PERSPECTIVE REFORMULATION AND APPLICATIONS

OKTAY GÜNLÜK* AND JEFF LINDEROTH†

Abstract.
In this paper we survey recent work on the perspective reformulation approach that generates tight, tractable relaxations for convex mixed integer *nonlinear* programs (MINLP)s. This preprocessing technique is applicable to cases where the MINLP contains binary indicator variables that force continuous decision variables to take the value 0, or to belong to a convex set. We derive from first principles the perspective reformulation, and we discuss a variety of practical MINLPs whose relaxation can be strengthened via the perspective reformulation. The survey concludes with comments and computations comparing various algorithmic techniques for solving perspective reformulations.

Key words. Mixed-integer nonlinear programming, perspective functions

AMS(MOS) subject classifications. 90C11, 90C30

1. Introduction.

Over the past two decades, tremendous advances have been made in the ability to solve mixed integer linear programs (MILP)s. A fundamental reason for the vast improvement is the ability to build tight, tractable relaxations of MILPs. The relaxations are built either via problem reformulation (automatically during a preprocessing phase), or dynamically through the addition of cutting planes. In this paper we survey a collection of techniques for obtaining tight relaxations to (convex) Mixed Integer Nonlinear Programs (MINLP)s. We call these preprocessing techniques the *perspective reformulation*, since they rely on replacing the original convex function in the formulation with its so-called *perspective*.

1.1. Motivation.

Consider a 0-1 MINLP, of the form

$$\min_{(x,z)\in\mathcal{F}} c(x,z) \tag{1.1}$$

where $\mathcal{F} = R \cap (\mathbb{R}^{n-p}_+ \times \mathbb{B}^p)$, $\mathbb{B}$ denotes $\{0,1\}$, and

$$R \stackrel{\text{def}}{=} \{(x,z) \in \mathbb{R}^{n-p}_+ \times [0,1]^p \mid f_j(x,z) \leq 0 \ \forall j \in M\}.$$

We call the set R a *continuous relaxation* of $\mathcal{F}$, and we emphasize that R is not unique in the sense that $\mathcal{F}$ can have many different continuous relaxations. Throughout, we will be interested in sets R that are convex. Even under the convexity requirement, the set R is not unique, and any convex

*Mathematical Sciences Department, IBM T.J. Watson Research Center, P.O. Box 218, Yorktown Heights, NY 10598, USA, gunluk@us.ibm.com

†Department of Industrial and Systems Engineering, University of Wisconsin-Madison, 1513 University Avenue, Madison, WI 53706, USA, linderoth@wisc.edu. The second author was supported by the US Department of Energy under grants DE-FG02-08ER25861 and DE-FG02-09ER25869, and the National Science Foundation under grant CCF-0830153.

set R' with the property that $R' \cap \mathbb{R}^{n-p}_+ \times \mathbb{B}^p = \mathcal{F}$ is a valid continuous relaxation of $\mathcal{F}$. If we let $\text{conv}(\mathcal{F})$ to denote the convex hull of $\mathcal{F}$, clearly all continuous relaxations of $\mathcal{F}$ that are convex have to contain $\text{conv}(\mathcal{F})$ and therefore $\text{conv}(\mathcal{F})$ is the smallest convex continuous relaxation of $\mathcal{F}$.

In the field of MILP, significant effort is spent on obtaining tight relaxations of the feasible set of solutions. This effort is justified by the fact that the optimization problem simply becomes a linear programming (LP) problem if $\text{conv}(\mathcal{F})$ can be explicitly described with linear inequalities. Notice that as the objective function is linear, it is easy to find an optimal solution that is an extreme point of $\text{conv}(\mathcal{F})$, which is guaranteed to be integral as all extreme points of $\text{conv}(\mathcal{F})$ are integral.

In MINLP, on the other hand, the optimal solution to the relaxation may occur at a point interior to $\text{conv}(\mathcal{F})$ and as such, it is not guaranteed to be integral. It is, however, easy to transform any problem to one with a linear objective function by moving the nonlinear objective function into the constraints. Specifically, the problem (1.1) can be equivalently stated as

$$\min\{\eta \mid \eta \in \mathbb{R},\ (x, z) \in \mathcal{F},\ \eta \geq c(x, z)\}. \tag{1.2}$$

We can, therefore, without loss of generality assume that the objective function of (1.1) is linear. Notice that, under this assumption, it is possible to solve the MINLP as a convex nonlinear programming (NLP) problem if $\text{conv}(\mathcal{F})$ can be explicitly described using convex functions.

In this paper, we review the perspective reformulation approach that, given a MINLP with an associated continuous relaxation R (perhaps after applying the transformation (1.2)), produces a smaller (tighter) continuous relaxation R' that contains $\text{conv}(\mathcal{F})$. The advantage of having tight relaxations is that as they approximate $\mathcal{F}$ better, they give better lower bounds, and they are more effective in obtaining optimal integral solutions via an enumeration algorithm.

1.2. The Importance of Formulation. To emphasize the importance of a tight relaxation, consider the well-known uncapacitated facility location problem (UFLP). Modeling the UFLP as a MILP is a canonical example taught to nearly all integer programming students to demonstrate the impact of a "good" versus "poor" formulation. In the UFLP, each customer in a set J must have their demand met from some facilities in a set I. A binary variable z_i indicates if the facility i is open, and the continuous variable x_{ij} represents the percentage of customer j's demands met from facility i.

The logical relationships that a customer may only be served from an open facility may be written algebraically in "aggregated" form

$$\sum_{j \in J} x_{ij} \leq |J| z_i \quad \forall i \in I$$

or in "disaggregated" form

$$x_{ij} \leq z_i \quad \forall i \in I, j \in J. \tag{1.3}$$

Writing the constraints in disaggregated form (1.3) makes a significant difference in the computational performance of MILP solvers. For example, in 2005, Leyffer and Linderoth [19] experiment with a simple branch and bound based MILP solver and report that on the average it took the solver 10,000 times longer when the aggregated formulation is used. For modern commercial MILP solvers, however, both formulations solve almost simultaneously. This is because modern MILP software *automatically* reformulates the aggregated (weak) formulation into the disaggregated (strong) one. We strongly believe that similar performance improvements can be obtained by MINLP solvers by performing automatic reformulation techniques specially developed for MINLP problems, and we hope this survey work will spur this line of research.

A common reformulation technique used by MILP solvers is to recognize simple structures that appear in the formulation and replace them with tight relaxations for these structures. The tightening of these structures leads to the tightening of the overall relaxation. We will follow this same approach here to derive tighter (perspective) relaxations through the study and characterization of convex hulls of simple sets.

1.3. The Perspective Reformulation. We are particularly interested in simple sets related to "on-off" type decisions. To that end, let z be a binary indicator variable that controls continuous variables x. The perspective reformulation is based on strengthening the natural continuous relaxation of the following "on-off" set:

$$S \stackrel{\text{def}}{=} \left\{ (x,z) \in \mathbb{R}^n \times \mathbb{B} \;\middle|\; \begin{array}{ll} x = \hat{x} & \text{if } z = 0 \\ x \in \Gamma & \text{if } z = 1 \end{array} \right\},$$

where $\hat{x}$ is a given point (ex: $\hat{x} = 0$), and

$$\Gamma \stackrel{\text{def}}{=} \{x \in \mathbb{R}^n \mid f_j(x) \leq 0 \; \forall j \in C, \; u \geq x \geq \ell\}$$

is a bounded convex set. (Note that Γ can be convex even when some of the functions f_j defining it are non-convex.)

In this paper we study the convex hull description of sets closely related to S. We present a number of examples where these simple sets appear as substructures, and we demonstrate that utilizing the convex hull description of these sets helps solve the optimization problem efficiently. Closely related to this work is the effort of Frangioni and Gentile [10, 12], who derive a class of cutting planes that significantly strengthen the formulation for MINLPs containing "on-off" type decisions with convex, separable, objective functions, and demonstrate that these inequalities are quite useful in practice. The connection is detailed more in Section 5.3.

The remainder of the paper is divided into 6 sections. Section 2 gives a review of perspective functions and how they can be used to obtain strong MINLP formulations. Section 3 first derives the convex hull description of two simple sets and then uses them as building blocks to describe the convex hull of more complicated sets. Section 4 describes a number of applications where the perspective reformulation technique can be successfully applied. Section 5 contains some discussion about computational approaches for solving relaxations arising from the perspective reformulation. Section 6 demonstrates numerically the impact of perspective reformulation and approach on two applications. Concluding remarks are offered in Section 7.

2. Perspective Functions and Convex Hulls. The *perspective* of a given function of $f : \mathbb{R}^n \to \mathbb{R}$ is the function $\tilde{f} : \mathbb{R}^{n+1} \to \mathbb{R}$ defined as follows:

$$\tilde{f}(\lambda, x) = \begin{cases} \lambda f(x/\lambda) & \text{if } \lambda > 0 \\ 0 & \text{if } \lambda = 0 \\ \infty & \text{otherwise} \end{cases} \tag{2.1}$$

An important property of perspective functions is that $\tilde{f}$ is convex provided that f is convex. A starting point for the use of the perspective function for strong formulations of MINLPs is the work of Ceria and Soares [8].

2.1. Using Perspective Functions to Obtain Convex Hulls. Ceria and Soares characterize the closure of the convex hull of the union of convex sets using the perspective transformation. The main result of Ceria and Soares is stated (in a simplified form) in Theorem 2.1.

THEOREM 2.1 (Ceria and Soares [8]). *For $t \in T$, let $G^t : \mathbb{R}^n \to \mathbb{R}^{m_t}$ be a vector-valued function with the property that the corresponding sets*

$$K^t = \{x \in \mathbb{R}^n \ : \ G^t(x) \leq 0\}$$

are convex and bounded. Let $K = \text{conv}(\cup_{t\in T} K^t)$. Then $x \in K$ if and only if the following (nonlinear) system is feasible:

$$x = \sum_{t\in T} x^t; \quad \sum_{t\in T} \lambda_t = 1; \quad \tilde{G}^t(\lambda_t, x^t) \leq 0, \quad \lambda_t \geq 0, \quad \forall t \in T. \tag{2.2}$$

Theorem 2.1 provides an extended formulation that describes the set K in a higher dimensional space. The work extends a well-known result from Balas in the case that all the K^t are polyhedral [2]. Also note that G^t being a convex function is sufficient, but not necessary for K^t to be convex.

A similar argument using the perspective functions was used by Stubbs and Mehrotra to formulate a convex programming problem to generate disjunctive cutting planes [23]. Later, Grossmann and Lee apply these same concepts to more general logic-constrained optimization problems known a generalized disjunctive programs [14].

2.2. Computational Challenges. There are a number of challenges in using the convex hull characterization of Theorem 2.1 in computation. One challenge is determining an appropriate disjunction such that $\mathcal{F} \subseteq \cup_{t\in T} K^t$. For example, using the disjunction that $z \in \{0,1\}^p$, requires $|T|$ to be exponential in p. In this case, a cutting plane algorithm, like the one suggested by Stubbs and Mehrotra may be appropriate [23].

A second challenge occurs when K^t is not bounded. In this case, the convex hull characterization is more complicated, and a closed form solution may not be known. Ceria and Soares address these complications and suggest a log-barrier approach to its solution [8]

A third challenge arises from the form of the perspective function. By definition, there is a point of nondifferentiability at $\lambda_t = 0$. This may cause difficulty for solvers used to solve the relaxation. Grossmann and Lee suggest to use the perturbed perspective inequalities

$$(\lambda + \epsilon) f(x/(\lambda + \epsilon)) \leq 0$$

for a small constant $\epsilon > 0$, which is valid if $f(0) \leq 0$. An improved perturbed expression is suggested by Furman, Sawaya, and Grossmann [13]:

$$((1-\epsilon)\lambda + \epsilon) f(x/((1-\epsilon)\lambda + \epsilon)) \leq \epsilon f(0)(1-\lambda). \tag{2.3}$$

Notice that both expressions give an exact approximation of the original constraints as $\epsilon \to 0$. In addition, inequality (2.3) has the very useful property that it gives the original constraints at $\lambda = 0$ and $\lambda = 1$, for any value of $0 < \epsilon < 1$. Furthermore, it preserves convexity as the left hand side of the inequality is convex if f is convex, and the inequality always forms a relaxation for any choice of $0 < \epsilon < 1$.

When the sets K^t are defined by conic quadratic inequalities (CQI), it is easier to deal with the nondifferentiability issue as the perspective of the associated functions are known to be representable by CQI [3]. We discuss this further in Section 5.2 and give CQI representation of different perspective functions that arise in the MINLP applications considered in Section 4.

3. Simple Sets. We next apply Theorem 2.1 to the special case where $T = 2$, and the sets K^0 and K^1 have a specific, simple, structure. More precisely, we consider the cases when K^0 is either a single point of a ray and K^1 is defined by convex functions. We then use these sets as building blocks to describe the convex hull of more complicated sets which appear as sub-structures in some MINLP models. We also note that there is ongoing work on other special cases by Bonami, Cornuéjols, and Hijazi [5].

3.1. The Convex Hull of a Point and a Convex Set. Consider the set $W = W_0 \cup W_1$ which is defined using the indicator variable $z \in \{0,1\}$

as follows: $W^0 = \big\{(x,z) \in \mathbb{R}^{n+1} \ : \ \ x = 0,\ z = 0\big\}$, and

$$W^1 = \big\{(x,z) \in \mathbb{R}^{n+1} \ : \ f_i(x) \leq 0 \text{ for } i \in I,\ u \geq x \geq l,\ z = 1\big\}$$

where $u, l \in \mathbb{R}^n_+$, and $I = \{1, \ldots, t\}$. Clearly, both W^0 and W^1 are bounded, and W^0 is a convex set. Furthermore, if W^1 is also convex then we may write an extended formulation as

$$\begin{aligned} \text{conv}(W) \ = \ \Big\{(x,z) \in \mathbb{R}^{n+1} \ : \quad & 1 \ \geq \ \lambda \ \geq \ 0, \\ & x = \lambda x^1 + (1-\lambda)x^0, \\ & z = \lambda z^1 + (1-\lambda)z^0, \\ & x^0 = 0,\ z^0 = 0,\ z^1 = 1 \\ & f_i(x^1) \ \leq 0 \text{ for } i \in I, \ \ u \geq x^1 \geq l \ \Big\}. \end{aligned}$$

We next give a description of W without the additional variables.

LEMMA 3.1. *If W^1 is convex, then* $\text{conv}(W) = W^- \cup W^0$*, where*

$$W^- \ = \ \Big\{(x,z) \in \mathbb{R}^{n+1} : \ f_i(x/z) \leq 0 \ i \in I, \ \ uz \geq x \geq lz, \ \ 1 \geq z > 0\Big\}.$$

(notice that z is strictly positive.)

Proof. As x^0, z^0 and z^1 are fixed in the extended formulation above, it is possible to substitute out these variables. In addition, as $z = \lambda$ after these substitutions, we can eliminate λ. Furthermore, as $x = \lambda x^1 = z x^1$, we can eliminate x^1 by replacing it with x/z provided that $z > 0$. If, on the other hand, $z = 0$, clearly $(x, 0) \in \text{conv}(W)$ if and only if $(x, 0) \in W^0$. □ It is also possible to show that W^0 is contained in the closure of W^- (see [16]) which leads to the following observation.

COROLLARY 3.1. $\text{conv}(W) = \text{closure}(W^-)$. We would like to emphasize that even when $f(x)$ is a convex function $f_i(x/z)$ may not be convex. However, for $z > 0$ we have

$$f_i(x/z) \leq 0 \quad \Leftrightarrow \quad z^t f_i(x/z) \leq 0 \tag{3.1}$$

for any $t \in \mathbb{R}$. In particular, taking $t = 1$ gives the perspective function which is known to be convex provided that $f(x)$ is convex.

When all $f_i(x)$ that define W^1 are polynomial functions, the convex hull of W can be described in closed form in the original space of variables. More precisely, let

$$f_i(x) = \sum_{t=1}^{p_i} c_{it} \prod_{j=1}^{n} x_j^{q_{itj}}$$

for all $i \in I$ and define $q_{it} = \sum_{j=1}^{n} q_{itj}$, $q_i = \max_t\{q_{it}\}$ and $\bar{q}_{it} = q_i - q_{it}$. If all $f_i(x)$ are convex and bounded in $[l, u]$, then (see [16])

$$\text{conv}(W) = \Big\{(x,z) \in \mathbb{R}^{n+1} : \sum_{t=1}^{p_i} c_{it} z^{\bar{q}_{it}} \prod_{j=1}^{n} x_j^{q_{itj}} \le 0 \text{ for } i \in I, \\ zu \ge x \ge lz, \qquad 1 \ge z \ge 0\Big\}.$$

3.2. The Convex Hull of a Ray and a Convex Set. It is possible to extend Lemma 3.1 to obtain the convex hull description of a ray and a convex set that contains the ray as an unbounded direction. More precisely consider the set $T = T_0 \cup T_1$ where

$$T^0 = \big\{(x,y,z) \in \mathbb{R}^{n+1+1} \ : \ \ x = 0,\ y \ge 0,\ z = 0\big\},$$

and

$$T^1 = \big\{(x,y,z) \in \mathbb{R}^{n+1+1} : f_i(x) \le 0\ i \in I,\ g(x) \le y, u \ge x \ge l,\ z = 1\big\}$$

where $u, l \in \mathbb{R}^n_+$, and $I = \{1, \ldots, t\}$.

LEMMA 3.2. *If T^1 is convex, then* $\text{conv}(T) = T^- \cup T^0$*, where*

$$T^- \ = \ \Big\{(x,y,z) \in \mathbb{R}^{n+1+1} : \ f_i(x/z) \le 0\ i \in I, \ \ g(x/z) \le y/z, \\ uz \ge x \ge lz, \qquad 1 \ge z > 0 \qquad \Big\}.$$

Proof. Using the same arguments as in the proof of Lemma 3.1, it is easy to show that $\text{conv}(T) = P \cup T^0$, where the following gives an extended formulation for the set P:

$$P \ = \ \Big\{(x,y,z) \in \mathbb{R}^{n+1+1} : \ f_i(x/z) \le 0\ i \in I,\ uz \ge x \ge lz,\ 1 \ge z > 0 \\ g(x/z) \le y/z - \frac{1-z}{z} y^0, \quad y^0 \ge 0\Big\}.$$

As $(1-z)/z > 0$ and $y^0 \ge$ for all feasible points, y^0 can easily be projected out to show that $P = T^-$. □

Similar to the proof of Corollary 3.1, it is possible to show that T^0 is contained in the closure of T^-.

COROLLARY 3.2. $\text{conv}(T) \ = \ \text{closure}(T^-)$.

In addition, we note that when all $f_i(x)$ for $i \in I$ and $g(x)$ are polynomial functions, the convex hull of T can be described in closed form by simply multiplying each inequality in the description of T^- with z raised to an appropriate power. We do not present this description to avoid repetition.

3.3. A Simple Quadratic Set. Consider the following mixed-integer set with 3 variables:

$$S = \Big\{(x, y, z) \in \mathbb{R}^2 \times \mathbb{B} \ : \ y \geq x^2, \ \ uz \geq x \geq lz, \ x \geq 0\Big\}.$$

Notice that $S = S^0 \cup S^1$ where $S^0 = \big\{(0, y, 0) \in \mathbb{R}^3 \ : \ y \geq 0\big\}$, and

$$S^1 = \ \big\{(x, y, 1) \in \mathbb{R}^3 \ : \ y \geq x^2, \ u \geq x \geq l, \ x \geq 0\big\}.$$

Applying Lemma 3.2 gives the convex hull of S as the perspective of the quadratic function defining the set.

LEMMA 3.3. $\text{conv}(S) \ = \ S^c$ *where*

$$S^c \ = \ \big\{(x, y, z) \in \mathbb{R}^3 \ : \ yz \geq x^2, \ uz \geq x \geq lz, \ 1 \geq z \geq 0, \ x, y \geq 0\big\}.$$

Notice that $x^2 - yz$ is not a convex function and yet the set $T^c = \{(x, y, z) \in \mathbb{R}^3 \ : \ yz \geq x^2, \ x, y, z \geq 0\}$ is a convex set.This explains why the set S^c, obtained by intersecting T^c with half-spaces, is convex.

3.4. A Larger Quadratic Set. Using the convex hull description of the set S, it is possible to produce a convex hull description of the following set

$$Q = \Big\{(w, x, z) \in \mathbb{R}^{n+1} \times \mathbb{B}^n : \ w \geq \sum_{i=1}^{n} q_i x_i^2, \ u_i z_i \geq x_i \geq l_i z_i, \ i \in I\Big\}, \tag{3.2}$$

where $I = \{1, \ldots, n\}$ and $q, u, l \in \mathbb{R}^n_+$. The convex hull description of Q is closely related to the convex envelope of the function $\sum_{i=1}^n q_i x_i^2$ over a mixed integer set. This set was first considered in the Ph.D. thesis of Stubbs [24].

Now consider the following extended formulation of Q

$$\bar{Q} \stackrel{\text{def}}{=} \Big\{(w, x, y, z) \in \mathbb{R}^{3n+1} \ : \ w \geq \sum_i q_i y_i, (x_i, y_i, z_i) \in S_i, \ \ i \in I\Big\}$$

where S_i has the same form as the set S discussed in the previous section except the bounds u and l are replaced with u_i and l_i. Note that if $(w, x, y, z) \in \bar{Q}$ then $(w, x, z) \in Q$, and therefore $\text{proj}_{(w,x,z)}(\bar{Q}) \subseteq Q$. On the other hand, for any $(w, x, z) \in Q$, letting let $y'_i = x_i^2$ gives a point $(w, x, y', z) \in \bar{Q}$. Therefore, $\bar{Q}$ is indeed an extended formulation of Q, or, in other words, $Q = \text{proj}_{(w,x,z)}(\bar{Q})$.

Before we present a convex hull description of $\bar{Q}$ we first define some basic properties of mixed-integer sets. First, remember that given a closed set $P \subset \mathbb{R}^n$, a point $p \in P$ is called an *extreme point* of P if it can not be represented as $p = 1/2p_1 + 1/2p_2$ for $p_1, p_2 \in P$, $p_1 \neq p_2$. The set P is called *pointed* if it has extreme points. A pointed set P is called *integral*

with respect to (w.r.t.) a subset of the indices J if for any extreme point $p \in P$, $p_i \in \mathbb{Z}$ for all $i \in J$.

LEMMA 3.4 ([16]). *For $i = 1, 2$ let $P_i \subset \mathbb{R}^{n_i}$ be a closed and pointed set which is integral w.r.t. indices I_i. Let $P' = \{(x, y) \in \mathbb{R}^{n_1+n_2} : x \in P_1,\ y \in P_2\}$.*

(i) P' is integral with respect to $I_1 \cup I_2$.

(ii) $\text{conv}(P') = \{(x, y) \in \mathbb{R}^{n_1+n_2} : x \in \text{conv}(P_1),\ y \in \text{conv}(P_2)\}$.

LEMMA 3.5 ([16]). *Let $P \subset \mathbb{R}^n$ be a given closed, pointed set and let $P' = \{(w, x) \in \mathbb{R}^{n+1} : w \geq ax,\ x \in P\}$ where $a \in \mathbb{R}^n$.*

(i) If P is integral w.r.t. J, then P' is also integral w.r.t. J.

(ii) $\text{conv}(P') = P''$ *where* $P'' = \{(w, x) \in \mathbb{R}^{n+1} : w \geq ax,\ x \in \text{conv}(P)\}$.

We are now ready to present the convex hull of $\bar{Q}$.

LEMMA 3.6. *The set*

$$\bar{Q}^c = \Big\{(w, x, y, z) \in \mathbb{R}^{3n+1} : w \geq \sum_i q_i y_i,\ (x_i, y_i, z_i) \in S_i^c,\ i \in I\Big\}.$$

is integral w.r.t. the indices of z variables. Furthermore, $\text{conv}(\bar{Q}) = \bar{Q}^c$.

Proof. Let $D = \{(x, y, z) \in \mathbb{R}^{3n} : (x_i, y_i, z_i) \in S_i,\ \ i \in I\}$ so that $\bar{Q} = \{(w, x, y, z) \in \mathbb{R}^{3n+1} : w \geq \sum_{i=1}^n q_i y_i,\ (x, y, z) \in D\}$. By Lemma 3.5, the convex hull of $\bar{Q}$ can be obtained by replacing D with its convex hull in this description. By Lemma 3.4, this can simply be done by taking convex hulls of S_i's, that is, by replacing S_i with $\text{conv}(S_i)$ in the description of D. Finally, by Lemma 3.5, $\bar{Q}^c$ is integral. □

A natural next step is to study the projection of the set $\bar{Q}^c$ into the space of (w, x, z). One possibility is to substitute the term x_i^2/z_i for each variable y_i, resulting in the inequality $w \geq \sum_i q_i x_i^2/z_i$. This formula, however, may not be suitable for computation as it is not defined for $z_i = 0$, and $z_i = 0$ is one of the two feasible values for Z_I. We next present an explicit description of the projection that uses an an exponential number of inequalities. Let

$$Q^c = \Big\{(w, x, z) \in \mathbb{R}^{2n+1} : w \prod_{i \in S} z_i \geq \sum_{i \in S} q_i x_i^2 \prod_{l \in S \setminus \{i\}} z_l,\ \forall S \subseteq I \qquad (\Pi)$$
$$u_i z_i \geq x_i \geq l_i z_i,\ \ x_i \geq 0,\ \ i \in I\Big\}$$

Notice that a given point $\bar{p} = (\bar{w}, \bar{x}, \bar{z})$ satisfies the nonlinear inequalities in the description of Q^c for a particular $S \subseteq I$ if and only if one of the following conditions hold: (i) $\bar{z}_i = 0$ for some $i \in S$, or, (ii) if all $z_i > 0$, then $\bar{w} \geq \sum_{i \in S} q_i \bar{x}_i^2/\bar{z}_i$. Based on this observation it is possible to show that these (exponentially many) inequalities are sufficient to describe the convex hull of Q in the space of the original variables.

LEMMA 3.7 ([16]). $Q^c = \text{proj}_{(w,x,z)}(\bar{Q}^c)$. Note that all of the exponentially many inequalities that are used in the description of Q^c are indeed

necessary. To see this, consider a simple instance with $u_i = l_i = q_i = 1$ for all $i \in I = \{1, 2, \ldots, n\}$. For a given $\bar{S} \subseteq I$, let $p^{\bar{S}} = (\bar{w}, \bar{x}, \bar{z})$ where $\bar{w} = |\bar{S}| - 1$, $\bar{z}_i = 1$ if $i \in \bar{S}$, and $\bar{z}_i = 0$ otherwise, and $\bar{x} = \bar{z}$. Note that $p^{\bar{S}} \notin Q^c$. As $\bar{z}_i = q_i \bar{x}_i^2$, inequality ($\Pi$) is satisfied by $\bar{p}$ for $S \subseteq I$ if and only if

$$(|\bar{S}| - 1) \prod_{i \in S} \bar{z}_i \geq |S| \prod_{i \in S} \bar{z}_i.$$

Note that unless $S \subseteq \bar{S}$, the term $\prod_{i \in S} \bar{z}_i$ becomes zero and therefore inequality (Π) is satisfied. In addition, inequality (Π) is satisfied whenever $|\bar{S}| > |S|$. Combining these two observations, we can conclude that the only inequality violated by $p^{\bar{S}}$ is the one with $S = \bar{S}$. Due to its size, the projected set is not practical for computational purposes and we conclude that it is more advantageous to work in the extended space, keeping the variables y_i

3.5. A Simple Non-quadratic Set and its Generalization. The simple 3 variable mixed-integer set S introduced in Section 3.3 can be generalized to the following set, studied by Aktürk, Atamtürk, and Gürel:

$$C = \left\{(x, y, z) \in \mathbb{R}^2 \times \mathbb{B} \,:\, y \geq x^{a/b},\ uz \geq x \geq lz,\ x \geq 0\right\}$$

where $a, b \in \mathbb{Z}_+$ and $a \geq b > 0$. Clearly $C = C^0 \cup C^1$, with $C^0 = \{(0, y, 0) \in \mathbb{R}^3 \,:\, y \geq 0\}$, and $C^1 = \{(x, y, 1) \in \mathbb{R}^3 \,:\, y \geq x^{a/b}, u \geq x \geq l,\ x \geq 0\}$. By applying Lemma 3.2, the convex hull of C is given by using the perspective of the function $f(y, x) = y^b - x^a$.

LEMMA 3.8 (Aktürk, Atamtürk, Gürel [1]). *The convex hull of C is given by*

$$C^c = \left\{(x, y, z) \in \mathbb{R}^3 \,:\, y^b z^{a-b} \geq x^a,\ uz \geq x \geq lz,\ 1 \geq z \geq 0,\ x, y \geq 0\right\}.$$

In addition, it is possible to construct the convex hull of larger sets using the set C as a building block. See [1] for more details.

4. Applications. In this section, we six applications to which the perspective reformulation has been applied.

4.1. Separable Quadratic UFL. The Separable Quadratic Uncapacitated Facility Location Problem (SQUFL) was introduced by Günluk, Lee, and Weismantel [15]. In the SQUFL, there is a set of customers J, opening a facility $i \in I$. All customers have unit demand that can be satisfied using open facilities only. The shipping cost is proportional to the square of the quantity delivered. Letting z_i indicate if facility $i \in I$ is open, and x_{ij} denote the fraction of customer j's demand met from facility

i, SQUFL can be formulated as follows:

$$\begin{aligned} \min \quad & \sum_{i\in I} c_i z_i + \sum_{i\in I}\sum_{j\in J} q_{ij} x_{ij}^2 \\ \text{subject to} \quad & x_{ij} \le z_i \quad \forall i \in I, \forall j \in J, \\ & \sum_{i\in I} x_{ij} = 1 \quad \forall j \in J, \\ & z_i \in \{0,1\}, \qquad x_{ij} \ge 0 \quad \forall i \in I, \forall j \in J. \end{aligned}$$

To apply the perspective formulation, auxiliary variables y_{ij} are used to replace the terms x_{ij}^2 in the objective function and the constraints

$$x_{ij}^2 - y_{ij} \le 0 \quad \forall i \in I, j \in J \tag{4.1}$$

are added. In this reformulation, if $z_i = 0$, then $x_{ij} = 0$ and $y_{ij} \ge 0\ \forall j \in J$, while if $z_i = 1$, the convex nonlinear constraints (4.1) should also hold. Therefore, we can strengthen the formulation of SQUFL by replacing (4.1) by its perspective reformulation

$$x_{ij}^2/z_i - y_{ij} \le 0 \qquad \forall i \in I, \forall j \in J. \tag{4.2}$$

We demonstrate the impact of this reformulation on solvability of the MINLP in Section 6.2.

4.2. Network Design with Congestion Constraints. The next application is a network design problem with requirements on queuing delay. Similar models appear in the papers [6], [4], and [7]. In the problem, there is a set of commodities K to be shipped over a capacitated directed network $G = (N, A)$. The capacity of arc $(i, j) \in A$ is u_{ij}, and each node $i \in N$ has a net supply b_i^k of commodity $k \in K$. There is a fixed cost c_{ij} of opening each arc $(i, j) \in A$, and we introduce {0,1}-variables z_{ij} to indicate whether arc $(i, j) \in A$ is opened. The quantity of commodity k routed on arc (i, j) is measured by variable x_{ij}^k and $f_{ij} = \sum_{k\in K} x_{ij}^k$ denotes the total flow on the arc. A typical measure of the total weighted congestion (or queuing delay) is

$$\rho(f) \stackrel{\text{def}}{=} \sum_{(i,j)\in A} r_{ij} \frac{f_{ij}}{1 - f_{ij}/u_{ij}},$$

where $r_{ij} \ge 0$ is a user-defined weighting parameter for each arc. We use decision variables y_{ij} to measure the contribution of the congestion on arc (i, j) to the total congestion $\rho(f)$. The network should be designed so as to keep the total queuing delay less than a given value β, and this is to be accomplished at minimum cost. The resulting optimization model (NDCC)

can be written as

$$
\begin{aligned}
\min \quad & \sum_{(i,j)\in A} c_{ij} z_{ij} \\
\text{subject to} \quad & \sum_{(i,j)\in A} x^k_{ij} - \sum_{(j,i)\in A} x^k_{ji} = b^k_i && \forall i \in N,\ \forall k \in K, \\
& \sum_{k\in K} x^k_{ij} - f_{ij} = 0 && \forall (i,j) \in A, \\
& f_{ij} \le u_{ij} z_{ij} && \forall (i,j) \in A, \quad (4.3) \\
& y_{ij} \ge \frac{r_{ij} f_{ij}}{1 - f_{ij}/u_{ij}} && \forall (i,j) \in A, \quad (4.4) \\
& \sum_{(i,j)\in A} y_{ij} \le \beta, \\
& x \in \mathbb{R}^{|A|\times|K|}_+,\ y \in \mathbb{R}^{|A|}_+,\ f \in \mathbb{R}^{|A|}_+, \quad z \in \{0,1\}^{|A|}.
\end{aligned}
$$

In this formulation of NDCC, note that if $z_{ij} = 0$, then $f_{ij} = 0$ and $y_{ij} \ge 0$. On the other hand, if $z_{ij} = 1$, then f_{ij} and y_{ij} must satisfy $f_{ij} \le u_{ij}$ and constraint (4.4). Therefore, each constraint (4.4) can be replaced by its perspective counterpart:

$$
z_{ij} \left[\frac{r_{ij} f_{ij}/z_{ij}}{1 - f_{ij}/(u_{ij} z_{ij})} - \frac{y_{ij}}{z_{ij}} \right] \le 0. \tag{4.5}
$$

4.3. Scheduling with controllable processing times. Consider a scheduling problem where jobs are assigned to non-identical parallel machines with finite capacity. Let J denote the set of jobs and I denote the set of machines. In this problem, not all jobs have to be processed but if job $j \in J$ is assigned to a machine $i \in I$, a reward of h_{ij} is collected. The regular processing time of job j on machine i is p_{ij}, however by paying a certain cost, it can be reduced to $(p_{ij} - x_{ij})$ where $x_{ij} \in [0, u_{ij}]$. The cost of reducing the processing time of job i on machine j by x units is given by the expression

$$
f_{ij}(x) = k_{ij} x^{a_{ij}/b_{ij}}.
$$

This problem is called the *machine-job assignment problem with controllable times* and has been recently studied by Aktürk, Atamtürk and Gürel

[1]. A MINLP formulation for this problem is:

$$\max \quad \sum_{i\in I}\sum_{j\in J}(h_{ij}z_{ij} - f_{ij}(x_{ij}))$$

$$\begin{aligned}
\text{subject to} \quad & \sum_{j\in J}(p_{ij}z_{ij} - x_{ij}) \le c_i && \forall i \in I \\
& x_{ij} \le u_{ij}z_{ij} && \forall i \in I,\ \forall j \in J \quad (4.6)\\
& \sum_{i\in I} z_{ij} \le 1 && \forall j \in J \\
& z_{ij} \in \{0,1\},\ \ x_{ij} \ge 0 && \forall i \in I,\ \forall j \in J \quad (4.7)
\end{aligned}$$

where the variable z_{ij} denotes if job j is assigned to machine i and x_{ij} is the reduction on the associated processing time. The total processing time available on machine $i \in I$ is denoted by c_i. The objective is to maximize the sum of the rewards minus the cost of reducing the processing times.

As in the case of the SQUFL in Section 4.1, after adding a new variable y_{ij} and a new constraint

$$x_{ij}^{a_{ij}/b_{ij}} \le y_{ij} \tag{4.8}$$

for all $i \in I,\ j \in J$, it is possible to replace the objective function with the following linear expression:

$$\sum_{i\in I}\sum_{j\in J}(h_{ij}z_{ij} - y_{ij}/k_{ij}).$$

The inequality (4.8) together with inequalities (4.6) and (4.7) is the set C studied in Section 3.5 and therefore, inequality (4.8) can be replaced with its perspective counterpart

$$z\left(\frac{x_{ij}}{z_{ij}}\right)^{a_{ij}/b_{ij}} \le y \tag{4.9}$$

to obtain a stronger formulation. The authors of [1] raise both sides of inequality (4.9) to the bth power and multiply both sides by $z_{ij}^{a_{ij}-b_{ij}}$ to obtain equivalent inequalities

$$x_{ij}^{a_{ij}} \le y_{ij}^{b_{ij}} z^{a_{ij}-b_{ij}} \tag{4.10}$$

4.4. The unit commitment problem. One of the essential optimization problems in power generation is the so-called *the unit commitment problem* which involves deciding the power output levels of a collection of power generators over a period of time. In this setting, a generator (also called a *unit*) is either turned off and generates no power, or it is turned on

and generates power in a given range $[l, u]$. It is important to point out that $l > 0$ and therefore production levels are "semi-continuous". In most models, time horizon is divided into a small number of discrete intervals (ex: 48 half hour intervals for a daily problem) and the generators are required to collectively satisfy a given demand level in each interval. The operating cost of a generator is typically modeled by a convex quadratic function. There are also additional constraints including the most commonly used *min-up, min-down* constraints that require that a generator must stay on for a certain number of time periods after it is turned on, and similarly, it must stay down for a number of time periods after it is turned off. Letting I denote the set of generators and T denote the set of time periods under consideration, a MINLP formulation for this problem is the following:

$$\begin{aligned}
\min \quad & \sum_{i\in I}\sum_{t\in T} h_{it}z_{it} + \sum_{i\in I}\sum_{t\in T} f_{it}(x_{it}) \\
\text{subject to} \quad & \sum_{i\in I} x_{it} = d_t && \forall t \in T \\
& l_i z_{it} \;\le\; x_{it} \le u_i z_{it} && \forall i \in I,\ \forall t \in T \qquad (4.11) \\
& z \in P && \qquad (4.12) \\
& z_{ij} \in \{0,1\} && \forall i \in I,\ \forall t \in T \qquad (4.13)
\end{aligned}$$

where variable z_{it} denotes if generator $i \in I$ is turned on in period $t \in T$ and variable x_{it} gives the production level when the generator is on. There is a fixed cost h_{it} of operating a unit i in period t as well as a variable cost given by the convex quadratic function $f_{it}(x) = a_{it}x^2 + b_{it}x$ for some given $a_{it}, b_{it} \in \mathbb{R}_+$. The demand requirement in period t is given by d_t. Finally, the constraint (4.12) above expresses some other constraints involving how generators can be turned on and off.

To obtain a stronger formulation, it is again possible to introduce new variables y_{it} and constraints

$$a_{it}x_{it}^2 + b_{it}x_{it} \;\le y_{it} \qquad (4.14)$$

for all $i \in I$ and $t \in T$ so that the new variable y_{it} can replace the term $f_{it}(x_{it})$ in the objective function. Using inequalities (4.11) and (4.13), we can now replace inequality (4.14) with its perspective counterpart

$$a_{it}x_{it}^2 + b_{it}x_{it}z_{it} \;\le y_{it}z_{it} \qquad (4.15)$$

to obtain a stronger formulation.

4.5. Stochastic Service System Design. Elhedhli [9] describes a stochastic service system design problem (SSSD) modeled as a network of M/M/1 queues. The instance is characterized by a sets of customers M,

facilities N, and service levels K. There are binary decision variables x_{ij} to denote if customer i's demand is met by facility j and y_{jk} to denote if facility j is operating at service level k. Customer i has a mean demand rate of λ_i and facility j has a mean service rate of μ_{jk} when operated at service level k. There is a fixed cost c_{ij} of assigning customer i to facility j, and a fixed cost f_{jk} of operating facility j at level k.

A straightforward formulation of problem is not convex, however, by introducing auxiliary variables v_j and z_{jk}, Elhedhli provides the following convex MINLP formulation:

$$\min \quad \sum_{i\in M}\sum_{j\in N} c_{ij}x_{ij} + t\sum_{j\in N} v_j + \sum_{j\in N}\sum_{k\in K} f_{jk}y_{jk}$$

$$\begin{aligned}
\text{subject to} \quad \sum_{i\in M} \lambda_i x_{ij} - \sum_{k\in K} \mu_{jk} z_{jk} &= 0 && \forall j \in N \\
\sum_{j\in N} x_{ij} &= 1 && \forall i \in M \\
\sum_{k\in K} y_{jk} &\le 1 && \forall j \in N \\
z_{jk} - y_{jk} &\le 0 && \forall j \in N, \forall k \in K \qquad (4.16)\\
z_{jk} - v_j/(1+v_j) &\le 0 && \forall j \in N, \forall k \in K \qquad (4.17)\\
z_{jk}, v_j \ge 0,\ x_{ij}, y_{jk} \in \{0,1\} \quad & && \forall i \in M, j \in N, \forall k \in K \qquad (4.18)
\end{aligned}$$

Instead of directly including the nonlinear constraints (4.17) in the formulation, Elhedhli proposes linearizing the constraints at points $(v_j, z_{jk}) = (v_j^b, 1), b \in B$, yielding

$$z_{jk} - \frac{1}{(1+v_j^b)^2} v_j \le \frac{(v_j^b)^2}{(1+v_j^b)^2}. \qquad (4.19)$$

Elhedhli uses a dynamic cutting plane approach to add inequalities (4.19).

Notice that if $y_{jk} = 0$, then $z_{jk} = 0$ and $v_j \ge 0$, and therefore inequality (4.17) can be replaced by their perspective counterpart

$$z_{jk} \le \frac{v_j}{1 + v_j/y_{jk}} \qquad (4.20)$$

yielding a tighter (non-linear) formulation. Furthermore, linearizing these inequalities at points $(v_j, y_{jk}, z_{jk}) = (v_j^b, 1, 1), b \in B$, gives

$$z_{jk} - \frac{1}{(1+v_j^b)^2} v_j \le \frac{(v_j^b)^2}{(1+v_j^b)^2} y_{jk} \qquad (4.21)$$

which dominate the inequalities used in Elhedhli [9]. Note that the inequalities (4.21) could also be derived by applying a logical integer strengthening argument to the inequalities (4.19). The linearized perspective inequalities are called *perspective cuts* [10], which are discussed in greater detail in Section 5.3. Computational results demonstrating the effect of the the perspective reformulation on this application are given in Section 6.2.

4.6. Portfolio Selection. A canonical optimization problem in financial engineering is to find a minimum variance portfolio that meets a given minimum expected return requirement of $\rho > 0$, see [20]. In the problem, there is a set N of assets available for purchase. The expected return of asset $i \in N$ in given by α_i, and the covariance of the returns between pairs of assets is given in the form of a positive-definite matrix $Q \in \mathbb{R}^{n\times n}$. There can be at most K different assets in the portfolio and there is a minimum and maximum buy-in thresholds for the assets chosen. A MINLP formulation of the problem is

$$\min\{x^T Q x \mid e^T x = 1, \alpha^T x \geq \rho, e^T z \leq K; \ell_i z_i \leq x_i \leq u_i z_i,\ z_i \in \mathbb{B}\ \forall i \in N\},$$

where the decision variable x_i is the percentage of the portfolio invested in asset i and z_i is a binary variable indicating the purchase of asset i. Unfortunately, direct application of the perspective reformulation is not possible, as the objective is not a separable function of the decision variables.

However, in many practical applications, the covariance matrix is obtained from a *factor model* and has the form $Q = B\Omega B^T + \Delta^2$, for a given *exposure matrix*, $B \in \mathbb{R}^{n\times f}$, positive-definite *factor-covariance matrix* $\Omega \in \mathbb{R}^{f\times f}$, and positive definite, diagonal *specific-variance* matrix $\Delta \in \mathbb{R}^{n\times n}$ [22]. If a factor model is given, a separable portion of the objective function is easily extracted by introducing variables y_i, changing the objective to

$$\min x^T (B\Omega B^T) x + \sum_{i\in N} \Delta_{ii} y_i,$$

and enforcing the constraints $y_i \geq x_i^2\ \forall i \in N$. These constraints can then be replaced by their perspective counterparts $y_i \geq x_i^2/z_i$ to obtain a tighter formulation.

Even if the covariance matrix Q does not directly have an embedded diagonal structure from a factor model, it may still be possible to find a diagonal matrix D such that $R = Q - D$ is positive-definite. For example, Frangioni and Gentile [10] suggest using $D = \lambda_n I$, where $\lambda_n > 0$ is the smallest eigenvalue of Q. Frangioni and Gentile [11] subsequently gave a semidefinite programming approach to obtain a diagonal matrix D that may have desirable computational properties.

5. Computational Approaches. Algorithms to solve MINLP are based on solving a sequence of continuous relaxations of the formulation. In

this section, we discuss approaches and software for solving the perspective reformulation. The approaches for solving the perspective reformulation fall into three general categories, with tradeoffs in speed and generality of the approaches. The first approach, for use in the most general cases, is to simply give the reformulated problem to a general purpose NLP solver. The second approach is to use a solver that is specialized for second-order cone programming problems. A final approach is to linearize the nonlinear functions of the perspective reformulation and use a MILP solver. This approach is most effective if the linearizations are added in a dynamic manner.

5.1. NLP Solvers. Care must be taken when using a traditional solver for nonlinear programs (NLP)s to solve the perspective reformulation. Applying the perspective transformation to a constraint $f(x) \leq 0$ gives $zf(x/z) \leq 0$, which is not defined when $z = 0$. Often, the constraint $zf(x/z) \leq 0$ can be manipulated to remove z from the denominator, but this may result in new difficulties for the NLP solver. To illustrate this point, consider the inequalities (4.1) in the description of the SQUFL introduced in Section 4.1. Applying the perspective transformation to $x^2 - y \leq 0$ gives

$$x^2/z - y \leq 0, \tag{5.1}$$

which is not defined at $z = 0$. Multiplying both sides of the inequality by z gives

$$x^2 - yz \leq 0. \tag{5.2}$$

However, since the function $x^2 - yz$ is not convex, traditional NLP solvers cannot guarantee convergence to a globally optimal solution to the relaxation. A reformulation trick can be used to turn (5.1) into an equivalent inequality with a convex constraint function. Specifically, since $y, z \geq 0$, (5.1) is equivalent to

$$\sqrt{(2x)^2 + (y - z)^2} - y - z \leq 0, \tag{5.3}$$

and the constraint function in (5.3) is convex. This, however, may introduce yet a different obstacle to NLP software, as the constraint function in (5.3) is not differentiable at $(x, y, z) = (0, 0, 0)$. In Section 6 we will show some computational experiments aimed at demonstrating the effectiveness of NLP software at handling perspective constraints in their various equivalent forms.

5.2. SOCP Solvers. A second-order program (SOCP) is mathematical program that contains (conic) quadratic inequalities (CQI) of the form

$$\|Ax + b\|_2 \leq c^T x + d. \tag{5.4}$$

A rotated second-order cone constraint is of the form

$$x^2 \leq yz \text{ with } y \geq 0, z \geq 0. \tag{5.5}$$

As noted in Section 5.1, rotated second-order cone constraints (5.5) are equivalent to second-order cone constraints (5.4) since

$$\|(2x, y-z)^T\| \leq y+z \Leftrightarrow x^2 \leq yz, y \geq 0, z \geq 0. \tag{5.6}$$

The set of points x that satisfy (5.4) or (5.5) forms a convex set, and efficient and robust algorithms exist for solving optimization problems containing second-order cone constraints [25, 21]. An interesting and important observation from a computational standpoint is that the nonlinear inequalities present in all of the applications described in Section 4 can be described with second order cone constraints. Further, if a set of points is representable using second order cone constraints, then the perspective mapping of the set is also SOC-representable [3]. Therefore, quite often software designed to solve SOCPs can be used to solve the perspective relaxations arising from real applications.

To demonstrate the variety of reformulation techniques required to express nonlinear constraints in their SOC representation, we give the SOC representations for all of the nonlinear perspective constraints appearing in the applications in Section 4. The book of Ben-Tal and Nemirovski [3] contains a wealth of knowledge about the types of inequalities whose feasible regions are representable with CQI.

For the SQUFL described in Section 4.1, the nonlinear inequalities in the perspective reformulation (4.2) can be multiplied by z_i and then are evidently in rotated second order cone form (5.5). The nonlinear inequalities (4.5) in the perspective reformulation of the NDCC described in Section 4.2 can be put in the (rotated) second order cone form

$$(y_{ij} - r_{ij}f_{ij})(u_{ij}z_{ij} - f_{ij}) \geq r_{ij}f_{ij}^2, \tag{5.7}$$

which is a rotated SOC constraint as $y_{ij} \geq r_{ij}f_{ij}$ and $u_{ij}z_{ij} \geq f_{ij}$ for any feasible solution. Note that we multiply the inequality by z_{ij} to obtain the form (5.7) above. For the scheduling application of Section 4.3, the nonlinear inequalities (4.10) can be represented using SOC constraints. In this case, the transformation is more complicated, requiring $O(\log_2 a_{ij})$ additional variables and $O(\log_2 a_{ij})$ constraints. The details of the representation are provided in [1]. The nonlinear inequalities (4.15) arising from the perspective reformulation of the Unit Commitment Problem are also representable with CQI as

$$a_{it}x_{it}^2 \leq z_{it}(b_{it}x_{it} - y_{it}).$$

The inequalities (4.20) from the SSSD problem from Section 4.5 are representable as rotated SOC constraints using the relation

$$z_{jk}y_{jk} + z_{jk}v_j - y_{jk}v_j \leq 0 \Leftrightarrow v_j^2 \leq (-z_{jk} + v_j)(y_{jk} + v_j).$$

For the mean-variance problem of Section 4.6, the constraints $y_i \geq x_i^2 \ \forall i \in N$ and the perspective version $y_i \geq x_i^2/z_i \ \forall i \in N$ can be placed in SOC format in the same fashion as the SQUFL problem in Section 4.1.

Using SOC software is generally preferable to using general NLP software for MINLP instances whose only nonlinearities can be put in SOC form. We will provide computational evidence for the improved performance of SOCP solvers over general NLP software in Section 6.

5.3. LP Solvers. We next discuss how to use outer approximation cuts to solve the perspective formulation via linear programming (LP) solvers. As current LP solvers are significantly faster then both NLP and SOCP solvers, this approach may offer significant advantages. We will discuss this idea using a simple MINLP where the nonlinearity is restricted to the objective function. Consider

$$\min_{(x,z)\in\mathbb{R}^n\times\mathbb{B}} \Big\{ f(x) + cz \mid Ax \leq bz \Big\},$$

where (i) $X = \{x \mid Ax \leq b\}$ is bounded (also implying $\{x \mid Ax \leq 0\} = \{0\}$), (ii) $f(x)$ is a convex function that is finite on X, and (iii) $f(0) = 0$. Under these assumptions, for any $\bar{x} \in X$ and subgradient $s \in \partial f(\bar{x})$, the following inequality

$$v \geq f(\bar{x}) + c + s^T(x - \bar{x}) + (c + f(\bar{x}) - s^T\bar{x}))(z - 1) \tag{5.8}$$

is valid for the equivalent mixed integer program

$$\min_{(x,z,v)\in\mathbb{R}^n\times\mathbb{B}\times\mathbb{R}} \Big\{ v \mid v \geq f(x) + cz, Ax \leq bz \Big\}.$$

Inequality (5.8) is called the *perspective cut* and has been introduced by Frangioni and Gentile [10]. In their paper, Frangioni and Gentile use these cuts dynamically to build a tight formulation. It is possible to show[17] that perspective cuts are indeed outer approximation cuts for the perspective reformulation for this MINLP and therefore adding all (infinitely many) perspective cuts has the same strength as the perspective reformulation.

In subsequent computational work, Frangioni and Gentile compare the computational performance of a LP solver (using perspective cuts dynamically) with a second-order cone solver on instances of the unit commitment problem and the portfolio optimization problem discussed earlier [12]. They conclude that for these instances, the dynamic (linear) approximation approach is significantly better than a SOC approach. The LP approach offers significant advantages, such as fast resolves in branch-and-bound, and the extensive array of cutting planes, branching rules, and heuristics that are available in powerful commercial MILP software. However, a dynamic cutting plane approach requires the use of the callable library of the solver software to add the cuts. For practitioners, an advantage of

nonlinear automatic reformulation techniques is that they may be directly implemented in a modeling language. Here, we offer one simple way to obtain some of the strength of the perspective reformulation, while retaining the advantages of MILP software to solve the subproblem and an algebraic modeling language to formulate the instance. An interesting observation is that the (linear) perspective cuts (5.8) are equivalent to building a linear outerapproximation of the nonlinear inequalities

$$v \geq f(x) + cz$$

and then strengthening the linear inequalities using a logical deductive argument. This allows an approximate version of perspective cuts to also be implemented directly in a modeling language.

For example, in the SQUFL problem described in Section 4.1, instead of writing $y_{ij} \geq x_{ij}^2$, a linear relaxation of these inequalities could be created by choosing a set of points $B = \{\chi_i^1, \chi_i^2, \ldots \chi_i^{|B|}\}$, with $0 \leq \chi_i^b \leq 1$ and writing inequalities

$$y_{ij} \geq 2(\chi_i^b)x_{ij} - (\chi_i^b)^2 \ \forall i \in N, \forall j \in N, \forall b \in B. \tag{5.9}$$

Using the observation that if $z_i = 0$, then $x_{ij} = y_{ij} = 0$, the inequalities (5.9) can be strengthened to

$$y_{ij} \geq 2(\chi_i^b)x_{ij} - (\chi_i^b)^2 z_i \ \forall i \in N, \forall j \in N, \forall b \in B. \tag{5.10}$$

The inequalities (5.10) are precisely the perspective cuts (5.8) of Frangioni and Gentile for this instance.

This observation may give a simple and effective heuristic for MINLPs of this form. The heuristic works by creating a piecewise linear underapproximation of the nonlinear functions, using a specified set of breakpoints B. The linear approximation can be strengthened using by adding multiplying the constant term by the binary variable as in (5.9). Solving this underapproximating MILP provides a lower bound on the optimal solution value of the MINLP. To obtain an upper bound, the integer variables may be fixed at the values found by the solution to the MILP, and a continuous NLP solved. We will demonstrate the effectiveness of this approach in the next section.

6. Computational Results. The improvement in computational performance that can be obtained by using the perspective reformulation is exhibited on two families of instances, SQUFL (described in Section 4.1) and SSSD (described in Section 4.5). We also demonstrate the behavior of the various methodologies available for solving the relaxations.

6.1. Separable Quadratic Uncapacitated Facility Location. Random instances of SQUFL were generated similar to the instances of Günlük, Lee, and Weismantel [15]. For each facility $i \in M$, a location p_i is generated uniformly in $[0, 1]^2$ and the variable cost parameter was calculated

as $q_{ij} = 50\|p_i - p_j\|_2$. The fixed cost c_i of opening a facility is generated uniformly in $[1, 100]$. Ten random instances were created for values of $m \in \{20, 30\}$ and $n \in \{100, 150\}$. Each instance was solved with four different methods.

1. The original formulation, was solved with CPLEX (v11.0) software for Mixed-Integer Quadratic Programming (MIQP);
2. The perspective reformulation, with the perspective inequalities put in rotated SOC form, was solved with the Mosek (v5.0) software for Mixed-Integer Second-Order Cone Programming (MISOCP);
3. The linear under-approximation using unstrengthened inequalities (5.9) was solved with the CPLEX (v11.0) software for Mixed-Integer Linear Programming (MILP); and
4. The linear under-approximation using the perspective cuts (5.10) was solved with the CPLEX (v11.0) software for MILP.

In methods (3) and (4), $|B| = 10$ breakpoints equally distributed between 0 and 1 were used to underestimate each nonlinear function. After solving the approximating MILP, the value of the integer variables were fixed and the resulting continuous relaxation (a convex quadratic program) were solved with CPLEX. All instances were solved on an Intel Core 2 2.4GHz CPU, with a CPU time limit of 8 hours

Nonlinear and SOC Solvers: Table 1 shows the average performance of the two nonlinear formulations, methods (1) and (2), run on an Intel Core 2 2.4GHz CPU, with a CPU time limit of 8 hours. In the heading of the table, $\bar{z}^*$ is the average optimal solution value, "# sol" denotes the number of instances out of 10 that were solved to optimality within the time limit, $\bar{T}$ is the average CPU time required by the solver, and $\bar{N}$ is the average number of nodes in the branch and bound search tree.

TABLE 1
Computational Behavior of Nonlinear Formulations on SQUFL

			Original			Perspective		
m	n	$\bar{z}^*$	#sol	$\bar{T}$	$\bar{N}$	#sol	$\bar{T}$	$\bar{N}$
20	100	408.31	10	307	6165	10	18	37
20	150	508.38	10	807	7409	10	33	29
30	100	375.86	10	4704	67808	10	33	53
30	150	462.69	7	16607	96591	10	56	40

From this experiment, the strong positive influence of the perspective reformulation is apparent—instances that cannot be solved in 8 hours with the original formulation are solved in around a minute using the strong perspective reformulation. In [17], the commercial solvers DICOPT [18] and BARON [26] were unable to solve small instances without the reformulation technique. This demonstrates that commercial MINLP solvers have yet to implement the perspective reformulation technique.

LP Solvers: Table 2 shows the results of solving the SQUFL instances using the two different linear approximations to the problem. In the table, the average of all lower bounds (obtained by solving the MILP outer-approximation to optimality) is given in the column $\bar{z}_{lb}$ of the table. The average upper bound on z^* is given in the column $\bar{z}_{ub}$. The average CPU time ($\bar{T}$) and average number of nodes ($\bar{N}$) is also given in the table. The results demonstrate that strengthening the linear approximation of the nonlinear functions (the perspective cuts) significantly strengthens the formulation, as indicated by the significantly reduced number of nodes. Solving the root node linear relaxation with the piecewise linear approximation requires a significant time for both the strengthen and unstrengthened formulations, so the time improvements are not as great as in the nonlinear case.

TABLE 2
Computational Behavior of Linear Formulations on SQUFL

					Original		Perspective	
m	n	$\bar{z}^*$	$\bar{z}_{ub}$	$\bar{z}_{lb}$	$\bar{T}$	$\bar{N}$	$\bar{T}$	$\bar{N}$
20	100	408.31	410.88	373.79	247	491	28	4
20	150	508.38	510.58	449.42	658	510	183	3
30	100	375.86	378.45	335.58	346	510	171	3
30	150	462.69	466.76	389.3	948	475	582	4

The lower bound provided by the solution to the approximating MILP is on average about 10% below the optimal solution value, and the upper bound found by fixing the integer variables is typically quite close to the true optimal solution. In order to reduce the gap between lower and upper bounds in this heuristic approach, a finer approximation of the nonlinear function can be created by increasing $|B|$. Table 3 shows the results of an experiment where the instances were approximated with more linear inequalities, and the strengthened (perspective) reformulation was solved. In the table, $\bar{G}(\%) = 100(\bar{z}_{ub} - \bar{z}_{lb})/\bar{z}_{ub}$ denotes the average gap between lower and upper bounds in the heuristic approach. For these instances, $|B| = 50$ breakpoints is typically sufficient to prove that the solution obtained is within 1% of optimality. Note, however, the increase in CPU time required to solve the linear relaxations.

6.2. Stochastic Service Design. Instances of the SSSD were randomly created following the suggestions of Elhedhli [9]. The demand rate of each customer λ_i was uniformly distributed between 0.5 and 1.5. The lowest service rate μ_{j1} was uniformly distributed between $(5|M|/8|N|, 7|M|/8|N|)$, and for the $|K| = 3$ service levels, $\mu_{j2} = 2\mu_{j1}$, and $\mu_{j3} = 3\mu_{j1}$. The fixed cost for the lowest service level was uniformly distributed between 250 and 500, and to induce economies of scale, if $u_j = f_{j1}/\mu_{j1}$, then fixed costs for the higher service levels were set to $f_{j2} = u_j^{2/3}\mu_{j2}$ and $f_{j3} = u_j^{1/2}\mu_{j3}$. We use $t = 100$ for the queueing weight delay parameter, as these instances

TABLE 3
Impact of Number of Piecewise Linear Approximation on Gap and Solution Time

m	n	$\|B\|$	$\bar{G}(\%)$	$\bar{T}$	$\bar{N}$
20	100	10	9.12%	28	4
20	100	25	1.23%	122	2
20	100	50	0.31%	367	3
20	150	10	11.98%	183	3
20	150	25	1.45%	841	6
20	150	50	0.41%	2338	6
30	100	10	11.32%	171	3
30	100	25	1.35%	1000	9
30	100	50	0.39%	1877	5
30	150	10	16.6%	582	4
30	150	25	2.09%	1433	6
30	150	50	0.48%	3419	6

appear to be the most difficult computationally in the work of Elhedhli. [9]. Instances with $|M| = 15, 20, 25$ and $|N| = 4, 8$ were created. The number of breakpoints used in the linearization for each instance was $|B| = 10$.

Each instance was solved six times with the following combination of formulation and software:

1. The original MINLP was solved with Bonmin (v0.9);
2. The perspective strengthened MINLP was solved with Bonmin;
3. The original instance was formulated using CQI and solved with Mosek (v5.0);
4. The perspective strengthened instance was formulated using CQI and solved with Mosek;
5. The linear under-approximation, not strengthened with perspective cuts, was solved with the CPLEX (v11.1) MILP solver. After fixing the integer variables to the solution of this problem, the continuous NLP problem was solved with IPOPT (v3.4);
6. The linear under-approximation, strengthened with perspective cuts, was solved with the CPLEX MILP solver. After fixing the integer variables to the solution of this problem, the continuous NLP problem was solved with IPOPT (v3.4).

NLP Solvers: Table 4 shows the results solving each instance, with and without the perspective strengthening, using the MINLP solver Bonmin. Bonmin uses the interior-point-based solver IPOPT to solve nonlinear relaxations. The table lists the optimal solution value (z^*) (or bounds on the best optimal solution), the CPU time required (T) in seconds, and the number of nodes evaluated ($\#N$). A time limit of 4 hours was imposed.

In all cases, the NLP solver IPOPT failed at a node of the branch and bound tree with the message "`Error: Ipopt exited with error Restoration failed`." For this instance, the NLP relaxation of SSSD

TABLE 4
Bonmin Performance on SSSD Instances

		Without Perspective			With Perspective		
$\|M\|$	$\|N\|$	z^*	T (sec.)	$\#N$	z^*	T (sec.)	$\#N$
15	4	5.76	1161	21357	Failed after 236 nodes		
15	8	(9.37,9.41)	14400	224500	Failed after 91 nodes		
20	4	3.71	282	6342	Failed after 786 nodes		
20	8	(4.391,4.393)	14400	188987	Failed after 144 nodes		
25	4	2.98	238	3914	Failed after 235 nodes		
25	8	Failed after 2468 nodes			Failed after 85 nodes		

(especially the perspective-enhanced NLP relaxation) appears difficult to solve. We performed a small experiment designed to test the impact of the formulation and NLP software. In this experiment, four different nonlinear formulations of the perspective constraints were used, and the root node NLP relaxation was solved by three different NLP packages: Ipopt (v3.4), Conopt (v3.14S), and SNOPT (v7.2-4). The root relaxation was also solved by Mosek (using the conic formulation) to obtain the true optimal solution value. The four different formulations of the perspective strengthening of the nonlinear constraints (4.17) were the following:

$$z_{jk}y_{jk} - z_{jk}v_j - v_j y_{jk} \leq 0, \tag{F1}$$

$$z_{jk} - \frac{v_j}{1 + v_j/y_{jk}} \leq 0, \tag{F2}$$

$$v_j^2 - (v_j - z_{jk})(v_j + y_{jk}) \leq 0, \tag{F3}$$

$$\sqrt{4v_j^2 + (y_{jk} + z_{jk})^2} - 2v_j + y_{jk} - z_{jk} \leq 0. \tag{F4}$$

In all cases, an initial iterate of

$$y_{jk} = 1/|K|, x_{ij} = 1/|J|, v_j = \frac{\sum_i \lambda_i x_{ij}}{\sum_k \mu_{jk} y_{jk} - \sum_i \lambda_i x_{ij}}, z_{jk} = \frac{v_j y_{jk}}{(1 + v_j)}$$

was used. Ten random instance of size $|M| = 100$, $|N| = 40$ were solved, and Table 5 shows the number of instances for which the root node was correctly solved to (global) optimality for each formulation and software package. The results of this experiment indicate that modeling conic (perspective) inequalities in their convex form (F4) has an appreciably positive impact on the solution quality. The perspective results for Bonmin in Table 4 were obtained with the formulation ($F4$), so even using the "best" formulation for the NLP solver was not sufficient to ensure the correct solution to the instance.

SOCP Solvers: Table 6 shows the results of solving the SSSD instances with the Mosek software (methods (3) and (4)). In some cases, using the perspective reformulation has a very positive impact, while in other

TABLE 5
Number of Successful SSSD Relaxation Solutions (out of 10)

	Formulation			
Solver	F1	F2	F3	F4
Ipopt	0	10	10	10
Conopt	0	0	10	10
SNOPT	0	3	0	7

cases, the difference is minor. It is interesting to note the difference in behavior between Bonmin and Mosek without perspective strengthening. For example, Bonmin solved the $|M| = 15, |N| = 4$ instance in 21357 nodes, while Mosek does not solve this instance in more than 1.9 million nodes. For these SSSD instances, Bonmin is able to add strong valid inequalities to improve performance, while Mosek does not these inequalities.

TABLE 6
Mosek Performance on SSSD Instances

		Without Perspective			With Perspective		
$\|M\|$	$\|N\|$	z^*	T (sec.)	$\#N$	z^*	T (sec.)	$\#N$
15	4	(4.88,5.76)	14400	1951983	5.76	250	26454
15	8	(8.53,9.41)	14400	2540284	(9.38,9.41)	14400	2060281
20	4	(2.77,3.71)	14400	3444732	3.71	52	12806
20	8	(4.305,4.393)	14400	2129261	(4.391,4.393)	14400	1501679
25	4	2.98	46	10128	2.98	19	3045
25	8	(6.143,6.146)	14400	1210722	(6.143,6.146)	14400	1182577

LP Solvers: Table 7 shows the results on the SSSD obtained when solving the linearized instances with CPLEX (methods (5) and (6)). In the table z_{lb} is the lower bound obtained by solving the MILP to optimality, and the time (T) and number of nodes ($\#N$) are also given for the search. The value z_{ub} is obtained by fixing the (optimal) integer solution and solving the NLP with Ipopt. The time require to solve the NLP is negligible. The results for the linear case are interesting. Specifically, strengthening the root relaxation by adding perspective cuts does not appear to help the computations in this case. For these instances, CPLEX is able to significantly improve the gap at the root node by adding its own cutting planes. Table 8 demonstrates the improvement in root lower bound value between the initial solve and final processing for both the strengthened and unstrengthened linear approximation. Note also the case $|M| = 25$, $|N| = 4$, where the solution obtained after fixing integer variables is far from optimal.

TABLE 7
Linear/CPLEX Performance on SSSD Instances

		Without Perspective				With Perspective			
$\|M\|$	$\|N\|$	z_{lb}	z_{ub}	T	$\#N$	z_{lb}	z_{ub}	T	$\#N$
15	4	5.75	5.76	1.1	7586	5.75	5.76	1.1	7755
15	8	9.11	9.71	242	1320484	9.11	9.71	818	4464252
20	4	3.40	7.07	0.7	908	3.4	7.07	0.3	1517
20	8	4.368	4.405	810	5589674	4.368	4.405	933	5930870
25	4	2.64	17.2	0.7	486	2.64	17.2	0.8	541
25	8	6.13	6.16	374	1753143	6.13	6.16	599	2929388

TABLE 8
Initial and Final CPLEX Root Lower Bounds

		Without Perspective		With Perspective	
$\|M\|$	$\|N\|$	Initial	Final	Initial	Final
15	4	1/74	3.66	4.02	4.21
15	8	2.06	5.68	6.11	6.41
20	4	1.33	1.87	1.89	2.13
20	8	1.55	2.41	2.86	2.99
25	4	1.05	1.45	1.45	1.54
25	8	2.15	3.53	3.97	4.20

7. Conclusions. The perspective reformulation is a tool to create strong relaxations of convex mixed integer nonlinear programs that have 0-1 variables to indicate special on-off logical relationships. The perspective reformulation can be derived as a special case of theorems of convex analysis or via techniques more familiar to MILP researchers: extended formulations, projection, and convex hulls of "simple" sets.

Many applications of MINLP could take advantage of this reformulation technique. In the applications described in this survey, the reformulated inequalities can be cast as second-order cone constraints, a transformation that can improve an instance's solvability.

We hope this survey has achieved its goals of introducing a wider audience to the perspective reformulation technique, incenting software developers to consider automatic recognition of the structures requires for the perspective reformulation, and spurring the research community to investigate additional simple sets occurring in practical MINLPs in the hope of deriving strong relaxations.

Acknowledgement. The authors would like to thank Jon Lee and Sven Leyffer for organizing the very fruitful meeting on MINLP at the Institute for Mathematics and its Applications (IMA) in November, 2008. The comments of Kevin Furman and Nick Sawaya were also helpful in

preparing Section 2.2.

REFERENCES

[1] S. Aktürk, A. Atamtürk, and S. Gürel, *A strong conic quadratic reformulation for machine-job assignment with controllable processing times*, Tech. Rep. BCOL Research Report 07.01, Industrial Engineering & Operations Research, University of California, Berkeley, April 2007.

[2] E. Balas, *Disjunctive programming and a hierarchy of relaxations for discrete optimization problems*, SIAM Journal on Algebraic and Discrete Methods, 6 (1985), pp. 466–486.

[3] A. Ben-Tal and A. Nemirovski, *Lectures on Modern Convex Optimization*, SIAM, 2001. MPS/SIAM Series on Optimization.

[4] D. Bertsekas and R. Gallager, *Data Networks*, Prentice-Hall, Endlewood Cliffs, NJ, 1987.

[5] P. Bonami, G. Cornuéjols, and H. Hijazi, *Mixed integer non-linear programs with on/off constraints: Convex analysis and applications*, 2009. Poster Presentation at MIP 2009 Conference.

[6] R. Boorstyn and H. Frank, *Large-scale network topological optimization*, IEEE Transactions on Communications, 25 (1977), pp. 29–47.

[7] B. Borchers and J. E. Mitchell, *An improved branch and bound algorithm for mixed integer nonlinear programs*, Computers & Operations Research, 21 (1994), pp. 359–368.

[8] S. Ceria and J. Soares, *Convex programming for disjunctive optimization*, Mathematical Programming, 86 (1999), pp. 595–614.

[9] S. Elhedhli, *Service system design with immobile servers, stochastic demand, and congestion*, Manufacturing & Service Operations Management, 8 (2006), pp. 92–97.

[10] A. Frangioni and C. Gentile, *Perspective cuts for a class of convex 0-1 mixed integer programs*, Mathematical Programming, 106 (2006), pp. 225–236.

[11] ———, *Sdp diagonalizations and perspective cuts for a class of nonseparable miqp*, Operations Research Letters, 35 (2007), pp. 181–185.

[12] ———, *A computational comparison of reformulations of the perspective relaxation: SOCP vs. cutting planes*, Operations Research Letters, (2009). To appear.

[13] K. Furman, I. Grossmann, and N. Sawaya, *An exact MINLP formulation for nonlinear disjunctive programs based on the convex hull*, 2009. Presentation at 20th International Symposium on Mathematical Programming.

[14] I. Grossmann and S. Lee, *Generalized convex disjunctive programming: Nonlinear convex hull relaxation*, Computational Optimization and Applications, (2003), pp. 83–100.

[15] O. Günlük, J. Lee, and R. Weismantel, *MINLP strengthening for separaable convex quadratic transportation-cost ufl*, Tech. Rep. RC24213 (W0703-042), IBM Research Division, March 2007.

[16] O. Günlük and J. Linderoth, *Perspective relaxation of mixed integer nonlinear programs with indicator variables*, Tech. Rep. RC24694 (W0811-076), IBM Research Division, November 2008.

[17] ———, *Perspective relaxation of mixed integer nonlinear programs with indicator variables*, Mathematical Programming Series B, (2009). To appear.

[18] G. R. Kocis and I. E. Grossmann, *Computational experience with DICOPT solving MINLP problems in process systems engineering*, Computers and Chemical Engineering, 13 (1989), pp. 307–315.

[19] S. Leyffer and J. Linderoth, *A practical guide to mixed integer nonlinear programming*, 2005. Short Course offered at SIAM Optimization Conference.

[20] H. M. Markowitz, *Portfolio selection*, Journal of Finance, 7 (1952), pp. 77–91.

[21] *The mosek optimization tools manual. version 5.0 (revision 84)*, 2008. `www.mosek.com`.

[22] A. F. Perold, *Large-scale portfolio optimization*, Management Science, 30 (1984), pp. 1143–1160.

[23] R. Stubbs and S. Mehrotra, *A branch-and-cut method for 0-1 mixed convex programming*, Mathematical Programming, 86 (1999), pp. 515–532.

[24] R. A. Stubbs, *Branch-and-Cut Methods for Mixed 0-1 Convex Programming*, PhD thesis, Northwestern University, December 1996.

[25] J. F. Sturm, *Using SeDuMi 1.02, a MATLAB toolbox for optimization over symmetric cones*, Optimization Methods and Software, 11-12 (1999), pp. 625–653.

[26] M. Tawarmalani and N. V. Sahinidis, *Global optimization of mixed integer nonlinear programs: A theoretical and computational study*, Mathematical Programming, 99 (2004), pp. 563–591.