sussex, 1973.

Williams, H. P., "Further Simplification of Linear Programming Problems," Research Report 75-1, University of Sussex, 1975.

Williams, H. P., "Modelling in Mathematical Programming," J. Wiley, New York, 1978.

Williams, H. P., "A Reduction Procedure for Linear and Integer Programming Problems," this volume, 1982.

Wolfe, P., "Reduction of Systems of Linear Relations (abstract)," in H. A. Antosiewicz, 1955, pp. 449-451.

Wolfe, P. and Cutler, L., "Experiments in Linear Programming," in Graves and Wolfe, ed, Recent Advances in Mathematical Programming, 1963.

Wright, W., "Automatic Identification of Network Rows in Large-Scale Optimization Models," M.S. Thesis, Naval Postgraduate School, May, 1980.

Zeleny, M., "Linear Multiobjective Programming," Lecture Notes in Economics and Mathematical Systems, No. 95, Springer-Verlag, 1974.

Zimmerman, H. J. and Gal, T., "Redundanz und ihre Bedeutung fur Betriebliche Optimierungsent-scheidungen," Zeitschrift fur Betriebswirtschaft, Vol. 45, No. 4, 1975, pp. 221-236.

Zionts, S., Size Reduction Techniques of Linear Programming and Their Application, Ph.D. Dissertation, Carnegie Institute of Technology, 1965.

Zionts, S., "Toward a Unifying Theory for Integer Linear Programming," Operations Research, Vol. 17, No. 2, 1969, pp. 359-367.

Zionts, S., "Implicit Enumeration in Integer Programming," Naval Research Logistics Quarterly, Vol. 19, No. 1, March, 1972, pp. 165-182.

Zionts, S., "Toward a Unifying Theory for Integer Linear Programming-Some Comments on a Paper by Rubin," Management Science, Vol. 19, No., 7, March, 1973, p. 837.

Zionts, S., Linear and Integer Programming, New York: Prentice-Hall, 1974.

Zionts, S. and Wallenius, J., "An Interactive Programming Method for Solving the Multiple Criteria Problem," Management Science, Vol. 22, No. 6, 1976, pp. 652-663.

Zionts, S. and Wallenius, J., "Identifying Efficient Vectors: Some Theory and Computational Results," Operations Research, Vol. 28, 1980, pp. 785-793.

Zionts, S. and Wallenius, J., "A Method for Determining Redundant Constraints and Extraneous Variables in Linear Programming Problems," this volume, 1982.

NAMES AND ADDRESSES OF AUTHORS AND CONTRIBUTORS

Dr. Arnon Boneh
Computer Science Department
University of California
Santa Barbara, California 93106
U.S.A.

Professor Gordon H. Bradley
Computer Science Department
Naval Postgraduate School
Monterey, California 93940
U.S.A.

Professor Gerald G. Brown
Operations Research Department
Naval Postgraduate School
Monterey, California 93940
U.S.A.

Professor Dr. Dr. Tomas Gal
Fachbereich Wirtschaftswisenschaften
Fernuniversität Hagen
Hagen BRD
West Germany

Professor Glenn W. Graves
University of California
Los Angeles, California 90024
U.S.A.

Professor Sören Holm
Odense University
Niels Bohrs Alle
Dk-5000 Odense
Denmark

Professor Mark Karwan
Department of Industrial Engineering
State University of New York at Buffalo
Buffalo, New York 14260
U.S.A.

Professor Dieter Klein
Department of Management
Worcester Polytechnic Institute
Worcester, Massachusetts 01609
U.S.A.

Professor Vahid Lotfi
School of Management
State University of New York at Buffalo
Buffalo, New York 14214
U.S.A.

Professor Theodore H. Mattheiss
Department of Management Science
University of Alabama
Tuscaloosa, Alabama 35401
U.S.A.

Professor David S. Rubin
Graduate School of Business Administration
University of North Carolina
Chapel Hill, North Carolina 27514
U.S.A.

Mr. Awanti P. Sethi
Graduate School of Business Administration
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213
U.S.A.

Dr. Jan Telgen
Department of Applied Mathematics
RABO Bank Nederland
3700 WB Zeist
The Netherlands

Professor Gerald L. Thompson
Grad School of Industrial Administration
Carnegie-Mellon University
Pittsburgh, Pennsylvania 15213
U.S.A.

Dr. Jyrki Wallenius
University of Jyväskylä
Department of Economics
Seminaarinkatu 15
Jyväskylä 10, Finland

Professor H. Paul Williams
Department of Business Studies
University of Edinburgh
50 George Square
Edinburgh EH8 9JY
United Kingdom

Professor Stanley Zionts
School of Management
State University of New York at Buffalo
Buffalo, New York 14214
U.S.A.