VALID INEQUALITIES FOR MIXED-INTEGER LINEAR PROGRAMMING PROBLEMS

BY EMRE YAMANGIL

A dissertation submitted to the

Graduate School—New Brunswick

Rutgers, The State University of New Jersey

in partial fulfillment of the requirements

for the degree of

Doctor of Philosophy

Graduate Program in Operations Research

Written under the direction of

Endre Boros

and approved by

New Brunswick, New Jersey

October, 2015

ABSTRACT OF THE DISSERTATION

Valid Inequalities for Mixed-integer Linear Programming Problems

by Emre Yamangil

Dissertation Director: Endre Boros

In this work we focus on various cutting-plane methods for Mixed-integer Linear Programming (MILP) problems. It is well-known that MILP is a fundamental hard problem and many famous combinatorial optimization problems can be modeled using MILP formulations. It is also well-known that MILP formulations are very useful in many real life applications.

Our first, rather theoretical, contribution is a new family of superadditive valid inequalities that are obtained from value functions of special surrogate optimization problems. Superadditive functions hold particular interest in MILP as they are fundamental in building integer programming duality, and all "deepest valid inequalities" are known to arise from superadditive functions. We propose a new family of superadditive functions that generate inequalities that are at least as strong as Chvatal-Gomory (CG-) inequalities. A special subfamily provides a new characterization of CG-cuts. Value functions of optimization problems are known to be super additive. We look at special surrogate optimization problems, and measure their complexity in terms of the number of integer variables in them. It turns out that the lowest possible nontrivial complexity class here includes all CG-cuts, and provides some stronger ones, as well.

Our next contribution is a practically efficient, polynomial time method to produce

"deepest" cuts form multiple simplex rows for the so called corner polyhedra. These inequalities have been receiving considerable attention lately. We provide a polynomial time column-generation algorithm to obtain such inequalities, based on an arbitrary (fixed) number of rows of the simplex tableau. We provide numerical evidence that these inequalities improve the CPLEX integrality gap at the root node on a well-known set of MILP instances, MIPLIB.

In the last chapter, we consider a particular MILP instance, Optimal Resilient Distribution Grid Design Problem (ORDGDP). This is a problem of critical importance to infrastructure security and recently attracted a lot of attention from various government agencies (e.g. Presidential Policy Directive of Critical Infrastructure Security 2013). We formulate this problem as a MILP and propose various efficient solution methods blending together well-known decomposition ideas to overcome the numerical intractability encountered using commercial MILP software such as CPLEX.

Acknowledgements

First of all, this work wouldn't be possible if it wasn't for Professor Endre Boros, my mentor. I will forever feel privilaged to have witnessed his outstanding knowledge that always made every problem simpler and easier to comprehend. Not only a natural teacher, his reinforcement has made me survive the occasional wildfires of graduate study.

Another large chunk of this work was done under the supervision of Dr. Russell Bent during my internship at Los Alamos National Laboratory. I only hope that I will be able to reciprocate his excellence and kindness throughout our future collaborations.

I also want to thank my family, Behiye, Fuat, and Elif Yamangil, from the bottom of my heart. Without my mom always believing in me, my father's indispensable words of wisdom, and my sister singlehandedly taking me to my Neverland, I wouldn't even be here.

Lastly, I want to thank Svetlana Soloveva for her unwavering support. It is such rare thing to find a person that is so unselfish and willing to share, but also sincerely caring and full of unconditional love.

This work was partially supported by the National Science Foundation (Grant IIS-1161476) and by the DOE-OE Smart Grid R&D Program in the Office of Electricity in the US Department of Energy, the DTRA Basic Research Program, and the Center for Nonlinear Studies (It was carried out under the auspices of the National Nuclear Security Administration of the U.S. Department of Energy at Los Alamos National Laboratory under Contract No. DE-AC52-06NA25396).

Dedication

To my family and my love.

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Mixed-integer Linear Programming (MILP) algorithms are a well-known method to solve many combinatorial optimization problems that are typically NP-complete. A few popular examples are Traveling Salesman Problem ([78]), Vehicle Routing Problem ([5]), Stable Set Problem ([7]), Satisfiability Problem ([19]), and many others. In these works, authors describe the problem as a MILP instance, and exploit the two fundamental ideas to solve a MILP: successive approximation of integral hull, i.e. generation of cutting planes ([43]), and to branch into two subproblems on a fractional variable, i.e. branch and bound ([62]). Even though Gomory's work on mixed integer rounding cuts has been one of the earlier methods in the literature that can be proved to be finite for pure integer linear programs, branch and bound methods have ruled the early era without much challenge. Until late 20th century cutting-planes application has been of rather theoretical use. However recently, computational MILP started to see some promising results in applying various cutting-planes (Gomory's mixed integer rounding cuts in particular [26]) as a way to achieve significant speed-ups to some of the benchmark problems that would take CPU years with the conventional branch and bound technique ([85]). The goal of such an approach is to strengthen the Linear Programming (LP) relaxation at various nodes of a branch and bound tree, and in particular the root node ([15]).

1.1 Motivation

MILP researchers of the past few decades focused on generating inequalities because of the following: MILP is NP-hard ([57]) because of the integrality of its variables.

Therefore a logical approach is to relax the complicating conditions and solve the problem, as the common practice in many relevant areas of Operations Research (Nonlinear Programming [61], Benders Decomposition [102], Lagrangean Relaxation [50]). When we relax these conditions (i.e. LP relaxation), we usually end up with an infeasible solution. The infeasibility implies that there exists a separating hyperplane that separates the set of feasible solutions, in the realm of integer programming, the integral hull (the convex hull of the integer solutions within the feasible region), from the infeasible solution. The ways to generate such inequalities wildly differ (Gomory cuts [43], Split cuts [25], Intersection cuts [6], Lift-and-Project cuts [7], and many others). But the core idea is always the same, valid inequalities.

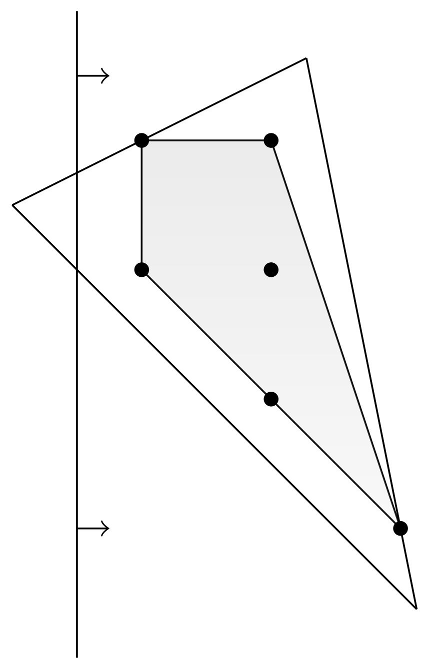


Figure 1.1: An example of a valid inequality for the integer core of a polyhedron, which we can use to strengthen the current formulation.

The key question is which one of these valid inequalities (see e.g. Figure 1.1) are better than the others and will help with the general branch and bound process more, as the goal is always the same, to decrease the size of the branch and bound tree as much as possible. In this work we will provide algorithms on how to generate a “depest” cut. Throughout this dissertation, we will use the polyhedron in Figure 1.1 to highlight various valid inequalities:

$$P = \left\{ x \in \mathbb{R}^2 : \begin{bmatrix} -1 & 2 \\ 5 & 1 \\ -2 & -2 \end{bmatrix} x \leq \begin{bmatrix} 6 \\ 21 \\ -9 \end{bmatrix} \right\}$$

1.2 Contribution

In this work, we focus on various ways to generate such valid inequalities. In particular Multi-row Simplex Cuts generation.

We first focus on super additive functions to generate valid inequalities for a general MILP. We make connections between super additive valid inequalities and CG-closure and characterize the CG-closure of a MILP in terms of a corresponding family of superadditive functions. We also show that the discrete Farkas' lemma of Lasserre is equivalent with super additive duality for integer programming.

Second we propose polynomial time methods to generate multi-row simplex cuts. Most famous inequalities used in state-of-the-art MILP solvers are usually cuts that are derived from multiple rows of the simplex table. In this work, we investigate ways to generate inequalities, that are as deep as possible, in the sense of a weighted sum of its coefficients. We develop our methods using the well-known intersection cut theory of Balas [6].

Finally we discuss an important problem in Power Distribution Resiliency. Modern society is critically dependent on the services provided by engineered infrastructure networks. When natural disasters (e.g. Hurricane Sandy) occur, the ability of these networks to provide service is often degraded because of physical damage to network components. One of the most critical of these networks is the electrical distribution grid, with medium voltage circuits often suffering the most severe damage. However, well-placed upgrades to these distribution grids can greatly improve post-event network performance. We formulate an optimal electrical distribution grid design problem as a two-stage, stochastic mixed-integer program with damage scenarios from natural disasters modeled as a set of stochastic events. This is a problem of critical importance to energy community where optimization and AI researchers can make significant contributions. AI has made many recent significant contributions to energy problems [49, 88, 23, 40, 89, 52, 87, 97, 99]. We develop computationally efficient algorithms for solving stochastic network design problems with discrete variables at each stage. The algorithms are based on hybrid optimization methods similar to recent work that combines Bender's Decomposition with heuristic master solutions [86].

Chapter 2

Preliminaries

In this chapter we survey some of the well-known and classical valid inequalities that are widely employed by MILP software. For a deeper understanding, the reader may refer to the book of Nemhauser and Wolsey [77] or the paper of Cornuéjols [27]. Here we will provide some of the fundamentals of Gomory Mixed Integer cuts, Lift-and-Project cuts, and Split cuts.

2.1 On Valid Inequalities

In this section we will focus on the fundamental theory of valid inequalities. Let $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ be a non-empty polyhedron, where $A \in \mathbb{Q}^{m \times n}$ and $b \in \mathbb{Q}^m$ are rational matrices. Let $P_I = \text{conv}(P \cap \mathbb{Z}^n)$ be the integral hull, in other words, the convex hull of integral feasible vectors in P.

Definition 2.1.1. *We call $\alpha^T x \leq \beta$ a valid inequality for $S \subset \mathbb{R}^n$ if $\alpha^T \bar{x} \leq \beta$ for all $\bar{x} \in S$.*

Now let us recall a key result from polyhedral theory:

Lemma 2.1.2 (Farkas' Lemma)**.** *Given $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$, one and only one of the following systems has a solution:*

i $\exists\, x \in \mathbb{R}^n$ *s.t.* $Ax = b$, $x \geq 0$,

ii $\exists\, y \in \mathbb{R}^m$ *s.t.* $A^T y \leq 0$ *and* $b^T y > 0$.

Farkas' Lemma readily implies that valid inequalities for a polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ are linear consequences of the set of inequalities $Ax \leq b$. In other words, an inequality for P, $\alpha^T x \leq \beta$ is valid, if and only if there exists $u \in \mathbb{R}^m_+$ such that

$u^T A \geq \alpha^T$ and $u^T b \leq \beta$. This further implies that we can generate valid inequalities using the following linear programming problem:

$$\begin{array}{lrcl} \max & \alpha^T w - \beta & & \\ \text{s.t.} & u^T A & \geq & \alpha \\ & u^T b & \leq & \beta \\ & u & \geq & 0 \end{array} \tag{2.1}$$

where $w \in \mathbb{R}^n$ is a vector of weights for the coefficient of the valid inequality to be generated (or a vector in $\mathbb{R}^n$ that we want to separate from the polyhedron).

Variations of the so-called cut generating Linear Programming Problem (2.1) appeared many times in the literature [26] and we will also use similar methods to generate valid inequalities for P_I that is the integer hull of a polyhedron. Though Farkas' Lemma proves to be powerful for polyhedral sets, P, when we move to the realm of integrality, this is quite a different story. First of all for many optimization problems, such as the Traveling Salesman Problem, the polyhedral description of P_I is known to be of exponential size. Even for problems such as Maximum Weighted Matching, where we know a polynomial time algorithm exists, the odd circuit constraints are exponential in the size of the graph. Which means that it is not likely that we will find a polyhedral description of the integral hull to characterize all valid inequalities for P_I. Instead researchers focused on generating valid inequalities using the integrality of variables in various ways that we will recall in the following sections.

An important issue to note before we move on, is the dominance of valid inequalities:

Definition 2.1.3. *Let ${\alpha^1}^T x \leq \beta^1$ and ${\alpha^2}^T x \leq \beta^2$ be two inequalities. We say ${\alpha^1}^T x \leq \beta^1$ dominates ${\alpha^2}^T x \leq \beta^2$ if $\alpha^1_i \geq \alpha^2_i$ $\forall i$ and $\beta^1 \leq \beta^2$ and at least one of these inequalities satisfied as a strict inequality.*

In this work we will apply the notion of dominance for the family of valid inequalities of a polyhedron, in particular we shall seek the un-dominated valid inequalities.

2.2 Gomory Mixed Integer Cuts

There are two well-known derivations of Gomory's Mixed Integer Rounding cuts. In this work we will follow Gomory's derivation. Now let us consider the following mixed integer set:

$$S = \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j=1}^{n} a_j x_j + \sum_{j=1}^{m} d_j y_j = b\} \tag{2.2}$$

Let us introduce $f_j = a_j - \lfloor a_j \rfloor$, $g_j = d_j - \lfloor d_j \rfloor$ and $f_0 = b - \lfloor b \rfloor$. Then we can rewrite (2.2) as follows:

$$\begin{aligned}
S &= \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j=1}^{n} a_j x_j + \sum_{j=1}^{m} d_j y_j = b\} \\
&= \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} (\lfloor a_j \rfloor + f_j) x_j + \sum_{j:f_j > f_0} (\lceil a_j \rceil + f_j - 1) x_j + \sum_{j=1}^{m} d_j y_j = \lfloor b \rfloor + f_0\} \\
&= \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} f_j x_j + \sum_{j:f_j > f_0} (f_j - 1) x_j + \sum_{j=1}^{m} d_j y_j = k + f_0\}
\end{aligned}$$

where $k = \lfloor b \rfloor - \sum_{j:f_j \le f_0} \lfloor a_j \rfloor x_j + \sum_{j:f_j > f_0} \lceil a_j \rceil x_j \in \mathbb{Z}$ for all $x \in \mathbb{Z}_+^n$, therefore either $k \le -1$ or $k \ge 0$. Consequently, we can write

$$\begin{aligned}
S &= \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} f_j x_j + \sum_{j:f_j > f_0} (f_j - 1) x_j + \sum_{j=1}^{m} d_j y_j = k + f_0\} \\
&\subseteq \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} f_j x_j + \sum_{j:f_j > f_0} (f_j - 1) x_j + \sum_{j=1}^{m} d_j y_j \le -1 + f_0\} \\
&\qquad \cup \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} f_j x_j + \sum_{j:f_j > f_0} (f_j - 1) x_j + \sum_{j=1}^{m} d_j y_j \ge f_0\} \\
&= \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : -\sum_{j:f_j \le f_0} \frac{f_j}{1 - f_0} x_j + \sum_{j:f_j > f_0} \frac{1 - f_j}{1 - f_0} x_j - \sum_{j=1}^{m} \frac{d_j}{1 - f_0} y_j \ge 1\} \\
&\qquad \cup \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} \frac{f_j}{f_0} x_j - \sum_{j:f_j > f_0} \frac{1 - f_j}{f_0} x_j + \sum_{j=1}^{m} \frac{d_j}{f_0} y_j \ge 1\} \\
&\subseteq \{(x, y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m : \sum_{j:f_j \le f_0} \max\{-\frac{f_j}{1 - f_0}, \frac{f_j}{f_0}\} x_j + \sum_{j:f_j > f_0} \max\{\frac{1 - f_j}{1 - f_0}, -\frac{1 - f_j}{f_0}\} x_j \\
&\qquad + \sum_{j=1}^{m} \max\{-\frac{d_j}{1 - f_0}, \frac{d_j}{f_0}\} y_j \ge 1\}
\end{aligned}$$

$$S \subseteq \{(x,y) \in \mathbb{Z}_+^n \times \mathbb{R}_+^m :$$
$$\sum_{j:f_j \le f_0} \frac{f_j}{f_0} x_j + \sum_{j:f_j > f_0} \frac{1-f_j}{1-f_0} x_j - \sum_{j:d_j \le 0}^{m} \frac{d_j}{1-f_0} y_j + \sum_{j:d_j > 0}^{m} \frac{d_j}{f_0} y_j \ge 1\}. \quad (2.3)$$

Last inequality is obviously valid for S and is known as the Gomory Mixed Integer Rounding cut. To compute a Gomory cut, let us consider our polyhedron in standard form:

$$P = \{x \in \mathbb{Z}_+^2, y \in \mathbb{R}_+^3 : \begin{bmatrix} -1 & 2 & 1 & 0 & 0 \\ 5 & 1 & 0 & 1 & 0 \\ -2 & -2 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 6 \\ 21 \\ -9 \end{bmatrix}\}$$
$$= \{x \in \mathbb{Z}_+^2, y \in \mathbb{R}_+^3 : \begin{bmatrix} 1 & 0 & -0.0909 & 0.1818 & 0 \\ 0 & 1 & 0.4545 & 0.0909 & 0 \\ 0 & 0 & 0.7273 & 0.5455 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3.2727 \\ 4.6364 \\ 6.8182 \end{bmatrix}\}$$

where the second representation is obtained using the Gaussian elimination operator $B^{-1} = [a_1, a_2, a_5]^{-1}$. Now if we apply formula (2.3) to the first row and we obtain the valid inequality shown in Figure 2.2:

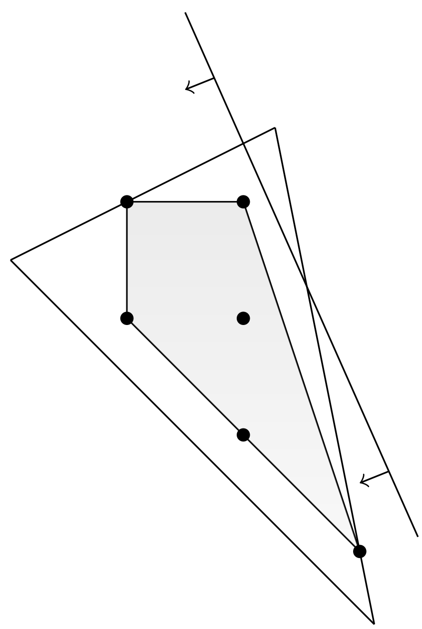


Figure 2.1: A Gomory mixed integer rounding cut

2.3 Split Cuts

Split cuts were introduced by Cook, Kannan and Scrijver [25] and it follows the following logic. Let us consider a polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$, $\pi \in \mathbb{Z}^n$ and $\pi_0 \in \mathbb{Z}$. It is straightforward to see that:

$$
\begin{aligned}
P_I \cap \mathbb{Z}^n &= P \cap \mathbb{Z}^n \\
&= \left(P \cap \mathbb{Z}^n \cap \{x \in \mathbb{R}^n : \pi^T x \leq \pi_0\}\right) \cup \left(P \cap \mathbb{Z}^n \cap \{x \in \mathbb{R}^n : \pi^T x \geq \pi_0 + 1\}\right) \\
&\subseteq \left(P \cap \{x \in \mathbb{R}^n : \pi^T x \leq \pi_0\}\right) \cup \left(P \cap \{x \in \mathbb{R}^n : \pi^T x \geq \pi_0 + 1\}\right) \\
&\subseteq \operatorname{conv}\left(\left(P \cap \{x \in \mathbb{R}^n : \pi^T x \leq \pi_0\}\right) \cup \left(P \cap \{x \in \mathbb{R}^n : \pi^T x \geq \pi_0 + 1\}\right)\right) = P_S
\end{aligned}
$$

An inequality valid for P_S is called a split cut. To compute a split cut let us first consider the following set:

$$
P = \{x \in \mathbb{Z}^2_+, y \in \mathbb{R}^3_+ : \begin{bmatrix} 1 & 0 & -0.0909 & 0.1818 & 0 \\ 0 & 1 & 0.4545 & 0.0909 & 0 \\ 0 & 0 & 0.7273 & 0.5455 & 1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 3.2727 \\ 4.6364 \\ 6.8182 \end{bmatrix} \}
$$

and the split:

$$
\pi^T \begin{bmatrix} x \\ y \end{bmatrix} \leq \pi_0 \text{ or } \pi^T \begin{bmatrix} x \\ y \end{bmatrix} \geq \pi_0 + 1
$$

where $\pi^T = [1, 0]$ and $\pi_0 = 3$. Derivation of Split cuts are based on the more general theory of intersection cuts (see Balas [6]). We shall describe this method in detail in Chapter 4. Here we provide some intuitive arguments for simplicity. The linear distance of the current simplex solution to the split planes, can be computed as $\epsilon_1 = \pi^T \bar{x} - \pi_0 = 0.2727$ and $\epsilon_2 = \pi_0 + 1 - \pi^T \bar{x} = 0.7273$, respectively. Let γ_j denote the nonbasic columns of simplex tableau (that only includes fractional rows that correspond to integral basic variables) and $\bar{x}^T = [3.2727, 4.6364]$ be the basic feasible solution we want to separate. Then we can calculate the magnitude of intersection points by $\alpha_j = -\frac{\epsilon_1}{\pi^T \gamma_j}$ if $\pi^T \gamma_j < 0$, otherwise by $\alpha_j = \frac{\epsilon_2}{\pi^T \gamma_j}$, for all nonbasic indices j (let us note that all valid inequalities can be written as a nonnegative summation of nonbasic variables that is greater than or equal to a nonnegative real, this will be discussed on Chapter 4). Which gives rise to $0.125y_1 + 0.667y_2 \geq 1$ or equivalently $3.2083x_1 + 0.9167x_2 \leq 13.75$. This inequality is depicted on Figure 2.3:

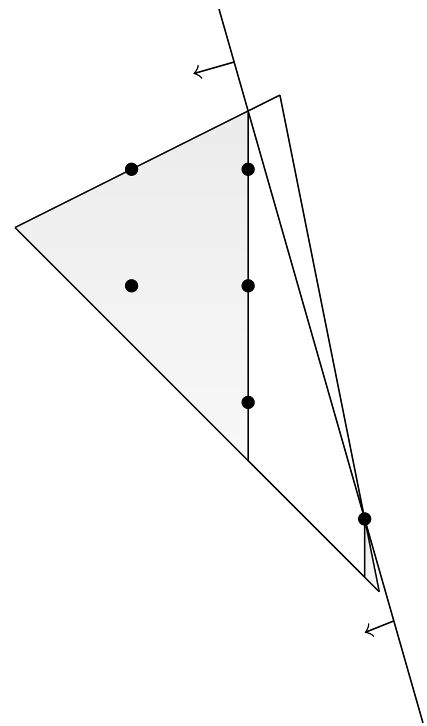


Figure 2.2: A split cut

Split cuts play an important role in today's mixed integer programming software (crooked cross split cuts in particular [29], or t-branch split cuts in general [69]). In general they are a special case of intersection cuts, where explicit formulas can be derived given a split. Though it is well-known that the separation over the split closure is in general NP-complete [17].

2.4 Lift-and-Project Cuts

For now let us consider a polyhedron $P = \{x \in \mathbb{R}^n : Ax \leq b\}$ and its binary hull $P_B = \mathrm{conv}\,(P \cap \mathbb{B}^n)$. Lift-and-Project is mainly an idea to find an intermediate set, M, between P and P_B (akin to CG-closure) by first representing P on higher dimensional space where it is strengthened by introducing more constraints (that are not necessarily linear) and by exploiting binary feasibility, tigthen the higher dimensional formulation formulation. Finally we project the lifted set to $\mathbb{R}^n$ to get a new approximation of P_B. Inequalities that fall between M and P are called Lift-and-Project cuts.

The most famous schemes to Lift-and-Project a polyhedral set is Sherali and Adams [98], Lovasz and Schrijver [71] and Balas, Ceria and Cornuejols [7]. Here we will provide the derivation of Balas, Ceria and Cornuejols family. However the other two families of Lift-and-Project methods follow similar lines and differ by the monomial multiplication of constraints.

First we strengthen the formulation by introducing nonlinearities:

$$M_k = \{x \in \mathbb{R}^n_+ : x_k(Ax - b) \leq 0, (1 - x_k)(Ax - b) \leq 0\}$$

Let us note that $M_k \subseteq P$ (by summing up $x_k(Ax - b) \leq 0$ and $(1 - x_k)(Ax - b) \leq 0$ we get $Ax - b \leq 0$). And let us further note that $P_B \subseteq M_k$ because all $x \in P_B$ implies $x_B \in M_k$ trivially. Considering binary variables, we know that $x_k = x_k^2$, for all j. Let us introduce new variables $y_j = x_k x_j$, for all j. Furthermore let A_k be the matrix A with the kth column, a_k, omitted.

$$M'_k = \{x \in \mathbb{R}^n_+, y \in \mathbb{R}^{n-1}_+ : A_k y + (a_k - b)x_k \leq 0, Ax - b - A_k y + (b - a_k)x_k \leq 0\}$$

Let $M'_k = \{x \in \mathbb{R}^n_+, y \in \mathbb{R}^{n-1}_+ : \hat{A}x + \hat{B}y \leq \hat{b}\}$. Considering M'_k is now linear in both x and y we can project it to x-space by using the null space of $\hat{B}^T$, i.e.:

$$M''_k = \{x \in \mathbb{R}^n_+ : u^T \hat{A}x \leq u^T \hat{b}, \forall u^T \hat{B} = 0, u \geq 0\}$$

M''_k is called the lift-and-project closure of P and all of its defining inequalities are called lift-and-project cuts. Now to demonstrate a lift-and-project cut, let us consider the following polyhedron:

$$A = \begin{bmatrix} 5 & 8 \\ 4 & 3 \end{bmatrix}, \quad b = \begin{bmatrix} 10 \\ 6 \end{bmatrix}$$

One can verify that:

$$M_1' = \{x \in \mathbb{R}^2_+, y \in \mathbb{R}_+ : \begin{bmatrix} 10 & 8 \\ 6 & 3 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} -8 \\ -3 \end{bmatrix} y \leq \begin{bmatrix} 10 \\ 6 \end{bmatrix}$$
$$\begin{bmatrix} -5 & 0 \\ -2 & 0 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} 8 \\ 3 \end{bmatrix} y \leq \begin{bmatrix} 0 \\ 0 \end{bmatrix} \}$$

and $u^T = [0, 0.7273, 0.2727, 0]$ falls into the null space of $\hat{B}^T = [-8, -3, 8, 3]$. Therefore generating the following valid inequality $(3x_1 + 2.1818x_2 \leq 4.3636)$:

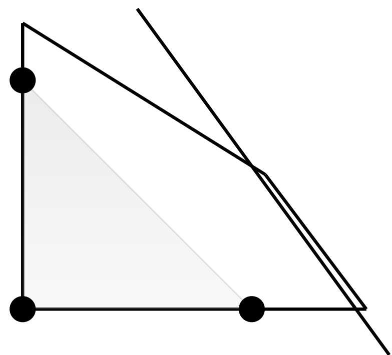


Figure 2.3: A lift-and-project cut

Chapter 3

On Superadditive Functions and Valid Inequalities

This chapter overviews some of the fundamental literature on value functions and their importance in superadditive duality of integer programming. We present a new representation of Chvátal-Gomory family of inequalities using linear programming tools. We further provide an elementary proof for the equivalence of discrete Farkas' Lemme presented in Lasserre [63] and superadditive duality.

3.1 Background

In this section we will recall important properties of IP. First of all let us define a superadditive function which plays central role in IP duality.

Definition 3.1.1. *A function $F : \mathbb{R}^m \to \mathbb{R}$ is called superadditive if*

1. $F(x) + F(y) \leq F(x + y)$ *for all* $x, y \in \mathbb{R}^m$,
2. $F(x) \leq F(y)$ *if* $x \leq y$,
3. $F(0) \leq 0$.

Superadditive functions in optimization context was first considered by Gilmore and Gomory [41]. Their results were later extended to the more general group problem by the work of Gomory [46]. Let us consider the polyhedron $P(A, b) := \max\{x : Ax \leq b, x \geq 0\}$ and the integer set $Q_I(A, b) = P(A, b) \cap \mathbb{Z}^n$. Let $A = [a_1, \ldots, a_n]$ denote the columns of A. An interesting observation is that any superadditive function generates a valid inequality for $Q_I(A, b)$:

Lemma 3.1.2. *If F is superadditive, then $\sum_{i=1}^{n} F(a_j)x_j \leq F(b)$ is valid for $Q_I(A, b)$.*

Proof. $\sum_{i=1}^{n} F(a_j)x_j \leq \sum_{i=1}^{n} F(a_j x_j) \leq F(\sum_{i=1}^{n} a_j x_j) \leq F(b)$ as wanted. First inequality follows because $x \in Q_I(A,b) \subset \mathbb{Z}^n$. Second inequality is a consequence of the first superadditive property and last inequality is because of the monotonicity of a superadditive function. □

Therefore a natural question to ask is *which superadditive functions in particular hold interest to generate stronger valid inequalities for* $Q_I(A,b)$. This function is called the value function of $Q_I(A,b)$ ([50]):

$$F_c(b) = \max\{c^T x : x \in Q_I(A,b)\} \tag{3.1}$$

and have been studied extensively by the integer programming community, in particular the duality (Jeroslow [53], Johnson [56], Gomory [104] and many others).

Claim 3.1.3. *$F_c(b)$ is superadditive.*

Proof. First of all, $F_c(b_1) + F_c(b_2) = c^T x_1 + c^T x_2 = c^T(x_1 + x_2) \leq F_c(b_1 + b_2)$, where the final inequality follows as $A(x_1 + x_2) \leq b_1 + b_2$. Secondly, $F_c(b_1) \leq F_c(b_2)$ whenever $b_1 \leq b_2$ by construction. □

An important thing to note is that given a valid inequality for $Q_I(A,b)$, $\alpha^T x \leq \beta$, we can use the value function of $Q_I(A,b)$ to generate another valid inequality that is at least as strong as $\alpha^T x \leq \beta$:

Lemma 3.1.4. *If $\alpha^T x \leq \beta$ is a valid inequality for $Q_I(A,b)$, then we have*

$$\alpha^T x \leq \sum_{j=1}^{n} F_\alpha(a_j)x_j \leq F_\alpha(b) \leq \beta \tag{3.2}$$

Proof. Considering e_j (the unit vector where jth component is 1, 0 otherwise) is feasible with respect to the optimization problem in $F_\alpha(a_j)$, we know that $F_\alpha(a_j) \geq c_j$. Second inequality and the final inequality follows by the first and second properties of a superadditive function, respectively. □

Corollary 3.1.5. *All tightest valid inequalities arise from superadditive functions.*

Now let us introduce a family of superadditive functions $\mathcal{F}_{A,c} = \{F \text{ is superadditive } : F(a_j) \geq c_j, \forall j\}$ for a rational matrix A and a rational vector c. Then we have the following ([56]):

Theorem 3.1.6 (Weak superadditive duality). *For every $F \in \mathcal{F}_{A,c}$ and $x \in Q_I(A,b)$ we have:*

$$c^T x \leq F(b) \tag{3.3}$$

Theorem 3.1.7 (Strong superadditive duality). *For every $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, we have:*

$$\max_{x\in Q_I(A,b)} c^T x = \min_{F\in\mathcal{F}_{A,c}} F(b) \tag{3.4}$$

The above result is known as Superadditive Duality for Integer Programming (Schrijver, [96]). Though very little is known about how to compute an optimal superadditive function $F \in \mathcal{F}_{A,c}$ that satisfies (3.4) ([60]), it is well-known that a combination of Chvatal-Gomory inequalities can be used to compute one [77]. Now let us define a superadditive function that gives rise to Chvatal-Gomory inequalities and has intimate ties to LP duality by a simple modification:

Definition 3.1.8. $F'_u(b) = \lfloor u^T b \rfloor$*, where $u \in \mathbb{R}^m_+$, is called as the Chvatal-Gomory function.*

It is an easy exercise to prove that F'_u is a superadditive function for $u \geq 0$. Therefore it generates a valid inequality that is known as the Chvátal-Gomory inequality, honoring the two seminal papers of Chvátal's [22] and Gomory's [44]:

$$\sum_{j=1}^{n} \lfloor u^T a_j \rfloor x_j \leq \lfloor u^T b \rfloor \tag{3.5}$$

It should be noted that if we remove the floor operation above, we get a linear consequence of $P(A,b)$, let F^{LP} represent this function. Now let $\mathcal{F}' = \{F'_u \text{ is a CG function }\}$ and $\mathcal{F}^{LP} = \{F^{LP}_u$ is a linear consequence of $P(A,b)\}$. Then we have the following:

Corollary 3.1.9. *For every $c \in \mathbb{R}^n$, $A \in \mathbb{R}^{m\times n}$ and $b \in \mathbb{R}^m$, we have:*

$$\max_{x\in Q_I(A,b)} c^T x = \min_{F\in\mathcal{F}_{A,c}} F(b) \leq \min_{F'_u\in\mathcal{F}'} F'_u(b) \leq \min_{F^{LP}_u\in\mathcal{F}^{LP}} F^{LP}_u(b) = \max_{x\in P(A,b)} c^T x \tag{3.6}$$

Proof. First equality follows from IP duality. First inequality is because CG-inequalities are a restricted family of superadditive functions. Second inequality is because of the removal of rounding operation, and final equality is known as the LP duality. □

The feasible region of the minimization problem in the middle is called the *Chvátal closure of* $P(A, b)$ (i.e. highlighted region in Figure 3.1), and Chvátal showed that finitely many closure operations need to be performed to get P_I, the integral hull ([22]). This is a way to compute an optimal superadditive function that satisfies 3.4. In the following section we will explicitly define this family as a collection of results from LP literature with finitely many integer variables.

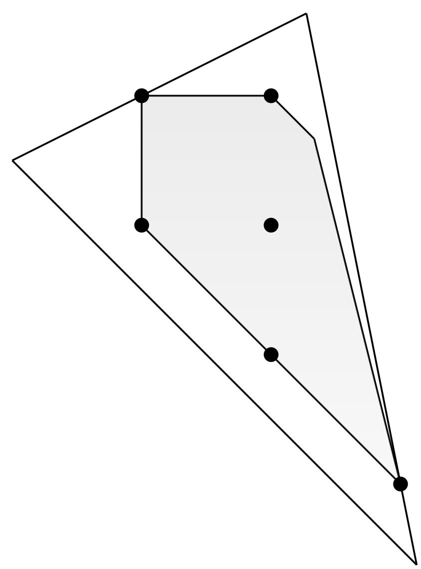


Figure 3.1: CG closure

3.2 A new family of Superadditive Inequalities

We consider the polytope $P = \{x \in \mathbb{R}^n_+ | Ax \leq b\}$, where $A \in \mathbb{R}^{m \times n}$ and $b \in \mathbb{R}^m$. Let $P_I = \text{conv}(P \cap \mathbb{Z}^n)$ denote the integral hull of P.

Let $F : \mathbb{R}^m \to \mathbb{R}$ be a superadditive (SA) function. Then, recalling Lemma 3.1.2:

$$\sum_{j=1}^{n} F(a_j)x_j \leq F(b) \tag{3.7}$$

is valid for P_I.

Let $Q \in \mathbb{R}^{m\times N},\ \Gamma \in \mathbb{R}^N,\ d \in \mathbb{R}^m$ and $I \subseteq \{1,\ldots,N\}$ where $N \in \mathbb{Z}_+$. Then the function

$$F_{Q,\Gamma,I}(d) = \left\{ \begin{array}{lccl} \max & \Gamma^T y & & \\ \text{s.t.} & Qy & \leq & d \\ & y_i & \in & \mathbb{Z} \quad \forall i \in I \end{array} \right\}. \tag{3.8}$$

is a SA function (The value function of a Mixed Integer Program is SA, [54]).

Theorem 3.2.1. *Every $F_{Q,\Gamma,\emptyset}$-inequality is satisfied by all points of P.*

Proof. By linear programming duality $F_{Q,\Gamma,\emptyset}(d) = \max\{\Gamma^T y | Qy \leq d\} = \min\{z^T d | z^T Q = \Gamma^T, z \geq 0\}$. Choose $u = z^*$ the optimal solution of the problem $\min\{z^T b | z^T Q = \Gamma^T, z \geq 0\}$. Therefore by feasibility of u we have $\min\{z^T a_j | z^T Q = \Gamma^T, z \geq 0\} = F_{Q,\Gamma,\emptyset}(a_j) \leq u^T a_j,\ \forall j = 1,\ldots,n$ and $F_{Q,\Gamma,\emptyset}(b) = u^T b$. □

3.3 Superadditive Closure

Let $\mathcal{F}(k,\ell)$ be the family of superadditive functions $F_{Q,\Gamma,I}$ obtained from $\Gamma \in \mathbb{R}^\ell$, $Q \in \mathbb{R}^{m\times\ell}$ and $I \subseteq \{1,\ldots,\ell\}$ such that $|I| \leq k$. Then $\mathcal{F}(k,\ell)$-closure is the polytope formed by the inequalities of type (3.7) for all $F \in \mathcal{F}(k,\ell)$.

Theorem 3.3.1. *Let $\Gamma \in \mathbb{R}^N$, $Q \in \mathbb{R}^{m\times N}$ and $I \subseteq \{1,\ldots,n\}$. If $\alpha_j = F_{Q,\Gamma,I}(a_j),\ j = 1,\ldots,n$ and $v^* = \min\{u^T b | u^T A \geq \alpha^T, u \geq 0\}$, then the continuous relaxation of $\max\{\Gamma^T z | Qz \leq b, z_i \in \mathbb{Z}, i \in I\}$ has optimum objective value of at least v^*.*

Proof. Take the optimum solution v_j of $F_{Q,\Gamma,I}(a_j)$ for all $j = 1,\ldots,n$. Let's relax $\max\{\Gamma^T z | Qz \leq b, z_i \in \mathbb{Z}, i \in I\}$ and take its dual $\min\{y^T b | y^T Q = \Gamma^T, y \geq 0\}$. Using v_j, $y^T a_j \geq y^T Q v_j = \Gamma^T v_j = \alpha_j,\ \forall j = 1,\ldots,n$ implies $u^T A \geq \alpha^T$, namely the constraints of $\min\{u^T b | u^T A \geq \alpha^T, u \geq 0\}$. Considering $y^T Q = \Gamma^T$ can only imply more restrictions on the feasible region and the integrality gap, objective value is at least v^*. □

For a given $A \in \mathbb{R}^{m\times n}$ and $u \geq 0$ s.t. $u \neq 0$ and $u^T A \in \mathbb{Z}^n$, we define the following matrix,

$$Q(A,u) = \left[\frac{1}{u^T\mathbf{1}}\mathbf{1}, a_1 - \frac{(u^TA)_1}{u^T\mathbf{1}}\mathbf{1}, \ldots, a_n - \frac{(u^TA)_n}{u^T\mathbf{1}}\mathbf{1}\right]$$

and the family of such matrices,

$$\mathcal{Q}(A) = \left\{Q(A,u) | u \geq 0, u \neq 0, u^TA \in \mathbb{Z}^n\right\}$$

Let us define the very restricted family of superadditive functions $f_A := \{F_{Q,\Gamma,\{0\}} | Q \in \mathcal{Q}(A)\}$ where $\Gamma^T = \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix}$. Obviously $\mathcal{F}(1, n+1) \supseteq f_A$.

Theorem 3.3.2. *f_A-closure is exactly the same as the CG-closure.*

Proof. We first show f_A-closure is at least as strong as the CG-closure. Let $u^TAx \leq \lfloor u^Tb \rfloor$ be an arbitrary CG-cut, where $u \in \mathbb{R}^m_+$. To majorize the CG-cut by an f-cut, where $f \in f_A$, we must have

$$v_\ell = \left\{\begin{array}{lllll} \max & y_0^\ell & & & \\ \text{s.t.} & \frac{1}{u^T\mathbf{1}}y_0^\ell & +\sum_{j=1}^n (a_{ij} - \frac{(u^TA)_j}{u^T\mathbf{1}})y_j^\ell & \leq & a_{i\ell}, \ \forall i \\ & y_0^\ell & & \in & \mathbb{Z} \end{array}\right\} \geq (u^TA)_\ell \tag{3.9}$$

for all $\ell = 1, \ldots, n$, and

$$v_0 = \left\{\begin{array}{lllll} \max & y_0^0 & & & \\ \text{s.t.} & \frac{1}{u^T\mathbf{1}}y_0^0 & +\sum_{j=1}^n (a_{ij} - \frac{(u^TA)_j}{u^T\mathbf{1}})y_j^0 & \leq & b_i, \ \forall i \\ & y_0^0 & & \in & \mathbb{Z} \end{array}\right\} \leq \lfloor u^Tb \rfloor \tag{3.10}$$

Clearly problem set (3.9) has feasible solutions $y_0^\ell = (u^TA)_\ell$, $y_\ell^\ell = 1$ and $y_j^\ell = 0, \ \forall j \neq \ell$, which shows the required inequality is attained in (3.9) for all $\ell = 1, \ldots, n$. Let us consider the dual of problem (3.10)'s LP relaxation.

$$\begin{array}{llcll} \min & b^Tz & & & \\ \text{s.t.} & \frac{1}{u^T\mathbf{1}}\mathbf{1}^Tz & = & 1 & \\ & (a_j - \frac{(u^TA)_j}{u^T\mathbf{1}}\mathbf{1})^Tz & = & 0, & j = 1, \ldots, n \\ & z & \geq & 0 & \end{array} \tag{3.11}$$

Now let us consider the second set of constraints of (3.11). First set of constraints imply $\frac{1}{u^T\mathbf{1}}\mathbf{1}^T z = 1$. Therefore $z^T a_j - (u^T a_j)(\frac{1}{u^T\mathbf{1}}\mathbf{1}^T z) = z^T a_j - u^T a_j = 0$ implies $z = u$ is a feasible solution. Therefore $v_0 \leq u^T b < \lfloor u^T b \rfloor + 1$. Which proves the first claim.

Second, we prove a CG-cut is at least as strong as an f-cut, where $f \in f_A$. We prove $v_\ell \leq (u^T A)_\ell,\ \ell = 1, \ldots, n$

$$\begin{aligned}
v_\ell &= \max\{\Gamma^T z^\ell | Q(A,u) z^\ell \leq a_\ell,\ z_0^\ell \in \mathbb{Z}\} \\
&\leq \max\{\Gamma^T z^\ell | Q(A,u) z^\ell \leq a_\ell\} \\
&= \min\{a_\ell^T y^\ell | \frac{1}{u^T\mathbf{1}}\mathbf{1}^T y^\ell = 1, a_j^T y^\ell - \frac{(u^T A)_j}{u^T\mathbf{1}}\mathbf{1}^T y^\ell = 0, \forall j, y \geq 0\} \\
&= \min\{a_\ell^T y^\ell | \frac{1}{u^T\mathbf{1}}\mathbf{1}^T y^\ell = 1, a_j^T y^\ell - (u^T A)_j = 0, \forall j, y \geq 0\} = (u^T A)_\ell
\end{aligned}$$

We note that $F_{Q,\Gamma,\{0\}}(b) \geq \lfloor F_{Q,\Gamma,\emptyset}(b) \rfloor$ because of the structure of objective function. Combining this with Theorem 3.3.1, implies $v_0 \geq \lfloor u^T b \rfloor$ which concludes the proof. $\square$

Theorem 3.3.2 implies that $\mathcal{F}(1, n+1)$-closure is at least as strong as the CG-closure. Next we show that $\mathcal{F}(1, n+1)$-closure involves inequalities stronger than any CG-cut. First of all, let us note that $\mathcal{F}(1,n) \subseteq \mathcal{F}(1, n+1)$, as we can have dummy columns extending Q. We provide an instance to prove this claim. Let us consider the following polytope,

$$A = \begin{bmatrix} -1 & 2 \\ 5 & 1 \\ -2 & -2 \end{bmatrix}, \quad b = \begin{bmatrix} 6 \\ 21 \\ -9 \end{bmatrix}$$

Let us consider the following Q and Γ matrices,

$$Q = \begin{bmatrix} -3 & 4 \\ 4 & 2 \\ -4 & -4 \end{bmatrix}, \quad \Gamma = \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \quad I = \{1\}$$

We calculate $F_{Q,\Gamma,I}(a_1) = 1.5$, $F_{Q,\Gamma,I}(a_2) = 0.5$ and $F_{Q,\Gamma,I}(b) = 6.75$. Therefore we find the valid inequality

$$6x_1 + 2x_2 \leq 27 \tag{3.12}$$

Claim 3.3.3. *(3.12) is stronger than any CG-cut.*

Proof. To majorize (3.12) by a CG-cut, we need a $u \geq 0$ s.t. $u^T A \geq (6,2)$ and $u^T b < 28$. To test this question we solve the linear programming problem $\min\{u^T b | u^T A \geq (6,2), u \geq 0\} = 28.9091$, hence such a u vector does not exist. $\square$

Corollary 3.3.4. $\mathcal{F}(1, n+1)$*-closure is stronger than the CG-closure.*

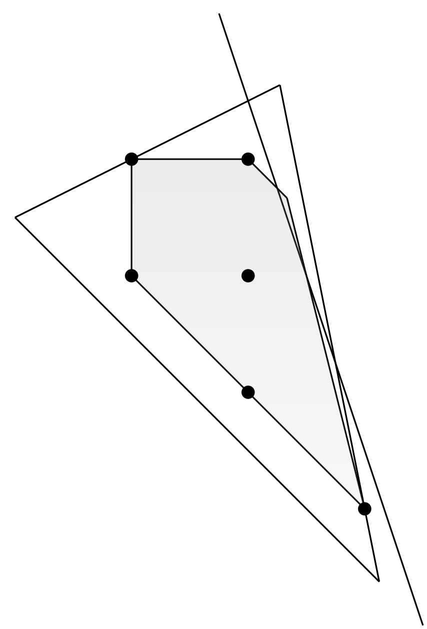


Figure 3.2: SA cut falling into CG closure

3.4 Superadditive closure of a mixed-integer program

In this section, we extend the superadditive closure into mixed-integer programs with equality constraints. First we remember classical theorem from valid inequalities:

Theorem 3.4.1. *Let* $T = \{x \in \mathbb{Z}^n_+, y \in \mathbb{R}^p_+ : \sum_{j \in \mathcal{I}} h_j x_j + \sum_{j \in \mathcal{J}} g_j y_j = b\}$*, where* $h_j, g_j, b \in \mathbb{R}$ *for all* j*. The inequality (also known as the Mixed Integer Gomory (MIG) cut)*

$$\sum_{j \in \mathcal{I}} F_\alpha(h_j) x_j + \sum_{j \in \mathcal{J}^-} \bar{F}_\alpha(g_j) y_j \leq F_\alpha(b)$$

is valid for T*, where* $\mathcal{J}^- = \{j \in \mathcal{J} : g_j < 0\}$*,* $0 < \alpha < 1$ *and*

$$F_\alpha(d) = \lfloor d \rfloor + \frac{(f_d - \alpha)^+}{1 - \alpha}$$
$$\bar{F}_\alpha(d) = \lim_{\lambda \to 0^+} \frac{F_\alpha(\lambda d)}{\lambda} = \frac{d}{1 - \alpha}$$

To derive the superadditive closure of a mixed-integer program let us consider the polyhedron $P = \{x \in \mathbb{R}^n_+ | Ax = b\}$, where $A = (a_j | j \in \mathcal{I})$. Also we modify $F_{Q,\Gamma,K}(d)$ for a set of equality constraints over non-negative variables as follows:

$$F'_{Q,\Gamma,K}(d) = \left\{ \begin{array}{llll} \max & \Gamma^T y & & \\ \text{s.t.} & Qy & = & d \\ & y_i & \in & \mathbb{Z} \quad \forall i \in K \end{array} \right\}. \tag{3.13}$$

Corollary 3.4.2. $F'_{Q(A,u),\Gamma,\{0\}}(u^T d) = \lfloor u^T d \rfloor$ *and* $F'_{Q(A,u),\Gamma,\emptyset}(u^T d) = u^T d$ *for* $u \in \mathbb{R}^m$*, where* $d \in \{a_j | j = 1, \ldots, n\} \cup \{b\}$ *and* $\Gamma^T = \begin{bmatrix} 1 & 0 & \ldots & 0 \end{bmatrix}$*.*

Say $\bar{x} \in P$ be an extreme point and B one it its corresponding basis. Therefore if we let $u^T = e_i^T B^{-1}$, $u^T A x = u^T b$ is the ith row of simplex table at this basic feasible solution.

Corollary 3.4.3. *Let* $\mathcal{I} \subseteq [n]$ *be the indices of integer variables and* $\mathcal{J} = [n] \setminus \mathcal{I}$*. The following inequality*

$$\sum_{j \in \mathcal{I}} (F'_{Q(A,u),\Gamma,\{0\}}(u^T a_j) + \frac{(F'_{Q(A,u),\Gamma,\emptyset}(u^T a_j) - F'_{Q(A,u),\Gamma,\{0\}}(u^T a_j) - \alpha)^+}{1 - \alpha}) x_j$$
$$+ \sum_{j \in \mathcal{J}^-} \frac{F'_{Q(A,u),\Gamma,\emptyset}(u^T a_j)}{1 - \alpha} x_j \leq F'_{Q(A,u),\Gamma,\{0\}}(u^T b)$$

is a valid inequality for $0 < \alpha < 1$ *and* $\Gamma^T = \begin{bmatrix} 1 & 0 & \dots & 0 \end{bmatrix}$.

3.5 The Equivalence of SA Duality and Discrete Farkas' Lemma

[63] explicitly describes a discrete variant of the Farkas' lemma by using the counting techniques based on generating functions as described by Brion and Vergne. [65] describes the defining inequalities of the integral hull using the discrete Farkas' lemma proposed in [63]. They reduce the integer programming problem into a set of equalities which equivalently represents the conditions of discrete Farkas' lemma. Using the extremal elements of this polyhedron, they describe the minimal valid inequalities, namely the facets, of the integral hull. They also provide an intuitive, constructional proof of discrete Farkas' lemma found in [65]. [64] links the Farkas' lemma, superadditive duality and integral hull of an integer program and does a quick overview of the duality relation proposed by Lasserre. We can see the motivation of Lasserre's work, where Jeroslow [54] states:

> ... by simply writing down the subadditivity relations (plus complementarity relations, if one wishes), and viewing the symbol $F(v)$ (for each element v of the group) as a variable in a linear system, one obtains a linear inequality system whose extreme points are facets of the group problem: this is surprising!

when he refers to the pioneering work of Gomory [45].

Given an integer programming problem, $\max\{c^Tx | Ax = b, x \in \mathbb{Z}^n_+\}$, Discrete Farkas' Lemma defined by [63] corresponding to this problem can be stated as follows:

$$\begin{array}{lrcl} \max & \displaystyle\sum_{j=1}^{n} \sum_{0 \le p \le b^*} c_j q_p^j & & \\ \text{s.t.} & \displaystyle\sum_{j=1}^{n} \left(q_{r-a_j}^j | r \ge a_j \right) - \sum_{j=1}^{n} q_r^j & = & \begin{cases} 1, & \text{if } r = 0 \\ -1, & \text{if } r = b \\ 0, & \text{otherwise} \end{cases}, \quad 0 \le r \le b^* \\ & q_p^j & \ge & 0, \quad 0 \le p \le b^*, \ j = 1, \dots, n \end{array} \tag{3.14}$$

If we take the dual of problem (3.14):

$$\begin{array}{llcll} \min & \gamma(b) \quad -\gamma(0) & & & \\ \text{s.t.} & \gamma(\alpha+a_j) \quad -\gamma(\alpha) & \geq & c_j & 0 \leq \alpha + a_j \leq b,\ j = 1,\ldots,n \end{array}$$

Let $\pi(\alpha) := \gamma(\alpha) - \gamma(0),\ 0 \leq \alpha \leq b$. Then we can rewrite the dual problem as follows:

$$\begin{array}{llcll} \min & \pi(b) & & & \\ \text{s.t.} & \pi(\alpha+a_j) - \pi(\alpha) & \geq & c_j & 0 \leq \alpha + a_j \leq b,\ j = 1,\ldots,n \end{array}$$

Let $D := \prod_{j=1}^{n}\{0,1,\ldots,b_j\}$ and $f_\pi(d) = \inf\limits_{v\in D}\{\pi(v+d)-\pi(v)\}$. [63] proves the following problem is equivalent to the dual of integer programming problem:

$$\begin{array}{llcll} \min & f_\pi(b) & & & \\ \text{s.t.} & f_\pi(a_j) & \geq & c_j, & j = 1,\ldots,n \end{array} \tag{3.15}$$

We will show the superadditive functions in problem (3.15) contains the superadditive functions that strengthen a valid inequality for P_I. In turn proving it is the set of defining superadditive functions for the SA dual problem.

Theorem 3.5.1. *Let $P = \{x | Ax \leq b, x \geq 0\}$ and $P_I = conv(P \cap \mathbb{Z}^n) \neq \emptyset$ and $\alpha^T x \leq \beta$ be valid for P_I. Let*

$$\pi(v) = \begin{cases} \max\{\alpha^T x | Ax \leq v, x \in \mathbb{Z}^n_+\} & \text{if } v \in D \\ \infty & \text{otherwise} \end{cases}$$

and $f_\pi(d) = \inf\limits_{v\in D}\{\pi(v+d)-\pi(v)\}$ is as defined above. Then we have $f_\pi(a_j) \geq \alpha_j,\ j = 1,\ldots,n$ and $f_\pi(b) \leq \beta$.

Proof. Let $v \in D$. We have $\pi(v) = \alpha^T x^*$ for some $x^* \in \mathbb{Z}^n_+$. If $v + a_j \notin D$ then we are done as $\pi(v+d)-\pi(v) = \infty$. Else $v+a_j \leq b$ implies $A(x^*+e_j) = Ax^*+a_j \leq v+a_j \leq b$. Therefore $x^* + e_j$ is a feasible solution for problem $\pi(v + a_j)$, implying $\pi(v + a_j) \geq \alpha^T(x^*+e_j) = \pi(v)+\alpha_j$. Hence we have $\pi(v+a_j)-\pi(v) \geq \alpha_j,\ \forall v \in D$, or equivalently $f_\pi(a_j) \geq \alpha_j$. Also we have $f_\pi(b) = \pi(b)-\pi(0) = \pi(b) = \max\{\alpha^T x | Ax \leq b, x \in \mathbb{Z}^n_+\} \leq \beta$ as $\alpha^T x \leq \beta$ is valid for P_I. Furthermore we note $\alpha^T x \leq \sum_{j=1}^{n} f_\pi(a_j)x_j \leq f_\pi(b) = \alpha^T \bar{x}$ for some $\bar{x} \in Q$. Therefore $\sum_{j=1}^{n} f_\pi(a_j)x_j \leq f_\pi(b)$ is tight. □

Now we show that if we disregard the integrality in the definition of $\pi(v)$, then we get linear consequences of P. Let

$$\bar{\pi}(v) = \begin{cases} \max\{\alpha^T x | Ax \leq v, x \geq 0\} & \text{if } v \in D \\ \infty & \text{otherwise} \end{cases}$$

or equivalently

$$\bar{\pi}(v) = \begin{cases} \min\{v^T y | A^T y \geq \alpha, y \geq 0\} & \text{if } v \in D \\ \infty & \text{otherwise} \end{cases}$$

Theorem 3.5.2. *The inequality $\sum_{j=1}^{n} f_{\bar{\pi}}(a_j)x_j \leq f_{\bar{\pi}}(b)$ is a linear consequence of P.*

Proof. Let y^* be optimal with respect to $\bar{\pi}(v+d)$ and $\bar{y}$ be optimal with respect to $\bar{\pi}(v)$

$$\begin{aligned}
f_{\bar{\pi}}(d) &= \inf_{v\in D} \{\bar{\pi}(v+d) - \bar{\pi}(v)\} \\
&= \inf_{v\in D} \left\{ \begin{array}{lrcl} \min & (v+d)^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} - \begin{array}{lrcl} \min & v^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} \right\} \\
&\geq \inf_{v\in D} \left\{ (v+d)^T y^* - v^T y^* \right\} \\
&= \inf_{v\in D} \left\{ d^T y^* \right\} \geq \inf_{v\in D} \left\{ \begin{array}{lrcl} \min & d^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} \right\} = \begin{array}{lrcl} \min & d^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} \\
&= \begin{array}{lrcl} \max & \alpha^T x & & \\ \text{s.t.} & Ax & \leq & d \\ & x & \geq & 0 \end{array}
\end{aligned}$$

On the other hand we have,

$$
\begin{aligned}
f_{\bar{\pi}}(d) &= \inf_{v \in D} \{\bar{\pi}(v+d) - \bar{\pi}(v)\} \\
&= \inf_{v \in D} \left\{ \begin{array}{lrcl} \min & (v+d)^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} - \begin{array}{lrcl} \min & v^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} \right\} \\
&\leq \inf_{v \in D} \left\{ (v+d)^T \bar{y} - v^T \bar{y} \right\} \\
&= \inf_{v \in D} \left\{ d^T \bar{y} \right\} = \begin{array}{lrcl} \min & d^T y & & \\ \text{s.t.} & A^T y & \geq & \alpha \\ & y & \geq & 0 \end{array} = \begin{array}{lrcl} \max & \alpha^T x & & \\ \text{s.t.} & Ax & \leq & d \\ & x & \geq & 0 \end{array}
\end{aligned}
$$

where second to last equality is not very straightforward. But essentially we are trying to minimize the linear form $d^T \bar{y}$ where $\bar{y}$ is coming from the linear program $\bar{\pi}(v)$, which is equivalent linear program following the expression. Therefore we have

$$
\alpha_j \leq f_{\bar{\pi}}(a_j) = \begin{array}{lrcl} \max & \alpha^T x & & \\ \text{s.t.} & Ax & \leq & a_j \\ & x & \geq & 0 \end{array}
$$

as e_j is a feasible solution to the above problem. Now let z^j be optimal to the problem $\max\{\alpha^T x | Ax \leq a_j, x \geq 0\}$. Therefore we have $Az^j \leq a_j$ and $z^j \geq 0$. Let $x \in P$ and $z(x) = \sum_{j=1}^n z^j x_j \geq 0$. Note that $z(x) \in P$ with this selection as $Az(x) = \sum_{j=1}^n (Az^j) x_j \leq \sum_{j=1}^n a_j x_j = Ax \leq b$. Let $\bar{z}$ be optimal to $\max\{\alpha^T x | Ax \leq b, x \geq 0\} = f_{\bar{\pi}}(b)$. Then the inequality

$$
\sum_{j=1}^n f_{\bar{\pi}}(a_j) x_j = \sum_{j=1}^n \alpha^T z^j x_j = \alpha^T z(x) \leq \alpha^T \bar{z} = f_{\bar{\pi}}(b)
$$

is valid for P, completes the proof. $\square$

Chapter 4

Deepest Valid Inequalities

In this chapter we will focus on valid inequality generation for arbitrary MILP instances. Our method will be based on generating valid inequalities for the corner polyhedron [45] using the intersection cut theory of Balas [6]. The efficiency of these inequalities will be demonstrated on a well-known problem set, MIPLIB, and we will prove some fundamental properties of these inequalities, in particular the polynomial time implementation.

4.1 Background

MILP literature studied multiple ways of generating a valid inequality from a given simplex table. The most well-known inequalities (i.e. Gomory mixed integer rounding cuts) are generated using a single row of the simplex table and have been the focus of state-of-the-art commercial MILP solvers. Though the theory was known for quite sometime, until recently, multi-row cut generation didn't receive a lot of attention in the literature.

Pioneering theoretical work on multi-row simplex cuts has been made by [6] as early as '71. Given a fractional vector $f \in \mathbb{R}^m$ and a lattice-free convex set, $S \subset \mathbb{R}^m$, around f such that $f \in \text{int}(S)$, then a valid inequality for the simplex tableau can be calculated using the gauge function of S. While this result has not received computational attention early on, there have been renewed interest in cut generation considering multiple rows of the simplex tableau recently ([4]). Where the authors show that if two rows of the simplex tableau are taken into account, facet-definining inequalities for convex hull of mixed-integer set are intersection cuts in $\mathbb{R}^2$. [16] later generalized this result on higher dimensional spaces in $\mathbb{R}^m$. The NP-completeness of separation for split cuts has been settled by [17]. Even though these authors provide invaluable examples where such cuts are helpful, the scope of their computational study have been only illustrational.

The first thorough computational study has been made by [35], where he considered multi row cuts that are the consequences of symmetric lattice-free convex sets with respect to the current simplex solution. He developes a heuristic scheme to mass-generate such inequalities at the root node of a branch and bound tree and provides very promising computational results by considering up to 15 rows of the simplex tableau simultaneously. More recently [9] and [33] also tackled the same question on 2-rows, where they consider heuristically defined lattice-free convex sets for mass generation of valid inequalities. Later [28] showed such cuts can strengthen the bounds obtained by Gomory's mixed integer rounding cuts based on single rows. A more sophisticated method has been proposed by [70] where the authors propose how to generate a *best* lattice-free convex set on $\mathbb{R}^2$. They compare single row non-lifted intersection cuts, with their 2-row counterparts and show promising results in terms of information that multiple rows carry. Even though these results are encouraging, they are essentially computed from relaxations of simplex tableau. [37] develop a scheme for strengthening these inequalities by using the non-negativity information on basic variables.

A deepest intersection cut, in the sense that it minimizes a general p-norm of its coefficients, was proposed in a semi-infinite programming context by [10]. Which has been used within the computational analysis of [70]. They also provide an oracle for the 2-row separation of this problem. Now, we focus on how to generalize these ideas. We show that if we consider a fixed number of simplex tableau rows, there exists an intuitive poly-time separation routine to solve the problem of finding a deepest intersection cut, *in the sense that it minimizes a weighted sum of its coefficients.*

4.2 Basic Definitions and Notation

Let us consider an integer programming problem and its set of feasible solutions

$$P_I \;=\; \operatorname{conv}\{x \in \mathbb{Z}^n \mid Ax = b, x \geq 0\} \tag{4.1}$$

where $A \in \mathbb{Q}^{m\times n}$, and $b \in \mathbb{Q}^m$. Let B be a basis of A that correspond to a basic feasible solution of the linear programming relaxation. Then we can rewrite P_I as follows

$$P_I \;=\; \operatorname{conv}\left\{ \begin{pmatrix} x_B \\ x_N \end{pmatrix} \in \mathbb{Z}^n \;\middle|\; \begin{array}{c} x_B = B^{-1}b - B^{-1}Nx_N \\ x_B \geq 0, x_N \geq 0 \end{array} \right\}. \tag{4.2}$$

Next, we relax the nonnegativity of basic variables and the integrality of nonbasic variables, and obtain the so called *continuous relaxation of corner polyhedron* (for sake of simplicity we will call this the corner polyhedron), introduced by [45]:

$$P_C \;=\; \operatorname{conv}\left\{ \begin{pmatrix} x_B \\ x_N \end{pmatrix} \in \mathbb{Z}^m \times \mathbb{R}^{n-m} \;\middle|\; \begin{array}{c} x_B = B^{-1}b - B^{-1}Nx_N, \\ x_N \geq 0 \end{array} \right\}$$

To simplify notation, let $k = n - m$. Let us reserve $x = x_B$ for the basic variables, denote by $s = x_N$ the nonbasic ones, and introduce $f = B^{-1}b$ and $(-B^{-1}N) = [r_1, r_2, ..., r_k]$ to get

$$P_C = \operatorname{conv}\left\{ \begin{pmatrix} x \\ s \end{pmatrix} \in \mathbb{Z}^m \times \mathbb{R}^k \;\middle|\; x = f + \sum_{i=1}^{k} r_i s_i, s \geq 0 \right\} \tag{4.3}$$

Assuming that f is not (yet) integral, we are interested in valid inequalities that separate the fractional corner f from P_C. Since variables x depend in a unique and linear way on the nonbasic variables, according to the system of equations in (4.3), it is enough to consider inequalities involving only s. An inequality of the form

$$\sum_{j=1}^{k} \alpha_j s_j \geq \beta$$

is called a *valid inequality* for the corner polyhedron if it is satisfied by all solutions to (4.3). We will be only interested in valid inequalities, which are violated by f. The following useful lemma claims that we do not loose generality by assuming $\beta > 0$ and $\alpha_j \geq 0$ for all $j = 1, ..., k$.

Lemma 4.2.1 ([4])**.** *Every non-trivial valid inequality for* (4.3)*, that is tight at some* $(\bar{x}, \bar{s}) \in P_I$ *can be written in the form*

$$\sum_{j=1}^{k} \alpha_j s_j \geq 1 \tag{4.4}$$

for some $\alpha_j \geq 0$, $j = 1, \ldots, k$. □

An inequality of the form (4.4) is *dominated* if there exists another valid inequality $\sum_{j=1}^k \alpha'_j s_j \geq 1$ for P_C such that $\alpha'_j \leq \alpha_j$ for all $j = 1, \ldots, k$, with strict inequality for some indices.

4.3 Intersection Cuts

[6] introduced the intersection cut for the corner polyhedron (4.3). Let, P_{LP} denote the LP relaxation of P_I where we further drop the integrality restrictions of basic variables. By using Minkowski-Weyl decomposition of a polyhedron, P_{LP} can be decomposed into $f + C$, where C is the polyhedral cone generated by rational rays r_i, $i = 1, \ldots, k$. Let us now consider a closed convex set $S \subset \mathbb{R}^m$ such that it contains the basic solution $f \in \text{int}(S)$ in its interior, but does not contain any point from $\mathbb{Z}^m$ in its interior. To such a closed convex set we associate the following function

$$\Phi_S(r) := \inf\{t > 0 \mid f + \frac{r}{t} \in S\}, \tag{4.5}$$

for $r \in \mathbb{R}^m$. Note that since f is in the interior of S, we have $\Phi_S(r)$ finite for all $r \in \mathbb{R}^m$. Furthermore, $\Phi_S(r) = 0$ only if $f + \lambda r \in S$ for all $\lambda \geq 0$. It was shown in [6] that

$$\sum_{j=1}^k \Phi_S(r_j) s_j \geq 1 \tag{4.6}$$

is a valid inequality for (4.3) that cuts off the vertex $(f, 0) \in P_C$ if S is lattice-free. [24] have shown that the converse holds true, too.

Theorem 4.3.1 ([24]). *If* $P_C \neq \emptyset$, *then every valid inequality for it is an intersection cut of the form* (4.6), *corresponding to a lattice-free convex set* S. □

4.4 Maximal Lattice-Free Convex Sets

Let us next recall the following characterization of maximal lattice-free convex sets in finite dimensional spaces:

Theorem 4.4.1 ([72]). *Let V be a rational affine subspace of $\mathbb{R}^n$ containing an integral point. A set $S \subset V$ is a maximal lattice-free convex set of V if and only if S is a polyhedron of the form $S = P + L$ where P is a polyhedron, L is a rational linear space,* $\dim(S) = \dim(P) + \dim(L) = \dim(V)$*, S does not contain any integral point in its interior and there is an integral point in the relative interior of each facet of S.* $\square$

Let us now return to our corner polyhedron P_C and denote by W the linear space spanned by the vectors $\{r_1, \ldots, r_k\}$. As we recalled above, every maximal lattice-free convex set in $f + W$ give rise to a valid linear inequality for P_C. Let S be a maximal lattice-free convex set in $f + W$ containing f in its interior. By Theorem 4.4.1, S is a polyhedron, and since $f \in \text{int}(S)$, there exists a finite integer ℓ and vectors $a_1, \ldots, a_\ell \in \mathbb{R}^m$ such that

$$S = \{x \in \mathbb{R}^m \mid a_i^T(x - f) \leq 1,\ i = 1, ..., \ell\}.$$

The following claim is easy to see (c.f. [9]):

Lemma 4.4.2. *For all $r \in \mathbb{R}^m$ we have*

$$\Phi_S(r) = \max_{i=1,\ldots,\ell} a_i^T r. \tag{4.7}$$

$\square$

Then (4.7) readily implies that Φ_S is subadditive and positively homogenous. These properties then provide a short proof for the fact that (4.6) is a valid inequality for P_C. Namely, we can write for $(x, s) \in P_C$, $x \in \mathbb{Z}^m$ that

$$\sum_{j=1}^{k} \Phi_S(r_j)s_j = \sum_{j=1}^{k} \Phi_S(r_j s_j) \geq \Phi_S(\sum_{j=1}^{k} r_j s_j) = \Phi_S(x - f) \geq 1$$

First equality follows because of positive homogeneity, second inequality follows by subadditivity, the third equality follows by (4.3), while the final inequality follows from (4.7) because S is a lattice-free convex set.

An important fact is to note that ℓ is not only finite, but also bounded by 2^m:

Theorem 4.4.3 ([11], [94] and [95]). *Let $A \in \mathbb{Q}^{\ell \times m}$, $b \in \mathbb{Q}^\ell$ and $Q(A, b) = \{x \in \mathbb{R}^m : Ax \leq b\}$. If $Q(A, b) \cap \mathbb{Z}^m = \emptyset$ and non of the inequalities describing $Q(A, b)$ can be dropped without changing the description of the polyhedron, then $\ell \leq 2^m$.* $\square$

Corollary 4.4.4. *Let $S = \{x \in \mathbb{R}^m : Ax \le b\}$ be a rational maximal lattice-free convex set with a minimal description, in other words, without redundant inequalities. Then S involves at most 2^m inequalities.*

Proof. We will prove the claim in three steps. First of all, by Theorem 4.4.1, there exists at least one lattice point in the relative interior of every facet of S. Furthermore, if we delete only one of the defining inequalities of S, say the ith one, there exists at least one integer point, $x^{(i)}$, which is in the interior of the polyhedron defined by the rest of the inequalities. Consequently, there exists $\epsilon > 0$ such that $x^{(i)}$ satisfies strictly all inequalities $Ax \le b - \epsilon\mathbf{1}$ except the ith one, which it violates, for all indices i. Let us consider $S' = \{x \in \mathbb{R} | Ax \le b - \epsilon\mathbf{1}\}$. Note that $S' \cap \mathbb{Z}^m = \emptyset$, but if we delete any one of the defining inequalities of S', then the rest of the inequalities has an integer feasible solution. Therefore, by Bell and Scarf's Theorem 4.4.3, S does not have more than 2^m inequalities. $\square$

4.5 Cutting off a Fractional Vertex

With the definitions and properties we recalled in the previous sections, we can now prove Theorem 4.3.1 in an important special case. Let us assume that we consider an integer programming problem of the form

$$\max c^T x$$

$$Ax \le b$$
$$x \in \mathbb{Z}^n$$

Let $P = \{x \in \mathbb{R}^n | Ax \le b\}$ be the set of feasible solutions in the continuous relaxation, and let $f \in P$ be a non-integral vertex of P. Let us denote by $Bx \le d$ the subset of inequalities, which are tight at f. Assuming A is of full column rank, and that there are no redundant inequalities, B is an $n \times n$ invertible matrix. Therefore the particular corner polyhedron we are considering is of the form

$$P_C \;=\; \{(x, s) \in \mathbb{Z}^n \times \mathbb{R}^n \mid x = f + (-B^{-1})s,\ s \ge 0\} \tag{4.8}$$

where s denotes the slack variables added to the system $Bx \leq d$.

Let us denote by $B^i \in \mathbb{R}^n$ the ith row of B, and by $r_j \in \mathbb{R}^n$ the jth column of $-B^{-1}$. Thus, e.g., we have $B^i r_j = -1$ if $i = j$ and $= 0$ otherwise.

Let us now consider a minimal valid inequality $\alpha^T s \geq 1$ for P_C. We know by Lemma 4.2.1 that all minimal valid inequalities are of this form, for some $\alpha \geq 0$.

We shall prove that there exists a maximal lattice free set S, such that the corresponding intersection cut (4.6) is equivalent with $\alpha^T s \geq 1$ (for a more comprehensive result the reader should consult [9]). Note that in this case we have $-B^{-1}s = x - f = x - B^{-1}d$, and hence the above inequality can be written also as $\alpha^T s = \alpha^T(-B)(-B^{-1}s) = (-\alpha^T B)(x - f) \geq 1$.

Theorem 4.5.1. *Assume that $a_0^T(x - B^{-1}d) \geq 1$ is a facet defining inequality of the corner polyhedron and that $B^{-1}d \notin \mathbb{Z}^n$. Then, there exists a lattice-free convex set S such that* (4.7) *is equivalent to $a^T(x - B^{-1}d) \geq 1$.*

Proof. Let $\hat{S} = \{x \in \mathbb{R}^n \mid Bx \leq d,\ a_0^T(x - B^{-1}d) \leq 1\}$. Since $a_0^T(x - B^{-1}d) \geq 1$ is a valid inequality for (4.8), the set $\hat{S}$ is a lattice-free convex set. We note that the relative interior of any facet of $\hat{S}$ other than $F = \{x \in \hat{S} \mid a_0^T(x - B^{-1}d) = 1\}$ is not containing any integral point. Therefore we can rotate any facet $F' = \{x \in \hat{S} \mid w^T x = u\}$ of $\hat{S}$ along the face $F \cap F'$ such that we keep $B^{-1}d$ inside the resulting new polyhedron, until we hit an integral point $x' \in \mathbb{Z}^n$. Let $F'' \supset F \cap F'$ be the obtained hyperplane. We replace the halfspace corresponding to F' with the one defined by F''. Repeating this with all the facets that do not contain integral points we can arrive to a maximal lattice-free convex set $S \supset \hat{S}$. Note that if $B^{-1}d \notin \mathbb{Z}^n$ we have $B^{-1}d \in \operatorname{int}(S)$. Figure 4.1 demonstrates this construction in detail.

Let us write the obtained maximal lattice free set as $S = \{x \in \mathbb{R}^n \mid a_i^T(x - B^{-1}d) \leq 1,\ i = 0, \ldots, \ell\}$ for some $a_1, \ldots, a_\ell$ obtained in this process. It is then easy to verify that we have

$$\Phi_S(r_j) = \max_{i=0,\ldots,\ell} a_i^T(r_j) = a_0^T r_j.$$

This is because the half line $f + \lambda r_j$, $\lambda \geq 0$ intersects the boundary of S within the facet F, for indices $j = 1, .., n$. Thus, the corresponding intersection cut (4.3) can be written as

$$\sum_{j=1}^{n}(a_0^T r_j)s_j \geq 1.$$

By using the facts that $s = d - Bx$, and $[r_1, ..., r_n] = -B^{-1}$ we can further rewrite the above as

$$a_0^T(-B^{-1})(d - Bx) \geq 1$$

which is the same as

$$a_0^T(x - B^{-1}d) \geq 1$$

as claimed. $\square$

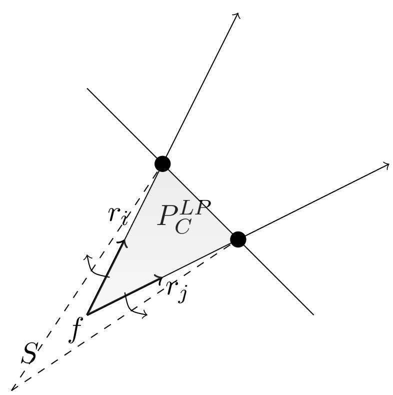


Figure 4.1: Lattice-free convex set construction

4.6 Deepest Cuts

Let us now turn back to a general corner polyhedron (4.3). In this section we consider the problem of generating a lattice-free convex set S such that a weighted sum of the coefficient of the corresponding intersection cut (4.6) is as small as possible. Let $w \in \mathbb{R}_+^k$ represent the weights. We can state our problem as follows:

$$\begin{aligned} \min \quad & \sum_{j=1}^{k} w_j \Phi_S(r_j) & \\ \text{s.t.} \quad & S \subset \mathbb{R}^m \qquad \text{is lattice-free, and} \\ & f \in \text{int}(S), \end{aligned} \tag{4.9}$$

which we can write equivalently, by applying the following steps:

1. introduce additional variables λ_j, $j = 1, ..., k$,

2. use the definition (4.7) of Φ_S, and represent S as a polyhedral set in terms of an unknown matrix $A \in \mathbb{R}^{\ell \times m}$ (where we can choose $\ell = 2^m$ by Corollary 4.4.4),

3. eliminate inf and represent lattice-freeness with infinitely many constraints, one for each lattice point.

Thus we get

$$\begin{array}{lll} \min & \displaystyle\sum_{j=1}^{k} w_j \lambda_j & \\ \text{s.t.} & A r_j - \lambda_j \mathbf{1} \leq 0 & j = 1, \dots, k, \\ & \displaystyle\max_{i=1,\dots,\ell} (A(x-f))_i \geq 1 & \forall\ x \in \mathbb{Z}^m, \\ & A \in \mathbb{R}^{\ell \times m}, & \\ & \lambda_j \geq 0 & j = 1, \dots, k. \end{array} \tag{4.10}$$

Note that problem (4.10) is a semi-infinite mixed-integer disjunctive programming problem, in $\ell m + k$ variables, which are the coefficients of matrix A and λ_j, $j = 1, ..., k$.

Let us remark here that to generate useful cuts for an integer programming problem we do not need to use all rows of the simplex tableaux. In fact current practice is to use only one or two rows, corresponding to some fractional components of the basic solution. Hence we can assume that p (where $p << m$) fractional rows of the simplex table will be taken into consideration (e.g., $p = 1$ for relaxed Gomory cuts or $p = 2$ for split cuts, triangle cuts, etc.).

Even though after the fixed dimensional representation of the problem (4.10), it still is a tough problem to solve using the general approach of disjunctive programming. Instead we solve the following equivalent problem:

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{k} w_j \lambda_j \\
\text{s.t.} \quad & R^T a_x - I\lambda \leq 0 && \forall\, x \in \mathbb{Z} \\
& a_x^T (x - f) \geq 1 && \forall\, x \in \mathbb{Z} \\
& a_x \in \mathbb{R}^p && \forall\, x \in \mathbb{Z} \\
& \lambda_j \geq 0 && j = 1, \ldots, k
\end{aligned}
\tag{4.11}
$$

Problem (4.11) is an infinite programming problem as opposed to the semi-infinite nature of problem (4.10). Though we will show that it is much easier to solve compared to problem (4.10). First, we will prove the relationship between these two problems.

Theorem 4.6.1. *Problem* (4.11) *and problem* (4.10) *share the same optimum objective value.*

Proof. Let $\bar{A} \in \mathbb{R}^{\ell \times p}$ be an optimal solution to (4.10). Let $\pi = [\pi_x \in \{1, \ldots, \ell\} : x \in \mathbb{Z}]$ be a vector s.t. $\bar{a}_{\pi_x}^T (x - f) \geq 1$, $\forall x \in \mathbb{Z}$. Let $A' \in \mathbb{R}^{\mathbb{Z} \times p}$ be such that $a'_x = \bar{a}_{\pi_x}$, $\forall x \in \mathbb{Z}$, then A' is feasible to problem (4.11). Considering the solution to problem (4.11) is a lattice-free convex set (not necessarily maximal), there exists a maximal-lattice free convex set that has at most 2^p facets. $\square$ $\square$

A common approach to solve semi-infinite (and infinite) optimization problems is to apply cutting-plane or column-generation. Initially we use a finite set $X \subseteq \mathbb{Z}^p$ (around f) and solve the restricted problem

$$
\begin{aligned}
\min \quad & \sum_{j=1}^{k} w_j \lambda_j \\
\text{s.t.} \quad & R^T a_x - I\lambda \leq 0 && \forall\, x \in X \\
& a_x^T (x - f) \geq 1 && \forall\, x \in X \\
& a_x \in \mathbb{R}^p && \forall\, x \in X \\
& \lambda_j \geq 0 && j = 1, \ldots, k
\end{aligned}
\tag{4.12}
$$

After we solve (4.12) and get an optimal matrix A, we try to extend X by solving the following integer programming problem in variables $x \in \mathbb{Z}^p$ and ϵ:

$$\begin{array}{llrcl} \max & & \epsilon & & \\ \text{s.t.} & Ax & +\epsilon\mathbf{1} & \leq & \mathbf{1} + Af \\ & & x & \in & \mathbb{Z}^p \\ & & \epsilon & \geq & 0. \end{array} \tag{4.13}$$

If the optimal value $\bar{\epsilon}$ to (4.13) is greater than 0, then the optimal $\bar{x}$ in the solution is an integral vector satisfying $A(\bar{x} - f) < 1$. In this case we extend $X \leftarrow X \cup \{\bar{x}\}$ and resolve problem (4.12). Otherwise we stop, since we have a solution to (4.11). This method is outlined at Algorithm 1.

Algorithm 1: Lattice point generation method

input: $f \in \mathbb{R}^m_+$, $R \in \mathbb{R}^{m \times k}$;

Let $X \subset \mathbb{Z}^m$ be the m-dimensional boolean cube around f;

while *true* **do**

- Solve (4.12) for X;
- Let $\bar{A}$ be the optimal solution;
- Solve (4.13) for $\bar{A}$;
- Let $(\bar{\epsilon}, \bar{x})$ be the optimal solution;
- **if** $\bar{\epsilon} = 0$ **then**
  - **return** $\bar{A}$;
- **else**
  - $X \leftarrow X \cup \bar{x}$;

Problem (4.11) boasts a number of adventages over problem (4.10). First of all, it is a pure linear programming problem in $p|X| + k$ variables.

Secondly, from an implementation point of view, to solve (4.10), at every new generated lattice point, we have to add $k(\ell + 1) + 1$ new constraints and $\ell(\ell p + 1)$ new variables (ℓ of which are integer). We note that $\ell = 2^p$ to consider all possible lattice-free convex sets on $\mathbb{R}^p$. However, when we add a new lattice point to (4.12), we create only $k + 1$ new constraints and p new linear variables. Thus by Theorem 4.6.1, we can solve problem (4.10) by solving (4.12) and (4.13) repeatedly. As our computational experiments in the following sections show a very small number of calls to (4.13) are needed in practice.

4.7 Computational Improvements

In this section we discuss implementation details on how to solve (4.12) and the important issue of how to choose the weights w_j, $j = 1, \ldots, k$ in (4.12). Let $\sum_{j=1}^{k} \alpha_j s_j \geq 1$ be a valid inequality for the relaxed simplex table, P_{LP}, and assume that $\alpha_j > 0$ for all j for simplicity. The lattice-free convex set associated with this inequality is $L_\alpha = \text{conv}\{f + \frac{r_j}{\alpha_j} : j = 1, \ldots, k\}$ ([4]). By using Corollary 4.4.4 and assuming L_α is a maximal lattice-free convex set, it has at most 2^p facets. Therefore we can heuristically say at most 2^p columns are actively carrying information (that are corresponding to the vertices of L_α). By using this fact, we reduce the number of columns selected to 2^p, or a similar fixed value, q, which is a small multiple of 2^p. We again note that p is the number of fractional simplex rows to be selected.

Now we need to make a justified decision on how to choose the most important q nonbasic columns of the simplex table. Let us return to problem (4.12). We say that two vectors, r_i and r_j are within each other's neighborhood if

$$r_i^T r_j \geq (1 - \epsilon)\|r_i\|\|r_j\|$$

where $\epsilon > 0$ is a small constant. We can eliminate r_j by summing $\|r_j\|/\|r_i\|$ to the objective coefficient of λ_i without changing the description of the problem. However, for numerical stability, we only add $\|r_j\|$. The following algorithm implements this detail:

Algorithm 2: Compute objective weights

input: A set nonbasic columns $r_1, \ldots, r_k$, and their priorities $w_1^0, \ldots, w_k^0$, and an ϵ;

for $j = 1, \ldots, k$ **do**
  $w_j \leftarrow \|r_j\| w_j^0$;
for $j = 1, \ldots, k$ **do**
  for $i = j+1, \ldots, k$ **do**
    if $r_i^T r_j \geq (1-\epsilon)\|r_i\|\|r_j\|$ **then**
      $w_i \leftarrow w_i + \|r_j\| w_j^0$;
      $w_j \leftarrow w_j + \|r_i\| w_i^0$;
return w;

Algorithm 2 sums the magnitude of all neighboring columns in one direction. A heuristic justification can be that if we have many columns in one direction, we want their coefficients in the valid inequality as small as possible. Then we sort the resulting objective vector, and choose the first q *distinct* columns (distinct in the sense that they are not within each other's neighborhood), where q is also a fixed value. This gives us a heuristically well chosen description of the actual problem (knowing that $q > 2^p$), which *completely fixes all dimensions.*

Lemma 4.7.1. *Assuming that the nonbasic columns columns are normalized and a fixed number of, p, rows of the simplex tableau have been chosen, we need at most $\frac{M}{\epsilon^{p-1}}$ columns to approximate problem* (4.11)*, where M is a fixed constant.*

Proof. The assumption of normalization implies that the nonbasic columns span the $p-$ball. The volume and surface area of a $p-$ball of R radius are $V_p(R) = \frac{\pi^{p/2}}{\Gamma\left(\frac{p}{2}+1\right)} R^p$ and $S_{p-1}(R) = \frac{p\pi^{p/2}}{\Gamma\left(\frac{p}{2}+1\right)} R^{p-1}$, respectively (where Γ is the gamma function). Therefore we are diving the surface area of $p-$unit ball to volumes of $(p-1)-$ball of radius ϵ:

$$\frac{\frac{p\pi^{p/2}}{\Gamma\left(\frac{p}{2}+1\right)}}{\frac{\pi^{(p-1)/2}}{\Gamma\left(\frac{p-1}{2}+1\right)}\epsilon^{p-1}} = \frac{p\pi^{1/2}}{\epsilon^{p-1}} \frac{\Gamma\left(\frac{p}{2}+1\right)}{\Gamma\left(\frac{p}{2}+\frac{1}{2}\right)} \leq \frac{M}{\epsilon^{p-1}}.$$

where last inequality follows from the fact that p is fixed. Therefore there exists a fixed

$M \geq p\pi^{1/2}\frac{\Gamma\left(\frac{p}{2}+1\right)}{\Gamma\left(\frac{p}{2}+\frac{1}{2}\right)}$ depending on only p. $\square$

Let us note that, since integer programming in a fixed number of variables can be solved in polynomial time (see [1, 67]), we can claim

Corollary 4.7.2. *Problem* (4.13) *can be solved in polynomial time when p is a fixed constant by [1, 67]. Therefore, problem* (4.12) *can be solved in polynomial time by [48].*

Let us further add that in practice we can start in X with the 2^p integer points we can obtain from f by rounding up and/or down its coordinates. Practical results show that in most cases the above method terminates just with a few calls to problem (4.13):

Remark 4.7.3. *On our test bed we computed a total of 10529, 10528, 10522 and 10506, 2,3,4 and 5 row cuts respectively. The mean numbers of lattice points generated were 1.3116, 4.6967, 9.3581 and 15.7244, respectively for problem* (4.12) *(while standard deviations were 1.3907, 3.8544, 7.2367 and 10.9894).*

4.7.1 A motivating example

As an illustration we consider the small integer program from [77] depicted in Figure 4.7.1. We can observe that there is a 2-row simplex inequality that cuts into the CG-closure, and in fact is a facet of the integer hull (the only facet missing from the CG-closure.)

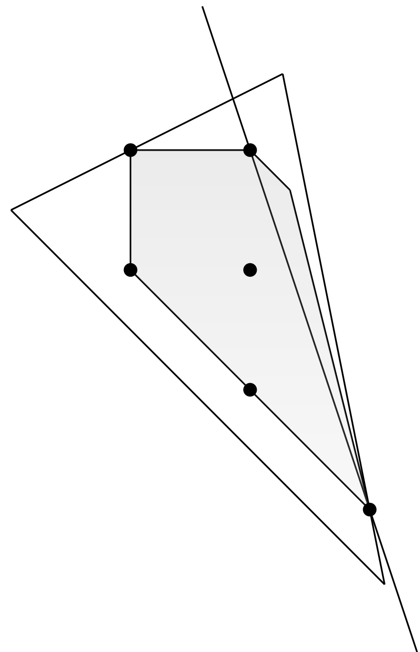


Figure 4.2: MVI cut

4.8 Numerical Results

We coded all our algorithms in C/C++ environment using g++ compiler with -O3 option and ran the experiment on a i7 2920xm @2.5GHz computer with 16GB 1600mhz memory installed. We use CPLEX v12.6 x64 for optimization purposes and coded our method as a cut-callback using CPLEX C API. We only apply cut-callback at the root node ([35, 70]). It is called right after CPLEX computes the root LP relaxation and before CPLEX generates any cutting planes for the root node. We also turn on CPLEX cut filter to make sure numerically stable and efficient cuts are generated. After our cut-callback, CPLEX applies well-known cutting planes such as Gomory mixed integer cuts, zero-half cuts, flow covers, knapsack covers, lift-and-project cuts, mixed integer rounding cuts and many others from the literature. It can also generate more inequalities as consequences of our inequalities that went through CPLEX filter. We further note that, we turn off CPLEX presolve to have easy access to the simplex tableau at the root node (this is easily justified as presolve doesn't have any implications on cut generation).

The much celebrated approach in valid inequality computation is to test cut generation against instances in the well-known problem database MIPLIB ([35, 70, 9]). Our testbed includes famous instances from MIPLIB 3 ([14]) and MIPLIB 2003 ([3]) as given in Table 4.2. Our main indicator of cut quality is the percentage integrality gap closed at the root node:

$$GC = \frac{Z_{CUTS} - Z_{LP}}{Z_{IP} - Z_{LP}} \times 100$$

where Z_{CUTS}, Z_{LP} and Z_{IP} represents objective value of the strengthened root relaxation with cutting planes, root LP relaxation, and integer optimal solution, respectively.

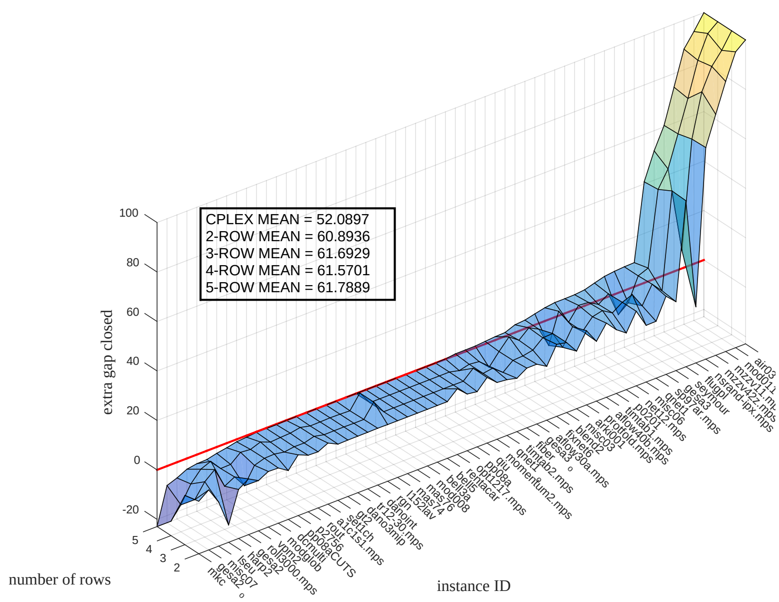


Figure 4.3: CPLEX + x-row gap closed

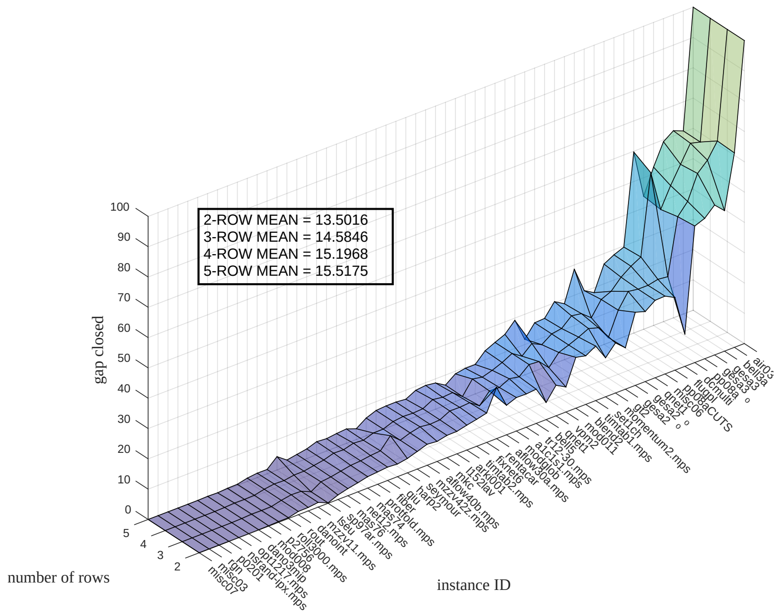


Figure 4.4: x-row only gap closed

We summarize our implementation as follows: Given a simplex tableau, and integer constants MAX_ROWS and MAX_CUTS, the number of rows to be taken into consideration and the number of cutting planes to be calculated respectively, we generate a random MAX_ROWS combination of the fractional row indices of the simplex tableau. Then we proceed into generating a cut from the selected rows of the simplex tableau. We repeat the process MAX_CUTS times. We also consider only the most fractional MAX_CONSIDERED rows of the simplex tableau. An interesting improvement of this method would consider rows in a similar fashion to pseudo-cost branching. As numerical evidence shows maximum infeasible branching doesn't seem to perform better than randomization on average ([2]).

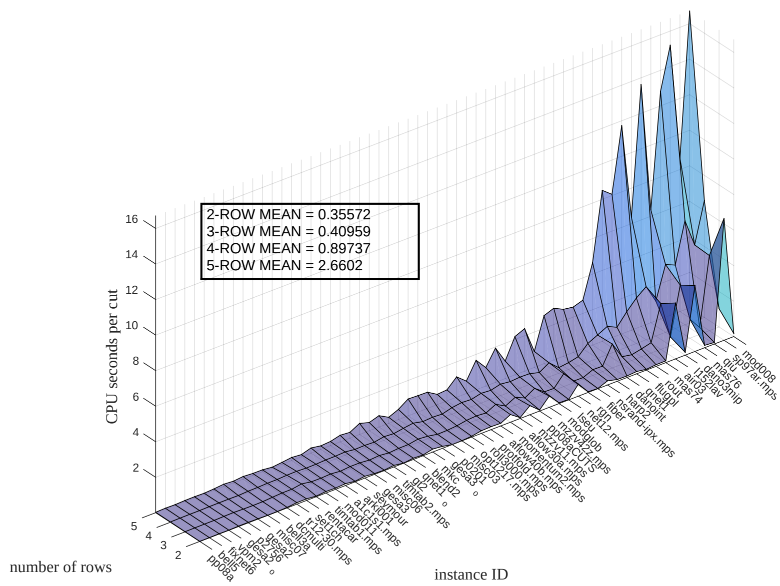


Figure 4.5: x-row cpu time per cut

Table 4.1: Omitted instances

Reason	**Instances**
Large	msc98-ip, atlanta-ip, ds, fast0507, nw04, rd-rplusc-21, stp3d, t1717
Solved at root	mod010, 10teams, khb05250, p0548, p0282, gen, vpm1, egout, p0033, mitre, disktom, manna81
Non-optimal	liu, momentum3
No improvement	pk1, markshare1, markshare2, air04, air05, swath, stein45, stein27, glass4
Not enough rows	cap6000
IP = LP relax.	noswot, enigma

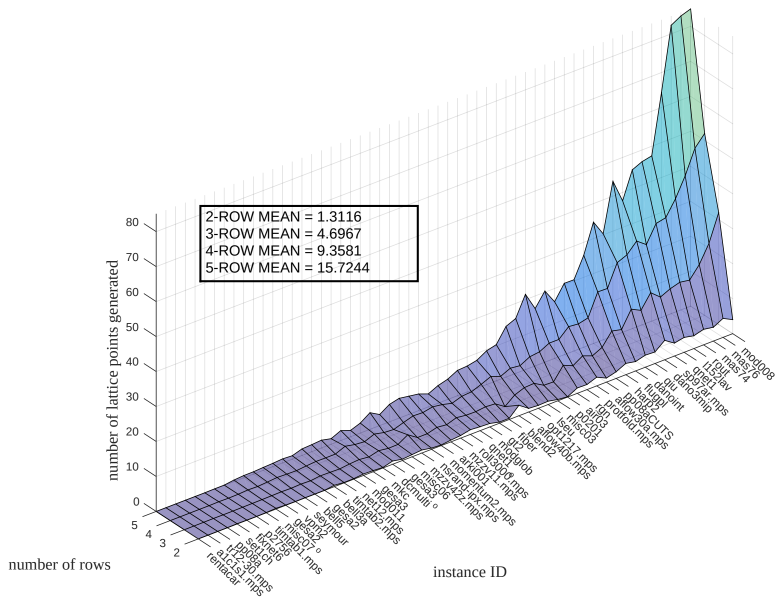


Figure 4.6: x-row lattice-points per cut

In this scnerio, we will compare bare CPLEX with CPLEX plus 2 to 5-row cuts, considering addition of up to 200 multi-row simplex cuts while taking into acocunt up to the most fractional 100 rows of simplex tableau considering upto 64 most impactful columns using Algorithm 2. While we add cuts to CPLEX, we will use CPLEX C API cut callback library.

We omit some instances from MIPLIB due to the following reasons provided at Table 4.1. Here *Large* stands for standard form representation of the problem doesn't fit computer memory. *Solved at root* is for problems which closes the full gap at the root node. *Non-optimal* used for problems which we don't know the optimal IP value. *No improvement* stands for neither CPLEX nor our method can improve on the LP relaxation value. *Not enough rows* is used for problems with less than MAX_ROWS infeasible rows and finally *IP = LP relax.* means the IP value of the problem being equal to the LP relaxation value, hence causing division by 0 for gap closed formula. After this filter, 56 problems have been taken into consideration.

Table 4.2: Results by instances

Instance	GC_0	GC_2	GC_3	GC_4	GC_5	**Instance**	GC_0	GC_2	GC_3	GC_4	GC_5
mkc	37.121403	35.437088	34.118243	20.027315	14.212249	pp08a	97.013341	96.434595	96.979056	97.037409	97.023902
gesa2_o	92.260628	93.603573	86.015417	80.249322	84.405002	qiu	2.349541	5.955159	6.022981	1.420135	2.478225
misc07	7.068154	1.971326	6.513410	1.971326	0.716846	momentum2	68.198163	68.246920	68.252021	68.276031	68.826410
lseu	97.183590	81.247454	100.000000	96.506626	92.831969	qnet1_o	96.047589	95.677526	96.734595	95.677526	96.698456
harp2	58.164106	55.069433	52.720641	59.125207	54.496216	timtab2	55.311219	53.981677	52.699296	51.096210	56.008246
gesa2	97.958662	97.589011	94.728808	94.469470	94.419895	fiber	93.871569	95.108410	94.568127	94.467831	94.917186
roll3000	51.466999	48.867330	43.159196	48.095515	48.894815	gesa3_o	72.799442	74.061419	73.742624	77.770232	74.195758
vpm2	83.337685	82.886578	81.699602	83.308005	81.471928	aflow30a	64.163892	63.020778	65.916099	66.248126	65.701268
modglob	96.353127	95.755694	96.018597	95.943012	95.369144	fixnet6	91.800889	98.022383	96.994860	97.376539	94.928160
dcmulti	86.655317	83.492665	86.492679	87.717548	85.990523	blend2	27.718236	33.095841	33.095841	31.192349	31.192349
pp08aCUTS	92.856341	94.504640	91.974733	92.338314	92.594441	misc03	24.268454	24.655811	22.510317	19.604087	28.722777
p2756	99.115418	99.057743	98.837487	98.908258	98.879489	arki001	13.137136	20.240006	18.190093	20.774198	18.510456
rout	0.322451	0.118232	0.014657	0.322451	0.094838	protfold	10.101708	11.385697	10.515550	10.755402	12.789627
a1c1s1	73.742148	75.516698	74.580045	74.508376	73.692688	aflow40b	55.376336	62.454200	61.360679	62.312818	61.314268
set1ch	99.841368	99.640676	99.788202	99.675352	99.802806	timtab1	58.762584	61.643081	65.689201	65.338511	64.792293
gt2	96.279940	96.190801	96.284959	96.188713	96.255520	p0201	68.519959	68.519959	72.972973	72.761565	74.975578
dano3mip	0.390377	0.330389	0.403919	0.382086	0.366099	net12	25.996575	33.259117	33.197300	32.953905	33.654799
tr12-30	99.623811	99.594997	99.571442	99.572680	99.604903	misc06	91.206025	91.206025	100.000000	88.190658	100.000000
danoint	3.244113	3.299521	3.200688	3.033587	3.235708	qnet1	90.701393	90.800590	93.401645	90.917251	100.000000
rgn	94.562227	94.562227	100.000000	100.000000	94.562227	sp97ar	-0.000000	8.676397	9.897638	9.604003	9.604045
l152lav	2.005540	2.005540	2.005540	2.005540	2.005540	gesa3	71.546513	76.489855	77.231138	82.766360	81.286557
mas74	8.462663	8.462663	8.462663	8.462663	8.462663	seymour	0.000000	44.219030	44.514932	41.468225	40.864537
mas76	13.091190	13.091190	13.091190	13.091190	13.091190	flugpl	13.903516	13.669800	32.812590	62.115725	65.748886
mod008	90.328434	90.328434	90.328434	90.328434	90.328434	nsrand-ipx	0.000000	62.635253	62.392968	60.838015	60.595760
bell3a	70.739634	70.739634	70.739634	70.739634	70.739634	mzzv42z	-0.000000	74.529338	79.973774	73.628254	74.504464
bell5	96.935281	96.935281	96.935281	96.935281	96.935281	mzzv11	0.020541	86.316451	88.714773	87.526535	88.355249
rentacar	38.967690	42.619560	38.967690	38.967690	38.967690	mod011	-0.000000	96.685612	93.639308	96.802085	93.927570
opt1217	36.131184	36.131184	36.131184	36.131184	36.131184	air03	0.000000	100.000000	100.000000	100.000000	100.000000
						MEAN	52.0897	60.8936	61.6929	61.5701	61.7889
						CPU	N\A	70.0610	76.7551	152.8512	456.2307

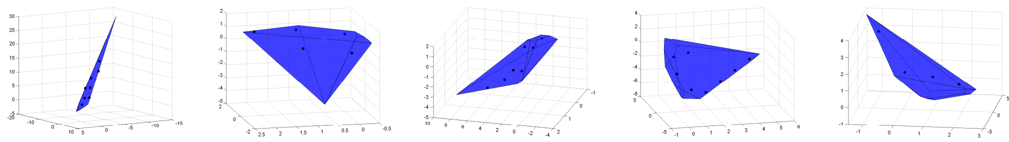


Figure 4.7: Lattice-free convex sets from aflow30a.

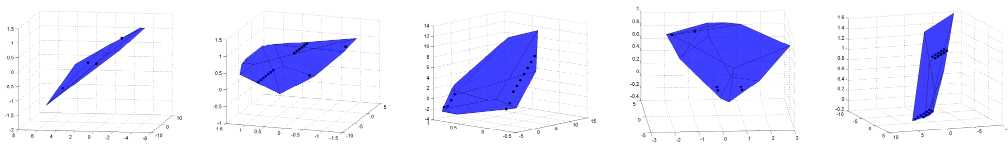


Figure 4.8: Lattice-free convex sets from protfold.

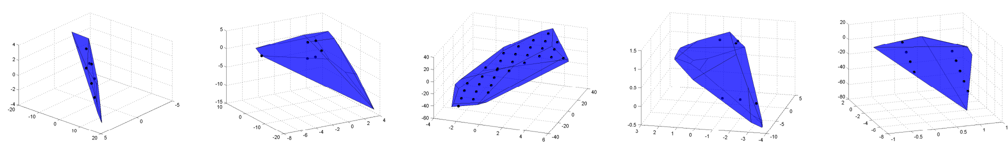


Figure 4.9: Lattice-free convex sets from sp97ar.

Our results are summarized by Table 4.2. Here GC_0, GC_2, GC_3, GC_4 and GC_5 represents the gap closed if we use CPLEX base, CPLEX plus 2 to 5-row cuts, respectively. We observe that CPLEX plus 5-row cuts performs the best, while CPLEX plus 4-row cuts performs worse than CPLEX plus 3-row cuts. However we note that CPLEX cut filter is ON during this experiment. Which can lead to unpredictable results such as this. To verify that x+1-row cuts performs better than x-row cuts on average, we carry out the experiment at Figure 4.4. Here we only measure the gap closed using x-row cuts only. It can be observed that 2 to 5-row cuts closes approximately, 13.5, 14.6, 15.2 and 15.5 percent gap at the root node respectively. We can see increasing performance with the increased information in this experiment, which falls inline with the notion of x-row cuts have fixed shaped on the x+1-row space (splits).

We note that the zero baseline at Figures 4.3 represents CPLEX performance. Hence positive value means increased closure of gap while negative value means hindering CPLEX performance. As we observed similar numbers in Table 4.2, the addition of x-row inequalities helps on a lot of instances, while not hurting CPLEX's performance (on average). We also provide various statistics like number of lattice points generated and the mean cpu time per cut at Figures 4.5 and 4.6. In these figures we note that the mean cpu time per cut is very acceptable and the number of lattice points generated is usually a very small number, which provides justifiable memory requirements.

4.9 Conclusion

Generation of valid inequalities is a crucial component of MIP software that can significantly improve the efficiency of the branch and bound procedure. Although the theory of multi-row simplex cuts has been known for many decades, state-of-the-art MIP software uses only single row cuts or variations of split inequalities.

In this study we provide a simple poly-time separation method to generate multi-row simplex cuts for a general MIP while considering a fixed number of tableau rows. Our computational experiment shows improvement over a standard problem set, MIPLIB 3 and 2003. We also observe that considering more rows of the simplex tableau improves the quality of the multi-row cuts on average. We hope that our positive computational experiment will encourage more and more researchers to discover compelling evidence of information that is hidden in the multiple rows of the simplex tableau, which has eluded the attention of industry for many decades.

Chapter 5

Optimal Resilient Distribution Grid Design

Natural disasters such as earthquakes, hurricanes, and other extreme weather pose serious risks to modern critical infrastructure including electrical distribution grids. At the peak of Hurricane Sandy, 65% of New Jersey's customers lost power [74]. Recent U.S. government sources [36, 101] suggest that new methodologies for improving system resilience to these events is necessary. Here, we focus on developing methods for designing and upgrading distribution grids to better withstand and recover from these threats that are inspired by techniques developed in the artificial intelligence and operations research communities. Our approach minimizes the upgrade budget while meeting a minimum standard of service by selecting from a set of potential upgrades, e.g. adding redundant lines, adding distributed (microgrid) generation (i.e. wind, solar, and combined heat and power), hardening existing components, etc.

We formulate our approach, i.e. Optimal Resilient Distribution Grid Design (ORDGDP), as a two-stage mixed-integer program. The first (investment) stage selects from the set of potential upgrades to the network. The second (operations) stage evaluates the network performance benefit of the upgrades against a set of damage scenarios sampled from a stochastic distribution. We first develop an exact solution method that exploits decomposition across the sampled scenarios. We also develop a metaheuristic that we call Scenario Based Variable Neighborhood Decomposition Search (SBVNDS) that is a hybrid of Variable Neighborhood Search [66] and the exact method. We present numerical evidence that our exact method is more efficient than out-of-the-box commercial mixed-integer programming solvers, and that our heuristic achieves near-optimal results in a fraction of the time required by exact methods.

5.0.1 Background

Network design problems and their variations are generally NP-complete [100, 76, 55]. However, recent work by [12] demonstrates that AI-based methods can lead to substantial improvement for realistic applications.

While the specific problem of designing resilient distribution systems is novel, a number of related problems exist. The flow of electric power in tree-like distribution networks is related to multi-commodity network flows making our problem similar to the design of multi-commodity flow networks with stochastic link and edge failures [93, 39]. However, the second stage of our formulation requires binary variables making our problem considerably more difficult than typical second-stage flow problems. The interdiction literature includes related max-min or min-max problems where the goal is to operate or design a system to make it as resilient as possible to an adversary who can damage up to k elements. Such models are similar to ours if a k is chosen that bounds the worst-case disaster [21, 20, 92, 30]. Binary variables at all stages make these models computationally challenging and solvable only for small k. Here, we exploit the probabilistic nature of our adversary to increase the size of tractable problems (eliminates a stage of binary variables).

In power engineering, papers have primarily focused on resilient system operation [42, 68, 59] using controls such as line switching. The ORDGDP is a fundamental generalization of the resilient operations problem because 1) this problem is embedded in our second stage and 2) minimizing the number of switch actions [68] can be thought of as a design problem for a single scenario. Finally, there is also a general power grid expansion planning problem for stochastic events [51] that is a variant of the single commodity flow problem, with the twist that flows are not directly controllable. Like stochastic multi-commodity flow, the second-stage variables are not binary.

5.1 Problem Description

Nomenclature

Parameters

$\mathcal{N}$	set of nodes (buses).
$\mathcal{E}$	set of edges (lines and transformers).
$\mathcal{S}$	set of disaster scenarios.
$\mathcal{D}_s$	set of edges that are inoperable during $s \in S$.
$\mathcal{D}'_s$	set of edges that are inoperable even though they are hardened during disaster $s \in S$.
c_{ij}	cost to build a line between bus i and j. 0 if line already exists.
κ_{ij}	cost to build a switch on a line between bus i and j.
ψ_{ij}	cost to harden a line between bus i and j.
$\zeta_{i,k}$	cost of generation capacity on phase k at bus i.
α_i	cost to build a generation facility at node i.
Q_{ijk}	line capacity between bus i and bus j on phase k.
$\mathcal{P}_{ij}$	set of phases for the line between bus i and bus j.
$\mathcal{P}_i$	set of phases allowed to consume or inject at bus i.
β_{ij}	parameter for controlling how much variation in flow between the phases is allowed.
$d_{i,k}$	demand for power at bus i for phase k.
$G_{i,k}$	existing generation capacity on phase k at node i.
$Z_{i,k}$	maximum amount of generation capacity on phase k that can be built at node i.
$\mathcal{C}$	the set of sets of nodes that includes a cycle.
λ	fraction of critical load that must be served.
γ	fraction of all load that must be served.
$\mathcal{L}$	set of buses whose load is critical.

Variables

x_{ij} determines if line i, j is built.

τ_{ij} determines if line i, j has a switch.

t_{ij} determines if line i, j is hardened.

$z_{i,k}$ determines the capacity for generation on phase k at node i.

u_i determines the generation capacity built at node i.

x_{ij}^s determines if line i, j is used during disaster s.

τ_{ij}^s determines if switch i, j is used during disaster s.

t_{ij}^s determines if line i, j is hardened during disaster s.

$z_{i,k}^s$ determines the capacity for generation on phase k at bus i during disaster s.

u_i^s indicates if the generation capacity is used at node i during disaster s.

$g_{i,k}^s$ generation produced for bus i on phase k during disaster s.

$l_{i,k}^s$ load delivered at bus i on phase k during disaster s.

$y_{i_j}^s$ determines if the jth load at bus i is served or not during disaster s.

$f_{ij,k}^s$ flow between bus i and bus j on phase k during disaster s.

$\bar{x}_{ij}^s$ determines if at least one edge between i and j is used during disaster s.

$\bar{\tau}_{ij}^s$ determines if at least one switch between i and j is used during disaster s.

$x_{ij,0}^s$ determines if there exists flow on line i, j from j to i, during disaster s.

$x_{ij,1}^s$ determines if there exists flow on line i, j from i to j, during disaster s.

5.1.1 Distribution Grid Modeling

A distribution network is modeled as a graph with nodes $\mathcal{N}$ (buses) and edges $\mathcal{E}$ (power lines and transformers). In the physical system, each edge is composed of one, two, or three circuits or "phases" and the electrical loads at the nodes are connected to and consume power from specific phases [38] ($\mathcal{P}$). In many papers, multiple phases are approximated as a single phase with a single edge flow. However, under the damaged and stressed conditions considered in this work, the flows on the phases are often unbalanced, i.e. unequal, making it important to model all phases to accurately evaluate flow constraints on each phase. The phase flows are *not* directly controllable, but are related to nodal voltages and power injections by non-convex, physics-based equations [38]. Incorporation of these equations into the current formulation increases the complexity, however, the structure of distribution networks enables a simplification.

The design of protection systems for the vast majority of distribution circuits is based on the these circuits having a tree-like structure. Therefore, although distribution grids are often designed to contain many possible loops, switches are used to ensure that these grids are operated in a tree or forest topology. While, the switches introduce binary variables that increase the complexity of the ORDGDP, a linearized version of the electrical power flow equations (i.e. DC power flow) on the resulting trees is equivalent to a commodity flow model. We use a multi-commodity flow model that models each phase separately (Fig. 5.1).

The linearization of the power flow equations assumes uniform voltage magnitude at all nodes and ignores reactive power flows. In practice, we expect these are reasonable approximations because, prior to being upgraded, the distribution grid is already feasible with respect to voltage and reactive power flows. By adding lines or distributed power sources, we put loads closer to generation thereby reducing voltage variability and reactive power flow and the potential for violating unmodeled constraints. In principle, it is possible to construct solutions where this is not the case, but the solutions to ORDGDP found by our algorithms has not resulted in these situations. However, this is an important area of future work, and we are developing methods to eliminate solutions that violate voltage or reactive power flow limits.

5.1.2 Damage Modeling

The ORDGDP is also defined by a set of scenarios, $\mathcal{S}$. These scenarios are provided by a user or are drawn from a probabilistic damage model (the case here). Each scenario is defined by the lines of the network that are damaged and are inoperable. Many networks consist of mix of line and pole types, i.e. overhead and underground lines. In our model, this is reflected by different damage probabilities. For the purposes of this paper, we assume a static (peak) demand profile for each scenario. However, we note that multiple load patterns can be included by creating scenarios representing the same damage set with different demand profiles.

5.1.3 Upgrade Options

We focus on four user-definable design options in distribution networks: 1) Hardening existing lines to lower the probability of damage, 2) Build new lines to add redundancy, 3) Build switches, to add operating flexibility, and 4) building distributed generation (sources of power). While deregulation has split network operation from generation ownership in transmission systems, in distribution systems (the focus here), this split varies from locale to locale and is our motivation for including generation as a design option. For example, Central Hudson has recently added generators for resilience and reliability [18].

$$\mathcal{Q}(s) = \{x^s, \tau^s, t^s, z^s, u^s :$$

$$-x^s_{ij,0} Q_{ijk} \leq f^s_{ij,k} \leq x^s_{ij,1} Q_{ijk} \qquad \forall ij \in \mathcal{E}, k \in \mathcal{P}_{ij} \tag{5.1}$$

$$x^s_{ij,0} + x^s_{ij,1} \leq x^s_{ij} \qquad \forall ij \in \mathcal{E} \tag{5.2}$$

$$(\tau^s_{ij} - 1) Q_{ijk} \leq f^s_{ij,k} \leq (1 - \tau^s_{ij}) Q_{ijk} \quad \forall ij \in \mathcal{E}, k \in \mathcal{P}_{ij} \tag{5.3}$$

$$\frac{\sum_{k \in \mathcal{P}_{ij}} f_{ij,k}}{\frac{|\mathcal{P}_{ij}|}{(1-\beta_{ij})}} \leq f^s_{ij,k'} \leq \frac{\sum_{k \in \mathcal{P}_{ij}} f_{ij,k}}{\frac{|\mathcal{P}_{ij}|}{(1+\beta_{ij})}} \qquad \forall ij \in \mathcal{E}, k' \in \mathcal{P}_{ij} \tag{5.4}$$

$$x^s_{ij} = t^s_{ij} \leq \begin{cases} 0 & \text{if } ij \in \mathcal{D}'_s \\ 1 & \text{else} \end{cases} \qquad \forall ij \in \mathcal{D}_s \tag{5.5}$$

$$l^s_{i,k} = \sum_{j=0}^{n_i} y^s_{i_j} d_{i_j,k} \qquad \forall i \in \mathcal{N}, k \in \mathcal{P}_i \tag{5.6}$$

$$0 \leq g^s_{i,k} \leq z^s_{i,k} + G_{i,k} \qquad \forall i \in \mathcal{N}, k \in \mathcal{P}_i \tag{5.7}$$

$$g^s_{i,k} - l^s_{i,k} - \sum_{j \in \mathcal{N}} f^s_{ij,k} = 0 \qquad \forall i \in \mathcal{N}, k \in \mathcal{P}_i \tag{5.8}$$

$$0 \leq z^s_{i,k} \leq Z_{i,k} u_i \qquad \forall i \in \mathcal{N}, k \in \mathcal{P}_i \tag{5.9}$$

$$\sum_{ij \in \mathcal{E}(C)} (x^s_{ij} - \tau^s_{ij}) \leq |V| - 1 \qquad \forall C \in \mathcal{C} \tag{5.10}$$

$$\tau^s_{ij} \leq x^s_{ij} \qquad \forall ij \in \mathcal{E} \tag{5.11}$$

$$\sum_{i \in \mathcal{L}, k \in \mathcal{P}_i} l^s_{i,k} \geq \lambda \sum_{i \in \mathcal{L}, k \in \mathcal{P}_i} d_{i,k} \tag{5.12}$$

$$\sum_{i \in \mathcal{N} \setminus \mathcal{L}, k \in \mathcal{P}_i} l^s_{i,k} \geq \gamma \sum_{i \in \mathcal{N} \setminus \mathcal{L}, k \in \mathcal{P}_i} d_{i,k} \tag{5.13}$$

$$x^s, y^s, \tau^s, u^s, t^s \in \{0, 1\}\} \tag{5.14}$$

Figure 5.1: Set of feasible distribution networks

5.1.4 Optimization model

Given a disaster $s \in S$, $\mathcal{Q}(s)$ in Fig. 5.1 defines the set of feasible distribution networks. The constraints of $\mathcal{Q}(s)$ involve a number of well-known constraints in the combinatorial optimization literature, including knapsacks, multi commodity flows, and tree constraints. In this model, Eq. 5.1 is a capacity constraint on phase flows. When the line is not built the flow is forced to 0 by x^s. Eq. 5.2 forces all phases to flow in the same direction, an engineering constraint. Eq. 5.3 states that the flow on a line is 0 when the switch is open. Eq. 5.4 limits the fractional flow imbalance between the phases to a value smaller than β_{ij}. Imbalance between phases cannot be extreme otherwise equipment may be damaged. Here, we use $\beta_{ij} = 0.15$ for transformers, and 1.0 otherwise. Eq. 5.5 removes components in the damage set from the network by linking the two damage sets with the hardening variables. Eq. 5.6 requires that all or none of the load at a bus is served. Once again, this an engineering limitation of most networks. Eq. 5.7 limits the distributed generation output by the generation capacity. Eq. 5.8 ensures flow balance at the nodes for all phases. Eq. 5.9 caps the generation capacity installed at the nodes. Eq. 5.10 eliminates network cycles, forcing a tree or forest topology. Eq. 5.11 states a switch is used only if the line exists. Eq. 5.12 ensures a minimum fraction λ of critical load is served. Here, we generally require $\lambda = 0.98$. Eq. 5.13 ensures that a minimum fraction of load is served. Here, $\gamma = 0.5$. Eqs. 5.12 and 5.13 are the resilience criteria that must be met by $\mathcal{Q}(s)$ and are similar to the $n - k - \epsilon$ criteria of [20]. Eq. 5.14 states which variables are discrete.

One of the more difficult constraints in this formulation is Eq. 5.10 due to possible combinatorics. There are different ways to implement cycle constraints, and we use the formulation in Fig. 5.2.

$$\sum_{ij \in \mathcal{E}(C)} (\bar{x}_{ij}^s - \bar{\tau}_{ij}^s) \leq |V| - 1 \qquad \forall C \in \mathcal{C} \tag{5.15}$$

$$x_{ij}^s \leq \bar{x}_{ij}^s \qquad \forall ij \in \mathcal{E} \tag{5.16}$$

$$\tau_{ij}^s \geq x_{ij}^s + \bar{\tau}_{ij}^s - 1 \qquad \forall ij \in \mathcal{E} \tag{5.17}$$

Figure 5.2: Cycle constraints

While the multi-graph structure introduces a large number of cycles, there is a relatively small number of cycles when the multi-edges are reduced to one edge. Thus, we introduce binary variables (linear number) for the edges of the corresponding single-edge graph and enumerate the possible cycles in that graph (Eq. 5.15). Then, Eqs 5.16 and 5.17 are used to pass information between artificial cycle variables and the actual line and switch variables.

For each $s \in \mathcal{S}$, $\mathcal{Q}(s)$ determines the set of feasible distribution networks. There are some redundant variables in this formulation that improve the separability of the problem. The ORDGDP is the minimum cost design that falls in the intersection of all the $\mathcal{Q}(s)$ (Fig. 5.3).

$$\begin{aligned}
\min \quad & \sum_{ij \in \mathcal{E}} c_{ij} x_{ij} + \sum_{ij \in \mathcal{E}} \kappa_{ij} \tau_{ij} + \sum_{ij \in \mathcal{E}} \psi_{ij} t_{ij} \\
& + \sum_{i \in \mathcal{N}} \alpha_i u_i + \sum_{i \in \mathcal{N}, k \in \mathcal{P}_i} \zeta_{i,k} z_{i,k} \qquad (5.18)
\end{aligned}$$

s.t.

$$x_{ij}^s \leq x_{ij} \qquad \forall ij \in \mathcal{E}, s \in \mathcal{S} \qquad (5.19)$$

$$\tau_{ij}^s \leq \tau_{ij} \qquad \forall ij \in \mathcal{E}, s \in \mathcal{S} \qquad (5.20)$$

$$t_{ij}^s \leq t_{ij} \qquad \forall ij \in \mathcal{E}, s \in \mathcal{S} \qquad (5.21)$$

$$z_{i,k}^s \leq z_{i,k} \qquad \forall i \in \mathcal{N}, k \in \mathcal{P}_i, s \in \mathcal{S} \qquad (5.22)$$

$$u_i^s \leq u_i \qquad \forall i \in \mathcal{N}, s \in \mathcal{S} \qquad (5.23)$$

$$z_{i,k} \leq M_{i,k} u_i \qquad \forall i \in \mathcal{N}, k \in \mathcal{P}_i \qquad (5.24)$$

$$(x^s, \tau^s, t^s, z^s, u^s) \in \mathcal{Q}(s) \qquad \forall s \in \mathcal{S} \qquad (5.25)$$

$$x, \tau, t, u \in \{0, 1\} \qquad (5.26)$$

Figure 5.3: Optimal Resilient Distribution Grid Design

Eq. 5.18 minimizes the cost of building lines and switches, hardening lines, and building facilities and generation. For notational simplicity, existing lines, switches, and generation are included as variables in the objective with 0 cost, however in practice these enter the formulation as constants. Eqs. 5.19 through 5.24 tie the first stage (construction) decisions with second stage variables ($\mathcal{Q}(s)$). Eq. 5.25 states that the mixed-integer vector $(x^s, \tau^s, t^s, z^s, u^s)$ constitutes a feasible distribution network for scenario s.

5.1.5 Generalizations

Without loss of generality, the formulation in Fig. 5.3 assumes the x_{ij}^s variables are treated as constants if the lines exist and are not in $\mathcal{D}_s$. Furthermore, Fig. 5.3 also assumes that hardened lines and new lines are built with switches. This is reflective of current industry practices and arises from the observation that switch costs are negligible when compared with the cost of the line itself. However, this assumption can be eliminated by modifying constraints 5.19 and 5.21 as follows:

$$\begin{aligned} x_{ij}^s &= x_{ij} \quad \forall ij \notin \mathcal{D}_s \\ x_{ij}^s &\leq x_{ij} \quad \forall ij \in \mathcal{D}_s \\ t_{ij}^s &= t_{ij} \quad \forall ij \notin \mathcal{D}'_s \\ t_{ij}^s &\leq t_{ij} \quad \forall ij \in \mathcal{D}'_s \end{aligned} \tag{5.27}$$

Finally, for notational simplicity, the formulation of Fig. 5.3 also assumes $ij \notin \mathcal{D}_s, ij \in \mathcal{D}'_s$ never occurs. However, if necessary this assumption can be relaxed by introducing auxiliary variables and additional constraints.

5.1.6 Linearized Distribution Flow

Although 3-phase AC power flow has a complicated form, we can use a simplification of it that is well-known to be accurate for distribution networks ([73]), namely the Distribution Flow of Baran and Wu ([8]). The derivation of these equations can be done in the following way. Let us consider Ohm's Law:

$$V_j = V_i - z_{ij} I_{ij}$$

here V_j is complex vector of voltage of bus j, z_{ij} is the complex impedance matrix, and I_{ij} is the complex vector of current on line ij. We multiply both sides by their Hermitian transpose to get:

$$\begin{aligned} V_j V_j^H &= (V_i - z_{ij} I_{ij})(V_i - z_{ij} I_{ij})^H \\ &= V_i V_i^H - V_i I_{ij}^H z_{ij}^H - z_{ij} I_{ij} V_i^H + z_{ij} I_{ij} I_{ij}^H z_{ij} \end{aligned}$$

we do the simplifications: $v_j = V_j V_j^H$ hermitian voltage matrix, $S_{ij} = V_i I_{ij}^H$ complex power flow matrix, $l_{ij} = I_{ij} I_{ij}^H$ are the complex losses on line ij. By eliminating loss terms, we get LinDistFlow:

$$\begin{aligned} v_j &= v_i - (S_{ij} z_{ij}^H + z_{ij} S_{ij}^H) \\ s_j &= \sum_{i:i\to j} \Lambda_{ij} - \sum_{k:j\to k} \Lambda_{jk} \end{aligned}$$

where s_j is the complex power injection of bus j, $S_{ij} = \gamma\Lambda_{ij}$ represents the complex power flow matrix on line ij, Λ_{ij} is a diagonal matrix of phase power flows,

$$\gamma = \begin{bmatrix} 1 & \alpha^2 & \alpha \\ \alpha & 1 & \alpha^2 \\ \alpha^2 & \alpha & 1 \end{bmatrix},$$

$\alpha = e^{-j2\pi/3}$ represents the 120 degrees rotation for phases. Let $\Lambda_{ij} = P_{ij} + iQ_{ij}$ be the active and reactive power flow representation of the power flow.

$$\begin{aligned}
S_{ij}z_{ij}^H &= \gamma(P_{ij} + iQ_{ij})(r_{ij} + ix_{ij})^H \\
&= \gamma(P_{ij} + iQ_{ij})(r_{ij}^T - ix_{ij}^T) \\
&= \gamma(P_{ij}r_{ij}^T + Q_{ij}x_{ij}^T + i(Q_{ij}r_{ij}^T - P_{ij}x_{ij}^T))
\end{aligned}$$

where

$$\begin{aligned}
\gamma P_{ij} r_{ij}^T &= \begin{bmatrix} 1 & \alpha^2 & \alpha \\ \alpha & 1 & \alpha^2 \\ \alpha^2 & \alpha & 1 \end{bmatrix} \begin{bmatrix} P_{ij}^a & & \\ & P_{ij}^b & \\ & & P_{ij}^c \end{bmatrix} \begin{bmatrix} r_{ij}^{aa} & r_{ij}^{ba} & r_{ij}^{ca} \\ r_{ij}^{ab} & r_{ij}^{bb} & r_{ij}^{cb} \\ r_{ij}^{ac} & r_{ij}^{bc} & r_{ij}^{cc} \end{bmatrix} \\
&= \begin{bmatrix} P_{ij}^a r_{ij}^{aa} + P_{ij}^b \alpha^2 r_{ij}^{ab} + P_{ij}^c \alpha r_{ij}^{ac} & \dots & \dots \\ \dots & P_{ij}^a \alpha r_{ij}^{ba} + P_{ij}^b r_{ij}^{bb} + P_{ij}^c \alpha^2 r_{ij}^{bc} & \dots \\ \dots & \dots & P_{ij}^a \alpha^2 r_{ij}^{ca} + P_{ij}^b \alpha r_{ij}^{cb} + P_{ij}^c r_{ij}^{cc} \end{bmatrix} \\
&= \begin{bmatrix} P_{ij}^a r_{ij}^{aa} + P_{ij}^b \bar{r}_{ij}^{ab} + P_{ij}^c \underline{r}_{ij}^{ac} & \dots & \dots \\ \dots & P_{ij}^a \underline{r}_{ij}^{ba} + P_{ij}^b r_{ij}^{bb} + P_{ij}^c \bar{r}_{ij}^{bc} & \dots \\ \dots & \dots & P_{ij}^a \bar{r}_{ij}^{ca} + P_{ij}^b \underline{r}_{ij}^{cb} + P_{ij}^c r_{ij}^{cc} \end{bmatrix}
\end{aligned}$$

where nondiagonal entries are omitted. If we put these equations together, we get LinDistFlow in explicit form:

$$
\begin{aligned}
v_j^a &= v_i^a - 2(r_{ij}^{aa}P_{ij}^a + x_{ij}^{aa}Q_{ij}^a + \bar{r}_{ij}^{ab}P_{ij}^b + \bar{x}_{ij}^{ab}Q_{ij}^b + \underline{r}_{ij}^{ac}P_{ij}^c + \underline{x}_{ij}^{ac}Q_{ij}^c)\\
v_j^b &= v_i^b - 2(\underline{r}_{ij}^{ba}P_{ij}^a + \underline{x}_{ij}^{ba}Q_{ij}^a + r_{ij}^{bb}P_{ij}^b + x_{ij}^{bb}Q_{ij}^b + \bar{r}_{ij}^{bc}P_{ij}^c + \bar{x}_{ij}^{bc}Q_{ij}^c)\\
v_j^c &= v_i^c - 2(\bar{r}_{ij}^{ca}P_{ij}^a + \bar{x}_{ij}^{ca}Q_{ij}^a + \underline{r}_{ij}^{cb}P_{ij}^b + \underline{x}_{ij}^{cb}Q_{ij}^b + r_{ij}^{cc}P_{ij}^c + x_{ij}^{cc}Q_{ij}^c)\\
p_j^l &= \sum_{i:ij\in E} P_{ij}^l - \sum_{k:jk\in E} P_{jk}^l, \quad \forall l \in \{a,b,c\}\\
q_j^l &= \sum_{i:ij\in E} Q_{ij}^l - \sum_{k:jk\in E} Q_{jk}^l, \quad \forall l \in \{a,b,c\}\\
0.95v_{ref}^l \leq v_j^l &\leq 1.05v_{ref}^l, \quad \forall l \in \{a,b,c\}
\end{aligned}
$$

where

$$
\begin{aligned}
z_{ij} &= r_{ij} + ix_{ij}\\
\bar{z}_{ij} &= \bar{r}_{ij} + i\bar{x}_{ij}\\
\underline{z}_{ij} &= \underline{r}_{ij} + i\underline{x}_{ij}
\end{aligned}
$$

$\bar{z}_{ij}$ and $\underline{z}_{ij}$ are $2\pi/3$ and $4\pi/3$ rotated z_{ij}, respectively. We can include these equations in our model and represent the flow with its real and reactive components easily that is a realistic approximation of a 3-phase AC power flow on radial networks.

5.1.7 Chance Constraints

In this chapter we will consider the probabilistic (chance) constrained variant of OR-DGDP. Without loss of generality let us consider an optimization problem of the form:

$$
\begin{array}{llcccl}
\min & c^T x & & & & \\
\text{s.t.} & Ax & & & = & b \\
 & P(Tx & \geq & \xi) & \geq & p \\
 & x & & & \geq & 0,
\end{array}
\tag{5.28}
$$

where $\xi = (\xi_1, \ldots, \xi_r)^T$ is a random vector with finite support. Problem 5.28 is called a chance constrained optimization problem and was first formulated and studied in [79],

[80] and [81], for the case of a discrete ξ. A detailed overview of the problem can be found in [82].

To solve (5.28) with discrete random variables, one usually needs to run either a column generation or cutting plane algorithm over a set of support vectors of the probability distribution, that are called *p-level efficient point* (pLEPs) (Figure 5.4). The notion of a pLEP of a discrete distribution was introduced in [81], where a dual method, for the solution of problem (5.28), was also proposed. Assuming the knowledge of all pLEPs, a cutting plane method was proposed in [84]. First we have to enumerate all pLEPs and then apply the cutting plane method that subsequently generates the facets of the cunvex hull of the pLEPs. Not long after that a column generation method, to generate tight lower and upper bounds was presented in [31], making the numerical solvability of problem (5.28) more realistic, as the enumeration of all pLEPs frequently led to intractability. The method was later extended to general convex programming with probabilistic constraints by the same authors [32]. Problem (5.28) was reformulated as a large scale mixed integer programming problem with knapsack constraints in [90]. Using bounds on the probability of the union of events, new valid inequalities for these mixed integer programming problems have been derived. A general framework to use probability bounds for the solution of chance constrained stochastic programming problem was presented in [83]. We also mention [103], where the algorithm in [31] was further explained.

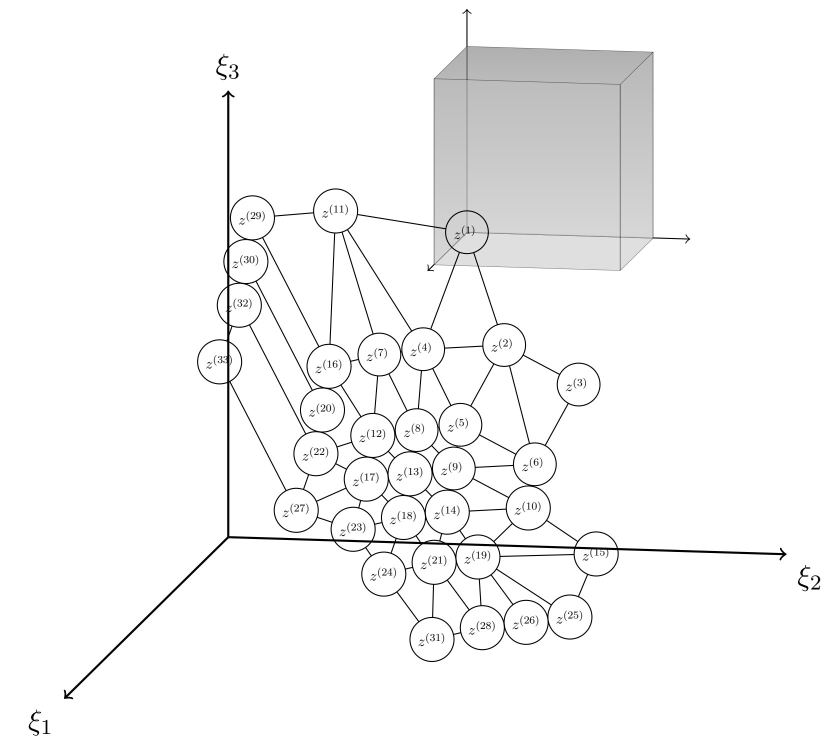


Figure 5.4: pLEP's on $\mathbb{R}^3$

Now let us return to the context of ORDGDP. For some networks, a very small number of scenarios in $\mathcal{S}$ may drive the total cost in Eq. 5.18. In real-world applications, the designer of the network may lower the total investment cost by accepting some risk of not always satisfying the resiliency criteria. In these situations, we can relax Eqs. 5.12 and 5.13 to a set of chance constraints:

$$P\left(\begin{array}{ll} \displaystyle\sum_{i\in\mathcal{L},k\in\mathcal{P}_i} l^s_{i,k} \geq \lambda \sum_{i\in\mathcal{L},k\in\mathcal{P}_i} d_{i,k}(1-v_s) & \forall s\in\mathcal{S} \\ \displaystyle\sum_{i\in\mathcal{N},k\in\mathcal{P}_i} l^s_{i,k} \geq \gamma \sum_{i\in\mathcal{N},k\in\mathcal{P}_i} d_{i,k}(1-v_s) & \forall s\in\mathcal{S} \\ v_s \leq 0 & \forall s\in\mathcal{S} \end{array}\right) \geq 1-\epsilon \tag{5.29}$$

Where v_s, $s \in S$ is a Bernoulli random variable, representing the probability that scenario s won't happen, or in other words, that we relax constraints (5.12) and (5.13) (therefore will be disregarded). Assuming scenarios happen independently uniform at random, (5.29) is equivalent to stating that these constraints are violated in $\epsilon|S|$ of the scenarios. Thus, we can restate these constraints as:

$$
\begin{aligned}
&\sum_{i\in\mathcal{L},k\in\mathcal{P}_i} l^s_{i,k} \geq \lambda \sum_{i\in\mathcal{L},k\in\mathcal{P}_i} d_{i,k}(1-v_s) && \forall s\in\mathcal{S}\\
&\sum_{i\in\mathcal{N},k\in\mathcal{P}_i} l^s_{i,k} \geq \gamma \sum_{i\in\mathcal{N},k\in\mathcal{P}_i} d_{i,k}(1-v_s) && \forall s\in\mathcal{S}\\
&\sum_{s\in S} v_s \leq \epsilon|S| \\
&v_s \in \mathbb{B} && \forall s\in\mathcal{S}
\end{aligned}
\tag{5.30}
$$

It is important to note that this is a critical observation to solve this problem, as it is well-known that chance constraints using discrete random variables is a very intractable problem.

5.2 Algorithms

In this section we discuss the algorithms we developed for solving the ORDGDP. ORDGDP is a two-stage mixed integer programming (MIP) problem with a block diagonal structure that includes coupling variables between the blocks. We developed an exact algorithm that is vastly more efficient than a commercial state-of-the-art MIP solver. We then used the exact algorithm to develop a hybrid with variable neighborhood search that is competitive with the exact solver and is better than a heuristic used by the industry.

5.2.1 Scenario-Based Decomposition (SBD)

Decomposition is often used for solving two-stage stochastic MIPs [102], and it can be applied to ORDGDP after the following key observation:

Observation 5.2.1. *The second stage variables do not appear in the objective function. Therefore any optimal first stage solution based on a subset of the second stage subproblems that is feasible for the remaining scenarios, is an optimal solution for the original problem.*

Based on this observation, we can apply SBD to solve the ORDGDP. At high level, Algorithm 3 solves problems with iteratively larger sets of scenarios until a solution is obtained that is feasible for all scenarios. The algorithm takes as input the set of disasters (scenarios) and an initial scenario to consider, S'. Line 2 solves ORDGDP on S', where $P(S')$ and σ^* are used to denote the problem and solution respectively. Line 3 then evaluates σ^* on the remaining scenarios in $S \setminus S'$. The function $l : P'(s, \sigma^*) \rightarrow \mathbb{R}_+$, is an infeasibility measure that is 0 if the problem is feasible, positive otherwise. This is implemented by maximizing the reliability constraints, i.e. total and critical demand satisfied. It measures the gap between the delivered and the required demand (the right hand side of the Eqs. 5.12 and 5.13). This function prices the current solution over $s \in S \setminus S'$. If all prices are 0, then the algorithm terminates with solution σ^* (lines 4-5). Otherwise, the algorithm adds the scenario with the worst infeasabilty measure to S' (line 7).

We also tested other decomposition strategies such as Benders and Dantzig-Wolfe, however, their performance was tempered by the ORDGDP structure. The ORDGDP has MIP formulations at both stages of the problem and does not contain optimality conditions in the second stage (only feasability conditions). These approaches rarely out performed the commerical MIP solver.

Algorithm 3: Scenario Based Decomposition

input: A set of disasters S and let $S' = S_0$;

1 **while** $S \setminus S' \neq \emptyset$ **do**

2     $\sigma^* \leftarrow$ Solve $P(S')$;

3     $I \leftarrow \langle s_1, s_2 \ldots s_{|S \setminus S'|} \rangle \; s \in S \setminus S' : l(P'(s_i, \sigma^*)) \geq l(P'(s_{i+1}, \sigma^*))$;

4   **if** $l(P'(I(0), \sigma^*)) \leq 0$ **then**

5   **return** σ^*;

6   **else**

7         $S' \leftarrow S' \cup I(0)$;

8   **return** σ^*

5.2.2 Greedy Algorithm

A computationally efficient way of generating feasible solutions to the ORDGDP relaxes the coupling first stage variables and solves each scenario $s \in \mathcal{S}$ individually. The solutions are combined by taking the maximum of each construction variable ($\mathcal{X} = x \cup \tau \cup t \cup z \cup u$) over all scenarios (Algorithm 2). The switch construction cost is determined by switches that are needed to reduce the network into a tree for every scenario (line 4). Although the Greedy Algorithm is simple and fast, it rarely results in an optimal investment decision. However, it is representative of the types of heuristics used by the industry: see Reference [75] for a survey.

Algorithm 4: Greedy

input: A set of disasters S;

1 **for** $s \in S$ **do**

2 $\quad \sigma^s \leftarrow Solve(P'(s))$;

3 $\sigma^*(x) = \max\{\sigma^s(x) | \forall s \in S\},\ \forall x \in \mathcal{X}$;

4 Update $\sigma^*(x_i)$ with switches to preserve feasibility;

5 **return** σ^*

5.2.3 Variable Neighborhood Search

To overcome the limitations of greedy heuristics like Algorithm 2, we developed an approach based on Variable Neighborhood Decomposition (VNS) Search [66]. The algorithm fixes a subset of first stage variables to their current value and searching the remaining variables for a better solution. If all the first stage variables are fixed, the problem decomposes into $|S|$ separate problems that are easily solved and provide heuristic justification for focusing on first stage variables. More formally, $P(\sigma, J)$ denotes the problem with first stage variables, $J \in \mathcal{X}$,fixed to σ, i.e. $x_j = \sigma(x_j)$, and P^{LP} is the LP relaxation of problem P.

Algorithm 3 describes the VNS procedure. Line 1 computes the solution to the LP relaxation of the ORDGDP, (σ^{LP}). Line 4 counts the number of variable assignments that are different between the solution to LP relaxation (σ^{LP}) and the best known solution σ^* ($\sigma(x)$ denotes the variable assignment of x in solution σ). Line 5 orders the variables of $\mathcal{X}$ by the difference between their assignments in σ^* and σ^{LP}. Heuristically, those variables whose assignments are furthest from their LP assignment represent good opportunities to improve σ^*. The algorithm updates the rate at which the neighborhood size is increased (*step*) based on whether or not the algorithm is in a restart situation (lines 8 and 11). If the algorithm is in a restart, the ordering of the variables is also randomized (line 9). Line 13 computes the best solution in the neighborhood of σ where the first k elements of J are fixed. If the resulting solution is better, then the algorithm proceeds with a new σ^* (lines 15-18)–f is used as shorthand for Eq. 5.18. Otherwise, the size of the neighborhood is increased (lines 20-23). The iterations terminate when the maximum number of restarts is reached (line 2), the maximum number of neighborhood resizings is reached (line 12), or a time limit is reached. In this paper, $\textsc{maxRestarts} = 10$, $\textsc{maxIterations} = 4$, $\textsc{maxTime} = 48$ CPU hours, and $d = 2$.

Algorithm 5: Variable Neighborhood Search

input: σ', maxTime, maxRestarts and maxIterations;

Let $\sigma^{LP} \leftarrow Solve(P^{LP})$, $\sigma^* \leftarrow \sigma'$, $restart \leftarrow false$;

while $t <$ maxTime *and* $i <$ maxRestarts **do**

    $j \leftarrow 0$;

    $n \leftarrow |x \in \mathcal{X} : |\sigma^*(x) - \sigma^{LP}(x)| \neq 0|$;

    $J \leftarrow \langle \pi_1, \pi_2 \ldots \pi_{|J|} \rangle \in \mathcal{X} : |\sigma^*(\pi_i) - \sigma^{LP}(\pi_i)| \leq |\sigma^*(\pi_{i+1}) - \sigma^{LP}(\pi_{i+1})|$;

    if *restart* **then**

        $i \leftarrow i + 1$;

        $step \leftarrow \frac{4n}{d}$, $k = |\mathcal{X}| - step$;

        $shuffle(J)$

    else

        $step \leftarrow \frac{n}{d}$, $k = |\mathcal{X}| - step$;

    while $t <$ maxTime *and* $j \leq$ maxIterations **do**

        $\sigma' \leftarrow Solve(P(\sigma^*, J(1, \ldots, k))$;

        if $f(\sigma') < f(\sigma^*)$ **then**

            $\sigma^* \leftarrow \sigma'$;

            $i \leftarrow 0$;

            $restart \leftarrow false$;

            $j \leftarrow$ maxIterations;

        else

            $j \leftarrow j + 1$;

            $k = k - \frac{step}{2}$;

            if $j >$ maxIterations **then**

                $restart \leftarrow true$;

    return σ^*

In our experimentation, VNS outperformed other popular random walk heuristics, such as Simulated Annealing (SA). We conjecture that this is because the ORDGDP does not appear to have a concise neighborhood structure, which is generally a prerequisite for successful SA implementations. Here, we overcome this challenge by using a mixed-integer program as a neighborhood oracle within the local search step of VNS. A block diagram representation of Algorithm 5 is provided in Figure 5.5.

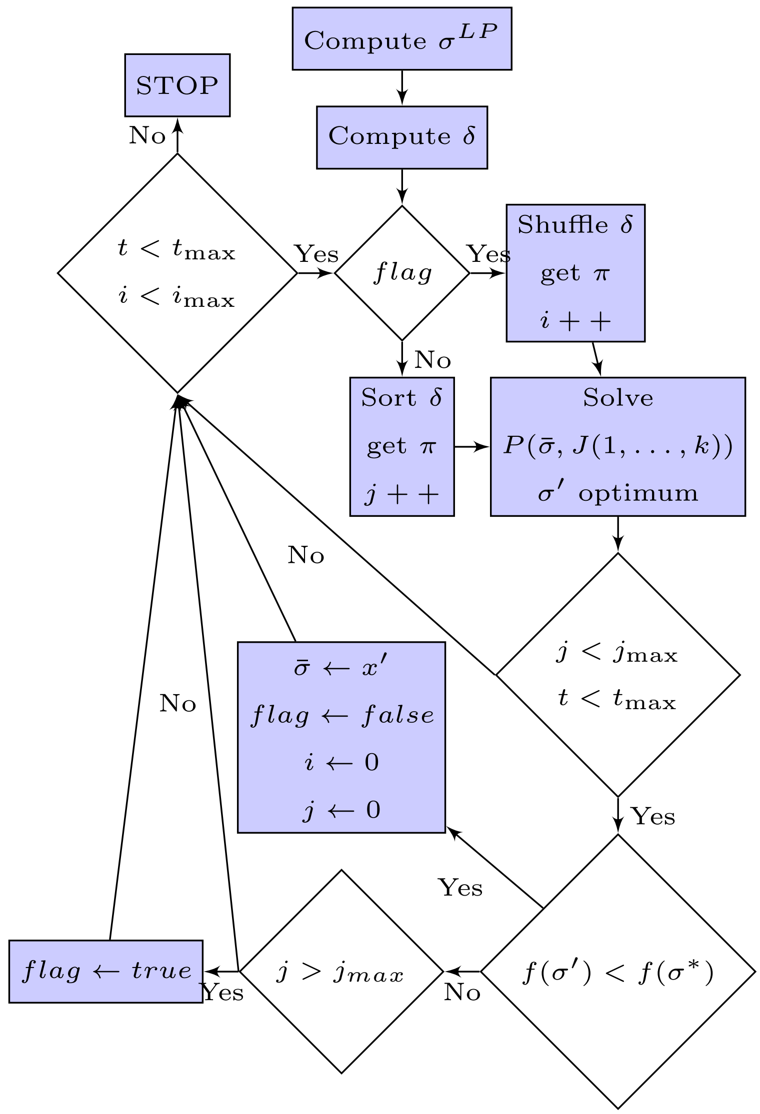


Figure 5.5: VNS block diagram

5.2.4 Scenario-based Variable Neighborhood Decomposition Search (SBVNDS)

Given that we have a powerful exact method in Algorithm 1 as well as a VNS in Algorithm 3, the natural algorithm hybridizes these approaches to get Algorithm, SBVNDS. The algorithm proceeds exactly the same as Algorithm 1, except that the exact solver for $Solve(P(S'))$ is replaced by VNS in line 2.

Algorithm 6: SBVNDS

input: A set of disasters S and let $S' = S_0$;

1 **while** $S \setminus S' \neq \emptyset$ **do**

2     $\sigma^* \leftarrow$ Solve $P(S')$ **using VNS**;

3     $I \leftarrow \left\langle s_1, s_2 \ldots s_{|S \setminus S'|} \right\rangle s \in S \setminus S' : l(P'(s_i, \sigma^*)) \geq l(P'(s_{i+1}, \sigma^*))$;

4   **if** $l(P'(I(0), \sigma^*)) \leq 0$ **then**

5   **return** σ^*;

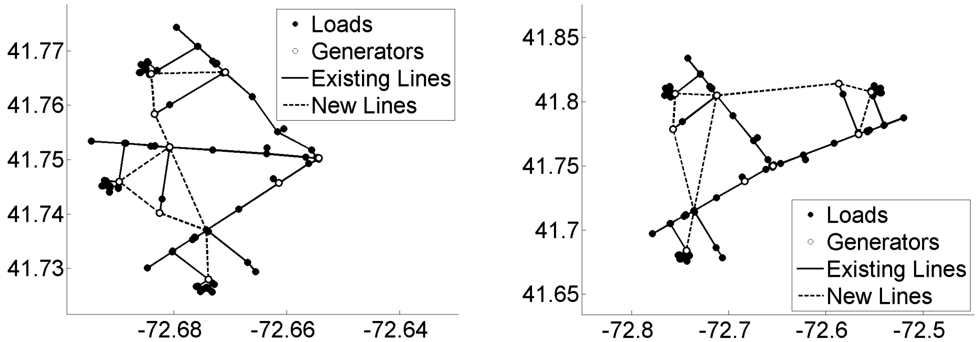
6   **else**

7         $S' \leftarrow S' \cup I(0)$;

8   **return** σ^*

5.3 Empirical Results

The algorithms were implemented using the CPLEX C++ API with Concert technology as a 32 threaded application on Intel XEON 2.29 GHz processors. Since these are planning problems, in principle, practitioners could utilize days of CPU time to produce a plan. However, in order to produce a wide range of results, we limited the algorithms to 48 hours of CPU time. Our problems are based on a modified version of the IEEE 34 bus systems [58] (see Fig. 5.6) that are representative of medium sized distribution systems.



(a) Urban (b) Rural

Figure 5.6: We generated two variations of the IEEE 34 bus problem. Each problem contains three copies of the IEEE 34 system to mimic situations where there are three normally independent distribution circuits that could support each other during extreme events. These problems include 100 scenarios, 109 nodes, 118 possible generators, 204 loads, and 148 edges, resulting in problems with $> 90k$ binary variables. The difference between rural (a) and urban (b) is the distances between nodes (expansion costs and line impedances). The cost of single and three phase underground lines is between \$$40k$ and \$$1500k$ per mile [47] and we adopt the cost of \$$100k$ per mile and \$$500k$ per mile, respectively. The cost of single and three phase switches is estimated to be \$$10k$ and \$$15k$, respectively [13]. Finally, the installed cost of natural gas-fired CHP in a microgrid is estimated to be \$$1500k$ per MW [34].

Urban, Hardened lines are not damageable (a)

	CPLEX		Greedy	SBD		SBVNDS	
	CPU	OBJ	OBJ	CPU	OBJ	CPU	OBJ
10%	19984.7	322.9	1044.5	465.8	322.9	289.9	353.7
25%	166352	635.4	1643.5	8028.3	635.4	811.4	635.4
50%	TO	X	2021.2	2840.7	647.7	791.3	647.7
75%	TO	X	1874.2	991.1	652.1	692.5	652.1
100%	TO	X	1934.4	712.7	654.1	662.5	654.1

Rural, Hardened lines are not damageable (b)

	CPLEX		Greedy	SBD		SBVNDS	
	CPU	OBJ	OBJ	CPU	OBJ	CPU	OBJ
10%	33083.5	2337.0	3274.8	1837.9	2337.0	503.3	2337.0
25%	32170.8	2390.3	3427.6	571.0	2390.3	457.8	2390.3
50%	20840.3	2397.6	3449.9	471.2	2397.6	421.2	2397.6
75%	15556.1	2400.4	3452.7	337.5	2400.4	299.8	2400.4
100%	17225.9	2400.6	2780.6	385.8	2400.6	346.9	2400.6

Urban, Hardened lines are damaged at a $\frac{1}{100}$ rate (c)

	CPLEX		Greedy	SBD		SBVNDS	
	CPU	OBJ	OBJ	CPU	OBJ	CPU	OBJ
10%	159166	445.8	1061.7	2232.9	445.8	2721.3	476.5
25%	TO	X	1441.9	14299.2	662.9	2994.7	701.5
50%	TO	X	1571.2	2848.7	646.0	1917.7	760.2
75%	TO	X	1787.3	16040.6	687.6	1481.4	687.6
100%	TO	X	2744.8	24270.3	1320.5	2157.5	1330.5

Rural, Hardened lines are damaged at a $\frac{1}{100}$ rate (d)

	CPLEX		Greedy	SBD		SBVNDS	
	CPU	OBJ	OBJ	CPU	OBJ	CPU	OBJ
10%	77947.9	2363.0	3375.4	759.0	2363.0	576.9	2363.0
25%	TO	X	8238.6	TO	X	919.4	6744.3
50%	TO	X	12336.0	TO	9288.9	4361.8	7121.0
75%	TO	X	23099.5	TO	X	23142.6	11500.0
100%	TO	X	16600.7	TO	X	5879.5	9797.3

Urban, Hardened lines are damaged at a $\frac{1}{10}$ rate (e)

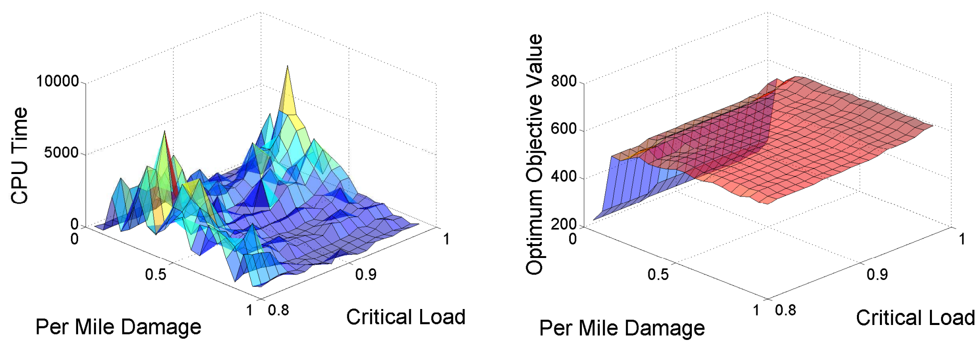
	CPLEX		Greedy	SBD		SBVNDS	
	CPU	OBJ	OBJ	CPU	OBJ	CPU	OBJ
10%	TO	X	859.1	5265.1	460.8	2505.7	594.1
25%	TO	X	1742.2	12530.3	961.2	2843.2	961.2
50%	TO	X	3133.8	34822.7	1417.2	3363.5	1555.2
75%	TO	X	3472.0	TO	X	7486.5	1894.2
100%	TO	X	10479.1	TO	X	32289.8	7959.4

Rural, Hardened lines are damaged at a $\frac{1}{10}$ rate (f)

	CPLEX		Greedy	SBD		SBVNDS	
	CPU	OBJ	OBJ	CPU	OBJ	CPU	OBJ
10%	TO	X	7503.3	141718.0	4325.9	7756.8	4424.8
25%	TO	X	18021.3	TO	X	21993.5	7371.9
50%	TO	X	28865.0	TO	12017.7	74729.0	12031.2
75%	TO	X	31887.0	TO	13522.2	107165.0	13500.8
100%	TO	X	32901.9	TO	16794.4	114354.0	16778.2

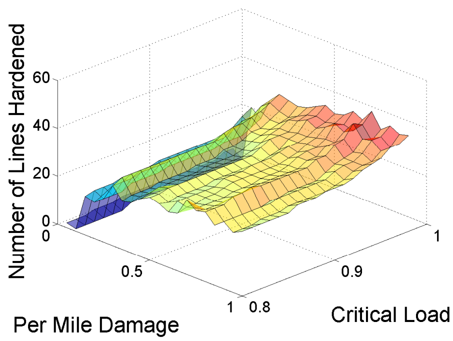
Table 5.1: These tables compare the performance of the algorithms when hardened lines cannot be damaged (a, b), are damaged at $\frac{1}{100}$ the rate of unhardened lines (c, d), and damaged at $\frac{1}{10}$ the rate of unhardened lines (e, f). The columns denoted by CPU and OBJ refer to CPU time and objective value, respectively. We omit the CPU time of Greedy as it is always less than 60 CPU seconds. The rows refer to the probability a 1 mile segment of a line is damaged.

Scenarios for this paper are based on damage caused by ice storms, whose intensity tends to be homogeneous on the scale of distribution systems [91]. Intensities are modeled as damage rates per mile on power poles and are transformed into the probability a power line segment of one mile length is damaged (a pole has failed). Empirically, we find that 100 randomly created scenarios is sufficient to capture the salient features of the distribution. Each scenario contains two sets of line failures, one for hardened lines ($\mathcal{D}'_s$) and a second for lines that are not hardened ($\mathcal{D}_s$).

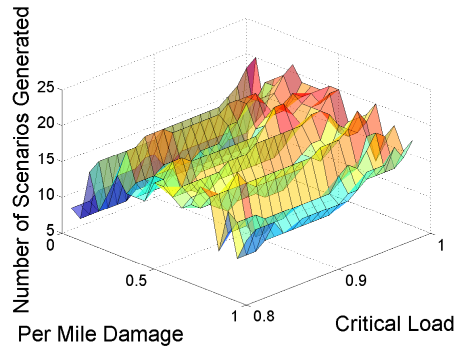


(a) CPU time (b) Objective value

Figure 5.7: Sensitivity of the CPU time and objective value to changes in λ on the Urban problem for SBD when hardened lines are not damageable.



(a) # of Hardened

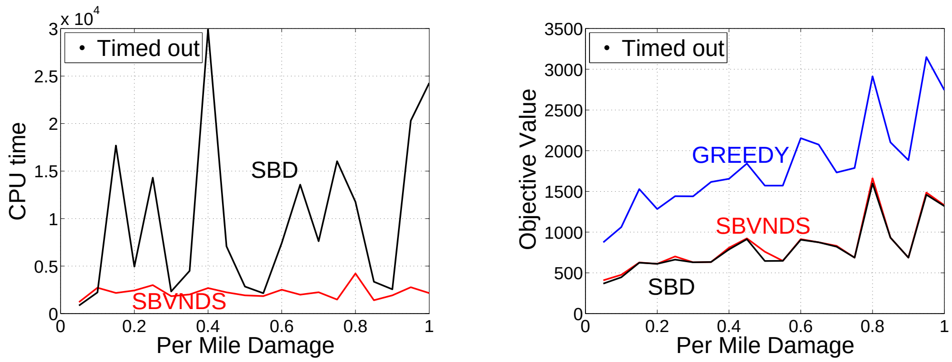


(b) # of Scenarios

Figure 5.8: Sensitivity of the number of lines hardened and the number of scenarios generated to changes in λ on the Urban problem for SBD when hardened lines are not damageable. Due to short distances, the solution favors hardening many lines. The required hardening is relatively insensitive to the amount of damage and λ. However, there are spikes in problem difficulty at transitions in λ that require additional load service.

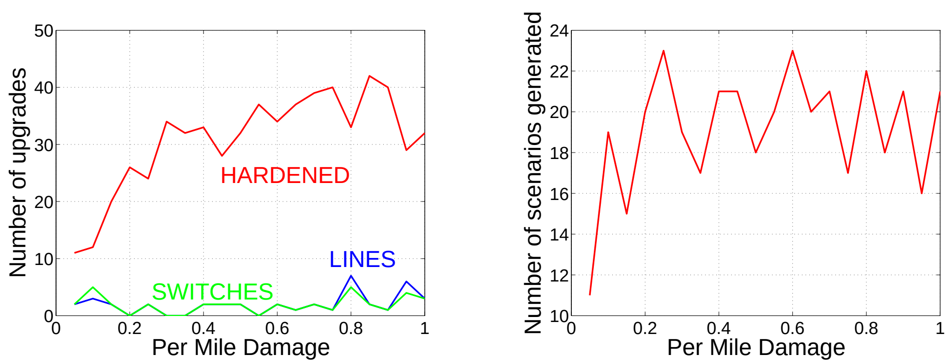
Table 5.1 provides results when hardened lines are not damaged or are damaged at rates of $\frac{1}{100}$ or $\frac{1}{10}$ of the unhardened rate. There are a number of important observations in these tables. First, CPLEX by itself is computationally uncompetitive. Only when the hardened lines are not damaged does CPLEX complete within the time limit. These problems are "easier" because hardened lines are robust and relatively inexpensive, enabling CPLEX to eliminate many solutions. The objective value for Greedy is always worse than optimal. The exact method SBD is much more computationally efficient than CPLEX and is able to solve many more problems to optimality indicating that CPLEX is unable to recognize the scenario structure in the problems. However, SBD is sensitive to which scenarios are included (function l), and if poor choices are made, it begins to resemble CPLEX. However, the meta-heuristic SBVNDS is able to overcome these limitations. It is much faster than SBD, and almost always achieves the optimal solution. This indicates that heuristic methods based on combining powerful techniques like VNS with strong exact algorithms are very good on this type of 2-stage mixed integer programming problems. Figures 5.9 through 5.16 provides more statistics that we will further analyze.

First let's take a closer look the the instances where a hardened asset has a failure probability of one hundredth original failure probability of a line. Therefore hardened assets are fairly likely to survive a disruption, however there is still non-negligible probability that they may fail.



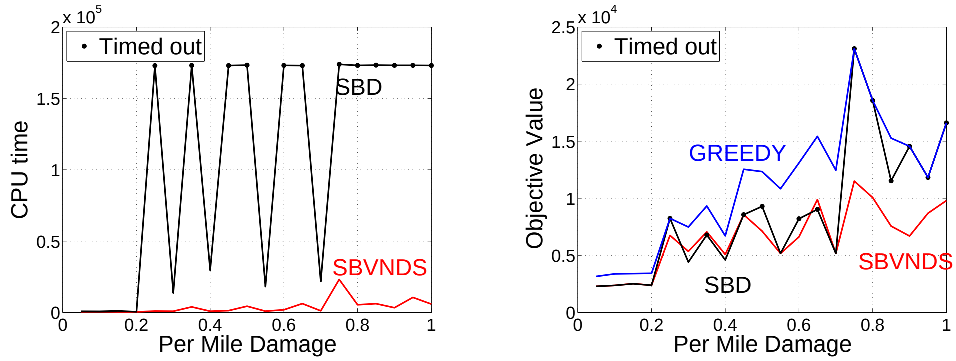
(a) CPU time (b) Objective value

Figure 5.9: $\frac{1}{100}$ Urban Comparison



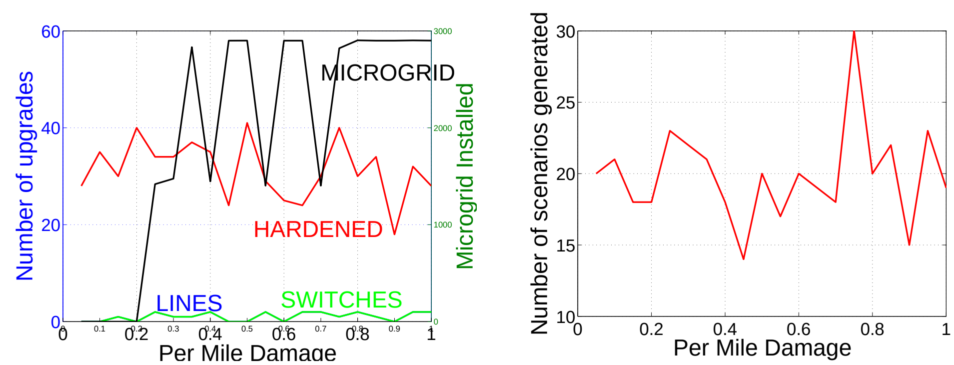
(a) # of Upgrades (b) # of Scenarios

Figure 5.10: $\frac{1}{100}$ Urban SBVNDS solution



(a) CPU time (b) Objective value

Figure 5.11: $\frac{1}{100}$ Rural Comparisons

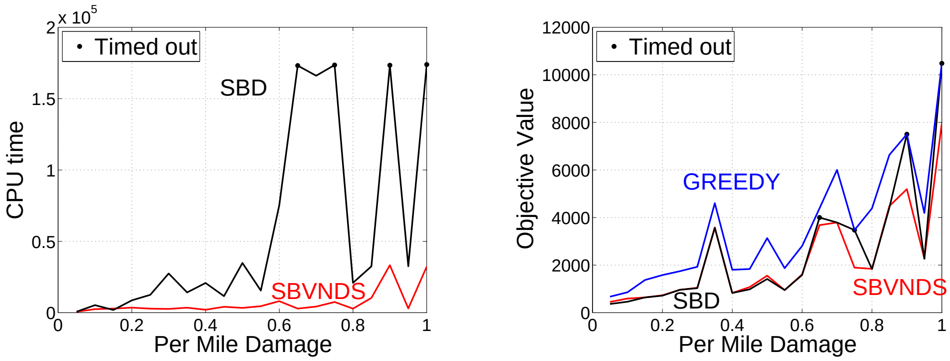


(a) # of Upgrades (b) # of Scenarios

Figure 5.12: $\frac{1}{100}$ Rural SBVNDS solution

We obserb that because hardened assets are likely to survive and hardening a line is cheaper in comparison to introducing microgrid generation, we prefer hardening major components of the network to keep the radial operations.

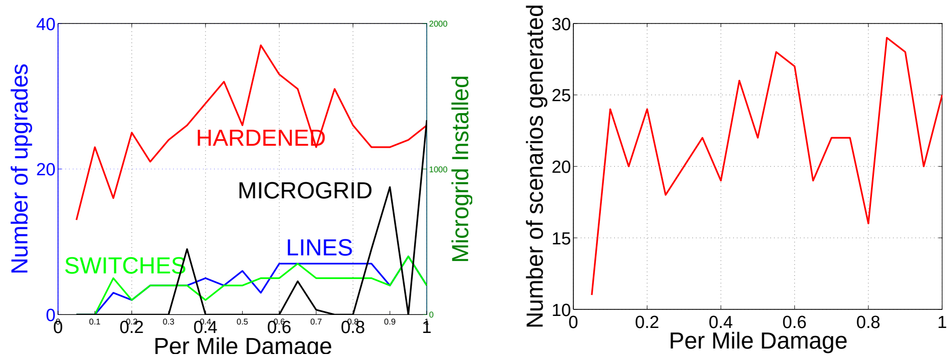
Next we take a look to the case of a hardened asset to be damaged with one tenth the original probability of line being damaged, therefore there is significant probability a line being damaged even though it might be hardened. These are the toughest instances of our data set and even SBVNDS runs into CPU time limit among one of these instances. However SBVNDS still provides a quality solution within reasonable CPU time compared to SBD.



(a) CPU time

(b) Objective value

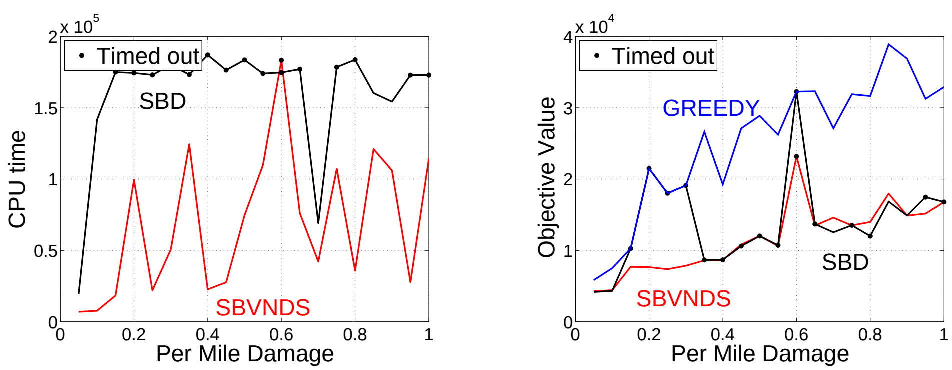
Figure 5.13: $\frac{1}{10}$ Urban Comparison



(a) # of Upgrades

(b) # of Scenarios

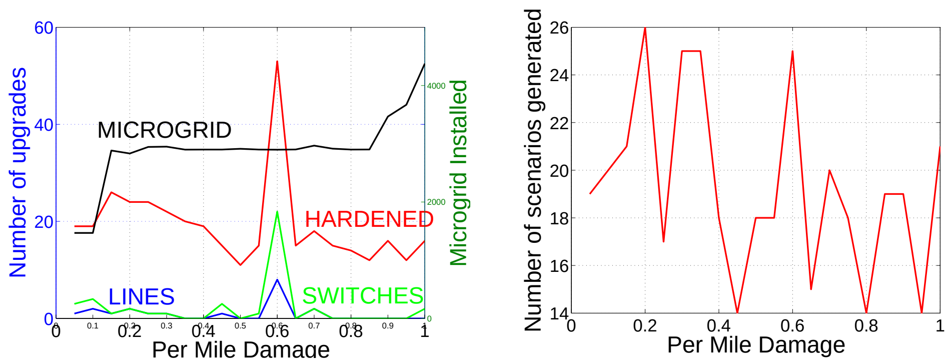
Figure 5.14: $\frac{1}{10}$ Urban SBVNDS solution



(a) CPU time

(b) Objective value

Figure 5.15: $\frac{1}{10}$ Rural Comparisons



(a) # of Upgrades

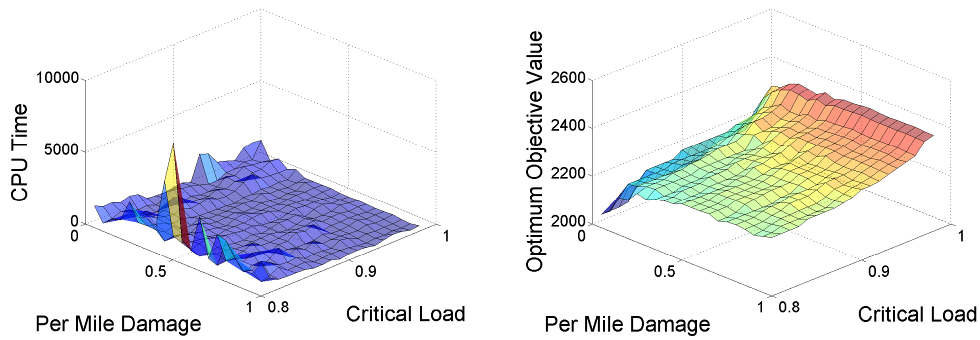
(b) # of Scenarios

Figure 5.16: $\frac{1}{10}$ Rural SBVNDS solution

We observe that the stronger the extreme event gets the more microgrid generation is added to the graph in exchange for the hardening assets. This is understandable considering as the probability that the hardened lines are damaged is increasing, the probability that there will be islanded costumers. If any of these customers have a critical demand, it is likely to be covered by a microgrid generation facility, as they are separated from the substation.

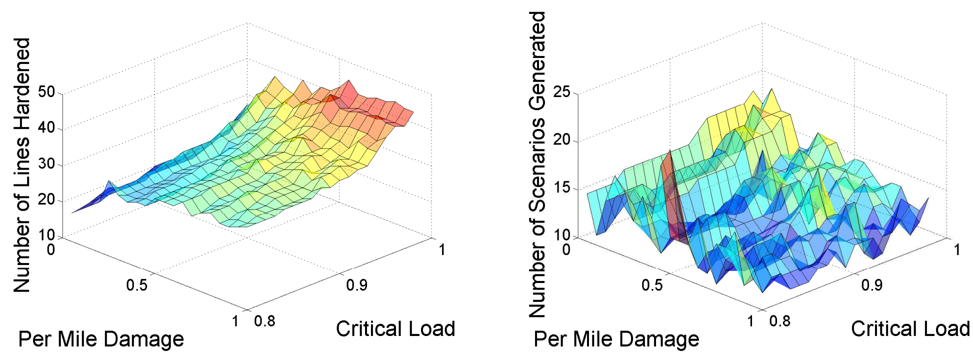
5.3.1 Critical load constraint

Figures 5.17 and 5.18 show some results for rural and urban problems when the required fraction of critical load served is varied. In general, peaks in CPU time correspond to discrete jumps in the amount of load served as λ increases.



(a) CPU time (b) Objective value

Figure 5.17: Sensitivity of the CPU time and objective value to changes in λ for SBD on the Rural problem when hardened lines are not damageable. Because of long distances, the solution favors adding generation and is sensitive to the amount of damage and λ.

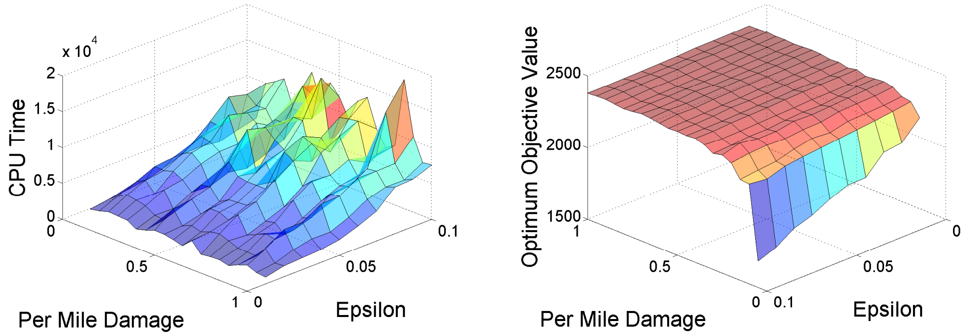


(a) # of Hardened (b) # of Scenarios

Figure 5.18: Sensitivity of the number of hardened lines and the number of scenarios to changes in λ for SBD on the Rural problem when hardened lines are not damageable.

5.3.2 Chance constraints

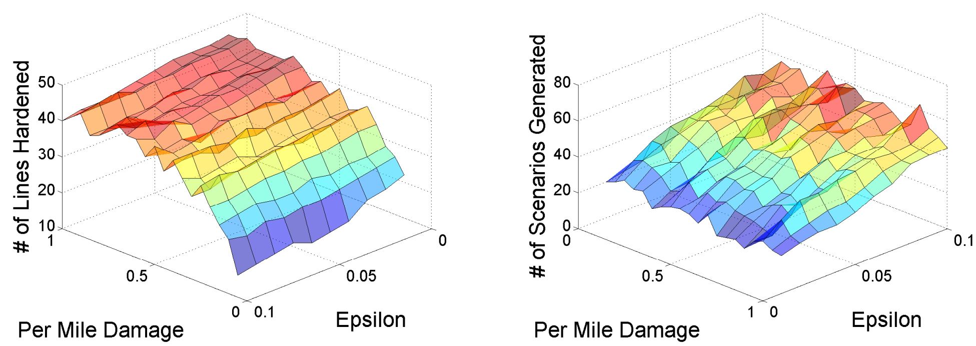
Fig. 5.19 and Fig. 5.20 shows results when the resiliency criteria are relaxed to the chance constraints in Eq. 5.30 and ϵ is varied. Interestingly, CPU time is not impacted too greatly by damage rates. Also, the solution is relatively insensitive to the choice of ϵ as damage rates increase, indicating that an "easier" problem with small ϵ could be used to approximate a solution to the harder problems.



(a) CPU time

(b) Objective value

Figure 5.19: These figures show how the CPU time and solution quality changes when chance constraints (ϵ) is modified for the Rural network, when hardened lines are not damageable. These plots are generated by SBD.



(a) # of Hardened

(b) # of Scenarios

Figure 5.20: These figures show how the number of hardened lines and the number of scenarios generated changes when chance constraints (ϵ) is modified for the Rural network, when hardened lines are not damageable. These plots are generated by SBD.

5.4 Conclusions

We formulated, proposed and tested new algorithms to solve the ORDGDP. Our primary contribution is an algorithm that combines the benefits of an exact method based on scenario decomposition with variable neighborhood search. This algorithm is shown to scale well to problems that are difficult for exact methods, without sacrificing solution quality. Future directions include:

1. Using a more accurate model of the 3-phase AC power flow equations to better exclude infeasible solutions, such as the DistFlow approximation of Baran and Wu that is discussed in Section 5.1.6.

2. Scaling to entire city-sized distribution networks. We considered a feeder system connected to a single substation in this paper. However, distribution grids in a city can span multiple substations. In general, we expect city-sized networks can be partitioned into subproblems to reduce complexity and is a topic of future work.

3. Including a variation of the restoration problem posed by [23].

4. Using various valid inequalities for ORDGDP, including the multi-row simplex inequalities of Chapter 4.

References

[1] Jr. A. K. Lenstra, H. W. Lenstra and L. Lovász. Factoring polynomials with rational coefficients. *Mathematische Annalen*, 261:513–534, 1982.

[2] Tobias Achterberg, Thorsten Koch, and Alexander Martin. Branching rules revisited. *Operations Research Letters*, 33(1):42 – 54, 2005.

[3] Tobias Achterberg, Thorsten Koch, and Alexander Martin. Miplib 2003. Technical Report 05-28, ZIB, Takustr.7, 14195 Berlin, 2005.

[4] Kent Andersen, Quentin Louveaux, Robert Weismantel, and Laurence A. Wolsey. Inequalities from two rows of a simplex tableau. In *IPCO*, pages 1–15, 2007.

[5] David Applegate, William J. Cook, Sanjeeb Dash, and André Rohe. Solution of a min-max vehicle routing problem. *INFORMS Journal on Computing*, 14(2):132–143, 2002.

[6] Egon Balas. Intersection cuts - a new type of cutting planes for integer programming. *Operations Research*, 19:19–39, 1971.

[7] Egon Balas, Sebastián Ceria, and Gérard Cornuéjols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Math. Program.*, 58:295–324, 1993.

[8] M.E. Baran and F.F. Wu. Optimal capacitor placement on radial distribution systems. *IEEE Transactions on Power Delivery*, 4:725–734, 1989.

[9] Amitabh Basu. *Corner Polyhedra and Maximal Lattice-free Convex Sets: A Geometric Approach to Cutting Planes*. PhD thesis, Carnegie Mellon University, May 2010.

[10] Amitabh Basu, Robert Hildebrand, and Matthias Köppe. Algorithmic and complexity results for cutting planes derived from maximal lattice-free convex sets. *CoRR*, abs/1107.5068, 2011.

[11] D.E. Bell. A theorem concerning the integer lattice. *Studies in Applied Mathematics*, 56:187–188, 1977.

[12] Russell Bent, Alan Berscheid, and G. Loren Toole. Transmission network expansion planning with simulation optimization. In *AAAI*. AAAI, 2010.

[13] Tom Bialek, December 2013. Private Communication with Tom Bialek of San Diego Gas & Electric.

[14] R. E. Bixby, S. Ceria, C. M. McZeal, and M. W. P Savelsbergh. An updated mixed integer programming library: MIPLIB 3.0. *Optima*, 58:12–15, 1998.

[15] Robert E. Bixby, Mary Fenelon, Zonghao Gu, Edward Rothberg, and Roland Wunderling. Mip: Theory and practice - closing the gap. In M. J. D. Powell and S. Scholtes, editors, *System Modelling and Optimization*, volume 174 of *IFIP Conference Proceedings*, pages 19–50. Kluwer, 1999.

[16] Valentin Borozan and Gérard Cornuéjols. Minimal valid inequalities for integer constraints. *Math. Oper. Res.*, 34(3):538–546, 2009.

[17] Alberto Caprara and Adam N. Letchford. On the separation of split cuts and related inequalities. *Math. Program.*, 94(2-3):279–294, 2003.

[18] Central Hudson Gas and Electric. `http://www.centralhudson.com/valueforourvalley/`, 2014. Accessed: Sept. 2014.

[19] Vijay Chandru and John N. Hooker. *Optimization Methods for Logical Inference*. Wiley-Interscience, New York, NY, USA, 1999.

[20] Richard Li-Yang Chen, Amy Cohn, Neng Fan, and Ali Pinar. Contingency-Risk Informed Power System Design. *IEEE Transactions on Power Systems*, 29(5):2087–2096, September 2014.

[21] Richard Li-Yang Chen and Cynthia A. Phillips. k-edge failure resilient network design. *Electronic Notes in Discrete Mathematics*, 41(0):375 – 382, 2013.

[22] Vasek Chvátal. Edmonds polytopes and a hierarchy of combinatorial problems. *Discrete Mathematics*, 4:305–337, 1973.

[23] Carleton Coffrin, Pascal Van Hentenryck, and Russell Bent. Last-mile restoration for multiple interdependent infrastructures. In *AAAI*, 2012.

[24] Michele Conforti, Gérard Cornuéjols, and Giacomo Zambelli. Equivalence between intersection cuts and the corner polyhedron. *Oper. Res. Lett.*, 38(3):153–155, 2010.

[25] William J. Cook, Ravi Kannan, and Alexander Schrijver. Chvátal closures for mixed integer programming problems. *Math. Program.*, 47:155–174, 1990.

[26] Gérard Cornuéjols. Revival of the gomory cuts in the 1990's. *Annals OR*, 149(1):63–66, 2007.

[27] Gérard Cornuéjols. Valid inequalities for mixed integer linear programs. *Math. Program.*, 112(1):3–44, 2008.

[28] Sanjeeb Dash, Oktay Günlük, and Juan P. Vielma. Computational experiments with cross and crooked cross cuts. *INFORMS Journal on Computing*, accepted 2013.

[29] Sanjeeb Dash, Oktay Günlük, and Juan Pablo Vielma. Computational experiments with cross and crooked cross cuts. *INFORMS Journal on Computing*, 26(4):780–797, 2014.

[30] A Delgadillo, J Arroyo, and N Alguacil. Analysis of Electric Grid Interdiction with Line Switching. *IEEE Transactions on Power Systems*, 25(2):633–641, 2010.

[31] D. Dentcheva, A. Prékopa, and A. Ruszczyński. Concavity and efficient points of discrete distributions in probabilistic programming. *Mathematical Programming Series A*, 89:55–77, 2000.

[32] D. Dentcheva, A. Prékopa, and A. Ruszczyński. On convex probabilistic programming with discrete distributions. *Nonlinear Analysis*, 47:1997–2009, 2001.

[33] Santanu S. Dey, Andrea Lodi, Andrea Tramontani, and Laurence A. Wolsey. Experiments with two row tableau cuts. In Friedrich Eisenbrand and F. Bruce Shepherd, editors, *IPCO*, volume 6080 of *Lecture Notes in Computer Science*, pages 424–437. Springer, 2010.

[34] EIA, 2010. http://www.meede.org/wp-content/uploads/Commercial-and-Industrial-CHP-Technology-Cost-and-Performance-Data-Analysis-for-EIA_June-2010.pdf.

[35] Daniel G. Espinoza. Computing with multi-row gomory cuts. In Andrea Lodi, Alessandro Panconesi, and Giovanni Rinaldi, editors, *IPCO*, volume 5035 of *Lecture Notes in Computer Science*, pages 214–224. Springer, 2008.

[36] Executive Office of the President. Economic benefits of increasing electric grid resilience to weather outages. Technical report, Executive Office of the President, August 2013.

[37] Ricardo Fukasawa and Oktay Günlük. Strengthening lattice-free cuts using non-negativity. *Discrete Optimization*, 8(2):229–245, 2011.

[38] P.A.N. Garcia, J.L.R. Pereira, S. Carneiro, V.M. da Costa, and N. Martins. Three-phase power flow calculations using the current injection method. *IEEE Transactions on Power Systems*, 15(2):508–514, May 2000.

[39] Manish Garg and J. Cole Smith. Models and algorithms for the design of survivable multicommodity flow networks with general failure scenarios. *Omega*, 36(6):1057–1071, December 2008.

[40] Vikas K. Garg, T. S. Jayram, and Balakrishnan Narayanaswamy. Online optimization with dynamic temporal uncertainty: Incorporating short term predictions for renewable integration in intelligent energy systems. In *AAAI*, 2013.

[41] P.C. Gilmore and R.E. Gomory. The theory and computation of knapsack functions. *Operations Research*, 14:1045–1074, 1966.

[42] Mehdi Golari, Neng Fan, and Jianhui Wang. Two-stage stochastic optimal islanding operations under severe multiple contingencies in power grids. *Electric Power Systems Research*, 114(0):68 – 77, 2014.

[43] Ralph E. Gomory. Outline of an algorithm for integer solutions to linear program. *Bulletin of the American Mathematical Society*, 64(5):275–278, September 1958.

[44] Ralph E. Gomory. Solving linear programming problems in integers. In R. E. Bellman and Jr. M. Hall, editors, *Combinatorial Analysis*, pages 211–216. American Mathematical Society, 1960.

[45] Ralph E. Gomory. Some polyhedra related to combinatorial problems. *Linear Algebra and Its Applications*, 2:451–558, 1969.

[46] R.E. Gomory. Faces of an integer polyhedrom. *Proceedings of the National Academy of Sciences*, 57:16–18, 1967.

[47] Governor and General Assembly of Virginia. Placement of utility distribution lines underground. Technical report, 2005.

[48] Martin Grötschel, László Lovász, and Alexander Schrijver. Corrigendum to our paper "the ellipsoid method and its consequences in combinatorial optimization". *Combinatorica*, 4(4):291–295, 1984.

[49] Pascal Van Hentenryck, Nabeel Gillani, and Carleton Coffrin. Joint assessment and restoration of power systems. In *ECAI*, pages 792–797, 2012.

[50] H. Everett III. Generalized Lagrange Multiplier Method for Solving Problems of Optimum Allocation of Resources. *Operations Research*, 11:399–417, 1963.

[51] R. A. Jabr. Robust Transmission Network Expansion Planning With Uncertain Renewable Generation and Loads. *IEEE Transactions on Power Systems*, 28(4):4558–4567, November 2013.

[52] Shweta Jain, Balakrishnan Narayanaswamy, and Y. Narahari. A multiarmed bandit incentive mechanism for crowdsourcing demand response in smart grids. In *AAAI*, pages 721–727, 2014.

[53] Robert G. Jeroslow. Cutting-plane theory: Algebraic methods. *Discrete Mathematics*, 23:121–150, 1978.

[54] Robert G. Jeroslow. Minimal inequalities. *Mathematical Programming*, 17:1–15, 1979.

[55] D. S. Johnson, J. K. Lenstra, and A. H. G. Rinnooy Kan. The complexity of the network design problem. *Networks*, 8(4):279–285, January 1978.

[56] Ellis Johnson. Cyclic groups, cutting planes and shortest paths. In S. Robinson T. Hu, editor, *Mathematical Programming*, pages 185–211. Academic Press, 1973.

[57] Richard M. Karp. Reducibility among combinatorial problems. In *Proceedings of a symposium on the Complexity of Computer Computations, held March 20-22, 1972, at the IBM Thomas J. Watson Research Center, Yorktown Heights, New York.*, pages 85–103, 1972.

[58] W.H. Kersting. Radial distribution test feeders. *IEEE Transactions on Power Systems*, 6(3):975–985, 1991.

[59] Sarika Khushalani, Jignesh M. Solanki, and Noel N. Schulz. Optimized restoration of unbalanced distribution systems. *IEEE Transactions on Power Systems*, 22(2):624–630, May 2007.

[60] Diego Klabjan. A new subadditive approach to integer programming. In *Integer Programming and Combinatorial Optimization, 9th International IPCO Conference, Cambridge, MA, USA, May 27-29, 2002, Proceedings*, pages 384–499, 2002.

[61] H. W. Kuhn and A. W. Tucker. Nonlinear programming. In Jerzy Neyman, editor, *Proceedings of the 2nd Berkeley Symposium on Mathematical Statistics and Probability*, pages 481–492. University of California Press, Berkeley, CA, USA, 1950.

[62] A. H. Land and A. G. Doig. An automatic method of solving discrete programming problems. *Econometrica*, 28(3):pp. 497–520, 1960.

[63] Jean B. Lasserre. A discrete farkas lemma. *Discrete Optimization*, 1(1):67–75, 2004.

[64] Jean B. Lasserre. Generating functions and duality for integer programs. *Discrete Optimization*, 1(2):167–187, 2004.

[65] Jean B. Lasserre. The integer hull of a convex rational polytope. *Discrete & Computational Geometry*, 32(1):129–139, 2004.

[66] Jasmina Lazic, Saïd Hanafi, Nenad Mladenovic, and Dragan Urosevic. Variable neighbourhood decomposition search for 0-1 mixed integer programs. *Comp. & OR*, 37(6):1055–1067, 2010.

[67] Jr. Lenstra, H. W. Integer programming with a fixed number of variables. *Mathematics of Operations Research*, 8(4):pp. 538–548, 1983.

[68] Juan Li, Xi-Yuan Ma, Chen-Ching Liu, and Kevin P. Schneider. Distribution System Restoration With Microgrids Using Spanning Tree Search. *IEEE Transactions on Power Systems*, PP(99):1–9, 2014.

[69] Yanjun Li and Jean-Philippe P. Richard. Cook, kannan and schrijver's example revisited. *Discrete Optimization*, 5(4):724–734, 2008.

[70] Quentin Louveaux and Laurent Poirrier. An algorithm for the separation of two-row cuts. *Math. Program.*, 143(1-2):111–146, 2014.

[71] L. Lovasz and A. Schrijver. Cones of matrices and set-functions and 0-1 optimization. *SIAM JOURNAL ON OPTIMIZATION*, 1:166–190, 1991.

[72] László Lovász. Geometry of numbers and integer programming. In *M. Iri and K. Tanabe, editors, Mathematical Programming: Recent Developments and Applications*, pages 177–201. Kluwer Academic Publishers, 1989.

[73] Steven Low. Convex relaxation of optimal power flow. *IEEE Transactions on Control of Network Systems*, 1:15–27, 2014.

[74] Matthew Mansfield and William Linzey. Hurricane sandy multi-state outage & restoration report. Technical Report 9308, National Association of State Eenergy Officials, February 2013.

[75] Francisco D. Munoz, Benjamin F. Hobbs, Jonathan L. Ho, and Saamrat Kasina. An Engineering-Economic Approach to Transmission Planning Under Market and Regulatory Uncertainties: WECC Case Study. *IEEE Transactions on Power Systems*, 29(1):307–317, January 2014.

[76] Dritan Nace, Michal Piro, Artur Tomaszewski, and Mateusz Zotkiewicz. Complexity of a classical flow restoration problem. *Networks*, 62(2):149–160, 2013.

[77] G. L. Nemhauser and L. A. Wolsey. *Integer and combinatorial optimization*. Wiley-Interscience, New York, NY, USA, 1988.

[78] Manfred Padberg and Giovanni Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM Rev.*, 33(1):60–100, February 1991.

[79] A. Prékopa. On probabilistic constrained programming. In *Proceedings of the Princeton Symposium on Mathematical Programming*, pages 113–138. Princeton University Press, 1970.

[80] A. Prékopa. Contributions to the theory of stochastic programming. *Mathematical Programming*, 4:202–221, 1973.

[81] A. Prékopa. Dual method for a one-stage stochastic programming problem with random rhs obeying a discrete probability distribution. *Zeitschrift f ur Operations Research*, 34:441–461, 1990.

[82] A. Prékopa. *Stochastic programming*. Kluwer Academic Publishers, Dordrecht, Boston, 1995.

[83] A. Prékopa. The use of discrete moment bounds in probabilistic constrained stochastic programming models. *Annals of Operations Research*, 85:21–38, 1999.

[84] A. Prékopa, B. Vizvári, and T. Badics. Programming under probabilistic constraint with discrete random variable. *New Trends in Mathematical Programming*, pages 235–255, 1998.

[85] E. Rothberg R. E. Bixby, Z. Gu and R. Wunderling. Mixed-integer programming : A progress report. In M. Gr otschel, editor, *The Sharpest Cut: The impact of Manfred Padberg and his work*, page 309326. MPS-SIAM Series in Optimization, 2004.

[86] Günther R. Raidl, Thomas Baumhauer, and Bin Hu. Speeding up logic-based benders' decomposition by a metaheuristic for a bi-level capacitated vehicle routing problem. In *HM 2014*, pages 183–197, 2014.

[87] Prashant P. Reddy and Manuela M. Veloso. Strategy learning for autonomous agents in smart grid markets. In *IJCAI*, pages 1446–1451, 2011.

[88] Prashant P. Reddy and Manuela M. Veloso. Factored models for multiscale decision-making in smart grid customers. In *AAAI*, 2012.

[89] Prashant P. Reddy and Manuela M. Veloso. Negotiated learning for smart grid agents: Entity selection based on dynamic partially observable features. In *AAAI*, 2013.

[90] A. Ruszczyński. Probabilistic programming with discrete distributions and precedence constrained knapsack polyhedra. *Mathematical Programming Series A*, 93:195–215, 2002.

[91] Yingshi Sa. *Reliability Analysis of Electric Distribution Lines.* PhD thesis, McGill University, Montreal, Canada, 2002.

[92] J Salmeron, K. Wood, and R Baldick. Worst-case interdiction analysis of large-scale electric power grids. *IEEE Transactions on Power Systems*, 24(1):96–104, 2009.

[93] Tjendera Santoso, Shabbir Ahmed, Marc Goetschalckx, and Alexander Shapiro. A stochastic programming approach for supply chain network design under uncertainty. Stochastic Programming E-Print Series. Institut fr Mathematik, 2003.

[94] H.E. Scarf. An observation on the structure of production sets with indivisibilities. *Proceedings of the National Academy of Sciences of the United States of America*, 74:3637–3641, 1977.

[95] A. Schrijver. *Theory of Linear and Integer programming.* Wiley-Interscience, 1986.

[96] Alexander Schrijver. On cutting planes. *Annals of Discrete Mathematics*, 9:291–296, 1980.

[97] Mike Shann and Sven Seuken. An active learning approach to home heating in the smart grid. In *IJCAI*, 2013.

[98] Hanif D. Sherali and Warren P. Adams. A hierarchy of relaxations between the continuous and convex hull representations for zero-one programming problems. *SIAM J. Discrete Math.*, 3(3):411–430, 1990.

[99] Sylvie Thibaux, Carleton Coffrin, Hassan Hijazi, and John K. Slaney. Planning with mip for supply restoration in power distribution systems. In *IJCAI*, 2013.

[100] Artur Tomaszewski, Michał Pióro, and Mateusz Żotkiewicz. On the complexity of resilient network design. *Networks*, 55(2):108–118, March 2010.

[101] US Department of Energy. U.S. Energy sector vulnerabilities to climate change and extreme weather. Technical Report DOE/PI-0013, US Department of Energy, July 2013.

[102] François Vanderbeck and Laurence A. Wolsey. Reformulation and decomposition of integer programs. In *50 Years of Integer Programming*, pages 431–502. 2010.

[103] B. Vizvári. The integer programming background of a stochastic integer programming algorithm of dentcheva, prékopa, ruszczyński. *Optimization Methods and Software*, 7:543–559, 2002.

[104] Lawrence Wolsey. Integer programming duality: price functions and sensitivity analysis. *Mathematical Programming*, 20:173–195, 1981.