Computer Sciences Department

Effective Separation of Disjunctive Cuts for Convex Mixed Integer Nonlinear Programs

Mustafa Kilinc
Jeff Linderoth
James Luedtke

Technical Report #1681

November 2010

UNIVERSITY OF WISCONSIN MADISON

Effective Separation of Disjunctive Cuts for Convex Mixed Integer Nonlinear Programs*

Mustafa Kılınç† Jeff Linderoth‡

James Luedtke§
Department of Industrial and Systems Engineering
University of Wisconsin-Madison

August 21,2010

Abstract

We describe a computationally effective method for generating disjunctive inequalities for convex mixed-integer nonlinear programs (MINLPs). The method relies on solving a sequence of cut-generating linear programs, and in the limit will generate an inequality as strong as can be produced by the cut-generating nonlinear program suggested by Stubbs and Mehrotra. Using this procedure, we are able to approximately optimize over the rank one simple disjunctive closure for a wide range of convex MINLP instances. The results indicate that disjunctive inequalities have the potential to close a significant portion of the integrality gap for convex MINLPs. In addition, we find that using this procedure within a branch-and-cut solver for convex MINLPs yields significant savings in total solution time for many instances. Overall, these results suggest that with an effective separation routine, like the one proposed here, disjunctive inequalities may be as effective for solving convex MINLPs as they have been for solving mixed-integer linear programs.

1 Introduction

The focus of this work is on the effective generation of disjunctive cutting planes for convex mixed-integer nonlinear programs. A mixed integer nonlinear pro-

*This research was supported in part by the U.S. Department of Energy under Grant DE-FG02-08ER25861 and the National Science Foundation under Grant CCF-0830153

†kilinc@wisc.edu

‡linderoth@wisc.edu

§jrluedt1@wisc.edu

gram is the optimization problem

$$\begin{aligned} z_{\text{MINLP}} = \text{minimize} \quad & c^T x \\ \text{subject to} \quad & g_j(x) \leq 0 \quad \forall j \in J, \\ & x \in X, \quad x_I \in \mathbb{Z}^{|I|}, \end{aligned} \tag{MINLP}$$

where J is the index set of nonlinear constraints, and $I \subseteq N \stackrel{\text{def}}{=} \{1, \ldots, n\}$ is the index set of discrete variables. The set $X = \{x \in \mathbb{R}^n \mid Ax \leq b\}$ is a polyhedral subset of $\mathbb{R}^n$. We define $g(x) : \mathbb{R}^n \to \mathbb{R}^{|J|}$ as the vector-valued function $g(x) = (g_1(x), g_2(x), \ldots, g_{|J|}(x))^T$ and $\nabla g(x)^T \in \mathbb{R}^{|J| \times n}$ as the Jacobian of g.

We focus on the case where the functions g_j are differentiable and convex, so by relaxing the constraints $x_I \in \mathbb{Z}^{|I|}$, a smooth convex program is formed. The fact that (MINLP) has a linear objective function is an important assumption in our work. Our aim will be to add valid linear inequalities that exclude the solution of a relaxation to (MINLP) from the feasible region. Without a linear objective function, the minimizer of the relaxation may lie in the strict interior of the feasible region, and thus may not be cut off using linear inequalities. However, there is no loss of generality by considering a linear objective function. A (convex) nonlinear objective function $f(x)$ can be included with the addition of an auxiliary variable η, changing the objective to minimize η and adding the constraint $f(x) - \eta \leq 0$.

If $J = \emptyset$, then (MINLP) is a mixed integer linear program (MILP), and over the past decades, algorithms for solving MILPs have improved by orders of magnitude. A crucial ingredient in the improvement of MILP software has been cutting planes. The reader is referred to [15] for a recent survey on cutting planes for MILP.

Research on the effective generation and use of cutting planes for MINLP is far less advanced, though some authors have shown that many common MILP cutting planes can be extended for use in nonlinear settings. For example, Çezik and Iyengar show how the classic Gomory cut [23] can be extended to the case of mixed integer second-order cone programs (MISOCP), which are (MINLP) where $g_j(x) = \|Hx - f\|_2 - r^T x - c \; \forall j \in J$. Atamtürk and Narayan [2] extended mixed integer rounding cuts [29] to the case of MISOCP.

Our work follows closely along the lines of Stubbs and Mehrotra [34], who demonstrated the applicability of the disjunctive inequalities of Balas [3] to the realm of convex MINLP. Like Stubbs and Mehrotra [34], we will generate disjunctive cuts only for simple, single variable, disjunctions of the form $(x_i \leq k \vee x_i \geq k + 1)$ for some $i \in I$. The ideas presented here can be extended to more general disjunctions as well.

For MILP, disjunctive cutting planes were pioneered by Balas, Ceria, and Cornuéjols [5]. A limitation of disjunctive inequalities is that in order to generate a valid cut, one must solve an auxiliary (separation) problem that is two times the size of the original relaxation. In the case of MILP, clever intuition of Balas and Perregaard [8] allows this separation problem to be solved in the original space of variables. No such extension is known in the case of MINLP.

Thus, the separation procedure of Stubbs and Mehrotra is quite computationally expensive. Stubbs and Mehrotra [34] report computational results only on four instances, the largest of which has $n = 30$ variables. Further, they report numerical difficulties in generating the disjunctive inequalities using the separation procedure that relies on solving a (nondifferentiable) nonlinear program (NLP).

An implementation of the Stubbs and Mehrotra procedure for the case of MISOCP appears in the Ph.D. thesis of Drewes [18]. Stubbs and Mehrotra [33] have also explored methods for generating convex polynomial cuts for mixed 0-1 convex programs. Zhu and Kuno [36] have suggested to replace the nonlinear feasible region of the separation problem of Stubbs and Mehrotra by a linear approximation taken about the solution to a specific relaxation.

In this work, we introduce a new separation procedure for generating disjunctive inequalities for convex MINLPs. Our method solves a sequence of cut-generating linear programs rather than solving a convex NLP in a higher dimensional space as suggested in [34]. The first iteration of our procedure yields a disjunctive inequality similar to what would be obtained by the procedure of Zhu and Kuno [36]. However, the disjunctive cut we obtain is improved at each iteration by strategically adding more linear structure derived from the nonlinear constraints of (MINLP). We demonstrate that the inequalities generated from our procedure are as strong as the disjunctive inequalities of Stubbs and Mehrotra [34] in the limit. A key advantage of our approach is that it produces a valid inequality at every iteration, so that we can terminate early in the event of "tailing off" in the cut generation procedure. We thus have a procedure that can effectively exploit the middle ground between the procedure of [36] which is computationally cheap but may generate weak inequalities, and the approach of [34] which can generate all possible disjunctive inequalities, but is computationally expensive.

A recent trend in mixed-integer linear programming has been to (approximately) optimize over the relaxation obtained by using all possible valid inequalities from a given class, referred to as a *closure*. This line of work began with Fischetti and Lodi [21] who optimized a broad class of instances over the *Chvátal-Gomory closure*, the class of all inequalities that can be obtained using the Chvátal-Gomory rounding procedure; see also [10] for the *lift-and-project* closure and [9, 16] for other MILP closure studies. These studies provide valuable insights into the relaxation strength that is possible using the associated classes of inequalities. Following these examples, we use our disjunctive cut separation procedure to approximately optimize a set of test instances over the *rank one disjunctive cut closure*, which includes all valid inequalities that can be obtained from single variable disjunctions. Only constraints in the original problem description are used to generate the disjunctive inequalities making up the closure, thus the name "rank one." We find that on many instances in our test set the rank one disjunctive cut closure reduces significant portion of the relaxation gap; on average 69% over our test set of 207 convex MINLP instances, and by as much as 100%.

Encouraged by these closure results, we incorporated our disjunctive cut sep-

aration procedure into FilMINT, a linearization-based solver for convex MINLPs. In this preliminary implementation, relatively simple strategies were used to limit the computational effort expended in generating the cuts. This disjunctive cut separation procedure enabled the solution of several instances that previously could not be solved in a three hour time limit by FilMINT and significantly reduced the solution time on many other instances.

The remainder of the paper is organized as follows. In §2, we give an overview of the original disjunctive cutting plane method of Stubbs and Mehrotra [34] for convex MINLP. We describe our new method for generating disjunctive inequalities and prove its equivalence to the approach of [34] in §3. Implementation details are described in §4. In §5, we give the disjunctive cut closure results and computational results on a broad suite of convex MINLPs are presented in §6. Conclusions are offered in §7.

2 Disjunctive Inequalities for MINLPs

In this section, we briefly review the basic theory on disjunctive inequalities for convex MINLPs. For more details, the interested reader is referred to [34]. We denote the feasible region of the continuous relaxation of (MINLP) as

$$R = \{x \in X \mid g(x) \leq 0\}.$$

Disjunctive inequalities are linear inequalities that are feasible for the convex hull of the union of two sets. Specifically, for a discrete variable $x_i, (i \in I)$, consider the two sets

$$R_i^{k-} = \{x \in R \mid x_i \leq k\} \quad \text{and} \quad R_i^{k+} = \{x \in R \mid x_i \geq k+1\},$$

for $k \in \mathbb{Z}$. Simple disjunctive inequalities are valid inequalities for the set

$$\mathcal{R}_i^k = \text{conv}(R_i^{k-} \cup R_i^{k+}).$$

Since $\mathcal{R}_i^k$ is a relaxation of the feasible region to (MINLP), valid inequalities for $\mathcal{R}_i^k$ are also valid for (MINLP). There is an extended formulation of $\mathcal{R}_i^k$ in terms of convex, nonlinear, inequalities using a variable transformation and the *perspective function*. Define the perspective function $\tilde{h}(\tilde{x}, \lambda)$ [26] related to a given convex function $h(x) : C \subset \mathbb{R}^n \to \mathbb{R}$ as follows:

$$\tilde{h}(\tilde{x}, \lambda) = \begin{cases} \lambda h(\tilde{x}/\lambda) & \text{if } \tilde{x}/\lambda \in C, \lambda > 0, \\ 0 & \text{if } \lambda = 0, \\ \infty & \text{otherwise.} \end{cases}.$$

Note that $\tilde{h}(\tilde{x}, \lambda)$ may be a nondifferentiable function, even if h itself is differentiable.

Stubbs and Mehrotra [34] (see also [14]) showed that the convex set

$$\tilde{\mathcal{M}}_i^k = \left\{ (x, \tilde{y}, \tilde{z}, \lambda, \mu) \;\middle|\; \begin{array}{ll} x = \tilde{y} + \tilde{z}, & \lambda + \mu = 1, \\ \tilde{g}(\tilde{y}, \lambda) \le 0, & \tilde{g}(\tilde{z}, \mu) \le 0 \\ A\tilde{y} \le \lambda b, & A\tilde{z} \le \mu b, \\ \tilde{y}_i \le \lambda k, & \tilde{z}_i \ge \mu k + \mu, \\ \lambda \ge 0, & \mu \ge 0 \end{array} \right\}$$

provides an extended formulation for $\mathcal{R}_i^k$. In other words,

$$\text{proj}_x(\tilde{\mathcal{M}}_i^k) = \{x \mid (x, \tilde{y}, \tilde{z}, \lambda, \mu) \in \tilde{\mathcal{M}}_i^k\} = \text{conv}(R_i^{k-} \cup R_i^{k+}).$$

Given a point $\bar{x} \notin \text{conv}(R_i^{k-} \cup R_i^{k+})$, we can find a linear inequality separating $\bar{x}$ from $\text{conv}(R_i^{k-} \cup R_i^{k+})$ by minimizing the distance $d(x) = \|x - \bar{x}\|$ between the point and the set, in any norm $\|\cdot\|$. Specifically, consider the minimization problem

$$d_{\tilde{\mathcal{M}}_i^k}(\bar{x}) = \min_{(x,\tilde{y},\tilde{z},\lambda,\mu) \in \tilde{\mathcal{M}}_i^k} d(x). \tag{1}$$

The following theorem states that a disjunctive inequality may be obtained using the subgradient of $d(\cdot)$ at an optimal solution to (1).

Theorem 1 ([34]). *Let $\bar{x} \notin \text{conv}\,(R_i^{k-} \cup R_i^{k+})$, x^* be an optimal solution of (1), and let $\xi \in \partial d(x^*)$. Then, $\xi^T(\bar{x} - x^*) < 0$ and $\xi^T(x - x^*) \ge 0$ $\forall x \in \text{conv}\,(R_i^{k-} \cup R_i^{k+})$.*

The downside of using (1) to generate inequalities is twofold. First, one must solve a nonlinear program that is twice the size of the original problem in order to generate a valid inequality. Second, the description of the set $\tilde{\mathcal{M}}_i^k$ contains nondifferentiable functions, so (as also noted by Stubbs and Mehrotra [34]), numerical difficulties in nonlinear programming software designed for differentiable functions may lead to the generation of invalid inequalities. In the next section, we aim to address these issues by replacing nonlinear inequalities with linearizations, obtaining a computationally viable method for generating disjunctive inequalities.

3 Effective Disjunctive Cut Generation for MINLP

Our idea is to avoid solving the difficult nonlinear program (1) by instead solving a sequence of linear programs that approximates (1). Let $B \supseteq R$ be a relaxation of the original relaxation feasible region R, and let

$$B_i^{k-} = \{x \in B \mid x_i \le k\} \quad \text{and} \quad B_i^{k+} = \{x \in B \mid x_i \ge k+1\}.$$

Inequalities valid for $\text{conv}(B_i^{k-} \cup B_i^{k+})$ are also valid for $\text{conv}\,(R_i^{k-} \cup R_i^{k+})$. Further, if B is restricted to be a polyhedron, and an appropriate norm is used in the definition of $d(\cdot)$, then the separation problem (1) reduces to a linear

program (LP). Zhu and Kuno [36] propose to replace the nonlinear inequalities in R with their linearizations at $\bar{x}$ to a get polyhedral relaxation B. Their preliminary results show that their method is effective for small test problems. Note that if the solution to the continuous relaxation of MINLP lies in the convex hull of the union of the two relaxed sets B_i^{k-} and B_i^{k+} ($\bar{x} \in \text{conv}(B_i^{k-} \cup B_i^{k+})$, then it cannot be separated with a linear inequality. This implies that in order to be able to separate more points, one should seek to obtain tight relaxations of the sets R_i^{k-} and R_i^{k+}. In what follows, we demonstrate that by iteratively updating polyhedral outer approximations of R_i^{k-} and R_i^{k+} in an appropriate manner, one may obtain a disjunctive inequality of the same strength as if one directly solved the nonlinear separation problem (1).

The method relies on generating polyhedral relaxations of the sets R_i^{k-} and R_i^{k+} iteratively. At iteration t, we define two finite sets of points $\mathcal{K}_-^t, \mathcal{K}_+^t \subset \mathbb{R}^n$ and linearize the nonlinear functions at these points, creating the sets

$$\mathcal{F}_-^t \stackrel{\text{def}}{=} \left\{ x \in X \mid x_i \leq k, \quad g(\bar{x}) + \nabla g(\bar{x})^T (x - \bar{x}) \leq 0 \ \forall \bar{x} \in \mathcal{K}_-^t \right\} \text{ and}$$
$$\mathcal{F}_+^t \stackrel{\text{def}}{=} \left\{ x \in X \mid x_i \geq k+1, \quad g(\bar{x}) + \nabla g(\bar{x})^T (x - \bar{x}) \leq 0 \ \forall \bar{x} \in \mathcal{K}_+^t \right\}.$$

By construction, $R_i^{k-} \subseteq \mathcal{F}_-^t$ and $R_i^{k+} \subseteq \mathcal{F}_+^t \ \forall t$. (For notational convenience, we suppress the dependence on the variable index i and branching value k in the notation of the relaxations $\mathcal{F}_-^t$ and $\mathcal{F}_+^t$.)

Since the sets $\mathcal{F}_-^t$ and $\mathcal{F}_+^t$ are polyhedral, the theory of disjunctive programming for the union of polyhedral sets [3] yields the following extended formulation for $\text{conv}(\mathcal{F}_-^t \cup \mathcal{F}_+^t)$:

$$\tilde{\mathcal{M}}^t = \left\{ (x, \tilde{y}, \tilde{z}, \lambda, \mu) \;\middle|\; \begin{array}{ll} x = \tilde{y} + \tilde{z}, & \lambda + \mu = 1 \\ \lambda g(\bar{x}) + \nabla g(\bar{x})^T (\tilde{y} & -\lambda \bar{x}) \leq 0 \ \forall \bar{x} \in \mathcal{K}_-^t \\ \mu g(\bar{x}) + \nabla g(\bar{x})^T (\tilde{z} & -\mu \bar{x}) \leq 0 \ \forall \bar{x} \in \mathcal{K}_+^t \\ A\tilde{y} \leq \lambda b, & A\tilde{z} \leq \mu b \\ \tilde{y}_i \leq \lambda k, & \tilde{z}_i \geq \mu k + \mu \\ \lambda \geq 0, & \mu \geq 0 \end{array} \right\}.$$

Note that all inequalities describing $\tilde{\mathcal{M}}^t$ are linear, so if $d(x) = \|x - \bar{x}\|_1$ or $d(x) = \|x - \bar{x}\|_\infty$, then the separation problem

$$d_{MDP(t)}(\bar{x}) = \min_{(x^t, \tilde{y}^t, \tilde{z}^t, \lambda^t, \mu^t) \in \tilde{\mathcal{M}}^t} d(x^t) \qquad \text{(MDP}(t)\text{)}$$

may be written as linear program. Using the solution x^* to this linear program, a linear inequality valid for $\text{conv}(\mathcal{F}_-^t \cup \mathcal{F}_+^t)$ may be derived using Theorem 1.

The algorithm starts with $\mathcal{K}_-^0 = \mathcal{K}_+^0 = \emptyset$ and augments the sets based on the solution of the linear program (MDP(t)). For stating the algorithm and proving its convergence, we will make the standard assumptions that the set X is compact and that either R_i^{k-} or R_i^{k+} is non-empty. (These assumptions ensure that (MDP(t)) always has an optimal solution). At iteration t, the separation problem (MDP(t)) is solved, yielding three points: $\tilde{y}^t, \tilde{z}^t$, and x^t. The point x^t

will converge to an optimal solution of (1), and this limit point will serve as the point x^* for generating the disjunctive inequality using Theorem 1. The points $\tilde{y}^t$ and $\tilde{z}^t$ are used to define the new points at which linearizations will be taken in the following manner. Observe that if $\lambda > 0$ and $\mu > 0$ and we define the points y^t and z^t by

$$y^t = \tilde{y}^t/\lambda^t,\ z^t = \tilde{z}^t/\mu^t,$$

then $y^t \in \mathcal{F}_-^t$ and $z^t \in \mathcal{F}_+^t$. If $y^t \in R_i^{k-}$ and $z^t \in R_i^{k+}$, then $x^t \in \text{conv}(R_i^{k-} \cup R_i^{k+})$ since $x^t = \tilde{y}^t + \tilde{z}^t = \lambda y^t + \mu z^t$ and hence we can terminate the cut separation procedure. On the other hand, if $y^t \notin R_i^{k-}$, then we add a linearization that excludes y^t from $\mathcal{F}_-^t$ by updating $\mathcal{K}_-^{t+1} \leftarrow \mathcal{K}_-^t \cup \{y^t\}$. Similarly, we update $\mathcal{K}_-^{t+1}$ if $z^t \notin R_i^{k+}$. Our separation procedure simply repeats this process; for concreteness, we provide pseudo-code of the method in Algorithm 1.

$x^0 \leftarrow \bar{x}$, $\mathcal{K}_-^1 \leftarrow \emptyset$, $\mathcal{K}_+^1 \leftarrow \emptyset$, $t \leftarrow 1$
repeat
  Solve MDP(t) for $(x^t, \tilde{y}^t, \tilde{z}^t, \lambda^t, \mu^t)$
  if $\lambda^t \neq 0$ **then**
    $y^t \leftarrow \tilde{y}^t/\lambda^t$
  else
    $y^t \leftarrow y^{t-1}$
  end if
  if $\mu^t \neq 0$ **then**
    $z^t \leftarrow \tilde{z}^t/\mu^t$
  else
    $z^t \leftarrow z^{t-1}$
  end if
  $\mathcal{K}_-^{t+1} \leftarrow \{y^t\} \cup \mathcal{K}_-^t$, $\mathcal{K}_+^{t+1} \leftarrow \{z^t\} \cup \mathcal{K}_+^t$, $t \leftarrow t+1$
until $\lambda^t g(y^t) + \mu^t g(z^t) \leq 0$

Algorithm 1: Iterative generation of MDP(t).

The quality of the cut generated by Algorithm 1 may be measured by the objective value of MDP(t), since $d_{\text{MDP}_{(t)}}(\bar{x})$ is the distance between $\bar{x}$, the point we want to separate, and the separating hyperplane. In particular, we successfully exclude the point $\bar{x}$ if and only if $d_{\text{MDP}_{(t)}}(\bar{x}) > 0$. Using this measure, the following theorem states that, in the limit, the cut generated by Algorithm 1 is as strong as the disjunctive cut proposed by Stubbs and Mehrotra [34]. Our proof of convergence of Algorithm 1 is in the spirit of the proof of Kelley's cutting plane procedure [27].

Theorem 2. *If (MDP(t)) is updated by the Algorithm 1, then*

$$\lim_{t\to\infty} d_{MDP(t)}(\bar{x}) \to d_{\tilde{\mathcal{M}}_i^k}(\bar{x})$$

Proof. By construction, $\mathcal{F}_-^{t+1} \subseteq \mathcal{F}_-^t$ and $\mathcal{F}_+^{t+1} \subseteq \mathcal{F}_+^t$ for $\forall t$ and thus $\{d_{MDP(t)}(\bar{x})\}$ forms a monotonically increasing sequence,

$$d_{MDP(1)}(\bar{x}) \leq d_{MDP(2)}(\bar{x}) \leq \ldots.$$

Since $R_i^{k-} \subseteq \mathcal{F}_t^-$ and $R_i^{k+} \subseteq \mathcal{F}_t^+$ for $\forall t$, then $\tilde{\mathcal{M}}_i^k \subseteq \tilde{\mathcal{M}}_t$ and so $d_{MDP(t)}(\bar{x}) \leq d_{\tilde{\mathcal{M}}_i^k}(\bar{x})$. Therefore, the sequence converges. The proof continues by showing that $\{x^t\}$ converges to a point in conv $(R_i^{k-} \cup R_i^{k+})$, so that $\{d_{MDP(t)}(\bar{x})\}$ converges to $d_{\tilde{\mathcal{M}}_i^k}(\bar{x})$.

To show the convergence of $\{x^t\}$ to a point in conv $(R_i^{k-} \cup R_i^{k+})$, let us consider the sequence $\{\lambda^t\}$. There are three cases to consider:
Case 1: For every N, there exists $t_1 > N$ such that $\lambda^{t_1} \neq 0$ and there exists $t_2 > N$ such that $\lambda^{t_2} \neq 1$.

First, we show $\{y^t\}$ converges to a point in R_i^{k-}. Consider the subsequence $\{y^{t_s}\}$ where $\{t_s\}$ corresponds to subset of indices such that $y^{t_s} \notin R_i^{k-}$ and $\lambda^{t_s} > 0$. If the subsequence $\{y^{t_s}\}$ is finite then clearly $\{y^t\}$ converges to a point in R_i^{k-}. Otherwise, for every y^{t_s} in $\{y^{t_s}\}$ there exists a $j \in J$ such that $g_j(y^{t_s}) > 0$ implying that the linearization of g at y^{t_s}

$$g_j(y^{t_s}) + \nabla g_j(y^{t_s})^T(y - y^{t_s}) \leq 0$$

is an inequality violated at y^{t_s}. Thus, y^{t_s} is excluded from $\mathcal{F}_-^{t_{s+1}}$. It follows that

$$y^{t_{s+1}} \neq y^{t_p}, p = 1, \ldots, s.$$

On the other hand, since $y^{t_s} \in \mathcal{F}_-^{t_s}$, it satisfies the inequalities

$$g(y^{t_p}) + \nabla g(y^{t_p})^T(y^{t_p} - y^{t_s}) \leq 0 \quad (p = 1, \ldots, s-1).$$

If $\{y^{t_s}\}$ converges to a point in R_i^{k-}, then $g(y^{t_s})$ must possess a subsequence that converges to a vector of nonpositive elements. If the desired convergence does not occur, then there exists an $\epsilon > 0$ and $j \in J$ such that

$$\epsilon \leq g_j(y^{t_p}) \leq \nabla g_j(y^{t_p})^T(y^{t_s} - y^{t_p}) \leq M\|y^{t_s} - y^{t_p}\| \quad (p = 1, \ldots, s-1)$$

where M is a finite constant with $M \geq \nabla g_j(x)$ for all $x \in X$. This sequence of inequalities implies that

$$\|y^{t_q} - y^{t_p}\| \geq \frac{\epsilon}{M} \quad (p > q)$$

which means that $\{y^{t_s}\}$ does not contain a Cauchy subsequence. However, this is impossible since the set X is compact. This completes the proof that $\{y^t\}$ converges to a point in R_i^{k-}.

Similarly, we can show that $\{z^t\}$ converges to a point in R_i^{k+}. Since $x^t = \lambda^t y^t + \mu^t z^t$ and $\lambda^t + \mu^t = 1$, then $\{x^t\}$ converges to a point in conv $(R_i^{k-} \cup R_i^{k+})$.
Case 2: There exists an N such that $\lambda^t = 1$ for $\forall t > N$.

By a similar construction to Case 1, we can show that $\{y^t\}$ converges to a point in R_i^{k-}. In this case, $\{\lambda^t\}$ converges to 1, so $\{\mu^t\}$ converges to 0. Since $x^t = \lambda^t y^t + \mu^t z^t$ and $\{y^t\}$ converges to a point in R_i^{k-}, then $\{x^t\}$ converges to a point in R_i^{k-} thus a point in $\text{conv}\,(R_i^{k-} \cup R_i^{k+})$.
Case 3: There exists an N such that $\lambda^t = 0$ for $\forall t > N$.
This is similar to Case 2, as in this case $\{\mu^t\}$ converges to 1. □ □

A consequence of this result is that if $\bar{x} \notin \text{conv}\,(R_i^{k-} \cup R_i^{k+})$, then the algorithm will find a valid inequality for $\text{conv}\,(R_i^{k-} \cup R_i^{k+})$ that cuts off $\bar{x}$ in finitely many iterations. Furthermore, in the limit, the valid inequality will be supported by a point in $\text{conv}\,(R_i^{k-} \cup R_i^{k+})$.

4 Implementation Details

We first introduce some notation to facilitate the discussion of how to implement our separation procedure. For the linearization sets $\mathcal{K}_-^t = \{y^1, y^2, \ldots, y^p\}$ and $\mathcal{K}_+^t = \{z^1, z^2, \ldots, z^q\}$, we define the matrices $A^- \in \mathbb{R}^{m^- \times n}$, $A^+ \in \mathbb{R}^{m^+ \times n}$, $b^- \in \mathbb{R}^{m^-}$, and $b^+ \in \mathbb{R}^{m^+}$ as

$$A^- = \begin{bmatrix} -A \\ -\nabla g(y^1)^T \\ -\nabla g(y^2)^T \\ \vdots \\ -\nabla g(y^p)^T \end{bmatrix}, \quad b^- = \begin{bmatrix} -b \\ g(y^1) - \nabla g(y^1)^T y^1 \\ g(y^2) - \nabla g(y^2)^T y^2 \\ \vdots \\ g(y^p) - \nabla g(y^p)^T y^p \end{bmatrix},$$

$$A^+ = \begin{bmatrix} -A \\ -\nabla g(z^1)^T \\ -\nabla g(z^2)^T \\ \vdots \\ -\nabla g(z^q)^T \end{bmatrix}, \quad b^+ = \begin{bmatrix} -b \\ g(z^1) - \nabla g(z^1)^T z^1 \\ g(z^2) - \nabla g(z^2)^T z^2 \\ \vdots \\ g(z^q) - \nabla g(z^q)^T z^q \end{bmatrix}.$$

When solving the separation problem MDP(t), any norm may be used in the objective function. However, if either the 1-norm or ∞-norm is used, then MDP(t) can be reformulated as a linear program. Our preliminary experiments indicated that using the ∞-norm was superior to the 1-norm, as the generated inequalities tend to be sparser with the ∞ norm. (A similar observation was also made in [34]). The ∞-norm objective function can be linearized by introducing a new variable θ to represent the value of the objective function and ensuring that

$$-\theta \leq x_j - \bar{x}_j \leq \theta, \ \forall j \in N.$$

Thus, with the ∞-norm, MDP(t) can be written as the linear program

$$
\begin{aligned}
\text{minimize} \quad & \theta \\
\text{subject to} \quad & -\theta \leq \tilde{y}_j + \tilde{z}_j - \bar{x}_j \leq \theta, && \forall j \in N \\
& A^- \tilde{y} - \lambda b^- \geq 0, && A^+ \tilde{z} - \mu b^+ \geq 0, \\
& -\tilde{y}_i + \lambda k \geq 0, && \tilde{z}_i - \mu(k+1) \geq 0, \\
& \lambda + \mu = 1, && \lambda \geq 0,\ \mu \geq 0,
\end{aligned}
\tag{MLP}
$$

where the substitution $x_j = \tilde{y}_j + \tilde{z}_j$ was performed.

From one iteration to the next, a new point is added to either $\mathcal{K}^t_-$ or $\mathcal{K}^t_+$, thus new constraints are added to (MLP) and the size of the basis of (MLP) increases when using simplex method. To avoid this, we instead solve the dual of (MLP):

$$
\begin{aligned}
\text{maximize} \quad & \beta - \bar{x}^T(\alpha^+ - \alpha^-) \\
\text{subject to} \quad & {A^-}^T u - u_0 e_i - (\alpha^+ - \alpha^-) = 0, \\
& {A^+}^T v + v_0 e_i - (\alpha^+ - \alpha^-) = 0, \\
& -u^T b^- + u_0 k + \beta \leq 0, \\
& -v^T b^+ - v_0(k+1) + \beta \leq 0, \\
& \sum_{j \in I \cup C} (\alpha^+_j + \alpha^-_j) = 1, \\
& \alpha^+ \geq 0,\ \alpha^- \geq 0,\ u \geq 0,\ v \geq 0,\ u_0 \geq 0,\ v_0 \geq 0,
\end{aligned}
\tag{CGLP}
$$

where e_i is i-th unit vector of appropriate size, $\alpha^+, \alpha^- \in \mathbb{R}^n$ are the dual variables for each side of the first constraint and $u \in \mathbb{R}^{m^-}$,$v \in \mathbb{R}^{m^+}$,$u_0, v_0, \beta \in \mathbb{R}$ are the dual variables for the remaining constraints in (MLP), respectively. In this case, whenever a new point is added to either $\mathcal{K}^t_-$ or $\mathcal{K}^t_+$, new columns are added to (CGLP), so the size of the basis remains the same. Further, (CGLP) can be effectively re-solved with the primal simplex method, since primal feasibility is conserved.

The linear program (CGLP) is well-studied in the literature [5, 22]. It is convenient to note that we can read the cut directly from the optimal solution of the (CGLP). Specifically, the dual variables $\alpha = \alpha^+ - \alpha^-$ give the cut coefficients, and β is right hand side of the cut—$\alpha^T x \geq \beta$. The objective of the (CGLP) is to maximize the violation of the point $\bar{x}$.

Multiplying cut coefficients and right hand side of the cut with any positive number will not change the actual hyperplane that defines the cut, but does scale the objective value in (CGLP). Thus, a normalization constraint is required to avoid having an unbounded problem in the case that $\bar{x}$ can be cut off. The constraint $\sum(\alpha^+_j + \alpha^-_j) = 1$ provides one such normalization.

A more commonly used normalization, first proposed in [4] and called the *standard normalization condition*, is defined as $\sum u_j + \sum v_j + u_0 + v_0 = 1$ which restricts the multipliers in (CGLP). This is equivalent to relaxing the

original constraints in the (MLP). Our method can also be adopted for this normalization. To do so, we modify the separation problem (MLP) as follows

$$\begin{aligned}
\text{minimize} \quad & \theta \\
\text{subject to} \quad & \tilde{y} + \tilde{z} = \bar{x} \\
& A^- \tilde{y} - \lambda b^- + \mathbf{1} \cdot \theta \geq 0, \\
& A^+ \tilde{z} - \mu b^+ + \mathbf{1} \cdot \theta \geq 0, \\
& -\tilde{y}_i + \lambda k + \theta \geq 0 \\
& \tilde{z}_i - \mu(k+1) + \theta \geq 0, \\
& \lambda + \mu = 1, \\
& \lambda \geq 0, \quad \mu \geq 0,
\end{aligned} \tag{2}$$

where $\mathbf{1}$ is the appropriate size vector of all ones. The dual of this linear program yields the linear program (CGLP) with the normalization constraint replaced by the standard normalization condition, $\sum u_j + \sum v_j + u_0 + v_0 = 1$. Algorithm 1 can also be applied to iterative generation of (2) with stopping criteria $\lambda^t g(y^t) \leq \mathbf{1} \cdot \theta^t$ and $\mu^t g(z^t) \leq \mathbf{1} \cdot \theta^t$ where θ^t is the optimal solution value of (2) at iteration t.

In our experiments, we used the ∞-norm and optimized (CGLP) with a column-generation method. Our method works by iteratively updating a polyhedral outer approximation of convex hull of the feasible points. Initially, the outer approximation is

$$C^0 = \{x \in X \mid g(x_R) + \nabla g(x_R)^T (x - x_R) \leq 0\}$$

where z_R is the objective value and x_R is the optimal solution of the continuous nonlinear programming relaxation. If $\bar{x}$ has fractional binary variables, they are sorted in non-decreasing order of their infeasibility, and cuts are generated for each variable in this order.

During the solution of (CGLP) by column generation procedure, only columns (linearizations of nonlinear constraints) that have a reduced cost greater than $\epsilon \cdot (\theta^t + 1)$ are included in the next (CGLP). The reduced cost of the column with respect to point y^t and constraint j can be calculated as $\lambda^t g_j(y^t)$, and the reduced cost with respect to point z^t and constraint j can be calculated as $\mu^t g_j(z^t)$. In Algorithm 2, we give the details for separation of a fractional point $\bar{x}$ with the disjunction on variable x_i. Algorithm 2 is terminated after at most τ iterations. This limits the total effort we spend generating cuts and helps to maintain numerical stability of (CGLP).

If a cut is generated by Algorithm 2 that separates $\bar{x}$ by at least δ, then it is added to our polyhedral outer approximation C^s and the linear program is solved to obtain a new point $\bar{x}$ that we wish to separate from the polyhedral outer approximation. (Note that this requires changing the objective function coefficients of (CGLP)). However, the fractional variable for which we will generate a disjunctive cut is still based on the original ordering. After generating disjunctive cuts for all of the fractional variables based on the original order, the

Input: $\bar{x}, i$
  $t \leftarrow 0$
  repeat
    Solve CGLP. Let $(\alpha^{+^t}, \alpha^{-^t}, \beta^t, u^t, v^t, u_0^t, v_0^t)$ and $(\tilde{y}^t, \tilde{z}^t, \lambda^t, \mu^t, \theta^t)$ be the primal and dual solutions respectively
    if $\lambda^t > \kappa$ **then**
      $y^t \leftarrow \tilde{y}^t / \lambda^t$
      for $j \in J$ **do**
        if $\lambda^t g_j(y^t) > \epsilon \cdot (\theta^t + 1)$ **then**
          $(-\nabla g_j(y^t), \mathbf{0}, \nabla g_j(y^t) y^t - g_j(y^t), 0, 0)^T$ as a column to CGLP
        end if
      end for
    end if
    if $\mu^t > \kappa$ **then**
      $z^t \leftarrow \tilde{z}^t / \mu^t$
      for $j \in J$ **do**
        if $\mu^t g_j(z^t) > \epsilon \cdot (\theta^t + 1)$ **then**
          $(\mathbf{0}, -\nabla g_j(z^t), 0, \nabla g_j(z^t)^T z^t - g_j(z^t), 0)^T$ as a column to CGLP
        end if
      end for
    end if
    $t \leftarrow t + 1$
  until No new columns are generated **or** $t > \tau$
  $\alpha \leftarrow \alpha^{+^t} - \alpha^{-^t}$, $\beta \leftarrow \beta^t$, $\theta \leftarrow \theta^t$
  return $\alpha x \geq \beta$ **and** t **and** θ

Algorithm 2: Iterative generation of CGLP.

variables are again ordered according to the current fractional solution $\bar{x}$. The procedure is repeated until the point $\bar{x}$ can no longer be separated by disjunctive cuts. Note that disjunctive cuts generated by the algorithm are never included in the formulation of (CGLP), so the cuts we generate are always of rank one. The details are given in Algorithm 3.

5 Computation of the Rank One Simple Disjunctive Closure

Significant advantages of the cut generation procedure outlined in Algorithm 2 are that it requires only the solution of linear programs and that it may be terminated at any step. If the objective value of the separation problem at termination is positive, then the current fractional point $\bar{x}$ is excluded from the feasible region by the generated inequality. These computational advantages allow us to perform the first (to our knowledge) significant computational study estimating the strength of disjunctive cuts for convex MINLP. Specifically, in

$\bar{x} \leftarrow \text{argmin}_{x \in C^0}\{c^T x\}$
$r \leftarrow 0$, $l \leftarrow 0$, $s \leftarrow 0$
repeat
  newcut $\leftarrow$ **false**
  $F \leftarrow \{x_i \mid \rho < \bar{x}_i < 1 - \rho, i \in B\}$
  Order $x_{i_1}, x_{i_2}, \ldots, x_{i_{|F|}}$ such that $\max\{\bar{x}_{i_p}, 1 - \bar{x}_{i_p}\} \geq \max\{\bar{x}_{i_q}, 1 - \bar{x}_{i_q}\}$ if $p < q$
  for $p = 1$ to $|F|$ **do**
    if $\rho < \bar{x}_{i_p} < 1 - \rho$ **then**
      Call Algorithm 2 with $\bar{x}$ and i_p to get $\alpha x \geq \beta$, t and θ
      if $\theta > \delta$ **then**
        newcut $\leftarrow$ **true**
        $C^{s+1} = \{x \in C^s \mid \alpha x \geq \beta\}$
        $s \leftarrow s + 1$
        $\bar{x} \leftarrow \text{argmin}_{x \in C^s}\{c^T x\}$
      end if
      $l \leftarrow l + t$
    end if
  end for
  $r \leftarrow r + 1$
until *newcut* = **false or** $r \geq R$ **or** $l \geq T$
return C^s

Algorithm 3: Outer Loop for the Generation of Rank One Simple Disjunctive Cuts.

this section, we will report results on an experiment aimed at estimating the strength of the *rank one disjunctive closure*.

For this experiment, disjunctive cuts were generated only for the set of binary decision variables $B \subseteq I$. In our test suite, only a small fraction of the instances contain general integer variables. In this case, the *rank one disjunctive closure* of R is

$$\mathcal{C} \stackrel{\text{def}}{=} \cap_{i \in B}(\text{conv}(R_i^{0-} \cup R_i^{0+})).$$

The set $\mathcal{C}$ can provide a much better approximation to the convex hull of feasible solutions than R. We measure the improvement in terms of the additional integrality gap closed. Specifically, let $z_R = \min_{x \in R}\{c^T x\}$ and $z_{\mathcal{C}} = \min_{x \in \mathcal{C}}\{c^T x\}$. The percentage gap closed by the rank one disjunctive cut closure is

$$100 \left(\frac{z_{\text{MINLP}} - z_{\mathcal{C}}}{z_{\text{MINLP}} - z_R} \right).$$

When reporting this improvement for instances for which the optimal solution value z_{MINLP} is not known, we use instead the best known value.

The strength of disjunctive cuts are tested on a suite of convex MINLPs from the MacMINLP collection [28], GAMS MINLP World[12], the collection on the website of the IBM-CMU research group [32] and instances we created ourselves.

The test suite comprises 207 convex problems covering a wide range of applications such as multi-product batch plant design problems [35], layout design problems [13], synthesis design problems [19], retrofit planning [31], stochastic service system design problems [20], cutting stock problems [25], quadratic uncapacitated facility location problems [24] and network design problems [11]. The computational experiments were run on a cluster of identical 64-bit Intel Core2 Duo microprocessors clocked at 3.00 GHz, each with 2 GB RAM.

In theory, the gap closed by closure should not depend on the mechanism by which it is generated. However, for practical reasons we must impose an overall time limit in our experiments, and hence the gap closed depends on the effectiveness of the separation procedure. We found that by mildly limiting the effort on the individual cut generation, more gap could be closed within the allowed time limit of 8 hours, leading to a better approximation of the true closure results. Thus, we set $\epsilon = 10^{-6}$, $\delta = 10^{-6}$, $\kappa = 10^{-2}$, $\rho = 10^{-4}$ and $\tau = 30$ in Algorithms 2 and 3. R and T in Algorithm 3 are set to very large numbers.

Characteristics of the instances and closure results are given in Table 1. The first 7 columns in Table 1 list the instance family, whether or not the instance has a nonlinear objective function, the number of instances in the family, average number of variables, average number of binary variables, average number of linear constraints, and average number of nonlinear constraints for the instances in the family. In the next 3 columns, the percentage gap closed by disjunctive cuts at the root node is given. The last column indicates for how many instances we hit the time limit for that family of instances.

Table 1: Closure results.

Instance Family	**NL Ob?**	**# of ins**	**Average**				**% gap closed**			**Hit limit**
			Var	**Bin**	**LC**	**NLC**	**min**	**ave**	**max**	
Batch	√	10	334.6	123	1089.1	1	44.87	53.53	64.19	10
CLay		12	116.7	35.3	138.3	40	7.47	40.81	72.74	0
FLay		10	158	28	183	4	28.72	50.72	99.99	3
fo-m-o		9	112.2	41.6	194.3	13.6	0.00	2.17	19.56	1
nd		5	574	37.6	283.8	37.6	73.03	85.00	93.05	0
RSyn		48	922.3	251	1716.3	34.2	60.49	88.50	100	0
safetyLay		3	120.7	38	111	34.7	100	100	100	0
SLay	√	14	336	92	437	0	37.61	69.35	86.78	6
sssd		14	162.4	135.5	50	20.1	99.40	99.68	99.82	0
Syn		48	366.3	95	660	34.2	95.83	99.32	100	0
trimloss		12	279.2	227.5	133.3	6	0.00	6.41	14.41	4
uflquad	√	10	1571	23.5	1613	0	0.00	6.41	13.21	10
others	√	12	205.4	86.4	206	3.3	0.00	47.57	100	3
Total		207	487.5	127.6	767.9	22.3	0.00	69.42	100	37

Results in Table 1 shows that on average, 69% of the gap is closed at the root node by adding rank one disjunctive cuts from single variable disjunctions. For specific instance families the results are even more striking. For example, disjunctive cuts close nearly the entire gap for instances in the RSyn, Syn, and sssd families.

6 Branch-and-Cut Framework Results

In this section, we report our preliminary experience using our disjunctive cut separation procedure within a general purpose solver for convex MINLP. We implemented our iterative cut generation method within FilMINT [1]. FilMINT is based on the LP/NLP-Based Branch-and-Bound algorithm of Quesada and Grossmann [30]. The algorithm works by creating a *reduced master-problem* that is a polyhedral relaxation of convex hull of the feasible points, $\text{conv}\{x | x \in R, x_I \in \mathbb{Z}^{|I|}\}$. Specifically, FilMINT solves the problem

$$\begin{aligned} z_{\text{MINLP}} = \text{minimize} \quad & c^T x \\ \text{subject to} \quad & g(\bar{x}) + \nabla g(\bar{x})^T (x - \bar{x}) \leq 0 \quad \forall \bar{x} \in \mathcal{K} \qquad (\text{MP}(\mathcal{K})) \\ & x \in X, \quad x_I \in \mathbb{Z}^{|I|} \end{aligned}$$

by branch-and-cut, where the set $\mathcal{K}$ is a dynamically updated collection of points about which linearizations are taken. In our experiment, we generated disjunctive cuts after FilMINT preprocessed the original instance and generated its default cuts.

To make disjunctive cuts practically effective, the effort spent on generation must be limited. To that end, we followed the procedure explained in Section 4 with the following modifications:

- We set the parameters $\epsilon = 10^{-4}$, $\delta = 10^{-6}$, $\kappa = 10^{-2}$, $\rho = 10^{-4}$ and $\tau = 30$ in Algorithms 2 and 3.
- At each round, we generate cuts for at most half of the binary variables, changing $|F|$ to $\min\{|B|/2, |F|\}$ in Algorithm 3.
- The number of rounds is limited to 30 ($R = 30$ in Algorithm 3).
- The total number of iterations for the entire cut generation procedure is limited to 15000 ($T = 15000$ in Algorithm 3).
- We also terminate Algorithm 3 if the objective function value was improved by less than 1% over the last three rounds of cut generation.

A final change was to disable the addition of linearizations coming from the extended cutting plane (ECP)-based method or the Fixfrac method in FilMINT [1]. These linearizations are used to capture the nonlinear structure of the

problem within the LP/NLP-Based Branch-and-Bound algorithm. Our computational experience indicated that the disjunctive cuts provide sufficient information about the nonlinear structure, and that generating these extra linearizations was typically not helpful in conjunction with disjunctive inequalities.

Out of 207 instances we collected, 148 of them can be solved by FilMINT in less than five minutes. For these "easy" instances, summary results for solution times are given in Table 2. We see that the extra effort spent generating disjunctive cuts does not pay off for these instances. However, the average increase in computing time is very small.

Table 2: Solution times (sec.) for FilMINT and FilMINT with disjunctive cuts on 148 easy instances.

	FilMINT	**FilMINT with DC**
Arithmetic mean	24.4	25.4
Geometric mean	2.9	4.5

The remaining 59 hard instances were run, enforcing a three hour time limit. Within the time limit, FilMINT could only solve 30 of the instances, but with the practical disjunctive inequality separation procedure, 42 of the instances were solved in less than 3 hours. Additionally, the instances are solved more than three times as fast, on average.

The characteristics of the 42 solved instances are given in the Appendix in Table 4, and the detailed performance of FilMINT and FilMINT with disjunctive cuts on each instance is listed in Table 3. The relative improvement obtained by using disjunctive cuts can also be dramatically seen in the performance profiles (see [17]) of Figure 1.

7 Conclusion

We have introduced a computationally effective mechanism for computing disjunctive inequalities for convex MINLPs. The methodology relies only on solving a sequence of linear programs, giving it a significant computational advantages over the separation approach that relies on solving a nonlinear, nondifferentiable program. We have proved that our methodology is "complete." Specifically, if $\bar{x}$ is a point not in the convex hull of the union of the two convex sets defined by the disjunction, then the procedure will find a linear inequality separating $\bar{x}$ from the feasible region. Using the new cutting-plane procedure allows us for the first time to report a significant computational study aimed at measuring the strength of simple disjunctive inequalities for convex MINLPs. For many families of convex MINLPs, the disjunctive inequalities close a significant fraction of the gap between the nonlinear relaxation and the optimal value. Finally, the inequalities have been successfully incorporated into the software FilMINT, resulting in often dramatic performance improvements. Continuing work will

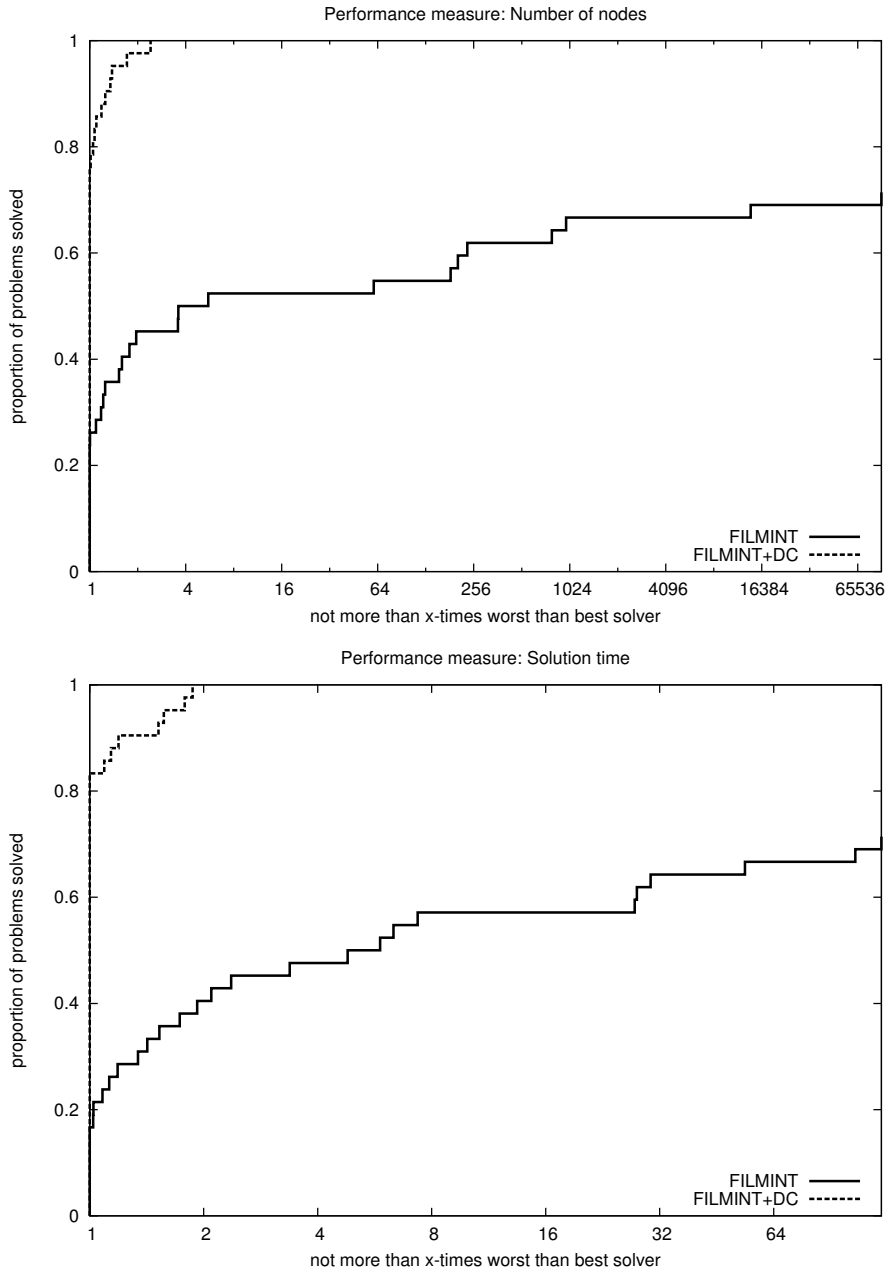

Figure 1: Performance profiles comparing FilMINT with and without disjunctive cuts.

aim to make the inequalities even more effective by testing additional normalization constraints, adding monoidal strengthening [7], and solving the separation problem for over only a subset of the variables [6].

Acknowledgements

The authors would like to thank Andrew Miller for his insightful comments on this work.

References

[1] K. Abhishek, S. Leyffer, and J. T. Linderoth. FilMINT: An outer-approximation-based solver for nonlinear mixed integer programs. *INFORMS Journal on Computing*, 2010. To appear.

[2] Alper Atamtürk and Vishnu Narayanan. Conic mixed integer rounding cuts. *Mathematical Programming*, 122:1–20, 2010.

[3] E. Balas. Disjunctive programming. *Annals of Discrete Mathematics*, 5:3–51, 1979.

[4] E. Balas. A modified lift-and-project procedure. *Mathematical Programming*, 79:19–31, 1997.

[5] E. Balas, S. Ceria, and G. Corneujols. A lift-and-project cutting plane algorithm for mixed 0-1 programs. *Mathematical Programming*, 58:295–324, 1993.

[6] E. Balas, S. Ceria, and G. Cornuejols. Mixed 0-1 programming by lift-and-project in a branch-and-cut framework. *Management Science*, 42:1229–1246, 1996.

[7] E. Balas and R. G. Jeroslow. Strengthening cuts for mixed integer programs. *European Journal of Operational Research*, 4:224–234, 1980.

[8] E. Balas and Michael Perregaard. A precise correspondence between lift-and-project cuts, simple disjunctive cuts. *Mathematical Programming, Series B*, 94:221–245, 2003.

[9] E. Balas and A. Saxena. Optimizing over the split closure. *Mathematical Programming*, 113:219–240, 2008.

[10] P. Bonami and M. Minoux. Using rank-1 lift-and-project closures to generate cuts for 0-1 MIPs, a computational investigation. *Discrete Optimization*, 2:288–307, 2005.

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

[12] M. R. Bussieck, A. S. Drud, and A.Meeraus. MINLPLib - a collection of test models for mixed-integer nonlinear programming. *INFORMS Journal on Computing*, 15(1), 2003.

[13] I. Castillo, J. Westerlund, S. Emet, and T. Westerlund. Optimization of block layout design problems with unequal areas: A comparison of MILP and MINLP optimization methods. *Computers and Chemical Engineering*, 30:54–69, 2005.

[14] S. Ceria and J. Soares. Convex programming for disjunctive optimization. *Mathematical Programming*, 86:595–614, 1999.

[15] M. Conforti, G. Cornuéjols, and G. Zambelli. Polyhedral approaches to mixed integer linear programming. In M. Jünger, T. Liebling, D. Naddef, W. Pulleyblank, G. Reinelt, G. Rinaldi, and L. Wolsey, editors, *50 Years of Integer Programming 1958—2008*. Springer, 2009.

[16] S. Dash, O. Günlük, and A. Lodi. MIR closures of polyhedral sets. *Mathematical Programming*, 121:33–60, 2010.

[17] Elizabeth Dolan and Jorge Moré. Benchmarking optimization software with performance profiles. *Mathematical Programming*, 91:201–213, 2002.

[18] S. Drewes. *Mixed Integer Second Order Cone Programming*. PhD thesis, Technische Universität Darmstadt, 2009.

[19] M. A. Duran and I. Grossmann. An outer-approximation algorithm for a class of mixed-integer nonlinear programs. *Mathematical Programming*, 36:307–339, 1986.

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

[21] M. Fischetti and A. Lodi. Optimizing over the first Chvátal closure. *Mathematical Programming, Series B*, 110:3–20, 2007.

[22] M. Fischetti, A. Lodi, and A. Tramontani. On the separation of disjunctive cuts. *Mathematical Programming*, 2010. To appear.

[23] R. E. Gomory. An algorithm for the mixed integer problem. Technical Report RM-2597, The RAND Corporation, 1960.

[24] O. Günlük, J. Lee, and R. Weismantel. MINLP strengthening for separable convex quadratic transportation-cost UFL. Technical Report RC24213 (W0703-042), IBM Research Division, March 2007.

[25] I. Harjunkoski, T. Westerlund, R. Porn, and H. Skrifvars. Different transformations for solving non-convex trim loss problems by MINLP. *European Journal of Operational Research*, 105:594–603, 1998.

[26] Jean-Baptiste Hiriart-Urruty and Claude Lemarechal. *Convex Analysis and Minimization Algorithms I: Fundamentals (Grundlehren Der Mathematischen Wissenschaften)*. Springer, October 1993.

[27] J.E. Kelley. The cutting-plane method for solving convex programs. *J. SIAM*, 8:703–712, 1960.

[28] S. Leyffer. MacMINLP: Test problems for mixed integer nonlinear programming, 2003. `http://www-unix.mcs.anl.gov/~leyffer/macminlp`.

[29] G. Nemhauser and L. Wolsey. A recursive procedure for generating all cuts for 0-1 mixed integer programs. *Mathematical Programming*, 46:379–390, 1990.

[30] I. Quesada and I. E. Grossmann. An LP/NLP based branch–and–bound algorithm for convex MINLP optimization problems. *Computers and Chemical Engineering*, 16:937–947, 1992.

[31] N. Sawaya. *Reformulations, relaxations and cutting planes for generalized disjunctive programming*. PhD thesis, Chemical Engineering Department, Carnegie Mellon University, 2006.

[32] N. W. Sawaya, C. D. Laird, and P. Bonami. A novel library of nonlinear mixed-integer and generalized disjunctive programming problems. In preparation, 2006.

[33] R. Stubbs and S. Mehrohtra. Generating convex polynomial inequalities for mixed 0-1 programs. *Journal of Global Optimization*, 24:311–332, 2002.

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

[35] A. Vecchietti and I. E. Grossmann. LOGMIP: a disjunctive 0-1 non-linear optimizer for process system models. *Computers and Chemical Engineering*, 23(4-5):555 – 565, 1999.

[36] Y. Zhu and T. Kuno. A disjunctive cutting-plane-based branch-and-cut algorithm for 0-1 mixed-integer convex nonlinear programs. *Industrial Engineering Chemistry Research*, 45(1):187–196, 2006.

Table 3: Solution statistics for FilMINT with and without disjunctive cuts.

Problem	**FilMINT**		**FilMINT with DC**	
	Node	**Time**	**Node**	**Time**
FLay05H	106823	1771.5	106467	1639.2
FLay05M	82677	714.8	86731	603.4
fo7	247017	302.3	203434	213.0
fo8	601114	995.0	480980	651.3
fo9	4729736	10800.0	1972581	4217.0
nd-12	19078	1119.8	5337	152.5
nd-13	18242	921.3	11953	389.8
nd-14	113635	9939.8	96427	9732.9
o7_2	2864359	4947.0	2884020	3196.0
o7	2433961	4284.1	6901545	9245.3
RSyn0810M03M	129829	488.3	2150	101.8
RSyn0810M04M	152101	869.8	747	137.2
RSyn0815M02M	515741	1286.7	2813	46.8
RSyn0815M03M	2397805	8324.6	10286	298.8
RSyn0815M04M	2812556	10800.0	6208	472.1
RSyn0820M02M	2335654	6999.2	2952	56.8
RSyn0820M03M	2369872	10800.0	3069	190.8
RSyn0820M04M	1383792	10800.0	7567	660.4
RSyn0830M02M	1685598	6502.3	1735	61.8
RSyn0830M03M	1508031	10800.0	4511	268.1
RSyn0830M04M	937614	10800.0	9099	1057.4
RSyn0840M02M	2777282	10800.0	1617	117.0
RSyn0840M03M	1681035	10800.0	2116	251.7
RSyn0840M04M	696771	10800.0	6902	1213.5
Safety3	2026677	5736.9	2729701	8716.0
SLay09H	53071	898.0	73197	1599.2
SLay10M	1220630	10800.0	437652	5087.8
sssd-16-7-3	795223	403.8	499476	233.6
sssd-16-8-3	3421746	1691.8	1750796	807.8
sssd-17-7-3	724435	380.3	408062	197.8
sssd-18-7-3	757942	367.2	692627	326.1
sssd-18-8-3	4773204	2468.1	1328315	731.7
sssd-20-8-3	1036620	535.2	1773151	839.6
sssd-22-8-3	10766956	6760.2	1946252	1156.2
Syn30M04M	276797	1365.0	3	45.1
Syn40M02M	153675	363.3	11	6.8
Syn40M03M	3508363	10800.0	94	37.3
Syn40M04M	1996038	10800.0	23	58.5
uflquad-15-60	2437	403.4	2681	480.5
uflquad-15-80	3647	1249.9	3693	1365.4
uflquad-20-40	4101	548.5	4401	536.2
uflquad-25-40	6083	1216.3	7617	1384.3
Average	1526856.4	4844.1	582595.2	1394.9

Table 4: Test set statistics.

Problem	**NL Obj**	**Vars**	**Ints**	**L Cons**	**NL Cons**
FLay05H		382	40	460	5
FLay05M		62	40	60	5
fo7		114	42	197	14
fo8		146	56	257	16
fo9		182	72	325	18
nd-12		600	40	289	40
nd-13		640	40	316	40
nd-14		816	48	369	48
o7_2		114	42	197	14
o7		114	42	197	14
RSyn0810M03M		615	252	1434	18
RSyn0810M04M		820	336	2116	24
RSyn0815M02M		470	188	959	22
RSyn0815M03M		705	282	1614	33
RSyn0815M04M		940	376	2386	44
RSyn0820M02M		510	208	1046	28
RSyn0820M03M		765	312	1767	42
RSyn0820M04M		1020	416	2620	56
RSyn0830M02M		620	248	1232	40
RSyn0830M03M		930	372	2091	60
RSyn0830M04M		1240	496	3112	80
RSyn0840M02M		720	288	1424	56
RSyn0840M03M		1080	432	2424	84
RSyn0840M04M		1440	576	3616	112
Safety3	$\surd$	259	98	294	0
SLay09H	$\surd$	810	144	1044	0
SLay10M	$\surd$	290	180	405	0
sssd-16-7-3		161	133	51	21
sssd-16-8-3		184	152	56	24
sssd-17-7-3		168	140	52	21
sssd-18-7-3		175	147	53	21
sssd-18-8-3		200	168	58	24
sssd-20-8-3		216	184	60	24
sssd-22-8-3		232	200	62	24
Syn30M04M		640	240	1488	80
Syn40M02M		420	160	756	56
Syn40M03M		630	240	1314	84
Syn40M04M		840	320	1992	112
uflquad-15-60	$\surd$	915	15	960	0
uflquad-15-80	$\surd$	1215	15	1280	0
uflquad-20-40	$\surd$	820	20	840	0
uflquad-25-40	$\surd$	1025	25	1040	0