

A SET COVERING APPROACH TO
INFEASIBILITY ANALYSIS OF LINEAR
PROGRAMMING PROBLEMS AND RELATED
ISSUES

by

Mark Richard Parker

B. A., University of Colorado at Denver, 1984

M. S., University of Colorado at Denver, 1992

A thesis submitted to the
Faculty of the Graduate School of the
University of Colorado at Denver
in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
Applied Mathematics
1995

This thesis for the Doctor of Philosophy

degree by

Mark Richard Parker

has been approved by

Jennifer Ryan

Harvey Greenberg

Gary Kochenberger

J. Richard Lundgren

Burt Simon

Date _____

Parker, Mark Richard (Ph. D., Applied Mathematics)

A Set Covering Approach to Infeasibility Analysis of Linear Programming
Problems and Related Issues

Thesis directed by Associate Professor Jennifer Ryan

ABSTRACT

With the increased sophistication of both computers and optimization software, linear programming problems of a size inconceivable to solve only a few years ago are now readily accessible. As model size and complexity increase, the issue of feasibility becomes a major issue in model development. This thesis explores the analysis of linear programming infeasibility through the use of **Irreducible Infeasible Subsystems**, or IISs. In particular, we develop a constraint generation algorithm for identifying the minimum weight IIS cover, which isolates a minimal weight set of constraints whose removal from the system yields a feasible system. It has been shown that this set covering problem has a very special structure. We also explore the facial structure of the set covering polyhedra and provide a generalization to a class of covering problems. The link between this problem and the linear discriminant problem is also explored.

This abstract accurately represents the content of the candidate's thesis.

I recommend its publication.

Signed _____

Jennifer Ryan

CONTENTS

Chapter

1. Introduction	1
2. Finding the Maximum Weight Feasible Subsystem of an Infeasible System of Linear Equations	3
2.1 Introduction and Notation	3
2.2 Background on IISs	7
2.3 Hypergraph Framework	13
2.4 Techniques in Approximating the MWFS	17
3. Extensions and Unifications of IIS Results	29
3.1 Examination of Van Loon's Method	29
3.2 Van Loon's Method and the Gleeson and Ryan Method	37
4. Facial Structure of the IIS Cover Polyhedron	40
4.1 Dimension of the Min Weight IIS Covering Polytope	40
4.2 Known Properties of the General Set Covering Polytope	42
4.3 Properties of the Min Weight IIS Covering Polytope	44
4.4 Generalizations	52
5. Algorithm Description and Computational Results	55
5.1 Introduction	55
5.2 Min IIS Cover Isolation in Practice	57
5.3 Formulation of P , the Alternative Polyhedron, for General Linear Programming Problems.	62
5.4 Finding IISs	64

5.5	Heuristic Solution of Covering Problems	67
5.6	Computational Results	68
6.	Extensions to Solving the Linear Discriminant Problem .	82
7.	Summary and Areas for Further Research	88
	<u>Bibliography</u>	90

1. Introduction

This thesis is organized as follows. Chapter 2 introduces the problem of finding a maximum weight feasible subsystem of an infeasible system of linear inequalities (the MWFS problem). We review past work on this problem, including the notion of an *Irreducible Inconsistent Subsystem*, or IIS. We use a constraint generation algorithm combined with branch and bound to solve the MWFS problem. Our approach is based on finding the minimum weight cover over all IISs, which is equivalent to finding the minimum weight set of inequalities to remove from the infeasible system in order to restore feasibility. We denote this covering problem as the MWIC problem.

Chapter 3 contains new extensions and unifications to the theory of IISs.

Chapter 4 contains new theoretical results obtained in examining the facial structure of the MWIC problem. In particular, we demonstrate that all covering constraints, which correspond to the IISs of the infeasible system, are facets of the MWIC problem. We also examine a generalization of these results to a category of problems called “polyhedral covering problems.”

Chapter 5 describes an algorithm for solving the MWIC problem, and presents computational results. Also, a comparison is made between the algorithm and a fixed charge integer program formulation of the MWIC problem.

Chapter 6 discusses further applications of the algorithm to the linear discriminant problem.

Chapter 7 summarizes the results and discusses areas still open for further research.

2. Finding the Maximum Weight Feasible Subsystem of an Infeasible System of Linear Equations

In this chapter, we will establish notation and present the problem of finding a maximum weight feasible subsystem of an infeasible system of linear equations.

2.1 Introduction and Notation

Let A be an $m \times n$ real matrix and let \mathbf{b} be a real m -vector. We then define $S = \{A\mathbf{x} \leq \mathbf{b}\}$ as the system of linear inequalities arising from A and \mathbf{b} . Let $M = \{1, \dots, m\}$ be the set indexing the rows of A . We define the following subsystems of S for a subset of the rows of A , $I \subseteq M$.

$$S(I) = \{A_i\mathbf{x} \leq b_i \text{ for } i \in I\}$$

$$S \setminus S(I) = \{A_i\mathbf{x} \leq b_i \text{ for } i \notin I\}$$

$$S \setminus i = S \setminus \{A_i\mathbf{x} \leq b_i\}$$

$$S^= = \{A_i\mathbf{x} \leq b_i \mid A_i\mathbf{x} = b_i \text{ for } i \in M \text{ and for all } \mathbf{x} \text{ satisfying } A\mathbf{x} \leq \mathbf{b}\}$$

If there exists an \mathbf{x} satisfying all inequalities in S , then S is feasible, and \mathbf{x} is feasible in S . If there is no \mathbf{x} satisfying all inequalities of S , we say that S is infeasible. Similarly, if $S(I)$ is feasible for $I \subseteq M$, then $S(I)$ is called a **feasible subsystem**. If $S(I)$ is infeasible for $I \subseteq M$, we call $S(I)$ an **infeasible subsystem**. If $S(I)$ is a feasible subsystem and $S(I) \cup \{A_i\mathbf{x} \leq b_i\}$ is infeasible for every $i \notin I$, then $S(I)$ is a **maximal**

feasible subsystem. Similarly, if $S(I)$ is an infeasible subsystem and $S(I')$ is a feasible subsystem for every $I' \subset I$, then $S(I)$ is a **minimal infeasible subsystem** or an **irreducible infeasible subsystem (IIS)**. If S is feasible, then $S^=$ is the set of **implicit equalities** of S .

In the general setting, we will also have non-negativity constraints on x . We define the set of functional constraints to be the set of constraints exclusive of all non-negativity constraints. In some instances, we will wish to distinguish between non-negativity constraints and functional constraints when we are identifying IISs. We introduce the following definition due to Chinneck and Dravnieks [13]: An **irreducible inconsistent set of functional constraints (IISF)** is the complete subset of functional constraints in an IIS.

Suppose that S is infeasible. We define a maximum cardinality feasible subsystem of S as $S(I_{mcf})$ where

$$I_{mcf} = \arg \max_{I \subseteq M} \{ |I| \mid S(I) \text{ is a maximal feasible subsystem} \}$$

This extends naturally to a maximum **weight** feasible subsystem of S . If we assign weights w_i to each row i of A , we seek to find a feasible subset of rows maximizing the sum of the weights. We denote this as $S(I_{mwf})$ where

$$I_{mwf} = \arg \max_{I \subseteq M} \left\{ \sum_{i \in I} w_i \mid S(I) \text{ is a maximal feasible subsystem} \right\}$$

We also wish to review the following basic polyhedral theory. We refer to the **dual system** of S as

$$S^d = \{ \mathbf{y}^T A = \mathbf{0}, \mathbf{y}^T \mathbf{b} \leq 0, \mathbf{y} \geq 0 \}.$$

Much of the theory in this paper is based upon theorems of the alternative. In particular, we will make use of the following variant of Farkas' Theorem of the Alternative:

Theorem 2.1 (Farkas [35]) Exactly one of the following holds:

- (1) $A\mathbf{x} \leq \mathbf{b}$ is consistent.
- (2) There exists $\mathbf{y} \in R^m$ such that $\mathbf{y}^T A = 0$, $\mathbf{y}^T \mathbf{b} < 0$, and $\mathbf{y} \geq 0$.

Note that the alternative system identified in (2) above is equivalent to $S^d \cup \{\mathbf{y}^T \mathbf{b} < 0\}$.

We will also make reference to the final tableau obtained via a simplex algorithm. Given a system

$$\begin{aligned} \max \quad & \sum_{i=1}^n c_i x_i \\ \text{subject to} \quad & \\ & A\mathbf{x} \leq \mathbf{b} \\ & \mathbf{x} \geq 0 \end{aligned}$$

where A is an $m \times n$ real matrix, we use the following standard LP tableau notation; \mathbf{x}_B refers the vector of m basic variables, \mathbf{x}_N refers to the vector of the remaining $n - m$ nonbasic variables, B refers to the columns of A corresponding to the basic variables, and N refers to the columns of A corresponding to the nonbasic variables. The vector \mathbf{y} corresponds to the dual variables (or reduced costs) of the system. We define the **rate of substitution** of a nonbasic variable x_j with respect to a basic variable x_i as the rate at which x_i must change with respect to a change in x_j to maintain the equations $A\mathbf{x} + \mathbf{s} = \mathbf{b}$ with all other nonbasics held fixed at 0. This is the negative of the updated tableau entry in column j , in the row where i is basic. Thus, if we speak of a negative rate of substitution

for nonbasic variable j with respect to basic variable i , we mean that the tableau entry in column j of row i is positive.

We also refer to the **support** of a vector. We define the support of a vector x as the set of nonzero elements of x .

A serious problem in developing or modifying large linear programming models is the identification of modeling errors and inconsistencies. Many researchers have proposed identifying an IIS and using this to assist the modeler in debugging her formulation (see section 2.2). However it is known that an IIS can contain as many as $n + 1$ inequalities, and this information is potentially useless because a large system is difficult to comprehend. We propose to go a step further, to identify the maximum cardinality feasible subsystem. A more practical version allows the analyst to weight the constraints according to importance or flexibility. Then the maximum weight feasible subsystem is sought. Equivalently, we want to identify the *minimum weight set of inequalities that covers all IISs*, which we shall call the *minimum weight IIS cover* (MWIC). If we knew what the IISs of a system S were, we could formulate the following set covering problem, where w_i is the weight on the i th constraint.

$$\begin{aligned} \min \quad & \sum_{i=1}^m w_i z_i \\ \text{subject to} \quad & \sum_{i \in J} z_i \geq 1 \quad \text{for all IISs, } J. \\ & z_i \text{ binary} \end{aligned}$$

Unfortunately, it has been shown (see [9]) that the number of IISs may be exponential in the size of the original problem, so we do not want to write down the whole problem at once. Instead, we will generate constraints

dynamically for the problem and solve it iteratively.

2.2 Background on IISs

IISs, also called *minimal infeasible systems* and *minimal unsolvable systems*, were first introduced in the context of linear inequality theory in the early part of this century [8].

In his doctoral dissertation [27], Motzkin provides an excellent review of the theory of linear inequalities. He extends the previous work by deriving several results pertaining to systems of inequalities, the most interesting in this context is a necessary condition for an inconsistent system to be an IIS.

Theorem 2.2 (Motzkin [27]) The coefficient matrix A of an IIS $\{Ax \leq \mathbf{b}\}$, where $A \in R^{m \times n}$, $\mathbf{x} \in R^n$, and $\mathbf{b} \in R^m$, has rank $m - 1$. Motzkin proves that if an infeasible system is irreducibly infeasible, then the rank of the coefficient matrix of the system is one less than the number of constraints in the system.

Fan [17] again consolidated many of the known results for systems of inequalities. Additionally, he presented the following strengthening of Motzkin's characterization of an IIS to provide necessary and sufficient conditions for a system of inequalities to be an IIS:

Theorem 2.3 (Fan [17]) The system $\{Ax \leq \mathbf{b}\}$ is an IIS if, and only if, the following conditions hold.

1. There exists exactly $m - 1$ linearly independent rows.
2. There exists $\mathbf{y} \in R^m$ such that $\mathbf{y}A = 0$, $\mathbf{y}\mathbf{b} < 0$, and $\mathbf{y} > 0$.

Fan extends Motzkin's results by noting that a necessary and

sufficient condition for a system to be infeasible is the existence of a “special” solution to the alternative system from Farkas’ Theorem (see Theorem 2.1). In order to guarantee the irreducibility of the infeasible system, the solution to the alternative system must have $\mathbf{y} > \mathbf{0}$, rather than $\mathbf{y} \geq \mathbf{0}$ as in Theorem 2.1.

Van Loon [36] was the first to identify IISs with linear programming infeasibility analysis. He interprets the result of Fan in light of the simplex method and proves the result given in Theorem 2.4. Again, we consider the system $A\mathbf{x} + \mathbf{s} = \mathbf{b}$, with $\mathbf{s} \geq \mathbf{0}$. We will solve the system in terms of a single slack variable, say s_1 . Thus, we consider this row the objective function of a linear program. In the notation below, we refer to the matrix A^1 as the remaining $m - 1 \times n$ submatrix of A after the removal of row 1. We additionally use the following standard LP tableau notation (see section 2.1); \mathbf{x}_B refers the vector of $m - 1$ basic variables, \mathbf{x}_N refers to the vector of the remaining $n - m + 1$ nonbasic variables, B refers to the columns of A^1 corresponding to the basic variables, and N refers to the columns of A^1 corresponding to the nonbasic variables. The vector \mathbf{y} corresponds to the dual variables of the system.

Theorem 2.4 (Van Loon [36]) The system $A\mathbf{x} + \mathbf{s} = \mathbf{b}$, $\mathbf{s} \geq \mathbf{0}$ is an IIS if and only if there is a slack variable, say s_1 , and vector \mathbf{y} , such that the system can be solved with respect to s_1 and a set \mathbf{x}_B of basic variables as follows:

$$(1) \quad s_1 = \mathbf{y}^T \mathbf{b} / y_1 - \sum_{j=2}^n y_j s_j / y_1,$$

$$(2) \quad \mathbf{x}_B = B^{-1}(\mathbf{b} \setminus b_1) - B^{-1}N\mathbf{x}_N - B^{-1}(\mathbf{s} \setminus s_1)$$

with $\mathbf{y}^T \mathbf{b} < 0$ and $y_1, \dots, y_m > 0$.

In other words, the system of inequalities is an IIS if and only if upon solution of a standard Phase 1 LP, we have a slack variable which is negative, say s_1 , and all other slack variables have a strictly negative rate of substitution with respect to s_1 and all problem variables x_i have a 0 rate of substitution with respect to s_1 .

We demonstrate this theorem on the following example:

1. $x_2 \leq 1$
2. $x_1 - x_2 \leq 0$
3. $-x_1 - x_2 \leq -4$

We begin with the all slack basis, and will consider row 3 of our tableau as the objective:

$$\left[\begin{array}{ccccc|c} x_1 & x_2 & s_1 & s_2 & s_3 & RHS \\ \hline 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & -1 & 0 & 1 & 0 & 0 \\ \hline -1 & -1 & 0 & 0 & 1 & -4 \end{array} \right]$$

Upon pivoting x_1 and x_2 into the basis, we have the following optimal tableau:

$$\left[\begin{array}{ccccc|c} x_1 & x_2 & s_1 & s_2 & s_3 & RHS \\ \hline 0 & 1 & 1 & 0 & 0 & 1 \\ 1 & 0 & 1 & 1 & 0 & 1 \\ \hline 0 & 0 & 2 & 1 & 1 & -2 \end{array} \right]$$

In this tableau, we have $s_3 = -2 - 2s_1 - s_2 = -2$ (which satisfies (1) from the theorem), $x_1 = 1 - s_1 - s_2 = 1$, and $x_2 = 1 - s_1 = 1$ (which satisfy (2) from the theorem). Note also that $\mathbf{y} > \mathbf{0}$ so all of the conditions of Theorem 2.4 are satisfied, and thus the system is an IIS.

Van Loon strengthens this theorem by observing that at least one IIS can be obtained from a non-minimally infeasible system $A\mathbf{x} + \mathbf{s} = \mathbf{b}$, $\mathbf{s} \geq \mathbf{0}$ by examining the final Phase 1 tableau. If there exists a slack variable, say s_1 , which is negative and all problem variables x_i have a 0 rate of substitution with respect to s_1 , then an IIS can be identified within this row as follows:

Theorem 2.5 (Van Loon [36]) The subsystem, consisting of the constraints corresponding to s_1 and all slack variables having strictly negative rate of substitution with respect to s_1 , is an IIS.

Van Loon also notes that by pivoting through other bases of the Phase 1 LP other IISs can be identified.

By using IISs as a tool in LP infeasibility analysis, Van Loon provides the foundation of much of the more recent work in the area.

Gleeson and Ryan [19] derive a geometric approach to IIS identification that is based upon Farkas' Theorem of the Alternative and basic polyhedral theory. They identify an alternative system whose extreme vertices are in one to one correspondence with the IISs of the original infeasible LP. In the absence of degeneracy of the alternative system, this improves upon the work of Van Loon in that *every* pivot in the alternative system identifies a *unique* IIS in the original system.

Theorem 2.6 (Gleeson-Ryan [19]) Let $A\mathbf{x} \leq \mathbf{b}$ denote an inconsistent set of inequalities. Then the IISs are in 1-1 correspondence with the extreme points of the polyhedron $P = \{\mathbf{y} \in \Re^m \mid \mathbf{y}^T A = \mathbf{0}, \mathbf{y}^T \mathbf{b} = -1, \mathbf{y} \geq \mathbf{0}\}$. In particular, the nonzero components of any extreme point of P index an IIS.

Once a basic feasible solution to the alternative system exists, through solution of a Phase 1 LP of the alternative system for instance, Gleeson and Ryan propose the use of an algorithm due to Dyer [16] to visit all extreme points. This algorithm, in the absence of degeneracy, provides an efficient means of visiting all extreme points. In this way, all IISs of an infeasible system S can be enumerated.

Since removal of **any** constraint from an IIS yields a feasible subsystem, finding the minimum cardinality cover over all IISs will identify the **minimum** number of constraints to modify or remove from the system to attain feasibility. This minimum set of constraints will provide a localization of the modeling error/inconsistency. Thus, this approach provides a framework with which not only to isolate the infeasibility, but also to diagnose the infeasibility. To date, this approach had only been explored in the theoretical sense.

Recently, Chinneck (e.g. [13], [10], [11]) has developed a set of software tools to bring IIS isolation into practical use in LP infeasibility analysis. The goal of these algorithms is to identify a small cardinality IISs, the idea being that the smaller the constraint set the infeasibility is isolated to the easier the actual infeasibility analysis will be. These tools are now implemented in the commercial LP solvers MINOS(IIS), CPLEX, and LINDO.

In [13], Chinneck and Dravnieks develop the foundations upon which these subroutines are based. The first of these routines is a *deletion filter*. The deletion filter works by sequentially removing constraints from the LP and testing for feasibility. If the LP is feasible, the last constraint

deleted is re-inserted. This process is repeated until all constraints have been checked. The resulting system is an IIS. This algorithm can be computationally expensive, and so is used as one of the final pieces of the integrated algorithm.

The second filter is an *elastic filter*, which is based upon the idea of elastic programming. Elastic programming involves the introduction of artificial variables which “stretch” the feasible region (as in Phase 1 LP solutions). Chinneck and Dravnieks use this approach to generate a series of LPs having various constraints relaxed. By looking at which constraints have nonzero artificial variables in each LP, a subsystem containing an IIS can be identified.

The final filter is a *sensitivity filter*. This filter is based upon performing sensitivity analysis upon either a Phase 1 or elastic solution. The set of constraints having nonzero shadow prices in the final tableau of a Phase 1 LP contains at least one IIS (see [28]).

These three filtering routines can be combined in a number of ways to create integrated algorithms. Sensitivity analysis is usually used first, since most infeasibility is diagnosed upon solution of a Phase 1 LP. Subsequent operations are based upon the objective of the analyst - diagnosis of single IIS or multiple IISs. Either a combination of deletion and sensitivity filtering, or a combination of elastic and sensitivity filtering is used (or both) along with a final deletion filter to identify the IIS.

In [22], [23], [24], [25], and [26], Greenberg explores techniques for diagnosing infeasibility in linear programming models. These studies

provide demonstrations of the utility of IIS identification in infeasibility analysis by exploring the strengths and weaknesses of traditional and IIS isolation techniques on both general and specific models. In [21], Greenberg considers theoretical and computational issues in the analysis of consistency, redundancy, and implied equalities in linear programming models. He provides an analysis and extension of necessary and sufficient conditions for infeasible subsystems to be irreducible, and organizes the results in the following theorem. We denote the alternative system of S (see Theorem 2.1) as $S^A = \{\mathbf{y}^T A = 0, \mathbf{y}^T \mathbf{b} < 0, \text{ and } \mathbf{y} \geq 0\}$.

Theorem 2.7 (Greenberg [21]) If the system S is infeasible, then the following are equivalent:

- (1) $S(I)$ is an IIS.
- (2) $\text{rank}[A(I)] = |I| - 1$ and there exists a solution \mathbf{y} to S^A such that I is the support of \mathbf{y} .
- (3) I is the support of an extreme point of $\{S^A \cup \mathbf{y}^T \mathbf{b} = -1\}$.
- (4) I is the support of an extreme ray of S^A .

In the same paper, Greenberg also explores issues in identifying maximal feasible subsystems. We will address these issues as we develop our algorithm in Chapter 5.

2.3 Hypergraph Framework

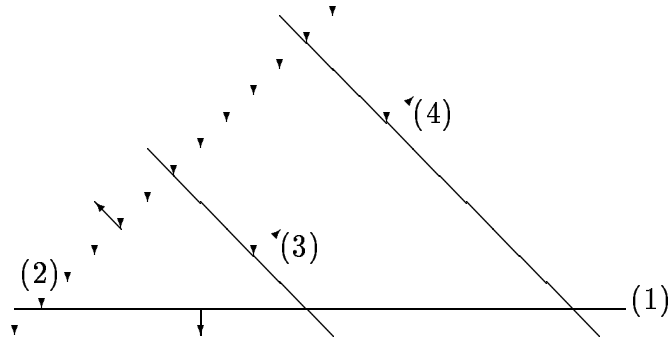
In [32] and [33], Ryan investigates the structure of the **IIS hypergraph**. A hypergraph is an n -dimensional abstraction of a graph. In a graph, we have a set of vertices and edges which connect pairs of vertices. In a hypergraph, we also have a set of vertices and edges; however, an edge is a subset of the vertices in V . We construct the IIS hypergraph

$H(V, E)$ as outlined below. Given the inconsistent system S , we let the set V index the constraints of the system, and let the elements of the set E index the the IISs of S . Thus an edge of H consists of the indices of the constraints of S which form an IIS. A transversal of H is a subset of V which intersects all edges of E . Thus, the problem of finding a minimum cardinality IIS cover is the same as finding a minimum cardinality transversal of H .

In [32], Ryan demonstrates an algorithm for identifying minimal transversals without the explicit enumeration of the edges of H . We note the following:

Lemma 2.8 (Ryan [32]) Let H be an IIS hypergraph arising from the infeasible system S . Then any minimal transversal of H corresponds to a minimal set of faces of P that intersect outside of P , and vice versa.

The heuristic Ryan presents is based upon using the simplex algorithm to find extreme points of P . Once an initial extreme point is found, some nonzero basic variable is set to 0 and this variable is included in the cover. Then, one can continue to re-solve the LP, set basic variables to 0, and update the cover until P is empty. We demonstrate that the cover obtained upon completion need not be minimal. Consider the following infeasible system for S :



$$\begin{aligned}
 1. \quad & x_2 \leq 1 \\
 2. \quad & x_1 - x_2 \leq -2 \\
 3. \quad & -x_1 - x_2 \leq -10 \\
 4. \quad & -x_1 - x_2 \leq -20 \\
 & x_1 \text{ and } x_2 \geq 0
 \end{aligned}$$

Then P is the system:

$$\begin{aligned}
 1. \quad & y_2 - y_3 - y_4 \geq 0 \\
 2. \quad & y_1 - y_2 - y_3 - y_4 \geq 0 \\
 3. \quad & y_1 - 2y_2 - 10y_3 - 20y_4 = -1 \\
 & \mathbf{y} \geq 0
 \end{aligned}$$

We find a solution to P of $y_1 = 0.2$, $y_2 = 0.1$, $y_3 = 0.1$, and $y_4 = 0.0$, yielding the IIS $\{1, 2, 3\}$. If we set $y_3 = 0$ and re-solve the LP, we find the solution $y_1 = 0.1$, $y_2 = 0.05$, $y_3 = 0$, and $y_4 = 0.05$, which yields the IIS $\{1, 2, 4\}$. If we also set $y_1 = 0$ and re-solve the LP, we find that it is now infeasible. Thus, we have identified the IIS cover $\{1, 3\}$. This is not minimal, since removal of constraint 1 from S yields a feasible system.

If H is a graph then it is 2-uniform, which means that every IIS of S has 2 elements. In this case, finding a hypergraph transversal is equivalent to finding a vertex cover. Since a 2-uniform IIS hypergraph can contain no cycle of odd length, a 2-uniform IIS hypergraph is a bipartite graph. We can find a minimum vertex cover of **any** bipartite graph in polynomial time; however, we can find a minimum vertex cover of a 2-uniform IIS hypergraph in linear time using a greedy algorithm.

Theorem 2.9 (Ryan [32]) Let H be a 2-uniform IIS hypergraph. Then the transversal obtained by successively picking the node having maximum degree among the currently uncovered edges is a minimum transversal.

General IIS hypergraphs do not share this property; however, they do generalize bipartite graphs according to the following:

Theorem 2.10 (Pulleyblank [30]) Let S be an infeasible linear system. Then S can be partitioned into two consistent subsystems.

A hypergraph is bicolored if there exists a partition of the nodes into two sets such that neither set contains an edge (all edges cross the partition). From the previous theorem, we have that an IIS hypergraph is bicolored.

In [33], Ryan details the relationship between IIS hypergraphs and other hypergraphs generalizing bipartite graphs. Ryan also presents an algorithm for finding the minimum transversal of H . This algorithm is based upon the following result:

Theorem 2.11 (Ryan [33]) Let T be any minimal transversal of an IIS hypergraph having corresponding polyhedron P . Then T is

indexed by the variables of P having positive tableau entries in some row having positive right hand side for some extreme point of P .

Ryan then provides an algorithm that in the absence of degeneracy in P , enumerates all minimal transversals in polynomial time with respect to the size of H , and so the minimum cardinality transversal can be found in time polynomial in the size of H . However, we should remember that the size of H may be exponential in the size of the original infeasible system S , since S may yield an exponential number of IISs.

Theorem 2.11 does provide an algorithm for identifying IIS covers (though they need not be minimal).

2.4 Techniques in Approximating the MWFS

We may wonder what other techniques are available for identifying a maximum cardinality feasible subsystem or a maximum weight feasible subsystem. This problem has been examined before (e.g. [26]). A standard technique is to convert the standard Phase 1 LP (note that we are including non-negativity constraints in this formulation):

$$\begin{array}{ll}
 \min & \sum_{i=1}^m s'_i \\
 \text{subject to} & \\
 & A\mathbf{x} + \mathbf{s} - \mathbf{s}' = \mathbf{b} \\
 & \mathbf{x}, \mathbf{s}, \mathbf{s}' \geq \mathbf{0}
 \end{array}$$

into a fixed charge problem:

$$\begin{aligned}
\min \quad & \sum_{i=1}^m z_i \\
\text{subject to} \quad & \\
& A\mathbf{x} + \mathbf{s} - \mathbf{s}' = \mathbf{b} \\
& \mathbf{s}' \leq M\mathbf{z} \quad \text{for constant } M \text{ sufficiently large} \\
& \mathbf{x}, \mathbf{s}, \mathbf{s}' \geq \mathbf{0} \\
& \mathbf{z} \text{ binary}
\end{aligned}$$

This fixed charge problem will minimize the number of violated constraints, and removal of this set of constraints yields a maximum cardinality feasible subsystem (see [26]). This problem is extremely difficult to solve, and a comparison with our algorithm is given in Chapter 5. There are many difficulties in implementing this formulation. If the constant M is not large enough, then the problem will still be infeasible. In addition, if the value of M is too large, then a large number of fractional solutions can be introduced into the LP relaxation. A large value for M can also cause stability problems as the z_i variables can take on values below the software tolerance for integers.

We may also look at the standard Phase 1 solution to approximate the maximum cardinality feasible subsystem. Using the Phase 1 formulation from above, we make the following observation about **any** Phase 1 solution:

Lemma 2.12 The constraints having artificial variable nonzero at the completion of Phase 1 comprise a set of functional constraints whose removal from the infeasible system $S = A\mathbf{x} \leq \mathbf{b}$ yields a feasible system. Further, this set of constraints is an IIS cover.

Proof: Let the set I consist of the constraints having artificial variable nonzero at the completion of Phase 1. For the Phase 1 solution \mathbf{x}^* , $S(I)$ is the subsystem consisting of those $i \in M$ with $A_i\mathbf{x}^* > b_i$, which is infeasible. All other constraints are of the form $A_i\mathbf{x}^* \leq b_i \forall i \notin I$. Thus removal of these constraints yields a feasible subsystem, and \mathbf{x}^* is a feasible solution. Further, since removal of all constraints in I restores feasibility in S , all IISs of S must have an element in I . Thus I also denotes an IIS cover. ■

How good a bound on the minimum cardinality IIS cover will the Phase 1 solution be? We can demonstrate that in general, we might not even obtain a **minimal** cardinality IIS cover from the Phase 1 solution. To illustrate, we give the following example.

Let S be the following system, which is depicted in Figure 2.1:

$$\begin{array}{rcl}
 1. & & x_2 \leq 1 \\
 2. & x_1 - & x_2 \leq -2 \\
 3. & -x_1 - & x_2 \leq -10 \\
 4. & -x_1 - & x_2 \leq -20 \\
 & & x_1 \text{ and } x_2 \geq 0
 \end{array}$$

We then add the artificial variables a_2 , a_3 , and a_4 to constraints 2, 3, and 4, obtaining the following Phase 1 LP:

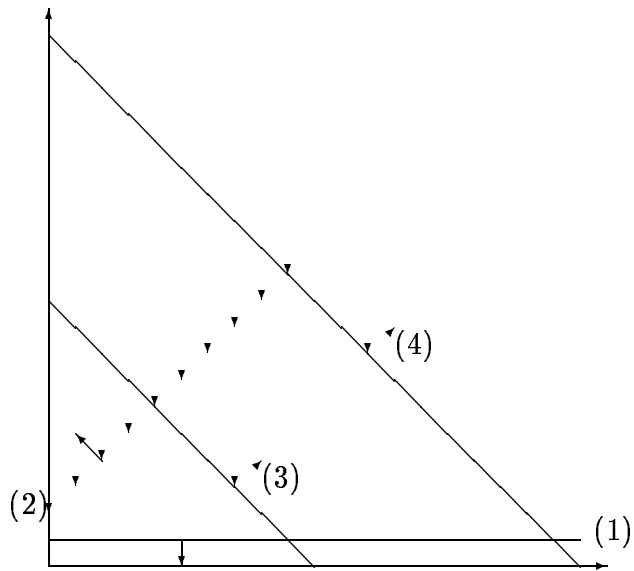


Figure 2.1. Infeasible system demonstrating Phase 1 solution does not necessarily provide a minimal IIS cover

$$\begin{aligned}
\min \quad & a_2 + a_3 + a_4 \\
1. \quad & x_2 + a_2 \leq 1 \\
2. \quad & x_1 - x_2 + a_2 = -2 \\
3. \quad & -x_1 - x_2 + a_3 = -10 \\
4. \quad & -x_1 - x_2 + a_4 = -20 \\
& x_1 \text{ and } x_2 \geq 0
\end{aligned}$$

An optimal solution to this LP is $x_1 = 9$, $x_2 = 1$, $a_2 = -10$, and $a_4 = -10$. Thus, removal of constraints 2 and 4 yields a feasible subsystem of S . We note that this is not a minimal set of constraints to remove, since removal of constraint 2 yields a subsystem satisfied by $x_1 = 19$ and $x_2 = 1$. Thus, we cannot guarantee that a Phase 1 LP solution will provide even a minimal IIS cover.

Recently, Chinneck [12] has developed a heuristic for identifying the MCIC which is based on recognizing this idea. The algorithm uses a greedy approach to build the cover, where the constraint affecting the Phase 1 objective the most is removed at a particular step of the algorithm. This process continues until the Phase 1 LP yields a feasible solution to the original problem, at which time the set of constraints removed from the problem corresponds to an IIS cover. This approach does not guarantee even a **minimal** cover upon completion. A comparison of this approach and an algorithm for solving the MCIC problem exactly is given in Chapter 5.

A Phase 1 LP can be viewed as the introduction of elastic variables to allow the “stretching” of each constraint. The objective is to then minimize the sum of the “stretching” over each constraint. By introducing

elastic variables for **each** constraint, rather than just those with negative RHS, and for each variable bound, can we then guarantee a minimal IIS cover as a solution to the “elastic” LP? Again, by a similar example, the answer is no:

Let S be the following system, depicted in Figure 2.2:

$$\begin{aligned}
 1. \quad & x_2 \leq 1 \\
 2. \quad & x_2 \leq 0.5 \\
 3. \quad & x_1 - x_2 \leq -2 \\
 4. \quad & -x_1 - x_2 \leq -10 \\
 5. \quad & -x_1 - x_2 \leq -20
 \end{aligned}$$

$$x_1 \text{ and } x_2 \geq 0$$

We then add the elastic variables $e_1, e_2, e_3, e_4, e_5, e_{x_1}$, and e_{x_2} to constraints 1, 2, 3, 4, and 5, and to the non-negativity bounds on x_1 and x_2 , obtaining the following elastic LP:

$$\begin{aligned}
 \min \quad & \sum_1^{x_2} e_i \\
 1. \quad & x_2 - e_1 \leq 1 \\
 2. \quad & x_2 - e_2 \leq 0.5 \\
 3. \quad & x_1 - x_2 - e_3 \leq -2 \\
 4. \quad & -x_1 - x_2 - e_4 \leq -10 \\
 5. \quad & -x_1 - x_2 - e_5 \leq -20 \\
 6. \quad & x_1 + e_{x_1} \geq 0 \\
 7. \quad & x_2 + e_{x_2} \geq 0
 \end{aligned}$$

An optimal solution to this LP is $x_1 = 4.25, x_2 = 5.75, e_1 = 4.75, e_2 = 5.25, e_3 = 0.5$, and $e_5 = 10$. Thus, removal of constraints 1, 2, 3,

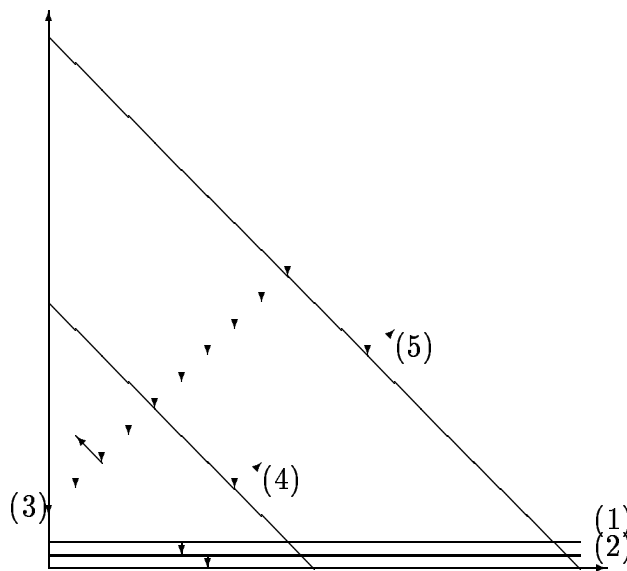


Figure 2.2. Infeasible system demonstrating full elastic Phase 1 solution does not necessarily provide a minimal IIS cover

and 5 yields a feasible subsystem of S . We note that this set is not minimal, since removal of constraints 1 and 2 yields a subsystem satisfied by $x_1 = 19.5$ and $x_2 = 0.5$. Thus, we cannot guarantee that a full elastic LP solution will provide even a minimal IIS cover.

So we obtain an IIS cover simply by solving the Phase 1 LP. In [12] Chinneck suggests that solving a full elastic programming LP may find a smaller cardinality cover. We have answered this question by demonstrating that by modifying our Phase 1 problem formulation to a full elastic programming LP, we still cannot guarantee a minimal cover; and in general the cover found by either of these techniques will not be of the same cardinality as the minimum cardinality IIS cover. This is not surprising, given the following results. In [9], Chakravarti proves that finding the maximum cardinality feasible subsystem of the infeasible system S is NP-hard.

Theorem 2.13 (Chakravarti [9]) The problem of finding the maximum cardinality feasible subsystem of the infeasible system S is NP-hard.

Chakravarti also demonstrates that there can exist an exponential number of IISs in an infeasible system by exhibiting a system having $2n + 1$ constraints, in n variables, with 2^n IISs.

Theorem 2.14 (Chakravarti II [9]) An infeasible system of constraints may possess exponentially many IISs.

Further analysis of the complexity of the problem is provided by Sankaran [34].

Theorem 2.15 (Sankaran [34]) The problem of finding the maximum cardinality feasible subsystem of the infeasible system S is NP-hard even when A is totally unimodular and \mathbf{b} is integer.

Sankaran proves the existence of a class of infeasible systems for which the maximum cardinality feasible subsystem can be found in polynomial time.

Theorem 2.16 (Sankaran [34]) If the augmented matrix $[A|\mathbf{b}]$ is totally unimodular, then the constraints having artificial variable nonzero at the completion of Phase 1 comprise a minimum cardinality set of functional constraints whose removal from the infeasible system $S = \mathbf{Ax} \leq \mathbf{b}$ yields a feasible system.

Thus, the Phase 1 LP solution will yield a maximum cardinality feasible subsystem for certain classes of LPs.

In the special case when only a single artificial or elastic variable has nonzero value at the conclusion of Phase 1, we note the following:

Lemma 2.17 If, at the conclusion of Phase 1, only a single artificial variable is nonzero, then the constraint corresponding to that artificial variable is a minimum cardinality IIS cover.

Proof: By Lemma 2.12, we have that the constraint corresponding to the nonzero artificial variable is an IIS cover. Since the system is infeasible, we have $0 < |\text{minimum IIS cover}| \leq 1$, so we have a minimum cardinality IIS cover. ■

Amaldi [1] and Amaldi and Kann [2], [3] also examine the complexity and approximability of both finding the maximum feasible subsystem and finding the minimum set of constraints to remove to attain

feasibility. While the two problems are equivalent in terms of complexity when solving optimally, they differ in complexity of approximation. We illustrate this with the following example. Suppose we are given an infeasible system having 2000 constraints with a MCIC consisting of 20 elements. If we find a set of 30 constraints whose removal from the system yields a feasible system, we have approximated the maximum cardinality feasible subsystem to within $(2000 - 30)/2000 = .985$; however, we have approximated the minimum set of constraints to remove in order to attain feasibility to within $(30 - 20)/20 = .5$. Thus, as noted by Amaldi, the second problem is more difficult to approximate due to its scale.

In particular, they prove the following (ignoring the trivial solution for the homogeneous case). APX denotes the class of problems in NP which can be approximated in polynomial time within some constant factor. MAX SNP-hard problems are a subclass of the APX class which are characterized by their approximability. MAX SNP-hard problems can be approximated in polynomial time within some constant, but there exists some constant $c > 1$ such that finding an approximate solution within c is NP hard. Thus, they can be approximated within some constant, but not all constants, in polynomial time.

Theorem 2.18 (Amaldi and Kann [2]) The problem of finding the maximum feasible subsystem of an infeasible system S having either $\{= \text{ or } \leq\}$ relations is MAX SNP-hard even when restricted to homogeneous systems with integer coefficients and no pairs of identical relations.

In other words, the maximum feasible subsystem can be approximated within a constant but not within every constant unless $P = NP$. Amaldi and Kann further define the approximability of the maximum feasible subsystem problem for infeasible systems S having equality constraints:

Theorem 2.19 (Amaldi and Kann [2]) Unless $P = NP$, there is a constant $\epsilon > 0$ such that the maximum feasible subsystem of the infeasible system $A\mathbf{x} = \mathbf{b}$ can not be approximated within m^ϵ , where m is the number of rows of A .

In the case of an infeasible system S having \leq constraints, Amaldi and Kann demonstrate that it is much easier to approximate the maximum feasible subsystem.

Theorem 2.20 (Amaldi and Kann [2]) The maximum feasible subsystem of the infeasible system $A\mathbf{x} \leq \mathbf{b}$ can be approximated in polynomial time within a factor of 2.

The proof for this result is constructive, and a polynomial time algorithm is provided which guarantees a 2-approximation.

If we restrict ourselves to identifying the MCIC, Amaldi and Kann present the following. $DTIME(T(n))$ denotes the class of problems which can be solved in time $T(n)$, where n is the size of the input.

Theorem 2.21 (Amaldi and Kann [3]) The problem of identifying the MCIC of the infeasible linear system $A\mathbf{x} \leq \mathbf{b}$ cannot be approximated within any constant unless $P = NP$ and within $c \log n$ for any $c < 1/8$ unless $NP \subseteq DTIME(n^{\log \log n})$, even when restricted to homogeneous systems with ternary coefficients in $\{-1, 0, 1\}$.

We know that certain cases can either be solved optimally in polynomial time, or approximated within a factor of 2 in polynomial time. What other bounds can we determine for the cardinality of the maximum feasible subsystem? We begin by making the following observations about the infeasible system $A\mathbf{x} \leq \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$.

If all $b_i \geq 0$, for $i = 1, m$, then the trivial solution $x_j = 0$, for $j = 1, n$ satisfies the system. Thus, in forming the Phase 1 LP, we need only add artificial variables to those rows having a negative right hand side and we can make the following observation.

Lemma 2.22 The cardinality of the minimum cardinality IIS cover will be less than or equal to the number of constraints with negative right hand side.

Proof: If we have k negative b_i 's, then we need k artificial variables for the Phase 1 LP. At most, all k artificial variables will be nonzero at the conclusion of Phase 1. Since removal of the constraints corresponding to these k artificial variables yields a feasible subsystem (the 0 solution satisfies the remaining constraints), the minimum cardinality IIS cover will be of cardinality equal to or less than k . ■

3. Extensions and Unifications of IIS Results

In this chapter, we will look beyond the surface theory at the strength behind several results and also explore the relationships among them.

3.1 Examination of Van Loon's Method

Several researchers have examined Van Loon's results (see [13], [19], [26]). A number of shortcomings of the method have been pointed out - the chief among these being that non-negativity constraints must be explicitly included in the formulation in order to obtain IISs. In this section, we will examine more closely the implications of Van Loon's result in an effort to illuminate a more general base theory which can be extracted.

Van Loon's method as described in Theorem 2.4 and Theorem 2.5 requires the explicit inclusion of all non-negativity constraints as functional constraints in the system $A\mathbf{x} \leq \mathbf{b}$; however, exclusion of non-negativity constraints leads to the identification of those IISFs which are IISs (if any exist); which is only a subset of all IISs of the system.

For example, consider the following infeasible system, displayed in Figure 3.1:

$$\begin{aligned} 1. \quad & x_1 + x_2 \leq 2 \\ 2. \quad & -x_1 \leq -3 \\ 3. \quad & -2x_1 - 3x_2 \leq -8 \\ & x_1, x_2 \geq 0 \end{aligned}$$

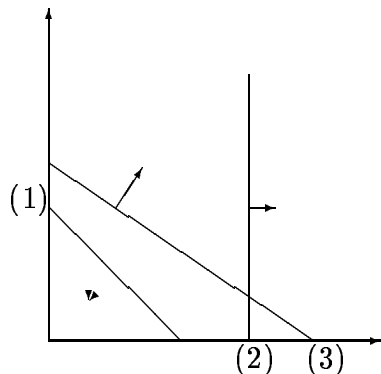


Figure 3.1: Infeasible system for example of Van Loon's method

If only the functional constraints are included in Van Loon's formulation (and x_1 and x_2 free), then we can pivot both x_1 and x_2 into the basis yielding the following tableau:

	x_1	x_2	s_1	s_2	s_3	RHS
x_1	1	0	3	0	1	-2
s_2	0	0	3	1	1	-5
x_2	0	1	-2	0	-1	4
	0	0	3	1	1	-5

Thus, using Theorem 2.5, we obtain the IIS $\{1, 2, 3\}$ (the notation meaning that constraints 1, 2, and 3 form an IIS). Pivoting to all other bases containing x_1 and x_2 yields no other IISs. By using Van Loon's theorem without explicitly including the non-negativity constraints in the functional constraints, we have identified all IISs which are IISFs. It is easy to demonstrate that we will find all IISs which are IISFs in this way; however, we may have to visit all bases in order to guarantee doing so. To

see why we will find all IISs which are IISFs, note that an IISF which is an IIS of $A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$ is still an IIS of the more general system $A\mathbf{x} \leq \mathbf{b}, \mathbf{x}$ free. Thus, by Theorem 2.5 we can identify such an IIS. Since we can find all IISs of the more general system by pivoting through its bases, we can find the subset of IISs which are IISFs of the original system. It is easily seen that we have other IISs in the system which rely on the non-negativity of x_1 or x_2 , for example $\{1, 3, x_1 \geq 0\}$ and $\{1, 2, x_2 \geq 0\}$. Using this restricted tableau, we cannot identify these IISs unless we modify Van Loon's method. We will look at modifications for doing this later. For now, we concentrate on determining when we can identify IISs without explicitly including non-negativity constraints in our model formulation.

Given an infeasible system $A\mathbf{x} \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}$, there exist certain conditions under which we know that a source of the infeasibility will lie strictly within the set of functional constraints. In other words, the system $A\mathbf{x} \leq \mathbf{b}$ has **no** solutions, let alone a non-negative solution. Murty [28] shows in general that an IIS is contained within the set of variables (problem and slack) having nonzero reduced cost upon completion of Phase 1. Further, at the conclusion of a standard Phase 1, if the reduced costs of all problem variables are 0 and the artificial objective is positive, then the system $A\mathbf{x} \leq \mathbf{b}$ is infeasible, and the source of the infeasibility is among the constraints corresponding to the slack variables having nonzero reduced costs in the final tableau. Thus, we know that at least one IISF which is itself an IIS lies among these constraints (see [28]).

We again consider the simple example of Figure 3.1.

$$\begin{aligned}
 1. \quad & x_1 + x_2 \leq 2 \\
 2. \quad & -x_1 \leq -3 \\
 3. \quad & -2x_1 - 3x_2 \leq -8 \\
 & x_1, x_2 \geq 0
 \end{aligned}$$

At the conclusion of the Phase 1 LP, we have the following tableau:

	x_1	x_2	s_1	s_2	s_3	RHS
x_1	1	1	1	0	0	2
s_2	0	1	1	1	0	-1
s_3	0	-1	2	0	1	-4
	0	0	3	1	1	-5

The reduced cost for problem variables x_1 and x_2 is 0, and all 3 slack variables have nonzero reduced cost. Thus, we have an IISF among the constraints $\{1,2,3\}$ - in fact, in this case the set corresponds exactly to an IISF.

Otherwise, if the reduced cost of at least 1 variable is nonzero, then $Ax \leq b$ *could* have a solution, but not a non-negative solution. Consider the following modification of the previous example:

$$\begin{aligned}
 1. \quad & x_1 + x_2 \leq 2 \\
 2. \quad & -x_1 \leq -3 \\
 & x_1, x_2 \geq 0
 \end{aligned}$$

We note that $x_1 = 3$ and $x_2 = -1$ is a solution to the system defined by (1) and (2) (thus a solution exists, but no non-negative solution exists).

This system has the following final Phase 1 tableau:

	x_1	x_2	s_1	s_2	RHS
x_1	1	1	1	0	2
s_2	0	1	1	1	-1
	0	1	1	1	-1

From Murty [28], we see that an IIS exists among the constraints $\{1, 2, x_2 \geq 0\}$ - again in this case, the set gives us an IIS, although it will not in general.

Looking beyond what Van Loon explicitly states in his paper, we see how to use his ideas in a more general setting. Suppose we are working with the infeasible system $A\mathbf{x} \leq \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$, and we choose **not** to explicitly include non-negativity constraints as functional constraints. Again, let us use the example of Figure 3.1:

$$\begin{aligned}
 1. \quad & x_1 + x_2 \leq 2 \\
 2. \quad & -x_1 \leq -3 \\
 3. \quad & -2x_1 - 3x_2 \leq -8 \\
 & x_1, x_2 \geq 0
 \end{aligned}$$

At the conclusion of the Phase 1 LP, we have the tableau depicted in Figure 3.2.

This is an optimal basis in the sense that all reduced costs are non-negative. Note that we are unable to pivot x_2 into the basis without pivoting x_1 out of the basis. Since we have a 1 in the (s_2, x_2) position of the tableau, we cannot use Theorem 2.5 to find an IIS in that row. We cannot identify an IIS in row s_3 because of the -1 in column x_2 . We

	x_1	x_2	s_1	s_2	s_3	RHS
x_1	1	1	1	0	0	2
s_2	0	1	1	1	0	-1
s_3	0	-1	2	0	1	-4
	0	0	3	1	1	-5

Figure 3.2. Final Tableau, Not Including Non-negativity in Functional Constraints

now compare this basis with that obtained from the following equivalent system:

1. $x_1 + x_2 \leq 2$
2. $-x_1 \leq -3$
3. $-2x_1 - 3x_2 \leq -8$
4. $x_1 \geq 0$
5. $x_2 \geq 0$

x_1, x_2 free

	x_1	x_2	s_1	s_2	s_3	s_4	s_5	RHS
x_1	1	1	1	0	0	0	0	2
s_2	0	1	1	1	0	0	0	-1
s_3	0	-1	2	0	1	0	0	-4
s_4	0	1	1	0	0	1	0	2
s_5	0	-1	0	0	0	0	1	0
	0	0	3	1	1	0	0	-5

From the all slack initial basis, we have pivoted in x_1 and pivoted out s_1 - the same pivot performed to obtain the tableau of Figure 3.2. As opposed to the tableau of Figure 3.2, we are able to pivot x_2 into the basis without pivoting out x_1 . The additional slack variable on the constraint $x_1 \geq 0$, s_4 , can be pivoted out of the basis. Our goal is to have a basis

similar to that in Figure 3.2, so we wish for x_2 to be basic with activity level 0. We force this to occur by pivoting in s_4 and pivoting out s_5 ; x_2 and s_5 **must** have the same activity level, so if one is not in the basis, the other must be 0 also.

	x_1	x_2	s_1	s_2	s_3	s_4	s_5	<i>RHS</i>
x_1	1	0	1	0	0	0	1	2
s_2	0	0	1	1	0	0	1	-1
s_3	0	0	2	0	1	0	-1	-4
x_2	0	1	0	0	0	0	-1	0
s_4	0	0	1	0	0	1	1	2
	0	0	3	1	1	0	0	-5

Figure 3.3. Final Tableau, Including Non-negativity in Functional Constraints

From row s_2 of the tableau of Figure 3.3, we obtain the IIS $\{s_1, s_2, s_5\}$. Looking back at row s_2 of the tableau in Figure 3.2, we see that it is identical, save that we have a 1 in the x_2 column rather than a 1 in the s_5 column (this column does not exist without explicitly including non-negativity constraints in the formulation). The key is to recognize that $x_2 - s_5 = 0 \rightarrow x_2 = s_5$ and thus it is irrelevant whether or not we include $x_2 \geq 0$ explicitly in the problem formulation. We have the following generalization of Theorem 2.5:

Theorem 3.1 (Extension of Van Loon's Theorem) Given the infeasible system $S = \{Ax \leq \mathbf{b}, \mathbf{x} \geq \mathbf{0}\}$, suppose there exists a Phase 1 tableau that satisfies the following:

- (1) a slack variable, s_i , which is negative, and basic in row i of the current tableau
- (2) positive value in the column (or columns) of one (several) x_j in row i
- (3) all x_j have 0 reduced cost

(4) all other slack variables have non-negative tableau entries in row i .

Then, the variables having positive tableau entries in row i index an IIS of S .

Proof: We note that if in the tableau row i described above there are no problem variables having nonzero entries, then the hypothesis holds trivially by Theorem 2.5. Otherwise, without loss of generality, assume that some problem variable, say x_1 has a positive tableau entry in row i . Since slack variable s_i is basic in row i and x_1 has a nonzero entry in row i , we know that x_1 is nonbasic and so $x_1 = 0$ and further the reduced cost for x_1 is also 0. Now consider the system where we include the constraints $-x_j + s_{x_j} = 0$ in the formulation. Here s_{x_j} represents the slack variable on the non-negativity of x_j (so $x_j = s_{x_j}$). We can add these constraints to the current tableau, making each newly added slack variable s_{x_j} basic. We denote the row added for the non-negativity of x_j as row $m + j$. Since each x_j is non-negative in the current tableau, the modified system will still be “feasible” (in the sense that only slack variables have negative value). The only modification to row i during this process is the addition of a 0 in the column for each new slack variable s_{x_j} . We wish to pivot x_1 into the basis, but leave its value at 0. Thus, we can pivot x_1 into the basis, and pivot s_{x_j} out of the basis. If we view the tableau columns for x_1 (prior to pivoting it into the basis) and s_{x_1} (after pivoting it out of the basis) as being the rate at which the basic variables change per unit change in the value of x_1 (resp. s_{x_1}), then since $x_1 = s_{x_1}$, we must have that the tableau column of nonbasic variable x_1 prior to pivoting it into the basis is identical to the tableau column of nonbasic variable s_{x_1} upon completion of the pivot.

In particular, we note that upon pivoting x_1 into the basis, there will be a 0 in the x_1 column of row i , and a nonzero value in the s_{x_1} column of row i . We must only demonstrate that we have modified no other entries in row i of the tableau by this pivot. Row $m + 1$ will have a 0 in every column excluding column s_{x_1} , which has a 1, and column x_1 , which has a -1. Because of this, when pivoting x_1 into the basis and pivoting s_{x_1} out of the basis, we do not affect any columns of row i excepting column x_1 and the previously discussed column s_{x_1} . We can repeat this argument for any such variable x_j having a positive tableau entry in row i . ■

3.2 Van Loon's Method and the Gleeson and Ryan Method

Let us look closer at the method of Gleeson and Ryan [19]. We note that the polyhedron P of Theorem 2.6 is a bounded form of the cone of the alternative system (2) from Theorem 2.1. The following form of Theorem 2.6 is also implied in [19]:

Theorem 3.2 (Conical version of Gleeson-Ryan Theorem)

Let $A\mathbf{x} \leq \mathbf{b}$ denote an inconsistent set of inequalities. Then the IISs are in 1-1 correspondence with the extreme rays of the cone

$$P' = \{\mathbf{y} \in \Re^m \mid \mathbf{y}^T A = 0, \mathbf{y}^T \mathbf{b} < 0, \mathbf{y} \geq 0\}.$$

In particular, the nonzero components of any extreme ray of P' index an IIS.

Thus, in the conical version of the Gleeson-Ryan Method, we are looking for the supports of extreme rays of the alternative system defined by Farkas' Theorem. The alternative system is simply the dual of our original (primal) system with the primal objective function zeroed out.

Thus, by ignoring the objective function, the Phase 1 solution of S can be used to identify an extreme ray of the unbounded dual system, or an IIS of the original system. In other words, we can identify one (or more) IISs with no more work than solving Phase 1 of our original system. In essence, this generalizes the result of Van Loon [36]. Assume that we are working with an infeasible system $A\mathbf{x} \leq \mathbf{b}$, where any non-negativity constraints are considered for this discussion to be included. Equivalently, we have the infeasible system $A\mathbf{x} + \mathbf{s} = \mathbf{b}$, where \mathbf{s} is a vector of slack variables. Consider the following LP:

$$\begin{aligned} \max \quad & \mathbf{c}^T \mathbf{x} \\ \text{subject to} \quad & \\ & A\mathbf{x} + \mathbf{s} = \mathbf{b} \\ & \mathbf{s} \geq 0 \end{aligned}$$

Suppose that at the termination of Phase 1 for the infeasible system we have a basic slack variable, say s_1 , which is strictly less than 0. Additionally, if the Phase 1 tableau entries for all nonbasic variables x_j are 0 and for all nonbasic slack variables s_i are non-negative in the row corresponding to s_1 , then in Theorem 2.5 Van Loon proves that the basic slack variable s_1 and all slack variables having negative rate of substitution with respect to s_1 form an IIS.

Now, from Theorem 3.2, we have that this same set of constraints which forms an IIS of P must be the supports of an extreme ray of the alternative cone P' . In other words, Van Loon's algorithm, without explicitly identifying it as such, finds extreme rays of the alternative system by pivoting through bases of the original infeasible system looking for tableau

rows satisfying the above.

This relationship should not be surprising, given the form of a solution row in Van Loon's method. We have a row with the following form:

$$[+ + \dots + 00 \dots 0 | -].$$

Since all reduced costs are 0 or positive, there exist no pivot which will change the sign of the RHS - if we increase any nonbasic variable from 0, we decrease the value of the RHS in this row. If we think in terms of a dual pivot, if we try to pivot this row out of the basis, there is no entering variable - we can increase unblocked and so we have an extreme ray.

4. Facial Structure of the IIS Cover Polyhedron

In this chapter, we derive some facets for the IIS covering problem. We will begin by introducing some preliminary concepts and previous results.

4.1 Dimension of the Min Weight IIS Covering Polytope

A few preliminaries will be presented first. We say that the **cardinality** of an IIS is the number of constraints and bounds contained in the IIS. In some instances, we will wish to speak of the **row-cardinality** of an IIS. This refers to the number of functional constraints contained in the IIS. For the remainder of this chapter, we assume, without loss of generality, that the infeasible system $A\mathbf{x} \leq \mathbf{b}$ has no row i of A with $a_{ij} = 0$ for all j and $b_i < 0$. If such a row exists, it can be preprocessed out of the system and dealt with separately. We immediately make the following observation which provides a lower bound on the size of an IIS from a system containing no trivial inequalities.

Observation 4.1 Given an infeasible system $A\mathbf{x} \leq \mathbf{b}$, every IIS of this system will be of cardinality ≥ 2 .

Recall that an inequality $\mathbf{A}_i\mathbf{x} \leq b_i$ is an **implicit equality** in the system $A\mathbf{x} \leq \mathbf{b}$ if $\mathbf{A}_i\mathbf{x} = b_i$ for all \mathbf{x} satisfying $A\mathbf{x} \leq \mathbf{b}$. We can thus partition the constraints of $A\mathbf{x} \leq \mathbf{b}$ into the following:

- (1) $A^=\mathbf{x} \leq \mathbf{b}^=$ is the system of implicit equalities in $A\mathbf{x} \leq \mathbf{b}$.
- (2) $A^<\mathbf{x} \leq \mathbf{b}^<$ is the system of all other inequalities in $A\mathbf{x} \leq \mathbf{b}$.

We also make use of the following well-known theorem (see [35] for example):

Theorem 4.2 ([35]) The dimension of the non-empty polyhedron $Q = \{\mathbf{x} \in \mathfrak{R}^n \mid A\mathbf{x} \leq \mathbf{b}\}$ is equal to $n - \text{rank}(A^=)$.

Using these results, as well as the fact that each IIS has cardinality at least 2, we can derive the dimensionality of the IIS covering polytope.

Theorem 4.3 Given an infeasible system S and the polytope $C = \{\mathbf{z} \in B^n \mid D\mathbf{z} \geq \mathbf{1}\}$ constructed such that each row i of D is the support vector of an IIS of S , then the dimension of C ($\dim(C)$) is n .

Proof: Since a valid solution can be an overcovering, we can set all z_i 's to be 1. From Observation 4.1, we have that every IIS has at least 2 elements, which implies that each row i of D has at least 2 nonzero d_{ij} 's. This implies that no constraint in our problem is implicit for all solutions, and so the rank of $D^=$ is 0. Thus the dimension of C is n . ■

A matrix A is called a **clutter** if the support of every row is not properly contained in the support of any other row. Since the support of each row of the 0 – 1 coefficient matrix of the MWIC problem consists of the indices of an IIS of S , and since each IIS is minimal by definition, we have that the coefficient matrix of the MWIC problem is a clutter.

We now examine known results on facets of general set covering polytopes.

4.2 Known Properties of the General Set Covering Polytope

Within this section, we will discuss the general set covering polytope defined as:

$$GSCP(D) = \text{conv}\{\mathbf{x} \in B^n \mid D\mathbf{x} \geq \mathbf{1}\}$$

Among the basic facts known about the polyhedron $GSCP(D)$ are the following theorems (see [4] for example).

Theorem 4.4 (GSCP Dimensionality [4]) The polyhedron $GSCP(D)$ is full dimensional if and only if D is a clutter and each row of D has 2 or more nonzero values.

As we have seen from Theorem 4.3, the MWIC polyhedron meets the conditions for full dimensionality. The remaining results in this section depend upon the dimensionality of the general set covering polytope. Thus, for the remainder of this section, we shall assume that the polyhedron $GSCP(D)$ is full dimensional. In general, this need not be true, however; if a row of D has only a single nonzero element or if a row of D is properly contained in another row of D , we can preprocess the covering problem to yield a subproblem which is full dimensional. In particular, the IIS covering problem is full dimensional.

Theorem 4.5 ([4]) All inequalities of the form $x_j \leq 1$ define facets of the polyhedron $GSCP(D)$

Thus, the variable upper bounds for the MWIC polyhedron are facet defining.

Theorem 4.6 ([4]) The inequality $x_j \geq 0$ defines a facet of the polyhedron $GSCP(D)$ if and only if the number of nonzero elements in row i not including the j^{th} element is at least 2 for all rows i of D .

Balas and Ng [4] provide necessary and sufficient conditions for a standard covering constraint to be a facet defining inequality of the general set covering polytope. We use the following notation. N^i is the set of nonzero coefficients in row i of D , and M is the row index set.

Theorem 4.7 (Balas-Ng [4]) The inequality

$\sum(x_j \mid j \in N^i) \geq 1$ defines a facet of $GSCP(D)$ if and only if

- (1) there exists no $k \in M$ with $N^k \subset N^i$; and
- (2) for each $k \in N \setminus N^i$, there exists $j(k) \in N^i$ such that $d_{hj(k)} = 1$ for all $h \in M^0(k)$, where $M^0(k) = \{h \in M \mid d_{hk} = 1 \text{ and } d_{hj} = 0, \text{ for all } j \in N \setminus N^i \cup \{k\}\}$

They prove that a covering inequality i is a facet of the general set covering polytope if and only if it is not properly contained in another row and for every 0 element k in row i there exists a column $j(k)$ having a 1 in row i and a 1 in every row that has a 1 in column k and a 0 in all other columns that row i has a 0 in.

We examine the implications of this result on a general set covering instance. Below, we have the coefficient matrix D of a general set covering problem:

	x_1	x_2	x_3	x_4	x_5	x_6
(1)	1	1	0	0	1	1
(2)	0	1	1	0	0	1
(3)	0	1	1	0	1	0
(4)	1	0	1	1	1	1

We observe that row (1) of this problem is a facet. We first note that row (1) is not properly contained in any other row of D , and thus row

(1) satisfies condition (1) of Theorem 4.7. We consider all columns where row (1) has 0's to show that condition (2) is satisfied. First, we consider column x_3 . Rows (2) and (3) have a 1 in column x_3 and 0's in all other columns where row (1) has a 0 (column x_4 is the only other column row (1) has a 0). These two rows define the set $M^0(x_3)$. We have that rows (1), (2), and (3) all have a 1 in column x_2 , thus $j(x_3) = x_2$ and condition (2) is satisfied for column x_3 . This condition is satisfied trivially for column x_4 since there are no rows having a 1 in column x_4 and a zero in column x_3 (so $M^0(x_4)$ is empty). Thus, row (1) forms a facet defining constraint of the general set covering problem $D\mathbf{x} \geq \mathbf{1}$. Similarly, we see that rows (2) and (3) are also facet defining; however, row (4) is not. Condition (1) of Theorem 4.7 holds for this row, but condition (2) fails. The set $M^0(x_2)$ contains rows (1), (2), and (3). However, we are unable to find a column $j(x_2)$ having 1's in all four rows and so condition (2) fails.

4.3 Properties of the Min Weight IIS Covering Polytope

We can interpret the meaning of Theorem 4.7 within the context of our infeasible system. Let the original infeasible system of linear inequalities be denoted by S . Let P be the polyhedron defined by the feasible alternative system given in Theorem 2.6,

$$P = \{\mathbf{y} \in \Re^m \mid \mathbf{y}^T A = 0, \mathbf{y}^T \mathbf{b} = -1, \mathbf{y} \geq 0\}$$

and let C denote the MWIC polyhedron

$$C = \{\mathbf{z} \in B^m \mid \sum_{i \in I_j} z_i \geq 1 \text{ for all } j\},$$

where $I_j = j^{\text{th}}$ IIS.

Note that we have the following correspondence between C and P : covering constraints of C correspond to the supports of the extreme points of P , and variables of C correspond to faces of P .

We will begin by proving the following general lemma.

Lemma 4.8 The support of an extreme point p of any polyhedron $A\mathbf{x} \leq \mathbf{b}$, $\mathbf{x} \geq \mathbf{0}$ consists of those faces of the polyhedron on which p does not lie.

Proof: The support of an extreme point p is the set of nonzero variables in an optimal basis for p . If $x_i \neq 0$, then the constraint $x_i \geq 0$ is not tight; and so p does not lie on the face $x_i = 0$. Similarly, if $x_i = 0$, then the constraint $x_i \geq 0$ is tight; and so p lies on the face $x_i = 0$. An identical argument for slack variables concludes the proof. ■

The following corollary is a translation of Theorem 4.7 using the relationship between C and P defined above.

Corollary 4.9 The support of an extreme point p of P is a facet defining inequality of C if and only if for every face k of P having extreme point p on it and having non-empty set of extreme points L_k of P which are not on face k but on all other faces that extreme point p is on, there exists another face $j(k)$ of P not containing extreme point p or any extreme point in L_k .

Proof: The support of an extreme point p of P forms a covering constraint $\sum x_j \geq 1$, for $j \in N^p$ in C . Since the covering constraints of C form a clutter, we satisfy condition (1) of Theorem 4.7. We note that the statement of this corollary is a direct translation of condition (2) of Theorem 4.7 in light of Lemma 4.8. This completes the proof. ■

Let us now examine the facial structure of the MWIC polyhedron. From Theorem 4.5 and the full dimensionality of the MWIC polyhedron, we have that constraints of the form $z_i \leq 1$ are facet defining for the MWIC polyhedron. Theorem 4.6 states necessary and sufficient conditions for constraints of the form $z_i \geq 0$ to be facet defining. We see that in the following instances, non-negativity constraints are facets of the MWIC polyhedron.

Corollary 4.10 If there exists a constraint j of S which is contained in no IIS, then $z_j \geq 0$ is a facet defining constraint of C .

Proof: If constraint j of S is contained in no IIS, then there are no covering constraints of C having coefficient 1 for variable z_j . Since there are at least 2 nonzero coefficients in each covering constraint of C , neither of which is the coefficient of z_j , we satisfy the conditions of Theorem 4.6 and the conclusion holds. ■

Of course, we note that if a such a constraint j of S exists, then we can delete variable z_j from the covering problem C . A non-negativity constraint will be a facet if one of two conditions are met:

Corollary 4.11 A non-negativity constraint $z_j \geq 0$ is a facet defining constraint of C if and only if one of the following holds:

- (1) Constraint j of S appears in no IISs of S .
- (2) Every IIS of S containing constraint j of S is of cardinality 3 or greater.

Proof: To prove the “only if” direction, we note from Theorem 4.6 that in order for the non-negativity constraint $z_j \geq 0$ to be facet defining, removal of the j^{th} column of the coefficient matrix of C must yield a system still

having at least two nonzero coefficients in each row. Assume that $z_j \geq 0$ is a facet defining constraint of C . Now either constraint j of S appears in an IIS of S or it doesn't.

Case 1 Assume constraint j of S appears in at least one IIS of S . By the argument above, each IIS that constraint j appears in must have at least 3 elements, since upon removal of j we must still have at least 2. Thus condition (2) holds.

Case 2 Assume constraint j of S appears in no IISs of S . Then condition (1) holds.

To prove the “if” direction, we look at the two conditions specified in the hypothesis.

Case 1 Assume constraint j of S appears in no IISs of S . Then by Corollary 4.10, the non-negativity constraint $z_j \geq 0$ is facet inducing in C .

Case 2 Assume constraint j of S appears in at least one IIS of S , and that the cardinality of every IIS j appears in is at least 3. Then upon removal of variable j from C , every row of C will contain at least 2 nonzero coefficients. Thus by Theorem 4.6 the non-negativity constraint $z_j \geq 0$ is facet inducing in C . ■

We have all non-negativity constraints as facet defining for the MWIC polyhedron under the following special condition:

Corollary 4.12 All non-negativity constraints $\mathbf{z} \geq \mathbf{0}$ are facet defining for the MWIC polyhedron if and only if each IIS is of cardinality 3 or larger.

Proof: This is an immediate extension of Corollary 4.11. ■

We now examine a special instance of the MWIC problem.

Corollary 4.13 If the cardinality of the minimum weight IIS cover is 1, then all covering constraints are facet defining for the polyhedron C .

Proof: Since the cardinality of the MWIC is 1, there exists a column j of the MWIC coefficient matrix which is $\mathbf{1}$. This implies that we have a face j of P which has no extreme points of P on it. By Corollary 4.9 the conclusion holds. ■

What more can be said about the covering constraints being facet defining for the MWIC polyhedron? First, let us look at the set L_k (defined in Corollary 4.9) more closely. Given an extreme point p of P which is contained on face k , L_k consists of those extreme points of P which are not on face k , but are on all other faces that extreme point p of P is on. In other words, L_k consists of those extreme points of P which can be visited from extreme point p when we move off of face k . We first note that the nonzero elements of the support of extreme point p will correspond to the subset of basic variables which are nonzero in all bases of extreme point p . In the case where p is nondegenerate, there will be only a single basis and the nonzero elements of the support of p will correspond exactly to the basic variables. We wish to characterize the extreme points in L_k ; in particular, we will do this by determining if the extreme points in L_k are adjacent to p . In terms of the simplex algorithm, this is equivalent to determining if each extreme point in L_k is one pivot away from some basis of p . This is straightforward for p not degenerate since there is only a

single feasible basis corresponding to p . In the case where p is degenerate, we have many bases corresponding to the same extreme point. In this case, we view adjacency as meaning that there exists a single pivot from a basis of p which will take us to the particular extreme point of L_k .

Lemma 4.14 Each extreme point L_k^j of the set L_k for extreme point p is adjacent to p .

Proof: We will demonstrate the existence of a face of dimension 1 which contains both extreme points p and L_k^j , thus implying the adjacency of the extreme points. Let the dimension of the alternative polyhedron P be d . Since an extreme point is a face of dimension 0, we know that extreme point p is determined by the intersection of at least d faces, say $e \geq d$ faces. We can pick a subset of size d of these e intersecting faces, always including face k , which is sufficient to define p . Now L_k^j shares exactly $d - 1$ of the intersecting faces in this subset, since it does not lie on face k . Thus there exists a face of dimension $d - (d - 1) = 1$ containing both p and L_k^j . This implies that p and L_k^j are adjacent. ■

Given this result, how large can the set L_k be? By noting the uniqueness of the extreme point we move to in a simplex pivot, we obtain the following result.

Lemma 4.15 The cardinality of the set L_k for extreme point p of P is at most 1.

Proof: From Lemma 4.14, we have that the extreme points of L_k are a subset of those adjacent to extreme point p . By the simplex algorithm, when we pivot in a specific nonbasic variable, say the k^{th} , at a nonzero value there is a unique extreme point which we pivot to. Thus, the set L_k

can have at most a single element. ■

Now, we can prove the following result on the covering constraints of the MWIC polyhedron.

Theorem 4.16 (Covering Constraint Facet Theorem) All covering constraints of the MWIC polyhedron C are facet defining.

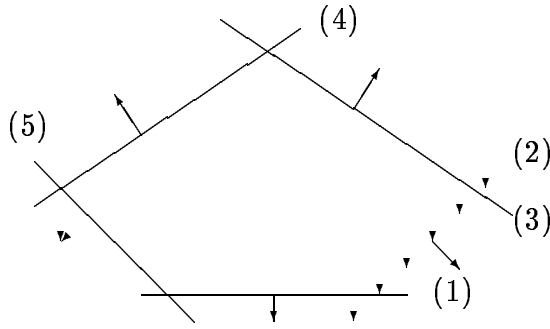
Proof: Choose an arbitrary extreme point p of P . Look at all faces of P that extreme point p is on. By Lemma 4.14, if we wish to pivot off of face k , then we can visit those elements of L_k in 1 pivot. By Lemma 4.15, L_k has at most 1 element. If L_k is empty, the hypothesis holds trivially via Corollary 4.9. Otherwise, L_k is not empty. Extreme point p is defined by the intersection of e faces (note e is not necessarily the dimension of P). The extreme point in L_k , call it L_k^1 , is defined by the intersection of $f \geq e$ faces, of which $e - 1$ also help define p . We note that their definitions will differ in at least 2 faces. L_k^1 is not on face k , and must be on a face that p is not (otherwise, the support of p is a proper subset of the support of L_k^1 which is not possible). Since each IIS of S must have at least 2 constraints, each extreme point of P must have 2 nonzero variables in its support. Since L_k^1 is on f faces, there must be at least $f + 2$ faces in P . Now L_k^1 is on f faces, p is on e faces, and they share $e - 1$ faces, so we have $f + e - (e - 1)$ faces which contain either L_k^1 or p or both. This leaves **at least** $f + 2 - (f + e - (e - 1)) = 2 - 1 = 1$ face which defines neither p nor L_k^1 . Thus there exists a face of P containing neither p nor L_k^1 ; and, by Corollary 4.9 we have the hypothesis. ■

Thus, all covering constraints of the MWIC problem are facet defining. If **all** facets of the MWIC problem are of this form, then the

integer hull of the MWIC polyhedron would be completely described, and the solution to the linear programming relaxation would give us the optimal cover. As the following example (from [32]) demonstrates, we have not defined all facets of this problem.

1. $x_1 \leq 2$
2. $-x_1 + x_2 \leq -10$
3. $-2x_1 - 3x_2 \leq -49$
4. $2x_1 - 3x_2 \leq -21$
5. $x_1 + x_2 \leq 6$

x_1, x_2 free



The IISs of this system are: $\{2, 4, 5\}$, $\{1, 3, 4\}$, $\{2, 3, 5\}$, $\{1, 2, 4\}$, and $\{1, 3, 5\}$. Thus, the MWIC has cardinality 2; there are several optimal solutions including $\{3, 4\}$. However, the optimal solution to the LP relaxation of the MWIC problem is to let each variable take the value $1/3$, and thus the LP relaxation solution is $5/3$.

This solution can be cut off using the family of facets derived by Balas and Ng in [4]. In this paper, all facets with coefficients in $\{0, 1, 2\}$ are identified for the general set covering problem. We have explored the facial structure of the MWIC polyhedron further but found no simplification of

the necessary and sufficient conditions for these constraints to be facets in the MWIC problem.

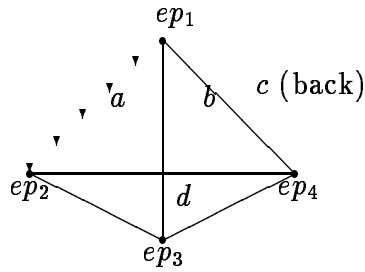
4.4 Generalizations

In this section we present an abstraction of the theorems in the previous sections to a new family of covering problems, called **Polyhedral Covering Problems**.

We first recognize that the MWIC problem is exactly one of identifying the minimum weight set of faces of P which cover all extreme points of the polyhedron P (or the minimum weight set of faces which intersect outside of P - see [32]). The special structure of the MWIC problem is a direct consequence of the polyhedral nature of the covering problem. In our case, the supports of the extreme points of P correspond to IISs of S . However, the only conditions we used in proving the results of Chapter 4 were the minimality of each IIS (thus having a clutter), that each IIS has at least 2 elements, and the extreme point relationship.

Suppose we are given the more general problem of identifying the minimum weight extreme point cover of the polyhedron GP - we denote this as the MWEPC problem. Let EP be the array whose rows are the support vectors of the extreme points of GP , so $EP\mathbf{v} \geq 1$ forms the covering constraints of MWEPC. By the minimality of an extreme point, we have that the rows of EP form a clutter.

We are not guaranteed that each extreme point of GP will have at least 2 nonzero elements in its support. Consider the following polyhedron:



Since ep_1 is defined by the intersection of faces a , b , and c , the support of extreme point ep_1 is $\{0001\}$. We will call such polyhedra “pyradmidal.” If GP is unbounded, then we can have extreme points whose supports have no nonzero elements. Consider the trivial case

$$1. \quad x_1 \geq 0$$

$$2. \quad x_2 \geq 0$$

$$x_1, x_2 \text{ free}$$

This system has only a single extreme point $x_1 = 0, x_2 = 0$, whose support vector is $\{00\}$.

If we wish to cover the extreme points of a polyhedron, these 2 cases can be trivially preprocessed out of the covering problem. If the support of an extreme point has a single element nonzero, say the j^{th} , then we simply remove this row and all other rows having a 1 in column j from EP and set the j^{th} covering variable to 1. If we have any extreme points having no nonzero elements in their supports, then there is no solution to the covering problem (there exists no faces intersecting outside of P).

With this type of preprocessing, all properties of the MWIC problem generalize to the MWEPC problem. In particular, all covering constraints of the MWEPC problem will be facet defining (this is analagous to Theorem 4.16).

5. Algorithm Description and Computational Results

5.1 Introduction

In this chapter we develop an algorithm for obtaining the MWIC based upon the IIS identification method of Gleeson and Ryan discussed previously. If we knew what the IISs of a system S were, we could formulate the following set covering problem, where w_i is the weight on the i th constraint.

$$\begin{aligned} \min \quad & \sum_{i=1}^m w_i z_i \\ \text{subject to} \quad & \\ & \sum_{i \in J} z_i \geq 1 \quad \text{for all IISs } J. \\ & z_i \text{ binary} \end{aligned}$$

Unfortunately, as previously discussed, the number of IISs may be exponential in the size of the original problem, so we do not want to write down the whole problem at once. Instead, we will generate constraints dynamically for the problem and solve it iteratively as outlined below:

- (1) Identify an initial set of IISs K . (K may be empty.)
- (2) Solve the covering problem

$$\begin{aligned} \min \quad & \sum_{i=1}^m w_i z_i \\ \text{subject to} \quad & \\ & \sum_{i \in J} z_i \geq 1 \quad \text{for all IISs } J \in K. \\ & z_i \text{ binary} \end{aligned}$$

Let T index the elements in the optimal cover.

- (3) Look for an IIS of S not covered by T . If there is none, STOP, T is an optimal cover of *all* IISs. Otherwise, add the new IIS to K and go to 2.

To identify IISs, we will make use of Theorem 2.6 due to Gleeson and Ryan [19]. We examine the utility of this theorem in terms of our algorithm presented above.

Let T denote the index set of a subset of the variables defining P . Then T will also denote a subsystem of the system S . Let $y_T = 0$ mean that all the variables in T have been set to 0. If we search for an extreme point of P with $y_T = 0$, there are two possibilities. The first is that we find an extreme point whose nonzero components index an IIS that does not intersect T . The second is that there is no such extreme point of P . In this case, the system $\{S \setminus T\}$ is feasible.

We can now modify step 3 of our algorithm (outlined above):

- (1) Identify an initial set of IISs K . (K may be empty.)
(2) Solve the covering problem

$$\begin{aligned} \min \quad & \sum_{i=1}^m w_i z_i \\ \text{subject to} \quad & \sum_{i \in J} z_i \geq 1 \quad \text{for all IISs } J \in K. \\ & z_i \text{ binary} \end{aligned}$$

Let T index the elements in the optimal cover.

- (3) Look for an extreme point of P having $y_T = 0$. If there is none, STOP, T is an optimal cover of *all* IISs. Otherwise, add the IIS corresponding to this extreme point to K and go to 2.

This implementation is generalized in Section 5.3 to operate on an infeasible linear programming problem given in a more general format. Later sections discuss enhancements including the selection of the initial K in Step 1, and the choice of extreme point in Step 3. We also discuss the efficacy of using a heuristic to solve the covering problem in Step 2 when possible. Now, we demonstrate the utility of the IIS cover isolation.

5.2 Min IIS Cover Isolation in Practice

Finding a minimum weight IIS cover will take more time than identifying a single IIS and attempting infeasibility diagnosis. Is the extra information obtained worth the extra time expense? We address this by looking at several examples, and making a number of simple observations. Observe that if there exists a constraint which is contained in **every** IIS, it is a minimum IIS cover. So, in the **best** case, infeasibility can be isolated to a single constraint.

This is illustrated by the following example from [36]: Consider the infeasible system, which is depicted in Figure 5.1:

$$\begin{array}{rcl}
 1. & x_1 & - x_2 \leq 0 \\
 2. & & 2x_2 \leq 1 \\
 3. & -x_1 & - x_2 \leq -2 \\
 4. & & - x_2 \leq -2 \\
 5. & -2x_1 & - x_2 \leq -4
 \end{array}$$

An IIS of this system is $\{1,2,3\}$. Based upon this information alone, out of the context of the entire system, a modeler would have a difficult time determining the source of the infeasibility. The unique min IIS cover is $\{2\}$, which indicates that every IIS in the system contains the second

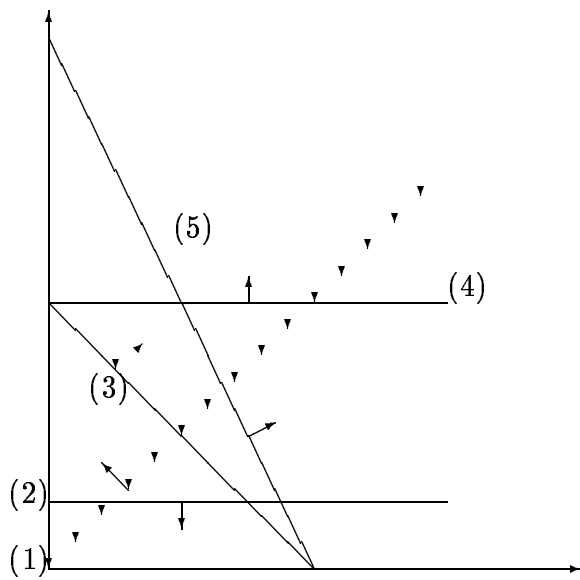


Figure 5.1: Infeasible system from Van Loon

constraint.

It is easily demonstrated, however, that the modeling error is not always isolated by a particular minimum IIS cover. Any infeasible system having only a single IIS will have a minimum IIS cover consisting of **any** element of the IIS. By weighting the objective function of the covering problem, alternative minima can be identified to aid in isolation. This can be done by setting the objective function coefficients for variables contained in the original cover to a suitably large integer, say n which is the number of variables in the covering problem. We then re-solve the covering subproblem one time with the current IIS set. In summary, the minimum IIS cover **itself** will not always provide a perfect isolation to the modeling error; however, the isolation information in using this technique to find the minimum IIS cover is always at least as useful as that obtained from a single IIS. Using our algorithm to find the min IIS cover always identifies at least one IIS. This combination of min IIS cover **and** IISs is available to the user in her debugging analysis. Additionally, the IIS cover will **always** give the modeler the means of eliminating infeasibility from the model. A single IIS does not provide this level of isolation in general.

We now examine the isolation power of this method on a larger problem. We look at a real industry example of using the minimum weight IIS cover to isolate infeasibility. This LP is a SONET network model having 845 rows, 1792 columns, and 8448 nonzero entries. The SONET model arises from a new family of problems in the telecommunications industry

– it is a multidimensional packing problem with many complicating constraints. Specifically, there are a large number of capacity and demand constraints and a large number of side constraints. These side constraints are generated by a number of different software codes depending upon the exact model being analyzed. An instance of the model was found to be infeasible, and was analyzed using our algorithm. The min cardinality IIS cover isolated the infeasibility to a single demand constraint. Since this portion of the model was known to be valid, weights were assigned to rows of the infeasible problem and the min *weight* IIS cover was found. A weight of 1 was assigned to all software generated constraints, which accounted for 32% of all rows, and a weight of 5 was assigned to all capacity and demand constraints. These weights become the objective function value in the covering problem for each covering variable. The covering variables correspond to constraints of the original infeasible system. Re-solving, we were able to isolate the infeasibility to an incorrectly generated constraint. Looking at the code used to generate this constraint led to the discovery of a family of 5 other improperly generated constraints and correction of a section of code. Using OSL, it initially took 3.26 seconds on the RS6000 to determine that the problem was infeasible. Using our algorithm built around the OSL library, the iterative procedure of identifying IISs and solving the minimum cardinality IIS cover to optimality took 7.28 seconds. Once we decided to re-solve the minimum weight IIS cover, our procedure took 7.21 seconds to conclude the iterative process of IIS and optimal cover identification. In order to compare our solution timeliness with state of the art IIS identification routines, we performed another

test to identify a single IIS. Again using the RS6000, we used the built-in CPLEX IIS identification routines (based upon the work of Chinneck) to identify a single IIS. This diagnosis took 86 seconds (after infeasibility was diagnosed), and yielded an IIS having 6 elements. In this example, we were able to find both minimum weight and minimum cardinality IIS covers in less time than the current state of the art IIS identifier found a **single** IIS. The analysis path used to debug this example has proven successful isolating infeasibility in several other problems.

Suppose a modeler takes the extra time during model development to generate weights for each constraint and includes these weights as an extra column in the LP formulation. As a practical consideration, when running the model the bounds for this column can be set to zero. Thus the supplemental debugging information is ignored when exercising the LP model. If the model is found to be infeasible, the weights will be used by the min weight IIS cover algorithm to weight the objective function of the covering subproblem in order to help isolate the infeasibility. As seen in this example, having constraint weights available can greatly increase the effectiveness of post-infeasibility analysis.

We present an example from the NETLIB infeasible library to demonstrate another key advantage of the min IIS cover. The WOOD-INFE problem is a small network example with 36 rows and 89 columns. An IIS infeasibility isolation from MINOS(IIS) is a single functional constraint (plus non-negativity constraints). However, the min IIS cover consists of 2 functional constraints. This problem has two disjoint modeling errors which **can not** both be identified by a single IIS. In infeasibility

instances with multiple modeling errors the covering approach is especially powerful.

5.3 Formulation of P , the Alternative Polyhedron, for General Linear Programming Problems.

We now turn our attention back to the problem of identifying IISs. Theorem 2.6 can be restated in a more general setting:

Theorem 5.1 (General Version of Gleeson-Ryan Theorem)

Given the inconsistent system

$$S = \{\mathbf{x} \in Q^n \mid A\mathbf{x} \leq \mathbf{b}, C\mathbf{x} = d, \mathbf{L} \leq \mathbf{x} \leq \mathbf{U}\},$$

the indices of the minimal infeasible subsystems of S are exactly the supports of the vertices of the polyhedron $P = \{\mathbf{y}, \mathbf{w}, \mathbf{v}, \mathbf{z} \in Q^m \mid \mathbf{y}^T A + \mathbf{w}^T C + \mathbf{v} - \mathbf{z} = 0, \mathbf{y}^T \mathbf{b} + \mathbf{w}^T \mathbf{d} + \mathbf{v}^T \mathbf{U} - \mathbf{z}^T \mathbf{L} = -1, \mathbf{y}, \mathbf{z}, \mathbf{v} \geq 0, \mathbf{w} \text{ unrestricted}\}$.

Note that if the only bounds on \mathbf{x} are non-negativity constraints, we can simplify the above formulation:

Theorem 5.2 (General Version II of Gleeson-Ryan Theorem)

Given the inconsistent system

$$S = \{\mathbf{x} \in Q^n \mid A\mathbf{x} \leq \mathbf{b}, C\mathbf{x} = \mathbf{d}, \mathbf{x} \geq 0\},$$

the indices of the minimal infeasible subsystems of S are exactly the supports of the vertices of the polyhedron $P = \{\mathbf{y}, \mathbf{w} \in Q^m \mid \mathbf{y}^T A + \mathbf{w}^T C \geq 0, \mathbf{y}^T \mathbf{b} + \mathbf{w}^T \mathbf{d} = -1, \mathbf{y} \geq 0, \mathbf{w} \text{ unrestricted}\}$.

In other words, we do not need to explicitly handle non-negativity constraints, but can simply check the status of the slack variables of the alternative system to determine if non-negativity of some x_i is included

in an IIS. Also, this helps reduce the size of the LP instance we solve at each step.

If the right hand side of the “cone-bounding” constraint were left as -1 for large problems, numerical instability would result. We observed on several problems that when we fixed at 0 those variables in the alternative system which are in the covering solution, we obtained an IIS identical to the one just previously generated. But this means that in trying to find an IIS not covered by the current solution, we have generated one that *is* covered by the current solution. Examine what can happen if P has many columns.

Assume that we have the following system for P : $\mathbf{y}^T A \geq 0$, where each $a_{ij} \geq 0$ for $i = 1, \dots, m$ and $j = 1, \dots, n$, and $\mathbf{y}^T \mathbf{b} = -1$, where $\mathbf{b} = \mathbf{1}$ (-1 is the m -vector of -1 's). Also assume that the zero tolerance cut-off is $1.0E-07$, so all values equal to or less than $1.0E-07$ are considered to be 0. Then, $\mathbf{y} = 1.0E - 07$ satisfies the first constraint, and if $m = 10,000,000$, then $\sum_1^m (1.0E - 07)(-1) = -1$, so the second constraint is also satisfied. Now each y_i is within epsilon of 0 and so should be considered 0. Instead, we have allowed a solution which is “numerically” feasible to be identified by the optimizer.

However, since the values of the basic variables are below the software imposed tolerance, they should be considered 0, and so we should have that $\sum_{i=1}^m (y_i * k_i) = 0$ for **any** \mathbf{k} ; which means that the solution to the alternative system is **not** basic feasible. When this “solution” is identified as being feasible, then we identify subsystems which are **not** IISs. In this example, we can alleviate this problem by setting the right hand side of

the second (or cone-binding) constraint to be equal to $\sum_i b_i = -m$ (the negative of the number of variables in the problem). This will eliminate the problem for tolerances greater than or equal to $1/m$ for the conditions outlined above - which are not necessarily worst case.

For the general case described in Theorem 5.1, we extend the logic used above to determine that a cone-binding constraint eliminating such problems could be:

$$\mathbf{y}^T \mathbf{b} + \mathbf{w}^T \mathbf{d} + \mathbf{v}^T \mathbf{U} - \mathbf{z}^T \mathbf{L} = -((\sum_k |b_k|) + (\sum_k |d_k|) + (\sum_j |U_j| + |L_j|)).$$

To see why absolute values are included on the sum, consider the following modification to the previous example. Assume that there is a 10,001st constraint, having a right hand side value of $-10,000$. Then, $\sum_i b_i = 0$, which does not eliminate the solution $\mathbf{y} = 0$. We have not yet encountered a problem for which this normalization fails to eliminate the problem.

5.4 Finding IISs

Two slightly different approaches may be used to identify IISs of S . First, we consider using the simplex algorithm to find extreme points of P . By bounding the objective function $\mathbf{y}^T \mathbf{b}$ of the alternative system and including it in the constraint set of P , we free ourselves to use a “surrogate” objective function to heuristically guide our search. We have experimented with using $\mathbf{1}$ as the objective weight. This should tend to find small cardinality IISs, although to find the true minimum cardinality IIS we must solve a fixed-charge integer programming problem (as demonstrated by Greenberg and Murphy [26]). Additionally, these weights can be modified at each step by incrementing the weight of each

variable appearing in the current IIS. In this way, we heuristically tend to find IISs which overlap in the fewest number of elements. This should help in the identification an optimal cover to the full MWIC problem by identifying sets of non-overlapping IISs.

We have also considered ignoring the objective function when looking for extreme points of P and using the first feasible solution found. Since every extreme point corresponds to an IIS, we still identify an IIS. As expected, this takes fewer pivots to find an IIS than solving to optimality with the surrogate objective described above. However, the savings in IIS identification time may be offset by the typically larger cardinality IISs found and the increased number of covering subproblems needed to solve.

Rather than generating IISs individually, we can generate many by solving a single LP if we instead search for extreme rays on the cone P' , as suggested by Greenberg [21]. We consider the following linear programming instance based upon our definition of P' :

$$\begin{aligned} \min \quad & \mathbf{y}^T \mathbf{b} \\ \text{subject to} \quad & \\ & \mathbf{y}^T A = 0 \\ & \mathbf{y} \geq 0 \end{aligned}$$

Since S is infeasible, Theorem 2.1 implies that P' is feasible and hence unbounded. Therefore, we must be able to find at least one nonbasic variable of non-optimal sign at termination of our new LP. If this nonbasic variable is not blocked, then the variable and all basic variables which correspond to non-zero tableau column entries for the nonbasic form the

support set of an IIS. In other words, the supports of the extreme rays of P' are IISs in the original infeasible system, S . In general, we will have more than a single such nonbasic variable, so we can find many IISs from one LP solution. By pivoting we can find all IISs of S by such a procedure.

In general, we will have to perform several pivots before identifying unboundedness due to the degeneracy of the $\mathbf{0}$ solution. In practice, this method can reduce the number of iterations to solve the minimum weight IIS cover. For example, let an IIS of S be identified as $\{1,2,3,4,5,8,10,15\}$ and suppose that constraint 15 is in every IIS (and so is the minimum IIS cover). If $\{1\}$ is identified as the cover, there could exist another IIS of the form $\{2,3,4,5,6,8,10,15\}$ and so forth. There could be potential to identify **many** IISs before the min IIS cover is solved optimally. If a conical solution yields multiple IISs, the total number of covering subproblems (and hence IIS identification problems) can be reduced. This becomes important for large infeasible problems which can have a single constraint as the optimal min IIS cover, while having IISs with hundreds of elements in them.

Since solving on the cone of the alternative system allows us the potential to identify multiple IISs from a single basis and in general takes fewer pivots to solve, the idea is to use this as a “jump-start” for the algorithm, and then to use the extreme point method to find IISs with specific characteristics.

We can illustrate each of these methods using the following example from [13] using LINDO. Suppose we are given the infeasible system

S defined as:

$$\begin{array}{rcll}
 1. & -0.5x_1 & + & x_2 & \geq & 0.5 \\
 2. & 2x_1 & - & x_2 & \geq & 3.0 \\
 3. & 3x_1 & + & x_2 & \leq & 6.0 \\
 4. & & & & & x_5 \leq 2.0 \\
 5. & & & & 3x_4 & - x_5 \leq 2.0 \\
 6. & & & & x_4 & \geq 5.0 \\
 7. & x_1 & & & + & x_5 \leq 10.0 \\
 8. & x_1 & + & 2x_2 & + & x_4 \leq 14.0 \\
 9. & & & x_2 & + & x_4 \geq 1.0
 \end{array}$$

with all $x_i \geq 0$

If we solve to optimality using objective weights 1 for all variables of the alternative system, after 6 pivots, we find $\{4,5,6\}$ as an IIS. If we solve to feasibility only (ignore the objective weights), we find the IIS $\{1,2,3\}$ after 5 pivots. Solving on the cone we find 3 IISs after only 4 pivots: $\{4,5,6\}$, $\{1,2,3\}$, and $\{1,2,5,6,7\}$. The min IIS cover is $\{1,5\}$, so by solving on the cone, we initialize our procedure with enough IISs to solve the cover problem only once.

5.5 Heuristic Solution of Covering Problems

The covering problems can be solved heuristically at intermediate steps. We can determine if these suboptimal covers are true covers of the min weight covering problem just as we did when solving this problem optimally in Step 3 of our algorithm. If a true optimal cover is desired, the covering problem must be solved optimally before terminating.

The greedy heuristic [14] can be used at intermediate steps.

When no IIS is found that is not covered by the greedy solution, we solve the covering problem optimally. If the cover is the same cardinality as that found by the greedy heuristic, we are done. Otherwise, we find an IIS not covered by the current solution and continue the algorithm. The greedy algorithm will not guarantee an optimal cover; however, it seems to work well in practice.

For the problems in the infeasible library on NETLIB, we have found that typically only a small fraction of total algorithm time (usually less than 10%) is spent solving the covering subproblems. This result is based on identifying IISs singly by searching for extreme point solutions of P , so the results could change significantly with improvements over the basic form of the algorithm. It should be noted that the difficulty of the covering problem has been displayed on a few relatively small problems. For the NETLIB problem MONDOU2, more than 97% of the CPU time (115 of 118 seconds) was spent solving a **single instance** of the covering subproblem having only 9 constraints. This demonstrates a need to both learn more about the structure of this special IIS covering problem and perhaps explore more fully using the greedy algorithm, or a modification of it, at intermediate steps.

5.6 Computational Results

All comparisons (unless otherwise stated) were run on an IBM RS6000 using the IBM OSL object library and FORTRAN code written by the author. All runs were made with approximately the same computer load – only a single user was on the system. We experimented with three versions of our algorithm. All three solve the covering subproblem to

optimality at each step using the OSL EKKMSSL subroutine. We have not yet completed implementation of a version which will solve on the cone, so all three versions presented here search for extreme point solutions on the bounded alternative system. We distinguish each version by the objective function used in solving the alternative system. The baseline algorithm, IIS_COV_1, weights the coefficients of the objective function to all 1's in an attempt to find small cardinality IISs. Version 2, IIS_COV_2, weights the coefficient of each variable according to how many generated IISs it already appears in. Intuitively, this will tend to find IISs which overlap as little as possible. Finally, version 3, IIS_COV_3, ignores the objective function, and the first extreme point found is the solution. This will tend to identify IISs faster, though perhaps at the trade off of finding larger cardinality IISs. Note that IIS_COV_2 will differ from IIS_COV_1 only when more than 1 IIS is found to optimally solve the covering problem. Therefore, we did not run IIS_COV_2 if we found that only 1 IIS was needed to find the optimal cover.

The test bed of problems consists of the infeasible LP library on NETLIB, originally set up by John Chinneck. A summary of the problems is given in Table 5.1.

In order to facilitate the analysis of results, we partitioned the problems into 3 sets according to their size. We considered “small” problems to be those having fewer than 100 rows and 100 columns. “Medium” problems are those with between 100 and 1000 rows and 100 to 1000 columns. The remainder are “large” problems.

Table 5.2 presents the results of our 3 approaches on the small

Problem	Original Problem			Alternative System		
	Constraints	Variables	Nonzeros	Constraints	Variables	Nonzeros
BGDBG1	348	407	1440	408	390	1737
BGETAM	400	688	2409	689	689	2957
BGINDY	2671	10116	65502	10117	2671	67589
BGPRTR	20	34	64	35	20	76
BOX1	231	261	651	262	492	1173
CERIA3D	3576	824	17602	825	3576	17851
CHEMCOM	288	720	1566	721	625	2180
CPLEX1	3005	3221	8944	3222	3223	10883
EX72A	197	215	467	216	412	897
EX73A	193	211	457	212	404	879
FOREST6	66	95	210	96	71	226
GALENET	8	8	16	9	16	38
GOSH	3792	10733	97231	10734	3792	97433
GREENBEA	2393	5405	30885	5406	2941	31926
ITEST2	9	4	17	5	9	26
ITEST6	11	8	20	9	11	31
KLEIN1	54	54	696	55	54	700
KLEIN2	477	54	4585	55	477	4600
KLEIN3	994	88	12107	89	994	12115
MEXP	1383	1500	5027	1501	1383	5755
MONDOU2	312	604	1208	605	1520	3088
PANG	361	460	2652	461	453	2587
PILOT4I	410	1000	5141	1001	717	5860
QUAL	323	464	1646	465	835	2662
REACTOR	318	637	2420	638	932	3699
REFINERY	323	464	1626	465	835	2641
VOL1	323	464	1646	465	835	2662
WOODINFE	35	89	140	90	69	208

Table 5.1: Problem Test Bed

problems. The measures we use in comparing the approaches are total solution time of the covering problem (CPU seconds), number of IISs identified while solving the covering problem optimally (**not** the number of IISs in the cover), number of pivots to first IIS, average number of pivots per IIS, and the smallest cardinality IIS. In analyzing infeasibility, it may be useful to measure the size of an IIS in terms of the number of actual **constraints** it contains, rather than constraints **and** variable bounds - both non-negativity and general upper and lower bounds. Note that IIS size refers to the number of constraints and variable bounds **exclusive of non-negativity bounds**. Information on non-negativity bounds is available with the solution approach, we have chosen to not include it in the IIS reporting. The columns for average IIS size and smallest cardinality IIS are thus subdivided into IIS size and number of actual constraints.

On most problems, very little time is spent in solving the min weight covering subproblem, and from 50% to 95% of the time is spent in identifying IISs - solving the alternative system LP. Therefore, another reasonable measure of how well each algorithm performs is the number of simplex pivots required to find an IIS.

The first few pivots made by OSL during Phase 1 will be random. This is done initially to speed up the simplex algorithm, and later other pricing schemes are used. This means that when re-running a problem with the same algorithm we will have variance in our solution times. On the problems we looked at, this variance was typically 5% or less. For this reason, we will consider time differences between algorithms to be

Problem	Algorithm	Cover Time	# IISs Generated	Pivots		Avg. IIS		Smallest IIS	
				IIS 1	Avg.	Size	Rows	Size	Rows
BGPRTR	IIS_COV_1	0.19	1	18	10.0	7.0	7.0	7	7
BGPRTR	IIS_COV_3	0.20	1	14	8.0	14.0	14.0	14	14
FOREST6	IIS_COV_1	0.73	2	124	50.3	64.0	59.0	64	59
FOREST6	IIS_COV_2	0.71	2	124	51.0	64.0	59.0	64	59
FOREST6	IIS_COV_3	0.61	2	81	29.3	69.5	64.5	69	64
GALENET	IIS_COV_1	0.19	1	7	3.5	6.0	3.0	6	3
GALENET	IIS_COV_3	0.20	1	5	3.0	6.0	3.0	6	3
ITEST2	IIS_COV_1	0.38	3	4	2.0	3.7	3.7	3	3
ITEST2	IIS_COV_2	0.35	3	4	2.0	3.7	3.7	3	3
ITEST2	IIS_COV_3	0.36	3	4	1.8	3.7	3.7	3	3
ITEST6	IIS_COV_1	0.65	3	6	2.8	3.3	3.3	3	3
ITEST6	IIS_COV_2	0.69	3	9	4.0	3.3	3.7	3	3
ITEST6	IIS_COV_3	0.65	6	3	1.7	3.7	3.7	3	3
KLEIN1	IIS_COV_1	0.89	1	162	81.5	51.0	51.0	51	51
KLEIN1	IIS_COV_3	0.82	1	116	71.5	51.0	51.0	51	51
WOODINFE	IIS_COV_1	0.46	2	24	12.6	2.0	1.0	2	1
WOODINFE	IIS_COV_2	0.41	2	24	12.6	2.0	1.0	2	1
WOODINFE	IIS_COV_3	0.48	2	9	11.0	2.0	1.0	2	1

Table 5.2: Results for Small Problems

significant only if they are greater than 10%. In this light, no significant time difference is found between the algorithms on the small problem test bed. On the 4 problems where all 3 algorithms were run, IIS_COV_2 was faster on 2, IIS_COV_3 and IIS_COV_1 were faster on 1 each, with all three indistinguishable on one. If we try to verify our assumptions on strengths of each algorithm, we see that IIS_COV_3 takes fewer pivots both to find the first IIS and on average; but requires more IISs to solve the cover problem only on ITEST6. IIS_COV_1 does find smaller IISs on average than the other algorithms. Not much more performance information can be obtained from the small problems.

Algorithmic comparisons for the “mid-sized” problems are presented in Table 5.3. We have several interesting problems in terms of the size of the min weight IIS covering problem solved. With these harder problems, we see a number of interesting trends. In 10 of the 12 problems, IIS_COV_3 takes fewer pivots to find an initial IIS as expected. However, this algorithm has the lowest average number of pivots per IIS in only 7 of the 12 problems. What we see is that both IIS_COV_1 and IIS_COV_2 typically take more pivots to find an initial solution. However, only a few pivots, typically fewer than 25, are needed to find IISs satisfying optimality conditions for the alternative system after the initial basic feasible solution is found. Thus some of the advantage of IIS_COV_3 is lost when finding many IISs. In fact, on those problems where IIS_COV_3 took fewer pivots to find its initial IIS but had a higher average pivot value, an algorithm needing to find a larger number of IISs to solve the covering problem had the lowest average pivot value. Thus the more IISs

one needs to find, the less important the initial number of pivots becomes in terms of solution time.

Additionally, we see that IIS_COV_3 solves the covering problem faster on 8 of the 9 problems with time distinctions. We also note that it finds the smallest average IIS on 3 of the 9 problems where distinction is made, and finds more IISs in solving the cover on only 4 problems and finds the least IISs in solving the cover 5 times (excluding ties). This seems to contradict the intuition that we would find IISs faster, but require more IISs to solve the covering problem.

As we look at the results for the large problems (see Table 5.4), we see that an undirected search for IISs is faster. IIS_COV_3 again dominates, being the fastest algorithm on 8 of the 9 problems, taking the fewest pivots to the first IIS on all problems, and taking fewest average number of pivots on 6 of 9 problems. It also finds the smallest average IIS on 4 of 6 problems where distinction is made.

We also compare (Table 5.5) the information provided by the IIS cover to that obtained from IIS isolation via MINOS(IIS) (from [11]). Many of the problems in this test bed were created by taking a feasible LP instance and modifying a single bound or constraint until the problem was infeasible. Four of the problems in the test bed we know to be infeasible in original form – BGDBG1, BGPTR, GREENBEA, and MONDOU2. Of these problems, only BGPTR has an IIS cover of 1 (and requires a single IIS to solve the cover). Thus although there are a large number of problems in the test bed having singleton IIS covers and requiring only a single IIS to find their cover, this may be an artifact of their construction.

Problem	Algorithm	Cover Time	# IISs Generated	Pivots		Avg. IIS		Smallest IIS	
				IIS 1	Avg.	Size	Rows	Size	Rows
BGDBG1	IIS_COV_1	8.05	33	112	12.4	8.1	7.4	2	2
BGDBG1	IIS_COV_2	6.15	24	112	14.6	7.0	6.5	2	2
BGDBG1	IIS_COV_3	7.74	36	34	8.6	6.9	6.1	2	2
BGETAM	IIS_COV_1	10.42	5	242	206.8	68.0	62.2	8	7
BGETAM	IIS_COV_2	11.79	6	150	145.4	59.5	51.7	13	12
BGETAM	IIS_COV_3	9.24	7	20	89.1	173.1	140.7	18	17
BOX1	IIS_COV_1	1.46	2	13	36.0	53.5	52.5	10	9
BOX1	IIS_COV_3	1.46	2	12	34.3	54.5	53.5	10	9
CHEMCOM	IIS_COV_1	3.74	2	177	136.0	26.0	16.0	25	15
CHEMCOM	IIS_COV_2	3.73	2	177	142.0	24.0	14.0	23	13
CHEMCOM	IIS_COV_3	3.63	1	137	204.5	27.0	15.0	27	15
EX72A	IIS_COV_1	1.35	3	104	27.0	69.0	68.0	59	58
EX72A	IIS_COV_2	1.66	3	104	39.5	70.3	69.3	61	60
EX72A	IIS_COV_3	1.22	2	108	36.0	66.5	65.5	59	58
EX73A	IIS_COV_1	0.92	1	88	44.0	28.0	27.0	28	27
EX73A	IIS_COV_3	0.97	1	79	42.5	26.0	25.0	26	25
KLEIN2	IIS_COV_1	8.03	13	597	85.4	53.2	53.2	53	53
KLEIN2	IIS_COV_2	5.46	4	574	173.8	54.0	54.0	53	53
KLEIN2	IIS_COV_3	5.38	6	476	113.9	55.2	55.2	53	53
PANG	IIS_COV_1	3.21	1	392	204.5	16.0	13.0	16	13
PANG	IIS_COV_2	2.59	1	250	127.5	16.0	13.0	16	13
PANG	IIS_COV_3	2.65	2	183	76.3	19.5	16.0	18	15
QUAL	IIS_COV_1	8.05	5	707	149.3	185.4	124.0	142	92
QUAL	IIS_COV_2	12.12	5	707	223.2	184.8	103.6	143	82
QUAL	IIS_COV_3	7.40	6	558	107.9	180.6	124.0	134	90
REACTOR	IIS_COV_1	5.58	2	152	192.0	5.0	1.0	5	1
REACTOR	IIS_COV_2	4.44	2	152	125.7	5.0	1.0	5	1
REACTOR	IIS_COV_3	3.89	2	100	119.3	5.0	1.0	5	1
REFINERY	IIS_COV_1	22.26	38	665	42.5	135.2	91.2	92	61
REFINERY	IIS_COV_2	19.99	31	665	48.1	134.0	89.0	93	63
REFINERY	IIS_COV_3	16.51	23	577	72.6	145.7	93.4	97	62
VOL1	IIS_COV_1	13.45	13	699	83.3	180.8	121.7	137	91
VOL1	IIS_COV_2	20.70	10	699	278.5	180.3	121.6	137	91
VOL1	IIS_COV_3	10.75	8	715	138.6	182.6	120.9	160	104

Table 5.3: Results for Mid-sized Problems

Problem	Algorithm	Cover Time	# IISs Generated	Pivots		Avg. IIS		Smallest IIS	
				IIS 1	Avg.	Size	Rows	Size	Rows
BGINDY	IIS_COV_1	139.19	1	1908	1140.0	3.0	3.0	3	3
BGINDY	IIS_COV_3	126.34	1	219	1114.0	3.0	3.0	3	3
CERIA3D	IIS_COV_1	56.32	3	1905	758.0	123.3	123.3	73	73
CERIA3D	IIS_COV_2	66.46	3	2406	921.0	141.3	141.3	74	74
CERIA3D	IIS_COV_3	26.45	8	538	99.0	144.8	144.8	73	73
CPLEX1	IIS_COV_1	38.50	1	1473	1214.5	5.0	5.0	5	5
CPLEX1	IIS_COV_3	33.83	1	212	1058.5	5.0	5.0	5	5
GOSH	IIS_COV_1	563.91	1	6009	3305.5	49.0	49.0	49	49
GOSH	IIS_COV_3	637.34	2	1885	2350.0	72.0	72.0	68	68
GREENBEA	IIS_COV_1	410.50	3	2983	1955.8	9.0	6.3	4	2
GREENBEA	IIS_COV_2	359.95	4	2983	1319.6	17.8	15.0	4	2
GREENBEA	IIS_COV_3	322.34	3	10	1442.8	8.3	5.3	4	1
KLEIN3	IIS_COV_1	97.16	29	1571	506.0	101.7	101.7	95	95
KLEIN3	IIS_COV_2	49.77	12	1560	538.8	87.4	87.4	86	86
KLEIN3	IIS_COV_3	36.34	21	1137	205.6	85.8	85.8	84	84
MEXP	IIS_COV_1	12.37	2	377	157.0	8.0	8.0	7	7
MEXP	IIS_COV_3	8.76	1	176	237.5	6.0	6.0	6	6
MONDOU2	IIS_COV_1	17.69	14	503	60.8	163.5	146.6	22	17
MONDOU2	IIS_COV_2	118.63	13	503	58.1	182.3	163.8	22	17
MONDOU2	IIS_COV_3	17.11	9	460	62.6	146.9	128.9	22	17
PILOT4I	IIS_COV_1	28.82	1	659	930.0	1.0	1.0	1	1
PILOT4I	IIS_COV_2	18.05	1	1010	629.0	1.0	1.0	1	1
PILOT4I	IIS_COV_3	7.20	1	86	202.5	1.0	1.0	1	1

Table 5.4: Results for Large Problems

We feel that the min weight IIS cover will be a valuable tool in debugging LPs at the model development and integration stage.

In order to gain a perspective on the solution time to solve the MCIC problem **exactly**, we make a comparison with the heuristic approach of Chinneck [12] on a subset of the NETLIB infeasible library. Chinneck's runs were made using a modified version of MINOS(IIS) on an IBM-compatible 66 MHz 80486DX2 microcomputer. Chinneck's results are presented in [12] in terms of a time ratio: the covering time divided by the Phase 1 solution time. We convert our results into the same form, so the comparison is as fair as possible (considering that we use a different optimization package). These results are presented in Table 5.6.

It is interesting to note that Chinneck's heuristic finds the minimum cardinality cover in all instances; however, as demonstrated previously, it will not **guarantee** a minimal cover upon termination. The times presented are particularly intriguing since the time ratio to solve exactly is **smaller** on 6 of the 14 problems! We should also note that in solving exactly a large set of IISs is determined. Using Chinneck's algorithm, **no** IISs are identified. In [12], an extension is given for doing so, but the CPU times presented are for cover identification only. Thus, more information is given in less time on a substantial portion of the test bed by solving optimally!

Problem	MINOS(IIS)	Min IIS Cover		Smallest IIS	
	Rows	Cardinality	IISs Found	Size	Rows
BGDBG1	2	12	38	2	2
BGETAM	7	1	5	8	7
BGINDY		1	1	3	3
BGPRTR	5	1	1	7	7
BOX1	8	1	2	10	9
CERIA3D	73	1	3	73	73
CHEMCOM	7	1	2	23	13
CPLEX1	5	1	1	5	5
EX72A	58	1	3	59	58
EX73A	24	1	1	26	25
FOREST6	55	1	2	64	59
GALENET	2	1	1	6	3
GOSH		1	1	49	49
GREENBEA	1	2	3	4	1
ITEST2	3	2	3	3	3
ITEST6	2	2	3	3	3
KLEIN1	50	1	1	51	51
KLEIN2	51	1	4	53	53
KLEIN3	74	1	21	84	84
MEXP	5	1	1	6	6
MONDOU2	15	3	9	22	17
PANG	11	1	1	16	13
PILOT4I	1	1	1	1	1
QUAL	76	1	6	134	90
REACTOR	1	1	2	5	1
REFINERY	47	1	38	92	61
VOL1	80	1	10	137	91
WOODINFE	1	2	2	2	1

Table 5.5: Min IIS Cover and MINOS(IIS) IIS Isolation Comparison

Finally, we compare this approach with the fixed charge formulation presented in Section 2.4:

$$\begin{aligned}
 \min \quad & \sum_{i=1}^m z_i \\
 \text{subject to} \quad & \\
 & A\mathbf{x} + \mathbf{s} - \mathbf{s}' = \mathbf{b} \\
 & \mathbf{s}' \leq M\mathbf{z} \quad \text{for constant } M \text{ sufficiently large} \\
 & \mathbf{x}, \mathbf{s} \geq \mathbf{0} \\
 & \mathbf{z} \text{ binary}
 \end{aligned}$$

The fixed charge IP formulations were solved on the RS6000 computer using version 3.0 of the CPLEX optimization library. Results for this comparison are given in Table 5.7. The time given is CPU seconds to completely solve the problems. The Min IIS Cover results presented are for the variant IIS_COV_3. The test bed consists of a subset of problems from the NETLIB infeasible library. For small problems (WOODINFE, ITEST2, and ITEST6), the fixed charge approach outperforms the MWIC approach. However, the largest of these problems is WOODINFE with 35 constraints and 89 variables. As we move to larger problems, we see that the fixed charge formulation becomes increasingly more difficult to solve. CPLEX was unable to prove the optimality of the integer solution for the BGDBG1 problem, which has 348 constraints and 407 variables, given a maximum of 20,000 nodes for the branch and bound tree. The time to solve this problem is thus greater than the 825 seconds taken to evaluate 20,000 nodes. IIS_COV_3 solves the largest problem in this test bed, MONDOU2 (312 constraints and 604 variables), in approximately 6% of

the time the fixed charge formulation takes. Thus, as problem size increases, the MWIC formulation becomes the preferred algorithm.

Problem	MINOS(IIS)		Min IIS Cover	
	Cardinality	Time Ratio	Cardinality	Time Ratio
BGDBG1	12	55.5	12	24.6
BGINDY	1	0.2	1	3.4
BGPRTR	1	1.0	1	1.5
CHEMCOM	1	0.6	1	4.3
CPLEX1	1	1.3	1	1.2
GREENBEA	2	2.1	2	3.1
ITEST2	2	1.0	2	35.0
ITEST6	2	2.2	2	16.3
KLEIN2	1	1.7	1	1.4
KLEIN3	1	6.0	1	3.1
MONDOU2	3	69.3	3	26.3
REACTOR	1	7.1	1	1.9
REFINERY	1	7.1	1	10.6
WOODINFE	2	2.5	2	6.8

Table 5.6: Min IIS Cover and MINOS(IIS) IIS Cover Comparison

Problem	Fixed Charge	Min IIS Cover
	Time	Time
BGDBG1	>825.00	7.74
ITEST2	0.18	0.36
ITEST6	0.30	0.65
KLEIN2	307.50	5.38
MONDOU2	297.21	17.11
WOODINFE	0.24	0.48

Table 5.7: Min IIS Cover and Fixed Charge IIS Cover Comparison

6. Extensions to Solving the Linear Discriminant Problem

We now consider the problem of finding a linear separator, or **separating hyperplane** for two sets of data. Given two point sets A^1 and A^2 (where the coordinates of a point correspond to a row of the appropriate matrix), we wish to identify a hyperplane $H = \{\alpha | \alpha \mathbf{h} = b\}$ such that $\alpha \mathbf{h} > b$ if $\alpha \in A^1$ and $\alpha \mathbf{h} < b$ if $\alpha \in A^2$. If such a hyperplane H exists, we say that the sets A^1 and A^2 are **linearly separable**. In the case where the sets are not linearly separable, for any \mathbf{h} and b there exists a point (WLOG) $\alpha \in A^1$ such that $\alpha \mathbf{h} \leq b$ and we say that α is misclassified for this \mathbf{h} and b . Thus, if the sets are not linearly separable, we wish to obtain the hyperplane that minimizes some measure of misclassification of points. This classification problem has a variety of applications across a large spectrum of disciplines including pattern recognition and neural networks. Figure 6.1 demonstrates a linear separable system with an optimal separating hyperplane (left) and a non-linearly separable system with a hyperplane minimizing the number of misclassifications (right).

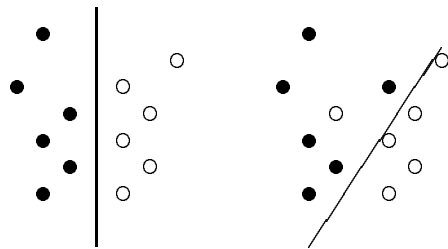


Figure 6.1: Linearly Separable (left) and Non-Separable (right) Sets

We can formulate the problem of finding a linear discriminant as

the following feasibility problem:

$$\begin{aligned}
 1. \quad & \alpha \mathbf{h} - b \geq \epsilon \quad \text{for all } \alpha \in A^1 \\
 2. \quad & \alpha \mathbf{h} - b \leq 0 \quad \text{for all } \alpha \in A^2 \\
 & \mathbf{h}, b \text{ free}
 \end{aligned}$$

Note that **any** feasible solution \mathbf{h} and b to the linear program will define a hyperplane separating the sets A^1 and A^2 . We force a true separation of the sets by restricting the points of A^1 to be distance at least ϵ from the hyperplane. This also eliminates the trivial solution $\mathbf{h} = \mathbf{0}$, $b = 0$. We need not force the points of set A^2 off of $\alpha \mathbf{h} = b$, since our “true” separator is the hyperplane $\alpha \mathbf{h} = b + (\epsilon/2)$. Also note that the choice of objective is arbitrary at this point - if the sets are separable, we care only to find a separating hyperplane. There are many modifications we can make to this formulation in order to “bias” the hyperplane towards one set or the other (for example, see Glover [20] or Bennett and Mangasarian [5]).

This linear programming formulation is infeasible if there is no separating hyperplane. If we find the minimum weight IIS cover, we have found the minimum weight subset of points which must be removed from the system in order for a feasible solution to be found. In other words, we will find the minimum weight subset of points which must be misclassified in order that a separator be found for the remaining points.

Using Theorem 5.1, we obtain the following alternative system

for the discriminant problem:

$$\begin{aligned}
1. \quad & \mathbf{y}^T A^1 + \mathbf{z}^T A^2 = 0 \\
2. \quad & \mathbf{y}^T - \mathbf{z}^T = 0 \\
3. \quad & -\mathbf{y}^T \boldsymbol{\epsilon} = -1 \\
& \mathbf{y} \leq \mathbf{0}, \mathbf{z} \geq \mathbf{0}
\end{aligned}$$

Since the problem of minimizing the number or weight of misclassified points is NP hard, research efforts have focused on identifying linear programming formulations which minimize different measures of the misclassification and on heuristic approaches to approximating the true minimum cardinality/weight set of misclassified points.

We can compare the performance of this exact formulation with a linear programming formulation due to Bennett and Mangasarian [5]. Let us assume that there are m points in A^1 and k points in A^2 . Then Bennett and Mangasarian solve the following:

$$\begin{aligned}
\min \quad & \sum_{i=1}^m e_i/m + \sum_{i=1}^k f_i/k \\
\text{subject to} \quad & \\
& \boldsymbol{\alpha} \mathbf{h} - b + e_i \geq \epsilon \quad \text{for all } \boldsymbol{\alpha} \in A^1 \\
& \boldsymbol{\alpha} \mathbf{h} - b - f_i \leq 0 \quad \text{for all } \boldsymbol{\alpha} \in A^2 \\
& \mathbf{e}, \mathbf{f} \geq \mathbf{0} \\
& \mathbf{h}, b \text{ free}
\end{aligned}$$

The principal property that distinguishes this formulation from other linear programming methods is that for the linearly inseparable case this formulation will always generate a nontrivial solution to the linear discriminant problem and does so without the addition of extra constraints. Note the similarity between this formulation and a standard Phase 1 LP;

the vectors \mathbf{e} and \mathbf{f} are the artificial or elastic variable vectors of a Phase 1 formulation, and the objective seeks to minimize a normalization of the sum of infeasibilities.

In [5], Bennett and Mangasarian compare this formulation with a number of others on samples consisting of data points from the Wisconsin Breast Cancer Database (see the same article for a complete description). This data was collected by by Dr. William H. Wolberg of University of Wisconsin Hospitals in Madison. The data base consists of malignant and benign samples collected in a 9-dimensional space. Of the formulations they examined, theirs performed the best in terms of both fewest misclassifications and computer time used.

This application lends itself to a deeper discussion of the practicality of the linear discriminant problem. What we are really looking for is a solution, or separating hyperplane, which can be found on a small sample set of the data. This sample set is commonly called a training set in the neural network community. Once we have obtained this separating hyperplane, it can be used to make decisions about the category which other data points fall into. So, by looking at a representative sample of data points whose classification we know, we wish to identify a classification strategy and then test the “classifier” on expanded data sets. The Wisconsin Breast Cancer Database is being used to develop an intelligent tissue analyzer.

In order to evaluate the effectiveness of our MWIC algorithm, we made a set of comparisons on the same data set. We used the 353 data points of Group 1 of the data set, of which 165 are malignant and 188 are

benign. The data set is inseparable, so we seek to minimize the number of misclassified points. The main question to be answered here is how well do the current state of the art LP formulations perform in this task and does our exact approach solve the problem in a timely manner.

Our alternative system is based upon the previously given feasibility problem, which is identical to Bennett and Mangasarian's LP formulation with the exclusion of the elastic variables e and f .

We noted the similarity between the Bennett and Mangasarian formulation and that of a Phase 1 LP, and also decided to see how much better a solution they find over a standard Phase 1 LP. In our comparisons, we also included a version of their formulation without the elastic variables e and f . Upon completion of a Phase 1 LP, the artificial variables correspond to data points which have been misclassified.

The Phase 1 LP and Bennett and Mangasarian's formulation were run using the CPLEX optimization package. We modified a version of our best performing MWIC algorithm, IIS_COV_3. The algorithm was modified to find a set of disjoint IISs prior to solving the covering subproblem. The idea behind this modification was to reduce the amount of time needed to solve the MCIC problem optimally. Again, this code is based upon the IBM OSL optimization package object libraries. All comparisons were run on an IBM RS6000. The results are presented in Table 6.1

	IIS_COV_3	B-M	Phase 1
Misclassified	6	25	25
CPU Time (sec)	739.3	1.3	1.0

Table 6.1: Linear Discriminant Analysis Comparison

It is interesting to note that the Bennett-Mangasarian (B-M) formulation misclassifies the same number of points as a standard Phase 1 LP. From this example, it appears that the extra effort in solving these problems exactly is worth it from a solution quality viewpoint. Although the exact solution took approximately 570 times longer to reach than the B-M formulation, it still took less than 15 minutes of computing time - relatively small for a problem which you would be interested in solving only once.

Another interesting result is that all 6 points which were misclassified by the MCIC algorithm were benign - even without explicitly weighting misclassifications in this direction as being more favorable. In contrast, for both the Phase 1 and B-M formulations, 14 malignant and 11 benign points were misclassified. In a true classification system, it would be desirable to err on the side of classifying benign as malignant rather than the reverse.

It appears that this approach is a viable means of solving linear discriminant problems of fair magnitude.

7. Summary and Areas for Further Research

We have developed an algorithm for finding the MWIC of an infeasible system of linear equations. We have demonstrated the utility of this algorithm for infeasibility analysis of linear programming problems and in linear discriminant analysis where we choose to minimize the number of misclassified points.

In addition to this algorithmic development, we have noted that these covering problems have a structure to them which seems to make them more tractable than general set covering problems. We have explored the facial structure of these covering polyhedra and identified facet defining inequalities. We have also generalized these results to a larger class of covering problems, called polyhedral covering problems.

We have reviewed the theory of irreducible inconsistent subsystems and provided a generalization of Van Loon's result. Additionally, we have established a link between the method of Gleeson and Ryan with Van Loon's.

We have evaluated a number of prototypes of our algorithm on the NETLIB infeasible library and demonstrated the utility of these algorithms on a few industry examples of infeasible LPs. The problems in the NETLIB infeasible library in many instances are feasible problems which have been perturbed, perhaps artificially, into infeasible instances. At this point, it would be interesting to analyze other infeasible industry problems to determine on what types of problems the MWIC approach provides a

useful infeasibility isolation.

We have also compared this algorithm with a heuristic proposed by Chinneck, and found that on many problems solving optimally takes less time than solving heuristically. This may be due in part to the differences in implementation. Our algorithms solve the alternative system of the original infeasible system, which is essentially the dual of that problem. Chinneck's method involves solving many Phase 1 instances of subproblems of the original infeasible system. Such factors as cover size, size of individual IISs, and problem size probably contribute to making the optimal cover more tractable.

We are also interested in determining the utility of identifying IISs by searching for extreme rays of the alternative system directly (either in the alternative system, or in the original infeasible system). This may help speed up the algorithm a small amount. A version of our code which finds a maximal set of disjoint IISs before solving the covering subproblem also looks very promising in terms of reducing the number of IISs necessary to find the optimal cover (and thus reducing total CPU time).

We would also be interested in determining if inclusion of facets in a modified branch and cut algorithm would be beneficial to reducing the time needed in the covering subproblem. At the other extreme, we could try to minimize the amount of time spent on the covering subproblem by using a heuristic.

BIBLIOGRAPHY

- [1] Amaldi E., From Finding Maximum Feasible Subsystems of Linear Systems to Feedforward Neural Network Design, Ph.D. Dissertation, École Polytechnique Fédérale de Lausanne, 1994.
- [2] Amaldi E., and Kann V., The Complexity and Approximability of Finding Maximum Feasible Subsystems of Linear Relations, to appear in *Theoretical Computer Science*. (Also appearing as Technical Report ORWP-11-93, École Polytechnique Fédérale de Lausanne, 1993).
- [3] Amaldi E., and Kann V., On the Approximability of Removing the Smallest Number of Relations from Linear Systems to Achieve Feasibility, Technical Report ORWP-6-94, École Polytechnique Fédérale de Lausanne, 1994.
- [4] Balas E., and Ng S., On the Set Covering Polytope: I. All the Facets with Coefficients in $\{0, 1, 2\}$, *Mathematical Programming*, Vol. 43 (1989).
- [5] Bennett K.P., and Mangasarian O.L., Robust Linear Programming Discrimination of Two Linearly Inseparable Sets, *Optimization Methods and Software*, Vol. 1 (1992), pp. 23-34.
- [6] Berge C., *Hypergraphs*, North-Holland, Amsterdam, 1989.
- [7] Boneh A., Identification of Redundancy by a Set Covering Equivalence, in *Operational Research 84*, edited by Brans J.P., Elsevier Science Publishers B. V., Amsterdam, The Netherlands, 1984, pp. 407-422.
- [8] Carver W. B., Systems of Linear Inequalities, *Annals of Mathematics*, Vol. 23 (1921-1922), pp. 212-220.
- [9] Chakravarti N., Some Results Concerning Post-Infeasibility Analysis, *European Journal of Operational Research*, Vol. 73 (1994), pp. 139-143.

- [10] Chinneck J., Finding Minimal Infeasible Sets of Constraints in Infeasible Mathematical Programs, Technical Report SCE-93-01, Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada, 1993.
- [11] Chinneck J., Finding the Most Useful Subset of Constraints for Analysis in an Infeasible Linear Program, Technical Report SCE-93-07, Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada, 1993.
- [12] Chinneck J., An Effective Polynomial Time Algorithm for the Minimum Cardinality IIS Set Covering Problem, Technical Report SCE-95-02, Department of Systems and Computer Engineering, Carleton University, Ottawa, Canada, 1995.
- [13] Chinneck J., and Dravnieks E., Locating Minimal Infeasible Constraint Sets in Linear Programs, *ORSA Journal on Computing*, Vol. 3 (1991), pp. 157-168.
- [14] Chvátal V., A Greedy Heuristic for the Set Covering Problem, *Mathematics of Operations Research*, Vol. 4 (1979), pp. 233-235.
- [15] Debrosse C., and Westerberg A., A Feasible-Point Algorithm for Structured Design Systems in Chemical Engineering, *American Institute of Chemical Engineers Journal*, Vol. 19 (1973), pp. 251-258.
- [16] Dyer, M., The Complexity of Vertex Enumeration Methods, *Mathematics of Operations Research*, Vol. 8 (1983), pp. 381-402.
- [17] Fan K., On Systems of Linear Inequalities, in *Annals of Mathematical Studies Number 38: Linear Inequalities and Related Systems*, edited by Kuhn H.W. and Tucker A.W., Princeton University Press, Princeton, NJ, 1956, pp. 99-156.
- [18] Gary M., and Johnson D., *Computers and Intractability*, W.H. Freeman, New York, 1979.
- [19] Gleeson J., and Ryan J., Identifying Minimally Infeasible Subsystems of Inequalities, *ORSA Journal on Computing*, Vol. 2 (1990), pp. 61-63.
- [20] Glover F., Improved Linear and Integer Programming Models for Discriminant Analysis, in *Creative and Innovative Approaches to the*

Science of Management, edited by Ijiri Y., Quorum Books, Westport, CT, 1993.

- [21] Greenberg H.J., Consistency, Redundancy, and Implied Equalities in Linear Systems, UCD/CCM Report No. 14, Mathematics Department, University of Colorado at Denver, 1993.
- [22] Greenberg H.J., Diagnosing Infeasibility in Min-cost Network Flow Problems Part II: Primal Infeasibility, *IMA Journal of Mathematics Applied in Business and Industry*, Vol. 2 (1988/9), pp. 39-50.
- [23] Greenberg H.J., Diagnosing Infeasibility in Min-cost Network Flow Problems Part I: Dual Infeasibility, *IMA Journal of Mathematics in Management*, Vol. 1 (1987), pp. 99-109.
- [24] Greenberg H.J., An Empirical Analysis of Infeasibility Diagnosis for Instances of Linear Programming Blending Models, *IMA Journal of Mathematics Applied in Business and Industry*, Vol. 4 (1992), pp. 163-210.
- [25] Greenberg H.J., How to Analyze the Results of Linear Programs Part 3: Infeasibility Diagnosis, *Interfaces*, Vol. 23, No. 6 (1993).
- [26] Greenberg H.J., and Murphy F.H., Approaches to Diagnosing Infeasible Linear Programs, *ORSA Journal on Computing*, Vol. 3, No. 3 (1991), pp. 253-261.
- [27] Motzkin T.S., Contributions to the Theory of Linear Inequalities, Doctoral Dissertation, University of Basel, 1933, translated by D.R. Fulkerson, in *Theodore S. Motzkin: Selected Papers*, edited by Cantor D., Gordon B., and Rothschild B., Birkhauser, 1983.
- [28] Murty K., *Linear Programming*, John Wiley and Sons, 1983.
- [29] Nemhauser G., and Wolsey L., *Integer and Combinatorial Optimization*, John Wiley and Sons, 1988.
- [30] Pulleyblank W., Personal Communication to J. Ryan.
- [31] Roodman G., Post-Infeasibility Analysis in Linear Programming, *Management Science*, Vol. 25 (1979), pp. 916-922.
- [32] Ryan J., Transversals of HHS-Hypergraphs, *Congressus Numerantium*, Vol. 81 (1991), pp. 17-22.

- [33] Ryan J., IIS-Hypergraphs, preprint, 1993.
- [34] Sankaran J.K., A Note on Resolving Infeasibility in Linear Programs by Constraint Relaxation, *Operations Research Letters*, Vol. 13 (1993), pp. 19-20.
- [35] Schrijver, A., *Theory of Linear and Integer Programming*, John Wiley and Sons, 1986.
- [36] Van Loon J., Irreducibly Inconsistent Systems of Linear Inequalities, *European Journal of Operations Research*, Vol. 8 (1981), pp. 283-288.